# Reinforcement Learning for Machine Translation: from Simulations to Real-World Applications

Dissertation

zur Erlangung der Doktorwürde
der Neuphilologischen Fakultät
der Ruprecht-Karls-Universität Heidelberg

vorgelegt von

Julia Kreutzer

06. September 2020

Institut für Computerlinguistik
Ruprecht-Karls-Universität Heidelberg

| | |
|---|---|
| Betreuer und Erstgutachter: | Prof. Dr. Stefan Riezler<br>Institut für Computerlinguistik<br>Ruprecht-Karls-Universität Heidelberg |
| Zweitgutachter: | Prof. Dr. André Martins<br>Unbabel & Instituto Superior Técnico &<br>Instituto de Telecomunicações, Lissabon |
| Datum der Einreichung: | 28.01.2020 |
| Datum der Disputation: | 30.03.2020 |

Hierbei handelt es sich um eine Heidelberger Dissertation.

Für Opa Tom und Oma Eka

# Abstract

If a machine translation is wrong, how we can tell the underlying model to fix it? Answering this question requires (1) a machine learning algorithm to define update rules, (2) an interface for feedback to be submitted, and (3) expertise on the side of the human who gives the feedback. This thesis investigates solutions for machine learning updates, the suitability of feedback interfaces, and the dependency on reliability and expertise for different types of feedback.

We start with an interactive online learning scenario where a machine translation (MT) system receives bandit feedback (i.e. only once per source) instead of references for learning. Policy gradient algorithms for statistical and neural MT are developed to learn from absolute and pairwise judgments. Our experiments on domain adaptation with simulated online feedback show that the models can largely improve under weak feedback, with variance reduction techniques being very effective.

In production environments offline learning is often preferred over online learning. We evaluate algorithms for counterfactual learning from human feedback in a study on eBay product title translations. Feedback is either collected via explicit star ratings from users, or implicitly from the user interaction with cross-lingual product search. Leveraging implicit feedback turns out to be more successful due to lower levels of noise. We compare the reliability and learnability of absolute Likert-scale ratings with pairwise preferences in a smaller user study, and find that absolute ratings are overall more effective for improvements in down-stream tasks. Furthermore, we discover that error markings provide a cheap and practical alternative to error corrections.

In a generalized interactive learning framework we propose a self-regulation approach, where the learner, guided by a regulator module, decides which type of feedback to choose for each input. The regulator is reinforced to find a good trade-off between supervision effect and cost. In our experiments, it discovers strategies that are more efficient than active learning and standard fully supervised learning.

# Kurzfassung

Wie können wir maschinellen Übersetzungssystemen (MT) beibringen, von Fehlern zu lernen? Die Beantwortung dieser Frage erfordert (1) einen maschinellen Lernalgorithmus, (2) eine Schnittstelle für die Übermittlung von Feedback und (3) Fachwissen. Diese Dissertation untersucht Algorithmen für maschinelles Lernen mit schwachem Feedback, verschiedene Feedback-Schnittstellen, sowie die Zuverlässigkeit verschiedener Feedback-Arten.

In einem interaktiven Online-Lernszenario erhält ein MT Modell "Bandit"-Feedback (d.h. für nur je eine Übersetzung) anstelle von Referenzübersetzungen zum Lernen. Dafür werden Policy-Gradienten-Algorithmen für statistische und neuronale MT entwickelt, die von absoluten und paarweisen Bewertungen lernen. Simulationsexperimente zeigen, dass sich die Modelle selbst mit schwachem Feedback erheblich verbessern und von Varianzreduktionstechniken profitieren.

In Produktionsumgebungen wird Offline-Lernen allerdings oft dem Online-Lernen vorgezogen. Daher evaluieren wir Algorithmen für kontrafaktisches Lernens mit menschlichem Feedback für Übersetzungen von eBay-Produkten. Das Feedback dabei wird entweder explizit durch Nutzerbewertungen oder implizit über Interaktionen mit der Website gesammelt. Die Nutzung impliziten Feedbacks erweist sich aufgrund geringeren Rauschens als erfolgreicher. In einer weiteren Studie vergleichen wir die Zuverlässigkeit und Erlernbarkeit von absoluten mit relativen Bewertungen, wobei absoluten Bewertungen effektiver für Verbesserungen am MT Modell sind. Darüber hinaus stellen Fehlermarkierungen eine kostengünstige und praktische Alternative zu Fehlerkorrekturen dar.

Anstelle die Art der Feedbacksammlung im Voraus zu entscheiden, lassen wir das Modell diese Entscheidung in der letzten Studie selbst treffen. Wir entwickeln einen Selbstregulierungsansatz für interaktiven Lernen: Das lernende Modell entscheidet mithilfe eines Regulierungsmoduls, welche Art von Feedback für welche Eingabe gewählt wird. Der Regulator wird so trainiert, dass er einen guten Kompromiss zwischen Lerneffekt und Kosten findet. In unseren Experimenten verfolgt er Strategien, die effizienter sind als aktives Lernen und vollüberwachtes Lernen.

# Contents

## Appendix

# List of Figures

# List of Tables

# List of Algorithms

# List of Acronyms

| | |
|---|---|
| BIN | Binary |
| BEER | Better Evaluation as Ranking (Stanojević and Sima'an, 2014) |
| biLSTM | bidirectional LSTM |
| BLEU | Bilingual Evaluation Understudy (Papineni et al., 2002) |
| BPE | Byte-Pair Encoding |
| ChrF | Character F-score (Popović, 2015) |
| CONT | Continuous |
| CV | Control Variate |
| DC | Doubly-Controlled Estimation |
| DPM | Deterministic Propensity Matching |
| EP | Europarl |
| ER | Expected Reward |
| GRU | Gated Recurrent Unit |
| KSMR | Keystroke Mouse Action Ratio |
| LSTM | Long Short-Term Memory |
| LMEM | Linear Mixed Effects Model |
| METEOR | Metric for Evaluation of Translation with Explicit Ordering (Lavie and Denkowski, 2009) |
| MDP | Markov Decision Process |
| ML | Machine Learning |
| MLE | Maximum Likelihood Estimation |
| MRT | Minimum Risk Training |
| MT | Machine Translation |
| NC | News Commentary |
| NLP | Natural Language Processing |

| | |
|---|---|
| NMT | Neural Machine Translation |
| SMT | Statistical Machine Translation |
| PG | Policy Gradient |
| PR | Pairwise Ranking |
| RL | Reinforcement Learning |
| RNN | Recurrent Neural Network |
| sBLEU | Per-sentence BLEU |
| SGD | Stochastic Gradient Descent |
| sQE | sentence-level Quality Estimation |
| TER | Translation Error Rate (Snover et al., 2006) |
| UNK | Unknown word |
| WMT | Workshop/Conference on Machine Translation |

# Chapter 1

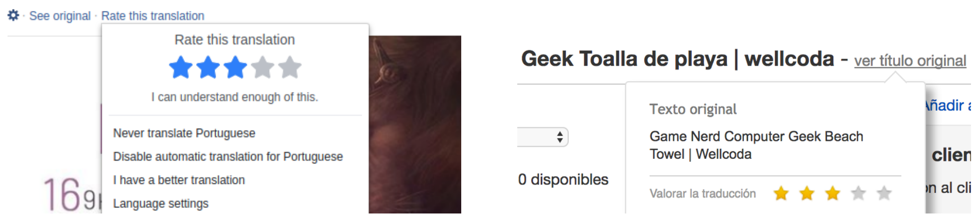
# Introduction

In Natural Language Processing (NLP), most models are trained with full supervision through gold standards. For the task of Machine Translation (MT), these are reference translations. The generation or collection of these references requires large amounts of expertise and cost, for example by professional translators or skilled bilinguals. With millions of translations collected e.g., from multilingual news, parliament proceedings or web crawls, the quality of MT models has now reached a level where they are widely deployed in products for end users, e.g., as a stand-alone translation service (Google Translate[1], Bing Translator[2], DeepL Translator[3]) or integrated into e-commerce websites (eBay, Amazon) or social media platforms (Facebook, Instagram).

Users translate custom requests, products, postings, websites—and whatever their use case is, they interact with the system and use its output for their purpose. In most of the cases, they have much lower expertise in the translation task than professional translators, but they might still be able to indicate whether the translation is good enough (*Does the output make any sense?*), point out obvious mistakes (*Does this word fit the context?*), or tell whether the translation helped them to reach their goal (*Does this translation serve its purpose?*). This type of *weak* feedback collected from non-experts and users is the learning opportunity that this thesis aims to exploit. We phrase it as a Reinforcement Learning (RL) problem: In interaction with the feedback provider, the model is trained to maximize the obtained reward.

Thanks to advances in Deep Learning (DL), the end-to-end training of deep artificial neural networks with backpropagation, recent progress in RL reported first successes in scaling to large action spaces (Mnih et al., 2015; Silver et al., 2016), and, as a result, got adopted for Sequence-to-Sequence learning (Seq2Seq) (Bahdanau et al., 2015) with exponential output spaces. In NLP, Seq2Seq with reinforcement learning signals was successfully applied to semantic parsing (Liang et al., 2017), summarization (Paulus et al., 2017), or the task in the focus of this thesis—machine translation (Bahdanau et al., 2017). However, these RL approaches *simulate* reward signals from gold standards with task-specific

[1] `https://translate.google.com/`
[2] `http://www.bing.com/translator`
[3] `https://www.deepl.com/translator`


See original · Rate this translation
Rate this translation
I can understand enough of this.
Never translate Portuguese
Disable automatic translation for Portuguese
I have a better translation
Language settings
Geek Toalla de playa | wellcoda - ver título original
Texto original
Game Nerd Computer Geek Beach Towel | Wellcoda
0 disponibles
Valorar la traducción

(a) Rating interface: Facebook. (b) Rating interface: eBay.

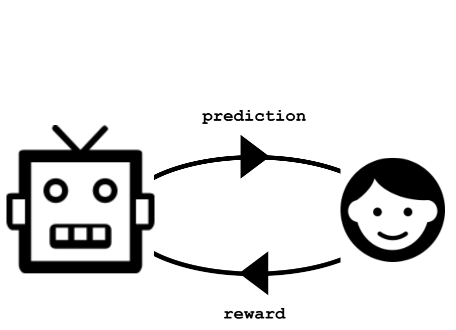
Figure 1: Example interfaces for Likert-scale ratings in commercially deployed MT systems: Facebook (a) and eBay (b).[4]

evaluation metrics such as BLEU, F1 score, or ROUGE as reward functions, in order to reduce the mismatch between training objective and evaluation metric (Ranzato et al., 2016).

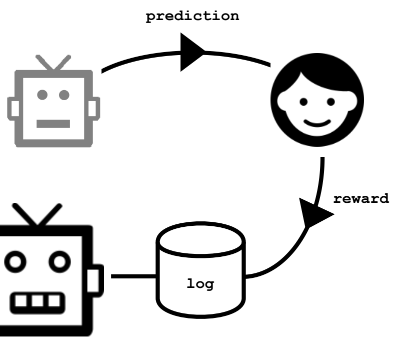
This thesis, in contrast, aims to improve Seq2Seq learning through reinforcement from *weak human feedback* directly. The central research question of this thesis is *how and when machine translation can be improved with weak feedback*. We approach this by developing policy learning algorithms suitable for online and offline feedback, with absolute and relative feedback of varying strength on sequence- and token-level, and test them in simulated and real-world applications. The goal is to enable users of deployed machine translation systems, such as in the examples provided above, to express the quality of the translation and their satisfaction through simple interfaces that do not require much time nor effort, and are flexible enough to account for various levels of expertise. Furthermore, these feedback mechanisms should offer a way to personalize MT systems to specific contexts and domains that general models are not built for, without having to provide reference translations. From a broader perspective, the integration of such feedback mechanisms pave the way to a more natural interaction with machine learning models, and a technique for humans to control their output.

At the current stage, interfaces for gathering weak feedback exist, for example for Facebook and eBay translations (Figure 1), but it is not transparent whether and how the feedback is used to improve the system. The collected feedback is often utilized for "static" evaluation purposes or market analysis. In scenarios where the human is considered the teacher or expert for an NLP system, for example in post-editing for Computer-Aided Translation (CAT), model adaptation from human feedback is adopted more widely, as for example in commercial CAT tools like MateCat[5] and Lilt[6] (Wuebker et al., 2015; Sanchis-Trilles et al., 2014, inter alia). This adaptation mode requires expert knowledge and is thus limited

[5] https://www.matecat.com/
[6] https://lilt.com/



(a) Collecting online feedback.



(b) Collecting offline feedback.

Figure 2: Human-in-the-loop machine learning with online or offline feedback. The robot head represents the machine learner and the human face the human that the machine is interacting with. Model predictions are shown to the human to receive feedback in the form of rewards. If the feedback is online, it can directly be used to update the model. If it is offline, it is first collected in a log. In that case logging and learning system might not be identical.

to the small group of translators. We show that weak feedback from non-experts can serve for model adaptation. We analyze the characteristics of this type of feedback and provide recommendations for a successful implementation in practice.

Figure 2 illustrates how the interaction with humans in the loop differs depending on the feedback availability: If feedback is available online, the machine learning system can directly use it to update its parameters. If it is collected offline, interacting and learning system might not be identical. It is first stored in a log and then used for updating the parameters of the learning system. The larger the gap between logging and learning system, the harder is the learning problem. We have to address the *counterfactual* question, which feedback the learning model would have gotten, had it been in place of the logging system.

The distinction between these two learning paradigms plays a crucial role in the design of our algorithms and user studies: Only offline feedback can easily be collected for MT systems deployed in the real world, due to the requirement of real-time feedback and updates. Thus our empirical studies with online feedback will be limited to simulations. In addition to the algorithms, we also emphasize the importance of the quality and characteristics of the collected feedback. We show how normalization and filtering techniques and the design of suitable interfaces can influence the success of the machine learner.

## 1.1 Contributions

The main contributions of this thesis can be summarized as the following:

1. We develop algorithms for bandit structured prediction in online and offline learning scenarios.
2. These algorithms are applied and evaluated in simulated and real-world human-in-the-loop machine translation tasks on standard public domains (news and talks) and specialized user-generated domains (e-commerce products).
3. We provide an analysis of the impact on feedback interfaces on the success of reinforcement learning from human feedback.
4. The cost and effectiveness of weak vs. strong supervision is quantified.
5. We propose a novel unified perspective on cost-sensitive interactive learning across supervision modes by approaching it as a meta-learning problem.

We believe that these contributions smooth the path towards larger scale adoption of human-in-the-loop learning paradigms for NLP applications. We provide a broad investigation of the human factor in interactive structured prediction and the importance of the role of interfaces in human-in-the-loop machine learning. Our algorithms are developed for and applied to a variety of domains, feedback modalities, and learning paradigms.

## 1.2 Publications

Large parts of this thesis have appeared in peer-reviewed publications. We list these publications below. The individual contribution of the author of this thesis to these publications is described in the beginning in each of the chapters.

1. Artem Sokolov, Julia Kreutzer, Christopher Lo, and Stefan Riezler. 2016a. Learning structured predictors from bandit feedback for interactive NLP. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany
2. Artem Sokolov, Julia Kreutzer, Stefan Riezler, and Christopher Lo. 2016b. Stochastic structured prediction under bandit feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*. Barcelona, Spain
3. Julia Kreutzer, Artem Sokolov, and Stefan Riezler. 2017. Bandit structured prediction for neural sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*. Vancouver, Canada

4. Artem Sokolov, Julia Kreutzer, Kellen Sunderland, Pavel Danchenko, Witold Szymaniak, Hagen Fürstenau, and Stefan Riezler. 2017. A shared task on bandit learning for machine translation. In *Proceedings of the Second Conference on Machine Translation (WMT)*. Copenhagen, Denmark

5. Julia Kreutzer, Shahram Khadivi, Evgeny Matusov, and Stefan Riezler. 2018a. Can neural machine translation be improved with user feedback? In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Industry Track (NAACL-HLT)*. New Orleans, LA, USA

6. Julia Kreutzer, Joshua Uyheng, and Stefan Riezler. 2018b. Reliability and learnability of human bandit feedback for sequence-to-sequence reinforcement learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*. Melbourne, Australia

7. Julia Kreutzer and Stefan Riezler. 2019. Self-regulated interactive sequence-to-sequence learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*. Florence, Italy

8. Julia Kreutzer, Nathaniel Berger, and Stefan Riezler. 2020. Correct me if you can: Learning from error corrections and markings. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation (EAMT)*. Virtual

The Joey NMT toolkit that was developed for the experiments in Chapters 6 and 7 was published in the following paper:

- Julia Kreutzer, Jasmijn Bastings, and Stefan Riezler. 2019. Joey NMT: A minimalist NMT toolkit for novices. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*. Hong Kong, China

## 1.3 Released Software and Data

Over the course of the dissertation, the landscape of tools and methods in machine translation changed drastically and rapidly. While the works from 2016 were based on the `cdec` framework for SMT (Dyer et al., 2010), later works were implemented on top of deep learning libraries (TensorFlow and PyTorch) and NMT toolkits based thereon, namely Neural Monkey (Helcl and Libovický, 2017) and Joey NMT (Kreutzer et al., 2019). The following three MT toolkits have been "banditized" over the course of the research for this dissertation:

1. `bandit-cdec`, a version of `cdec` (Dyer et al., 2010) that can learn online from absolute and relative feedback, used in (Sokolov et al., 2016a,b, 2017): `https://github.com/juliakreutzer/bandit-cdec`.

2. `bandit-neuralmonkey`, a version of `Neural Monkey` (Helcl and Libovický, 2017) that can learn from offline and online absolute feedback, used in (Kreutzer et al., 2017; Sokolov et al., 2017; Kreutzer et al., 2018b): `https://github.com/juliakreutzer/bandit-neuralmonkey`.

3. `bandit-joey`, a version of `Joey NMT` (Kreutzer et al., 2019) that can learn from offline absolute feedback on sequence and token-level, and meta-learn to balance various supervision strengths, used in (Kreutzer and Riezler, 2019) and (Kreutzer et al., 2020): `https://github.com/juliakreutzer/bandit-joeynmt`.

The feedback collected in (Kreutzer et al., 2018b) and (Kreutzer et al., 2020) was compiled into a data set of human machine translation ratings (five-star and pairwise ratings, corrections and error markings), called "HumanMT", available for download at `http://www.cl.uni-heidelberg.de/statnlpgroup/humanmt/`.

## 1.4 Outline

Each chapter approaches the central research question—how machine translation can be improved from weak human feedback—from a different angle, with the focus either on algorithms, human interaction, or applications, in order to find answers for smaller aspects of this general question (see Table 1 on page 14 for a systematic comparison). The following research questions may serve as a guidance to the individual chapters of this thesis:

1. Which algorithms are suitable for learning from weak feedback in MT?
   → Online learning in Chapter 3 vs. offline learning in Chapters 4 and 5.

2. How can one learn from feedback for MT in production?
   → Chapter 4

3. What is the influence of the feedback interface?
   → Chapter 5

4. How much cheaper is weak feedback than post-edits?
   And how effective is it in comparison? → Chapter 6

5. How can we find cost-efficient interactive learning strategies?
   → Chapter 7

| | Ch. 3 | Ch. 4 | Ch. 5 | Ch. 6 | Ch. 7 |
|---|---|---|---|---|---|
| **Policy** | | | | | |
| **Learning** | | | | | |
| online | x | | | | x |
| offline | | x | x | x | |
| **Parametrization** | | | | | |
| shallow | x | | | | |
| deep | x | x | x | x | x |
| **Feedback** | | | | | |
| **Source** | | | | | |
| simulation | x | x | x | x | x |
| human | | x | x | x | |
| **Type** | | | | | |
| absolute | x | x | x | x | x |
| pairwise | x | | x | | |
| **Granularity** | | | | | |
| token | | x | | x | x |
| sequence | x | x | x | x | |
| **Collection** | | | | | |
| explicit | x | x | x | x | x |
| implicit | | x | | | |
| **Data** | | | | | |
| **Domain** | | | | | |
| e-commerce | x | x | | | |
| TED | x | | x | x | x |
| news | x | x | | | |
| books | | | | | x |
| **Language** | | | | | |
| fr→en | x | x | | | |
| en→es | | x | | | |
| de→en | | | x | | x |
| en→de | | | | x | |

Table 1: Overview over policies, feedback, and data used in the experiments in each of the chapters. Gray entries mean that these were auxiliary experiments, e.g. for comparing the experiments with human feedback to simulated feedback.

We start with an overview of the background of this work in Chapter 2, including an outline of reinforcement and bandit learning and models of machine translation, and a summary of previous human-in-the-loop reinforcement learning studies. The remainder is grouped into three parts: Part I focuses on learning from online feedback in simulation, Part II investigates offline feedback, and Part III develops a meta-learning strategy that integrates reinforcement learning with fully supervised and unsupervised learning to find the most efficient interaction paradigm.

Chapter 3 presents the first reinforcement (or bandit) learning algorithms for machine translation that learn from online feedback in interaction with a simulated user. We first evaluate them on a range of simulated domain adaptation or personalization tasks, including a shared task on e-commerce data. We continue with the presentation of algorithms for learning from offline feedback and their evaluation on large-scale real human feedback data, elicited explicitly or implicitly from an e-commerce website in Chapter 4. Inspired by the difficulties caused by noise in the explicit ratings in these experiments, Chapter 5 investigates the effects of feedback interfaces (absolute vs relative feedback) on feedback reliability, learnability and effectiveness in a down-stream learning task in a smaller-scale experiment with semi-professional raters instead of users. Chapter 6 continues with a comparison of learning from strong supervision in the form of error corrections and weak supervision in the form of error markings, quantifying the trade-off between human supervision effort and machine improvement. Combining weakly-supervised learning with full supervision and self-training in one single training algorithm, Chapter 7 introduces a meta-learner for self-regulation that directly learns to balance supervision cost and effectiveness. We conclude this thesis with Chapter 8 discussing the limitations of the presented algorithms and training scenarios and pointing towards promising directions for future research. Table 1 gives an overview over the policies, feedback, and data used in each of the following chapters of this thesis.

# Chapter 2

# Background

This chapter describes the machine learning techniques and algorithms that this dissertation builds on. First, we introduce the task of machine translation and summarize the standard modeling approaches (Section 2.1). Then we discuss two paradigms for learning from interaction (Section 2.2), namely reinforcement and bandit learning, with a focus on human-in-the-loop approaches (Section 2.2.4). Finally, Section 2.3 discusses the challenges for applying reinforcement learning methods to the task of machine translation that will become relevant for the empirical studies conducted in later chapters.

## 2.1 Machine Translation

In the following, we will define the task of machine translation (Section 2.1.1), and then briefly describe two families of models, statistical and neural models, and their common training algorithms. The main focus hereby lies on neural approaches, as they now constitute state of the art and are used predominantly in this thesis. For a complete technical overview and in-depth details we refer the reader to the seminal book by Koehn (2009), and for an historic overview of early approaches to the book by Hutchins (2000).

### 2.1.1 Task Definition

**Modeling translation.** Given a source sentence $x$ consisting of a sequence of words, a translation model should produce the best translation of all possible translations $y \in \mathcal{Y}$ consisting of words in the target language (Brown et al., 1988, 1993):[1]

$$\arg\max_{y \in \mathcal{Y}} P(Y = y \mid X = x). \tag{1}$$

The notion of "the best" in translation is rather fuzzy since it varies from context to context, the task is simplified to produce one given reference translation. We

[1]The translation model was originally derived in a "noisy channel" model that describes the generative process of a translation with a target language model for modeling $p(y)$ and a noise model $p(x \mid y)$, such that: $\hat{y} = \arg\max_y p(y \mid x) = \arg\max_y p(x \mid y)p(y)$.

furthermore limit the scope to pairs of aligned sentences, which most of machine translation research has focused on.[2] From now on, we will write $p(y \mid x)$ in place for $P(Y = y \mid X = x)$.

**Decoding.** Decoding, the task of finding the best translation under the model, is NP-complete (Knight, 1999), since the number of possible translations grows exponentially with the length of the source sequence. Building a translation, one can choose one of many possible words of the target vocabulary for each of a number of positions in the sentence. Considering all possible options in all steps for producing a single translation is computationally too expensive, so instead of exact decoding, we resort to approximate decoding algorithms, such as greedy or beam search. In greedy decoding, search focuses on the single best scoring partial hypothesis in each step, while in beam search, a number of $k$ alternatives is considered and compared in each step.

**Discriminative training.** The objective for training a machine lerning model with parameters $\theta$ is to find the parameters that maximize the conditional log-likelihood of the reference translation given the source:

$$\arg\max_{\theta} \log p_{\theta}(y \mid x). \tag{2}$$

The trained models will be able to discriminate between good and bad translations, by assigning them either high or low probabilities. The probabilities are obtained from a scoring function that produces a scalar for a pair of $x$ and $y$. This scoring function is parametrized with weights $\theta$, building for example a log-linear or a neural model, as we will discuss in the following sections. The log-likelihood is differentiable with respect to the model weights, so they can be trained with gradient-based optimization, such as stochastic gradient ascent (Bottou, 2004). The highest scoring translation corresponds to the "best" translation under the translation model (the maximum a posterior translation, MAP). We speak of the training data for such models as parallel corpora $D$ which contain sentence-aligned pairs of sources and targets $(x, y) \in D$.

**Evaluation.** Using reference translations also enables us to use automatic metrics for translation quality evaluation instead of asking humans, which is costly and impracticable where frequent evaluations are needed. The metrics that we will use in the experiments of this thesis are listed below. Since none of these metrics in isolation is sufficient to judge translation quality (for example

[2]Document-level information has recently started to draw more attention, e.g. as it was introduced as dedicated shared task in the 2019 WMT evaluation (`http://www.statmt.org/wmt19/translation-task.html`).

BLEU neglects translations with a high recall)[3], we report several metrics instead of one to capture different aspects of the translation quality or to confirm general trends across metrics.

- **BLEU** (Bilingual Evaluation Understudy) was introduced by Papineni et al. (2002). It is a precision-based metric that builds a geometric mean of precision scores of $n$-grams up to a certain length ($N = 4$ in our case) with a penalty for proposed translations $y'$ that are longer than the reference translation $y$:

$$\text{BLEU}_4(y', y) = \min(1, \exp(1 - \frac{|y'|}{|y|}))(\prod_{n=1}^{4} \text{precision}_n)^{\frac{1}{N}}, \tag{3}$$

  where precision measures the ratio of matching $n$-grams in the proposed translation. BLEU returns a score between 0 and 1, where 1 expresses the perfect matching of the reference translation. It was introduced as a corpus-level metric aggregating the statistics over a whole corpus of (in the best case multiple) reference translations, but we will use its sentence-level variant for simulated quality judgments as well (sentence BLEU, sBLEU). On the sentence level, smoothing becomes important to prevent individual $n$-gram precisions from being zero, e.g., by replacing $n$-gram match counts of zero with a small number $\epsilon$ in the calculation of precision. The choice of smoothing technique has in influence of how well the sentence-level scores correlate with human judgments (Chen and Cherry, 2014).

- **TER** (Translation Error Rate) was proposed by Snover et al. (2006). It computes the ratio between the minimum number of edits (insertions, deletions, substitutions, shifts of word sequences) that the machine translation $y'$ would need to be transformed into the reference translation $y$:

$$\text{TER}(y', y) = \frac{\#\text{edits}(y', y)}{|y|} \tag{4}$$

  Due to the division by $|y'|$, the TER does not have an upper limit, but lower scores express higher quality due to fewer edits needed. The number of edits is approximated with dynamic programming for the minimal number of insertions, deletions and substitutions, and greedy search over shift locations.

- **METEOR** (Metric for Evaluation of Translation with Explicit Ordering) was developed by Lavie and Denkowski (2009). It is the only of these metrics that goes beyond surface-level matching and includes stemming and

[3]See Section 8.2.7 in (Koehn, 2009) for a deeper discussion.

synonyms. While it has better linguistic grounding than the surface-level metrics, it requires language-specific resources and higher computational effort.

- **BEER** (Better Evaluation as Ranking) was proposed by Stanojević and Sima'an (2014). It is a trained linear model that combines a set of dense features into one score. In contrast to the two previous metrics, it also includes character-level information about the character *n*-gram matches between proposed and reference translation. It correlates well with human judgments on sentence and corpus level (Bojar et al., 2016a).

**Domain Adaptation.** Large collections of parallel data are available for texts like parliament discussions (Tiedemann, 2012), news commentaries (Tiedemann, 2009), or patents (Wäschle and Riezler, 2012). The quality of machine translation systems is largely dependent on the training data size, so for these domains well-trained systems can be built. However, these domains might not capture the style, genre or topic, that end-users of translation are interested in. For that purpose, domain adaptation techniques exist that adapt an already trained MT system to new domains. Over the course of the thesis we will present several domain adaptation strategies with various level of supervision, to obtain translations for example for e-commerce product titles (Section 3.5).

### 2.1.2 Statistical Machine Translation

Whilst later chapters contain experiments solely with neural models, Chapter 3 reports experiments with a log-linear model based on a weighted synchronous context-free grammar (SCFG). Therefore we focus on this type of statistical machine translation (SMT) model in this introduction. As an instance of tree-based SMT it has the advantage over phrase-based SMT (Koehn et al., 2003) of being able to account for gaps in phrases.

**Tree-based SMT.** In tree-based SMT, source and target are represented according to their hierarchical structure as a pair of trees. An SCFG grammar consists of weighted production rules, where cross-lingual rules relate non-terminals of source and target language, i.e., syntactic constituents like noun (NP) or verb phrases (VP) or just abstract identifiers, and terminal symbols, i.e., the words itself. Consider for example the following excerpt from an SCFG for rules for a phrase in English ("the delicious bagel") and French ("le bagel délicieux"):

$$\begin{aligned}
\text{NP} &\xrightarrow{0.2} \text{DET}_1\ \text{NN}_2\ \text{JJ}_3 \mid\ \text{DET}_1\ \text{JJ}_3\ \text{NN}_2 \\
\text{NN} &\xrightarrow{0.9} \text{bagel} \mid \text{bagel} \\
\text{DET} &\xrightarrow{0.6} \text{le} \mid\ \text{the} \\
\text{JJ} &\xrightarrow{0.3} \text{délicieux} \mid \text{delicious}
\end{aligned}$$

Non-terminals are indexed so that they can be matched correctly in case of multiple occurrences. Each rule has a probabilistic score, and the product of all rules applied to a sentence form the score of the whole translation. This score can be combined with other scores, e.g., from language models. In practice, constraints for the construction of rules are applied to reduce parsing complexity, e.g., at most two non-terminal symbol per rule (Chiang, 2005). Before training the weights of the log-linear model (see below), hierarchical rules are extracted from a parallel corpus with word alignments, i.e., mappings between words of both languages, and optionally syntactic parses.

**Log-linear model.** In these type of models, a log-linear model produces scores for pairs of source and target sentences (Och and Ney, 2002). The feature mapping function $\phi(x, y) : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}^d$ of this log-linear model produces a vector from a range of feature extractors, e.g. the tree scores, language model scores for the whole sequences $p_{LM}(y)$ or $p_{LM}(x)$, or alignment scores. If the feature functions are parametric models themselves, they are trained before training the model parameters $w \in \mathbb{R}^d$. The feature extractors that we use in the experiments are described in Section 3.4.1.

$$p(y \mid x) = \frac{\exp(w^T \phi(x, y))}{\sum_{y' \in Y} \exp(w^T \phi(x, y'))} \tag{5}$$

In the study below we will speak of *dense* and *sparse* SMT models. Dense models have a low-dimensional $w$ expressing dense features, such as continuous language model scores. Sparse models, on the other side, have high-dimensional feature vectors with sparse entries, such as binary indicators for the occurrence of a word in the context.

**Training.** When we train the weights $w$ of the model to maximize the log-likelihood (2) of the references, we are not directly optimizing the weights for obtaining a high evaluation score (e.g. BLEU). Minimum-Error-Rate-Trainig (MERT) (Och, 2003) alleviates that by iteratively optimizing the individual entries of $w$ (while holding the others constant) to achieve an overall minimum error rate, or equivalently a maximum BLEU score, for a collection of $k$-best outputs (see details in (Lopez, 2008)). This approach is not feasible for models with a large number of features. For sparse models we therefore use the Margin Infused Relaxed algorithm (MIRA) instead (Crammer and Singer, 2003; Watanabe et al., 2007). This algorithm minimizes a structured hinge loss with online updates that increases the margin between better and worse translations.

**Re-ranking vs. re-decoding.** In domain adaptation experiments, we adapt one translation system from one domain to a new domain, e.g., a translation system trained on news to translate tweets. For SMT, there are two common options. One is to learn to *re-rank* the top $k$ translations by the original system by learning a new ranking function, such that a translation that was considered not that good on the old domain, will then get ranked higher for the new domain (or vice versa). Alternatively, one can adapt the model weights directly by switching the training data from the old to the new domain. This results in *re-decoding* with the new model, which means the new model has to generate completely new translations. In the experiments in Chapter 3 we will present results for both adaptation strategies.

### 2.1.3 Neural Machine Translation

The main difference between SMT and NMT is the parametrization of the translation model and the representations of inputs and outputs. In NMT, a artificial neural network (NN) with thousands to millions of weights organized in layers replaces the one-layer linear model in SMT. The scoring function becomes non-linear due to the use of non-linear activation functions that loosely mimic activations of neurons in the brain. The successful and wide-spread adoption of neural models, or *deep learning* was largely influenced first by the work of Bengio et al. (2003), who introduced a neural language model, then Collobert et al. (2011), who learned several NLP tasks with a neural network with a minimal amount of hand-defined features, and lastly by the invention of embeddings that represent words as continuous vectors (Mikolov et al., 2013b,a). The first successes with neural models for MT were published only a few years later (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2013, 2014; Cho et al., 2014; Bahdanau et al., 2015).[4] This section gives a brief introduction into the workings of an NMT system. For a general introduction of neural networks and deep learning for NLP we refer the reader to the primer by Goldberg (2016).

[4]Credits for earlier development of the idea go to (Neco and Forcada, 1997; Castano and Casacuberta, 1997).

**Sequence-to-sequence learning.** Neural models for machine translation are considered as an example for sequence-to-sequence learning (Seq2Seq). These models relate a sequence of input tokens $x = x_1 \ldots x_S$, to a sequence of output tokens $y = y_1 \ldots y_T$, with both sequences being of arbitrary length. A Seq2Seq translation model is trained to maximize the log-likelihood of the correct translation $y$ given the source $x$ (Equation 2). The conditional probability of a translation $p_\theta(y \mid x)$ is factorized into probabilities over single tokens conditioned on the history of preceding target tokens $y_{<t}$:

$$p_\theta(y \mid x) = \prod_{t=1}^{T} p_\theta(y_t \mid x, y_{<t}). \tag{6}$$

In contrast to tree-based SMT, we thus treat the input as a sequential structure without hierarchical sub-structures. The decoder acts like a target language model conditioned on the source. Note that there is no Markov assumption (unlike in traditional $n$-gram language models), i.e., each target token is conditioned on the complete history of previous target tokens.

**Local normalization.** The scores for each individual output token of the vocabulary $\mathcal{V}_{trg}$ form a discrete probability distribution, i.e., $\sum_{j=1}^{|\mathcal{V}|} p_\theta(y_t = j \mid x, y_{<t}) = 1$, by locally normalizing the outputs of the neural network for each decoding step with a softmax transformation (cf. Eq. 5)

$$p_\theta(y_t = i \mid x, y_{<t}) = \frac{\exp(o_i)}{\sum_{j=1}^{|\mathcal{V}_{trg}|} \exp(o_j)}, \tag{7}$$

where $\mathbf{o} \in \mathbb{R}^{|\mathcal{V}_{trg}|}$ is the output vector of a neural network that processes $x$ and $y_{<t}$ (e.g., a recurrent, convolutional or attentional network). Each entry $o_i$ represents the model score for a specific token of the target vocabulary $\mathcal{V}_{trg}$.

**End-to-end training.** In contrast to SMT, learning is done end-to-end, without much pre-processing or linguistic feature extraction pipelines. With the help of *back-propagation* (Rumelhart et al., 1986), the repeated application of the chain rule for weights along the computation path, the error signal obtained at the output of the network can be used to compute gradients in lower layers of the network. As long the loss function is differentiable with respect to the model weights, all weights can be adjusted with gradient updates. Suitable representations for input and output tokens, and intermediate levels of processing, such as the representation of a partial hypothesis, are learned on the fly. Due to the depth and non-convexity of the function and the size of the parameter space, training tricks like adaptive learning rates, momentum, gradient clipping, and carefully-tuned initialization techniques are required to find good local maxima and to speed up convergence (Sutskever et al., 2013).

**Embeddings.** Input and output tokens are represented as sparse one-hot vectors with indices over the source or target vocabulary. They further get *embedded*, i.e., represented as dense vectors $\mathbf{x}_s \in \mathbb{R}^{d_{src}}$, $\mathbf{y}_t \in \mathbb{R}^{d_{trg}}$ by multiplying them with either the source or the target embedding matrix $E_{src} \in \mathbb{R}^{|\mathcal{V}_{src}| \times d}$, $E_{trg} \in \mathbb{R}^{|\mathcal{V}_{trg}| \times d}$. The embedding matrices contain one vector representation for every token in the vocabulary. Instead of using a separate matrix for the source vocabulary and the target vocabulary, one can also use a joint one, for sharing representations across source and target language. These embedding matrices are part of the set of model weights $\theta$ that are trained and initialized randomly.

**Encoder-decoder architecture.** Seq2Seq is modeled with a neural architecture consisting of an encoder and a decoder (Cho et al., 2014; Sutskever et al., 2014; Bahdanau et al., 2015). The encoder reads the source sentence, and the decoder generates the target sentence word by word, conditioned on the encoded source. *Attention* modules allow learned connections between encoder and decoder or within encoder or decoder (Bahdanau et al., 2015; Vaswani et al., 2017). Recurrent (Cho et al., 2014; Sutskever et al., 2014; Bahdanau et al., 2015), convolutional (Gehring et al., 2017) or fully attentional neural networks (Vaswani et al., 2017) (or a mix thereof (Chen et al., 2018)) are common choices for encoder and decoder. In this thesis, we focus on recurrent architectures, but the proposed training algorithms for integrating feedback are in principle not limited to recurrent models.

**Recurrent encoder.** The input to the encoder is the sequence of embedded tokens $x = \mathbf{x}_1 \dots \mathbf{x}_S$ representing the source of length $S$. With a recurrent neural network (RNN) (Elman, 1990) the sequence of inputs is encoded into a sequence of hidden states:

$$\mathbf{h}_s = \text{RNN}([E_{src}\mathbf{x}_s, \mathbf{h}_{s-1}]) \tag{8}$$

$$\mathbf{h}_0 = \mathbf{0}. \tag{9}$$

*Gated RNNs* like Gated Recurrent Units (GRU) (Chung et al., 2014; Cho et al., 2014) or Long-Short-Term-Memory (LSTM) (Hochreiter and Schmidhuber, 1997) were shown to outperform the vanilla Elman-RNNs (Elman, 1990), since they prevent gradients from vanishing over long recursive gradient computations. The RNN reads the inputs sequentially and updates its internal state in every step. Gates control which parts of the inputs are written into memory, which parts of the memory are discarded and how the memory is updated. We refer the reader to (Lipton et al., 2015) for an in-depth introduction and discussion of RNN variants.

**Stacking.** Several RNN layers can be stacked by feeding the hidden state sequence $\mathbf{h}_1, \dots, \mathbf{h}_S$ as input to another RNN, which in turn outputs a new sequence of vectors. The sequential processing of the inputs from left to right may limit the expressiveness of the resulting hidden states, which *bi-directional* RNNs alleviate. They consist of two RNNs, where one receives the input sequence in the original order (the forward RNN) and the other in reversed order (the backwards RNN). The output of the forward and the backward RNN are then combined, e.g., by concatenation, and serve as inputs to higher layers.

**Recurrent decoder.** We define the attentional recurrent decoder following (Luong et al., 2015a) and borrow the notation from (Kreutzer et al., 2020). The decoder transforms the output of the encoder into a sequence of output vectors $\mathbf{o}_1, \dots \mathbf{o}_T$ of the size of the target vocabulary $\mathbf{o}_t \in \mathbb{R}^{|\mathcal{V}_{trg}|}$. The RNN decoder state is computed from the concatenation of the embedded previous target token and an attentional state $\tilde{\mathbf{s}}$, which in turn is the output of a linear layer combining decoder state and a context vector $\mathbf{c}_t$.

$$\mathbf{s}_t = \mathrm{RNN}([E_{trg}\mathbf{y}_{t-1}; \tilde{\mathbf{s}}_{t-1}], \mathbf{s}_{t-1}) \tag{10}$$

$$\tilde{\mathbf{s}}_t = \tanh(W_{att}[\mathbf{s}_t; \mathbf{c}_t] + \mathbf{b}_{att}) \tag{11}$$

There are multiple options to initialize the first decoder state $\mathbf{s}_0$, e.g., with zeros, or as a transformation (identical, linear or non-linear) of the last encoder state. The output vectors are computed from the attentional decoder states:

$$\mathbf{o}_t = W_{out}\tilde{\mathbf{s}}_t + \mathbf{b}_{out} \tag{12}$$

**Attention.** The context vector $\mathbf{c}_t$ serves the purpose of allowing individual connections between each decoder and encoder state. It is of a weighted average of all encoder states, where the weighting is learned.

$$\mathbf{c}_t = \sum_s a_{st} \cdot \mathbf{h}_s \tag{13}$$

$$a_{st} = \frac{\exp(\mathrm{score}(\mathbf{s}_{t-1}, \mathbf{h}_s))}{\sum_{k=0}^{S} \exp(\mathrm{score}(\mathbf{s}_{t-1}, \mathbf{h}_k))} \tag{14}$$

There are various options for the scoring function, such as a multi-layer-perceptron (MLP) (Bahdanau et al., 2015) or a bilinear transformation (Luong et al., 2015a). When generating an output at step $t$, the encoder states are weighted differently than for the steps before or after. Intuitively, this models something similar to the word alignments for SMT, but rather in a soft way. Figure 3 (page 25) visualizes the attention weights $a_{st}$ for each pair of source token (German) and target token (English). One can see that the attention when generating "a" is

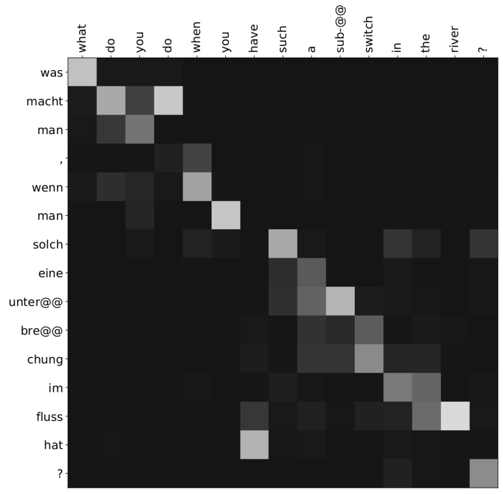


Figure 3: Learned attention weights for an example from the IWSLT (de-en) corpus. Bright: high weight, dark: low weight.[5]

widely spread across the second half of the source sentence, which does not fit the interpretation of classic word alignments.

**Maximum likelihood estimation.** Since this encoder-decoder architecture is fully differentiable with respect to its weights (with *back-propagation through time* (Mozer, 1995)), it can be trained with gradient descent methods. Given a parallel training set of $k$ source sentences and their reference translations $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{k}$, we define the token-level Maximum Likelihood Estimation (MLE) objective, which aims to find the parameters that maximize

$$J^{\text{MLE}}(\theta) = \sum_{i=1}^{k} \log p_\theta(y^{(i)} \mid x^{(i)}) \tag{15}$$

$$= \sum_{i=1}^{k} \sum_{t=1}^{T} \log p_\theta(y_t \mid x^{(i)}, y_{<t}^{(i)}). \tag{16}$$

In the context of this thesis, we refer to the MLE objective as *fully-supervised* training. It is non-convex for the case of neural networks. A trick of the trade is to ensemble several models with different random initialization to improve over single models (Luong et al., 2015a). While target token predictions are conditioned on reference tokens during training (*teacher forcing*), they are conditioned on model outputs during inference. Beam search expands, scores

[5]Example taken from the Joey NMT (Kreutzer et al., 2020) tutorial (`https://joeynmt.readthedocs.io/en/latest/tutorial.html`).

and prunes translation hypotheses token by token and usually produces outputs of higher quality than plain greedy decoding for every token.

## 2.2 Learning from Interaction

### 2.2.1 Reinforcement vs. Bandit Learning

Bandit and reinforcement learning both stand in contrast to fully-supervised training with reference or gold outputs. Both learning paradigms cover learning in interaction, where the model interacts with the world and learns from rewards it receives from it. For our use case these two paradigms are of great interest, since they will allow us to leverage weak feedback for MT training. This section highlights the commonalities and differences of both paradigms before we discuss how to apply them to machine translation.

#### Reinforcement Learning

In reinforcement learning (RL), the learner receives supervision signals by *interacting with an environment.* As opposed to supervised learning, there is no gold truth for the interactions available, that means that the learner does not know which action would have been the best choice in which situation. Instead, the learner has to rely on a scalar reward from the environment, that can be the increase of a score in a game, the accomplishment of a task goal, or an affirmative signal from a human. Only after the agent made a decision and executed an action will it receive feedback on the quality its action, sometimes even only after a sequence of interactions. The overall goal of the learner, or *agent*, is to maximize the cumulative reward.[6]

**Modeling decision making.** When the agent can fully observe the states, its sequential decisions making problem can be formalized as a Markov Decision Process (MDP). An MDP $< \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma >$ consists of

- A finite set of states $s \in \mathcal{S}$. The states are Markovian, i.e., future states are only conditionally dependent on the current state and not the whole trajectory.
- A finite set of actions $a \in \mathcal{A}$, which can be discrete or continuous.
- A state transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ that describes which state the agent ends up in when taking action $a$ from state $s$, either deterministic or stochastic.

[6]We refer the reader to the book by Sutton and Barto (1998) for a thorough introduction.

- A reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ from the environment that expresses whether a state-action pair should be encouraged (high reward) or discouraged (low reward).
- A discount factor $\gamma \in [0, 1]$ that is applied to future rewards.

**Trajectories.** A sequence of states, actions and rewards $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots, s_T, a_T, r_T)$ is called a *trajectory*. In applications, rewards are often not available after an atomic action, but only after a full trajectory of length $T$, e.g., after finishing a game or completion of a translation or utterance. In contrast to supervised learning, data is non i.i.d. because past actions influence future states, and the reward function is not transparent to the agent.

**Applications.** Reinforcement learning has been applied for a wide range of interactive tasks. These include agents that are developed for interactions with the real world such as in robotic manipulation (Gu et al., 2017), or autonomous driving (Michels et al., 2005), and agents that operate in virtual environments such as in games like Go (Silver et al., 2017) or computer games like Atari (Mnih et al., 2013). For Seq2Seq, the decoder can be seen as a reinforcement learning agent with continuous states and chooses discrete tokens from the vocabulary as actions. The transition function is deterministically defined by the decoder RNN weights. The reward-producing "environment" could be a user that reacts to partial translations (such as in interactive-predictive MT), or a database that returns the desired information after completing a query. We will explain in Section 2.3 how Seq2Seq learners can be trained as agents in an RL environment.

### Bandit Learning

Multi-armed bandit learning can be understood as a reduced instance of reinforcement learning with a one-step MDP. It does not have any notion of states or transitions.[7] Actions can be seen as arms of a range of one-armed bandit slot machines with unknown reward distributions. Just like in reinforcement learning, the agent has to balance *exploitation*, i.e., choosing the empirically most rewarding arm, and *exploration*, i.e., trying out arms that were not as rewarding or have not been chosen as frequently before. *Contextual bandits* (Agarwal et al., 2014), a subset of bandit learning algorithms, allow decisions to be made given a specific context, for example document or image features, and form the most suitable class of bandit algorithms for NLP. We refer the reader to (Szepesvári, 2009) for an overview of algorithms for reinforcement learning and their relation to bandit learning.

[7]We follow the definitions in the survey by Bubeck and Cesa-Bianchi (2012).

**Minimizing regret.** The agent's task is to choose the arm with the lowest *expected regret* over a horizon of $T$ learning steps,[8] i.e., the smallest expected difference in reward to the best arm[9]

$$\mathbb{E}[R_T] = \mathbb{E}\left[\max_{a' \in \mathcal{A}} \sum_{t=1}^{T} \mathcal{R}(a') - \sum_{t=1}^{T} \mathcal{R}(a_t)\right]. \tag{17}$$

The expectation is taken with respect to rewards and actions, omitted for brevity.

The challenge lies in learning with *partial feedback*, that means that the reward function is not fully revealed to the learner and that there is no access to rewards for alternative arms. We speak of *bandit feedback* when the reward function is only evaluated for exactly one arm (Swaminathan and Joachims, 2015a).

**Applications.** The greatest success of contextual bandit learning in practice has been to replace A/B-testing in production environments, for example in online advertisement (Chapelle et al., 2014), news recommendation (Li et al., 2010), or clinical trials (Durand et al., 2018). In NLP, they have been reported successful in extractive summarization (Dong et al., 2018) and structured prediction tasks like chunking and machine translation (Sokolov et al., 2016b,a; Sharaf and Daumé III, 2017), which we will discuss extensively in Chapter 3. However, for the task of sequence-to-sequence learning with combinatorial action spaces, i.e., an exponential number of arms, deep parametrized policies, and a need for strong generalization, popular bandit learning algorithms (UCB (Auer, 2002), EXP4 (Auer et al., 2002), $\epsilon$-greedy (Sutton and Barto, 1998), and Thompson sampling (Thompson, 1933)) fall short in terms of practical guarantees and scalability (Bubeck and Cesa-Bianchi, 2012). In the following, the main focus will thus lie on the reinforcement learning perspective on the problem of sequence-to-sequence learning. But since the SMT models produce one translation (one trajectory) at once without intermediate prediction steps or states, it resembles the bandit learning problem.

### 2.2.2 Policy Optimization

How does the agent decide which action to choose next? It follows a *policy*, which is a function $\pi : \mathcal{S} \mapsto \mathcal{A}$, that defines a distribution over actions depending on the current state $s$:

$$\pi(a \mid s) = P(A_t = a \mid S_t = s). \tag{18}$$

[8]The task might also be formulated in terms of cumulative regret or pseudo-regret minimization, see (Bubeck and Cesa-Bianchi, 2012).

[9]Since there is no notion of states, the state argument of the reward function is dropped.

The policy can choose actions either stochastically or deterministically. It needs to adjust the distribution over actions according to the feedback it receives from the environment. In *policy learning* the goal is to optimize the parameters $\theta$ that describe the policy for maximal expected reward:

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{a \sim \pi_\theta} \left[ \mathcal{R}(a, s) \right] \tag{19}$$

Alternatively, one could learn value functions for state-action pairs or actions (*policy evaluation*), and then infer the best policy from the value function. However, policy optimization scales better with growing action and state spaces, since it generalizes better for unseen state-action pairs.

One can either learn those parameters *online*, i.e., during the interaction with the environment, or *offline*, e.g., after the interaction is completed, or in a combination of both schemes. Comparing online or offline (aka *counterfactual*) policy learning there occurs a trade-off between bias and variance. It is harder to find methods for online learning with low variance (due to exploration) and methods for offline learning with little bias (due to the mismatch between logging and learning policy). The following section will present the most popular algorithm for online policy optimization. .

### Policy Gradient

The expected reward objective in Eq. (19) can be optimized with gradient ascent methods on the parameters of the policy.[10]

$$\begin{aligned} \nabla_\theta \mathbb{E}_{a \sim \pi_\theta} \left[ \mathcal{R}(a, s) \right] &= \nabla_\theta \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) \mathcal{R}(a, s) \\ &= \sum_{a \in \mathcal{A}} \nabla_\theta \pi_\theta(a \mid s) \mathcal{R}(a, s) \end{aligned} \tag{20}$$

In order to compute the gradient, the reward function does not have to be known, nor differentiable with respect to the policy parameters—as long as it can be evaluated for encountered state-action pairs. However, the computation of the full gradient (Equation 20) is intractable for large action spaces, and prohibited when learning from bandit feedback. The full gradient can be approximated stochastically in a *Monte Carlo simulation* by sampling actions from the

[10]If the goal is to minimize a task loss instead of a reward, gradient descent is used.

parametrized policy $a \sim \pi_\theta(a \mid s)$:

$$\begin{aligned}
\sum_{a \in \mathcal{A}} \nabla_\theta \pi_\theta(a \mid s) \mathcal{R}(a,s) &= \sum_{a \in \mathcal{A}} \nabla_\theta \pi_\theta(a \mid s) \frac{\pi_\theta(a \mid s)}{\pi_\theta(a \mid s)} \mathcal{R}(a,s) \\
&= \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) \frac{\nabla_\theta \pi_\theta(a \mid s)}{\pi_\theta(a \mid s)} \mathcal{R}(a,s) \\
&= \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s) \nabla_\theta \log \pi_\theta(a \mid s) \mathcal{R}(a,s) \\
&= \mathbb{E}_{a \sim \pi_\theta} \left[ \nabla_\theta \log \pi_\theta(a \mid s) \mathcal{R}(a,s) \right]. \qquad (21)
\end{aligned}$$

The policy is updated with stochastic gradients, which results in the REINFORCE algorithm proposed by Williams (1992):

$$\theta_{k+1} = \theta_k + \gamma \nabla_{\theta_k} \log \pi_\theta(a_k \mid s_k) \mathcal{R}(a_k, s_k). \qquad (22)$$

In every step, action $a_k$ is sampled from the current policy $\pi_{\theta_k}$, which changes after each update. Intuitively, the learner makes gradient steps towards the actions with the magnitude of the reward (and the learning rate $\gamma$ as a constant factor), reinforcing parameter values that lead to high rewards. Discovering actions that yield high rewards in a large action spaces only can take a long time and the variance of stochastic gradients is high, so the sample efficiency of policy gradient (PG) is in practice often insatisfactory. Methods for variance reduction of the stochastic gradient will be presented in the following section.

Many modifications of the original REINFORCE algorithm have been proposed, for example to include learned approximations for the reward function (Sutton et al., 2000b), to learn with deterministic outputs (Silver et al., 2014; Lillicrap et al., 2015), with natural gradients (Kakade, 2002), or with asynchronous updates (Mnih et al., 2016).

### 2.2.3 Variance Reduction by Control Variates

One issue with vanilla policy gradient is that its stochastic gradient updates suffer high variance, since they involve sampling from the model distribution, which leads to slow convergence. Various techniques and extensions of its plain version have since been developed to reduce the variance of the stochastic updates. Here, we focus on the use of control variates known from Monte Carlo estimation, as they are very effective for our applications (see Section 3.4.2).

#### Additive Control Variates

The idea of *additive* (or linear) control variates is to augment the random variable $X$ whose expectation is sought, here the noisy gradient from Equation (21), by another random variable $Y$, the control variate, which is highly correlated with $X$ (Wilson, 1984; Nelson, 1987; Ross, 2013). Subtracting the random variable from $X$ and adding its expectation leaves the estimator unbiased:

$$\begin{aligned}\mathbb{E}[X] &= \mathbb{E}[X - \hat{c}(Y - \mathbb{E}[Y])] \\ &= \mathbb{E}[X - \hat{c}\,Y] + \hat{c}\,\mathbb{E}[Y]. \end{aligned} \tag{23}$$

$\hat{c}$ is a constant that may be tuned, but is usually set to 1. The variance of this augmented estimate is dependent on the variance of the individual variables and their covariance:

$$\mathrm{Var}(X - \hat{c}\,Y) = \mathrm{Var}(X) + \hat{c}^2\mathrm{Var}(Y) - 2\hat{c}\,\mathrm{Cov}(X, Y). \tag{24}$$

The variance of the original estimate is reduced by the control variate if the covariance between $X$ and $Y$ is larger than the variance of $Y$:

$$\mathrm{Var}(X - \hat{c}\,Y) < \mathrm{Var}(X) \text{ iff } 2\,\mathrm{Cov}(X, Y) > \hat{c}\mathrm{Var}(Y). \tag{25}$$

The optimal scalar $c^*$ can be derived easily by taking the derivative of (24) with respect to $\hat{c}$, leading to $c^* = \frac{\mathrm{Cov}(X,Y)}{\mathrm{Var}(X)}$. A convenient choice for $Y$ in the context of policy gradient algorithms is the *baseline* control variate (Williams, 1992). Its practical implementation is simple: A running average of rewards is subtracted from the current reward. Details will be discussed in Section 3.3.

### Multiplicative Control Variates

*Multiplicative* (or ratio) control variates are combined with the original estimate by multiplication:

$$\mathbb{E}[X] \approx \mathbb{E}[\frac{X}{Y}]\ \mathbb{E}[Y]; \tag{26}$$

$$\mathbb{E}[Y] \neq 0. \tag{27}$$

In contrast to the additive control variate, this estimator is not unbiased, as $\mathbb{E}[\frac{Y}{X}] \neq \frac{\mathbb{E}[X]}{\mathbb{E}[Y]}$ (Swaminathan and Joachims, 2015b).

The variance of the combined estimator $m = \frac{X}{Y}$ (Kong, 1992) is

$$\mathrm{Var}(m) \approx \frac{1}{n}(m^2\mathrm{Var}(Y) + \mathrm{Var}(X) - 2m\mathrm{Cov}(Y, X)). \tag{28}$$

The variance of the original estimator will be reduced when the covariance is large.

We will see this control variate applied for self-normalization in counterfactual learning in Section 4.2.3 as proposed by Swaminathan and Joachims (2015b); Lawrence et al. (2017a).

There is a close connection to *importance sampling* in Monte Carlo estimation. When one can not sample from a distribution $p$, but from a proposal distribution $q$ instead, the estimator is reweighted by the likelihood ratio $\frac{p(X)}{q(X)}$ to correct the sampling bias. If the proposal distribution is chosen carefully, the variance of the original estimator is reduced. We will see an application of importance sampling in the Cross-Entropy Minimization objective in Chapter 3.

### 2.2.4 Human-in-the-Loop Reinforcement Learning

Humans can be integrated in the reinforcement learning process by either giving the rewards, or act as experts that demonstrate ideal behavior (from which rewards can be inferred), or correct the agent's behavior. However, most of the success in human-in-the-loop RL has been reported for problems with small action spaces and game-like environments, which we will outline in the following.

**Reward functions learned from humans.** Knox and Stone (2009) and Christiano et al. (2017) learn a reward function from human feedback and use that function to train an RL system. Knox and Stone (2009) apply their approach (TAMER) to Tetris and Mountain Car, where humans observe the game score and submit positive or negative feedback (see also (Celemin et al., 2019)), which resulted in faster convergence than with environmental rewards. Christiano et al. (2017) use a deep model to learn the reward function from human preference judgments (without rewards from the environment), aiming for faster convergence with less human effort. In their evaluation on simulated robotics task, human feedback is only on one task more sample-efficient than synthetic feedback or the true reward function, as well as for Atari, where the performance varies quite a lot across tasks. We will learn reward functions from human feedback in Chapters 4 and 5.

**Actor-critic learning from humans.** In actor-critic training, a critic model is trained to predict the environment's (true) reward. It is used to reinforce the agent (=actor) replacing the true reward in the policy gradient update. Pilarski et al. (2011) and MacGlashan et al. (2017) both adopt the actor-critic framework for interactive RL. For example, in the COACH framework proposed by (MacGlashan et al., 2017) a human teacher gives positive and negative feedback for a virtual dog training task with five actions. The feedback is used to adapt a shallow learner online, with an exponential decay over a window of time steps to account for the delay between actions and human feedback. However, in those methods learning of the agent happens only exactly when feedback is given, resulting in a sparse signal that might not be sufficient for tasks where many rounds of learning are needed. In our experiments, we first collect the feedback, and then move to a learning phase where the model is updated.

**Deep RL.** While the above approaches—with the exception of (Christiano et al., 2017)—employ shallow models as policies, Arumugam et al. (2019) equip COACH with a deep policy and find that it stabilizes and speeds up learning. In the Deep TAMER approach by Warnell et al. (2018), non-expert human trainers give a series of real-time critiques in the form of scalar feedback to an agent playing Atari Bowling, a game with a high-dimensional state space, and a small action space of four actions. The feedback is used to update the critic and also stored in a feedback replay buffer to allow more frequent updates than human reactions. They also achieve higher gains and faster learning than the shallow counterparts. In our experiments, all reward estimators will be deep and we will benefit from pre-trained embeddings for faster learning.

**Reliability.** None of these works systematically investigates the reliability of the feedback, so it is not clear what the human reward function expresses and whether any other reward function (or even self-supervised learning) would have achieved the same result. We will experience in Chapter 4 how unreliable feedback can hurt the reinforcement learner, and in Chapters 5 how a learned reward function on curated feedback can help to generalize.

**Cost and Availability.** The above discussed approaches found methods to cope with limited availability and slow reaction times of human trainers, for example by filling reward buffers, or distributing rewards over windows of actions. What has been missing in the analyses is the cost or effort it takes humans to provide this feedback. Unlike in simulated environments, where the feedback comes "for free", there is a cost to pay when employing human teachers. This concerns not only financial costs, but also costs one might pay if giving feedback is too strenuous and attention or accuracy is decreasing over time. In our experiments, we treat the choice of feedback interfaces as a hyperparameter for interactive learning that needs tuning for a good cost-benefit trade-off. While we choose it manually in Chapters 4 to 6, we leave it to a meta-learner in Chapter 7.

## 2.3 Reinforced Machine Translation

Similar to the applications for contextual bandits (see Section 2.2.1) or reinforcement learning (see Section 2.2.1), there are many interactive NLP applications, such as semantic parsing (Iyer et al., 2017; Lawrence and Riezler, 2018; Yao et al., 2019, inter alia), summarization (Leuski et al., 2003; Liu et al., 2009; Shapira et al., 2017; PV et al., 2018, inter alia) or dialogue (Wen et al., 2015; Li et al., 2016, 2017, inter alia), which may be modeled as sequence-to-sequence learning.

In interactive machine translation, user feedback in the form of post-edits of predicted translations is used for model adaptation (Bertoldi et al., 2014;

Denkowski et al., 2014; Green et al., 2014; Simianer et al., 2016; Karimova et al., 2018). Since post-editing feedback has a high cost and requires professional expertise (maybe even more so for the applications of summarization and semantic parsing), weaker forms of feedback are desirable, which makes reinforcement or bandit learning attractive.[11]

[11]When speaking of "weak" supervision, we do not refer to distant supervision or weak labeling approaches, but rather the "bandit"-type of feedback expressing the quality for a output.

This section establishes a bridge between reinforcement learning and sequence-to-sequence learning, focusing on the application of NMT. The task is to improve a machine translation model with bandit feedback. Instead of training on reference translations, it is adapted to user or annotator feedback. The feedback required by our methods can be provided by laymen users or can even be implicit, e.g., inferred from user interactions with the translated content on a web page, which makes it attractive for other sequence-to-sequence learning tasks as well.

The core idea is to understand the structured prediction, or generation models (just like the log-linear or neural MT models presented in Sections 2.1.2 and 2.1.3) as parametrized policy $\pi_\theta$. The output structure is produced by the policy either (1) in a one-step prediction of a contextual bandit where the whole structure corresponds to one action (as in log-linear models) or (2) in token-by-token generation process as a sequence of actions in reinforcement learning (as in neural models). While Chapter 3 will present detailed solutions for both, the following sections will elaborate the key challenges to overcome and discuss how far previous approaches have come.

### 2.3.1 Challenges

Regardless of the underlying model, the following three challenges have to be addressed when applying reinforcement/bandit learning to machine translation:

1. **Large action spaces**. The action space could potentially span the entire vocabulary of the target language, i.e., hundreds of thousands of words, which is magnitudes more than what classic RL algorithms are evaluated on (compare the three to five actions in the reported studies with human feedback in Section 2.2.4). This requires efficient exploration strategies, since a large portion of the action space should not even be considered. With those large action spaces one can generate exponentially many trajectories/translations, but only very few of them form valid translations. Syntactic sentence structure, semantic context and relations are required to be nearly correct for receiving any reward at all. Exploration gets even more challenging in offline learning (Langford et al., 2008; Lawrence et al., 2017b), which is required for real-world implementations of bandit machine translation, as we will further discuss in Chapter 4.

2. **Credit assignment**. In addition, the correctness of a translation can only be judged when near or at completion. Rewards are sparse, action spaces large and sequences long and structured. Credit assignment, knowing which individual actions led to the sequence-final reward, is hard. However, when interacting with humans, one might ask them for credit assignment in addition to rewards as well (e.g., by marking the words that are most relevant for their quality judgment, which we will propose in Chapter 6).

3. **Noisy rewards**. Judging MT quality is a complex task even for humans. There is usually low agreement on translation quality scoring, and getting reliable judgments requires filtering of annotators and a careful setup of interfaces (see Chapter 5). Automatic metrics also fall short in modeling translation quality and it is still an active field of research to find suitable metrics (e.g. in the WMT metrics task[12]). The collected feedback, human or simulated, will contain noise and might not express with what one is looking for in the final translations. One trick to avoid noise from human or automatic quality estimates is to collect feedback implicitly or directly embedded in a down-stream task and use the task success as feedback, which will explore in Chapter 4.

### 2.3.2 Previous Approaches

Both SMT and NMT have both most commonly been trained under supervised learning objectives with references, such as MLE. Previous or concurrent approaches to our work can be classified into two categories: The first group uses RL methods to overcome weaknesses of MLE in presence of reference translations. The second group shares our interest of improving MT in absence of reference translations. There is no prior work on RL from human feedback for MT.

#### Improving Fully-supervised Learning

Neural models trained with MLE were observed to suffer from *exposure bias*, since they learn to generate output words based on the history of given reference words (teacher forcing, see Section 2.1.3), not on their own predictions. This is why Ranzato et al. (2016) resort to techniques from reinforcement and imitation learning to learn from feedback to the model's own predictions.

Furthermore, they address the mismatch between word-level loss and sequence-level evaluation metric (e.g. BLEU) by using a mixture of REINFORCE and the standard maximum likelihood training to directly optimize a sequence-level loss.

With the help of policy gradients, models can be optimized for non-differentiable metrics without having to resort to differential approximations to the original metric, and as a result, usually show higher test scores.

These two aspects have been the main benefit of minimum risk training both for SMT (Och, 2003; Smith and Eisner, 2006; Gimpel and Smith, 2010; Yuille and He, 2012; He and Deng, 2012) and NMT (Shen et al., 2016; Wu et al., 2016), or later also in actor-critic (Bahdanau et al., 2017) or generative-adversarial Seq2Seq (Li et al., 2017; Yu et al., 2017a; Wu et al., 2018b; Yang et al., 2018). These works all share the concept of reinforcing the model's outputs, but not to interact with humans, but in order to overcome weaknesses of MLE training in

[12]`http://www.statmt.org/wmt19/metrics-task.html`

presence of reference translations. In our bandit setting, feedback to only a single sample per sentence is available, and it might not directly be the evaluation metric, which makes the learning problem much harder.

All these works approach the challenge of large action spaces by warm-starting the agent with a pre-trained model that was trained in a fully-supervised fashion on available parallel data. This results in peaked model distributions that ensure a relatively high quality of initial samples (Choshen et al., 2019). It fits naturally into the MT domain adaptation setting that we are addressing in the experiments. It is rare that no pre-training data for a given language is available (only for extremely low-resource languages), but usually this data does not match the domain of interest, so domain adaptation with bandit feedback is an attractive solution.

### Improving MT Without References

Concurrently to our introduction of the banditized machine translation training objectives in (Sokolov et al., 2016a,b), He et al. (2016) propose a *dual-learning* mechanism for training machine translation from weak feedback without the assumption of references. The idea is to train two translation models jointly on monolingual data with a reward signal from language models and a reconstruction error. This is attractive because the feedback can automatically be generated from monolingual data and does not require any human references. In practice, however, it required references to form a “soft-landing” (He et al., 2016) between standard likelihood training and reinforced training, just like in the mixed approach by Ranzato et al. (2016). The idea of leveraging back-translation for feedback has similarities to the approach we will present in Chapter 4 for learning from implicit feedback embedded in a cross-lingual information retrieval task.

Nguyen et al. (2017) propose a similar neural approach using a learned word-based critic in an advantage actor-critic reinforcement learning framework concurrently to our neural approach (Kreutzer et al., 2017). They investigate the impact of human-like noise in simulation experiments. In the online bandit learning shared task on e-commerce product descriptions their approach is compared to ours (Sokolov et al., 2017) (Section 3.5).

Sokolov et al. (2015) were the first to present a banditized objective for phrase-based SMT models for re-ranking, which is what we compare against in Chapter 3. Their algorithm suffers from slow convergence speed, meaning that impractically many rounds of user feedback would be necessary for learning in real-world interactive MT. Our initial approaches for SMT and NMT suffer from the same drawback (Chapter 3), but we will overcome this with means of off-policy learning, control variates and reward estimators, leading to successes even with little amounts of human feedback (Chapters 5 to 7).

Lawrence et al. (2017a,b) addressed the more practically-realizable problem of offline learning from logged bandit feedback, with special attention to the problem of exploration-free deterministic logging as is done in commercial MT systems. Their objectives for counterfactual SMT training employ variance reduction techniques, that we will adapt to neural models. We compare our neural adaptation to their SMT simulation results in Chapter 4, deploy it outside simulations, and apply it to learning from human feedback.

# Part I

# Online Learning with Simulated Bandit Feedback

# Chapter 3

# Online Bandit Structured Prediction for MT

This chapter presents the "banditization" of expected loss minimization (or expected reward maximization) approaches to structured prediction (Smith, 2011; Yuille and He, 2012) with log-linear and neural models. The resulting algorithms apply to online learning scenarios where gold standards are not available, but feedback to predicted structures can be obtained from users or teachers in the loop. As opposed to standard supervised learning approaches with gold standards, the learner does not know what the correct prediction looks like, nor what would have happened if it had predicted differently. As discussed in the introduction, this learning scenario has been investigated in bandit and reinforcement learning (see Sections 2.2).

The central challenge of balancing exploration and exploitation is approached by sampling from the model's Gibbs distribution over outputs, which results in simultaneous exploration and exploitation depending on the shape of the distribution—the flatter the distribution, the more is explored. The trick of the "banditization" is to use the feedback obtained for a single sample to construct a parameter update that is an unbiased estimate of the respective update rule for the corresponding full information objective.

Most closely related are RL approaches that use gradient-based optimization of a parametric policy for action selection (Bertsekas and Tsitsiklis, 1996; Sutton et al., 2000a) (see Section 2.2.2). Our case is related to contextual bandits (Langford and Zhang, 2007; Li et al., 2010) or combinatorial bandits (Dani et al., 2007; Cesa-Bianchi and Lugosi, 2012) in the sense that the parametric models receive contextual information and consist of a single state, but for structured prediction we have to manage structures over exponential output spaces instead of a small set of options. And while bandit learning is mostly formalized as online regret minimization with respect to the best fixed arm in hindsight, we investigate asymptotic convergence of our algorithms, motivated by the practical consideration of human feedback cost.

In comparison to previous work (Section 2.3), we aim to improve convergence speed for linear models by convexifying the expected reward objective (*bandit cross-entropy minimization*), and by merging two gradient updates into one with a pairwise preference objective (*bandit pairwise preference learning*). In order to investigate the connection of optimization for task performance with optimization-theoretic concepts of convergence, the algorithms are analyzed as stochastic first-order (SFO) methods in the framework of Ghadimi and Lan (2012). The pairwise preference objective is to our knowledge the first stochastic first-order approach to stochastic learning form pairwise feedback, since previous work on stochastic pairwise learning has been formalized as derivative-free stochastic zeroth-order optimization (Yue and Joachims, 2009; Agarwal et al., 2010; Ghadimi and Lan, 2012; Duchi et al., 2015).

The main contribution of this chapter is an experimental evaluation of the empirical performance and convergence speed of the different algorithms for SMT and NMT. We present an evaluation on several MT tasks with different loss functions and models where the reward signal is simulated by evaluating a task loss against gold standard structures without revealing them to the learning algorithm (Agarwal et al., 2014).[1]

We evaluate the resulting algorithms on the task of French-to-English translation domain adaptation where a seed model trained on Europarl data is adapted to the NewsCommentary and the TED talks domain with simulated weak feedback. In addition, we compare these algorithms in our newly-introduced bandit learning shared task at the Conference of Machine Translation (WMT) with e-commerce translations from German to English.

We find consistent improvements over the baseline for all objectives and models, with generally higher improvements and faster convergence for NMT than SMT, which demonstrates the success of the training algorithms across model families and exemplifies the potential of deep reinforcement learning for NLP applications. These first successes of online learning with weak simulated feedback across model families constitute the basis for bandit learning for structured prediction and the first step towards interactive learning with human feedback.

**Contributions.** The contributions of this chapter are the following:

1. Introduction of three objectives for online bandit structured prediction: expected reward maximization, cross-entropy minimization, and pairwise preference learning.

2. Realization and implementation of the three algorithms for shallow/linear and deep/neural models for structured prediction.

[1]More results for other structured prediction tasks, namely optical character recognition, named entity recognition and noun phrase chunking, are reported in (Sokolov et al., 2016a) and (Sokolov et al., 2016b).

3. Analysis of the empirical performance and convergence speed on machine translation domain adaptation tasks with simulated sentence-level feedback.

4. Measuring the effect of control variates on neural models for faster learning and improved generalization.

5. Comparison of competing algorithms in a bandit learning shared task at WMT. This is a novel evaluation paradigm that largely deviates from standard WMT evaluations with references.

**Publications.** The methods and results listed for SMT models were published in (Sokolov et al., 2016b) and (Sokolov et al., 2016a), where the author contributed the experiments and analysis on sparse machine translation models. The solutions for neural models published (Kreutzer et al., 2017) were all created by the author. The results on the shared task evaluation were published in (Sokolov et al., 2017), in a co-operation with Amazon, where the author had the initial idea and contributed neural baselines and bandit models.

**Outline.** Section 3.1 starts with the derivation of three algorithms for online reward optimization for SMT and NMT, and Section 3.2 analyses their convergence. The effect of reward baselines, which turn out very effective for reward maximization for NMT, is elaborated in Section 3.3. The experiments on domain adaptation for SMT and NMT, and the result of the WMT shared task are discussed in Sections 3.4 and 3.5, respectively. Section 3.6 summarizes the findings of this chapter.

## 3.1 Algorithms

### 3.1.1 Full information vs. bandit feedback

The objectives and algorithms presented in this chapter are based on the well-known expected loss (or expected reward) criterion for probabilistic structured prediction (Och, 2003; Smith and Eisner, 2006; Gimpel and Smith, 2010; Yuille and He, 2012; He and Deng, 2012). The objective is defined as a minimization of a task loss function (or maximization of the expectation of a given task reward respectively) with respect to the conditional distribution over structured outputs.[2] This criterion has the form of a continuous, differentiable, and in general, non-convex objective function.

[2]In previous work this has been presented as a loss minimization objective, but in the scope of this thesis we will focus on the maximization perspective to emphasize the similarities to reinforcement learning from rewards, details in Section 3.1.2.

More formally, let $\mathcal{X}$ be a structured input space, let $\mathcal{Y}(x)$ be the set of possible output structures for input $x$, and let $r : \mathcal{Y} \to [0,1]$ quantify the reward $r_y(y')$ received for predicting $y'$, where the gold standard structure $y$ should receive the highest reward $r_y(y) = 1$. In a full information setting, for an empirical data distribution $p(x,y)$, the learning problem is defined as maximization of the expected reward with respect to $\theta \in \mathbb{R}^d$:

$$\mathbb{E}_{p(x,y)p_\theta(y'|x)}\left[r_y(y')\right] = \sum_{x,y} p(x,y) \sum_{y' \in \mathcal{Y}(x)} r_y(y') p_\theta(y' \mid x). \tag{29}$$

Assume further that output structures given inputs are distributed according to an underlying Gibbs distribution (a.k.a. conditional exponential), which we will further specify in the next two sections.

Unlike in the full information scenario, bandit feedback in structured prediction means that the gold standard output structure $y$, with respect to which the objective function is evaluated, is not revealed to the learner. Thus we can neither evaluate the task reward $r_y$ nor compute the gradient (32) of the objective function (29).

A solution to this problem is to pass the evaluation of the reward function to the user, i.e., we access the reward directly through user feedback without assuming existence of a fixed reference $y$. We indicate this by dropping the subscript referring to the gold standard structure in the definition of $r$. In the following, we present algorithmic solutions to perform structured learning from this type of partial feedback, called *bandit structured prediction.*

### 3.1.2 Bandit Expected Reward Maximization

Algorithm 1 (page 44) shows the structure of the online learning methods analyzed in this chapter. It assumes a sequence of input structures $x^{(k)}, k = 0, \ldots, K$ that are generated by a fixed, unknown distribution $p(x)$ (line 3). For each input, an output $\tilde{y}^{(k)}$ is sampled from a Gibbs model to perform simultaneous exploitation and exploration on output structures (line 4). Then, feedback $r(\tilde{y}^{(k)})$ to the predicted structure is obtained (line 5). An update is performed by taking a step in the direction of the stochastic gradient $s_k$ (which usually involves the derivative of the sample probability and the reward), at a rate $\gamma_k$ (line 7). As a post-optimization step, a solution $\hat{\theta}$ is chosen from the list of parameters $\theta_k \in \{\theta_0, \ldots, \theta_K\}$ (line 8). The algorithm maximizes the objective below by stochastic gradient ascent optimization. It is non-convex for the specific instantiations in this chapter:

$$\mathbb{E}_{p(x)p_\theta(y|x)}\left[r(y)\right] = \sum_{x} p(x) \sum_{y \in \mathcal{Y}(x)} r(y) p_\theta(y \mid x). \tag{30}$$

**Algorithm 1** Bandit Expected Reward Maximization

```
Input: Sequence of learning rates $\gamma_k$
Output: Optimal parameters $\hat{\theta}$
1: Initialize $\theta_0$
2: for $k = 0, \ldots, K$ do
3:     Observe input structure $x^{(k)}$
4:     Sample output structure $\tilde{y}^{(k)} \sim p_\theta(y \mid x^{(k)})$
5:     Obtain feedback $r(\tilde{y}^{(k)})$
6:     Compute the stochastic gradient $s_k$ of ER: $\nabla_\theta J^{\mathrm{ER}} = \mathbb{E}[s_k^{\mathrm{ER}}]$
7:     Update the parameters $\theta_{k+1} = \theta_k + \gamma_k\, s_k$
8: Choose a solution $\hat{\theta}$ from the list $\{\theta_0, \ldots, \theta_K\}$
```

The core of the algorithm is the sampling: if the model distribution is very peaked, the model exploits, i.e., it presents the most probable outputs to the user. If the distribution is close to uniform, the model explores and presents random outputs to the user. The balance between exploitation and exploration is crucial to the learning process: in the beginning the model is rather uninformed and needs to explore in order to find outputs with high reward, while in the end it ideally converges towards a peaked distribution that exactly fits the user's needs. Pre-training the model, i.e. setting $\theta_0$ wisely, ensures a reasonable exploitation-exploration trade-off.

The "banditization" can be applied to any objective $J$ provided the stochastic gradients $s_k$ are unbiased estimators of the true gradient of the objective ($\nabla_\theta J = \mathbb{E}[s_k]$). The computation of the stochastic gradient is dependent on the underlying model. In the following we derive them for log-linear and neural auto-regressive models and present an intuitive interpretation of the affect of these updates. We refer to the algorithm for online maximization of the expected reward (Eq. 30) as Algorithm ER.

### Log-linear Models

For log-linear statistical models, the Gibbs distribution is based on the product of a joint feature representation of inputs and outputs $\phi : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^d$ and an associated weight vector $w \in \mathbb{R}^d$:

$$p_w(y \mid x) = \exp(w^\top \phi(x, y))/Z_w(x), \tag{31}$$

where $Z_w(x)$ is a normalization constant. For this model, the gradient of objective (30) is as follows:

$$\nabla_\theta \mathbb{E}_{p(x)p_w(y|x)}\left[r(y)\right] = \mathbb{E}_{p(x)p_w(y|x)}\Big[r(y)\left(\phi(x, y)\ - \mathbb{E}_{p_w(y|x)}[\phi(x, y)]\right)\Big]. \tag{32}$$

**Algorithm 2** Sampling Structures

```
Input: Model $\theta$, target sequence length limit $T$
Output: Sequence of sampled tokens $y = (y_1, \dots, y_T)$ and log-probability $q$
1: $y_0 = \text{START}$, $q = 0$
2: $y = (y_0)$
3: for $t \leftarrow 1 \dots T$ do
4:     Sample output token $y_t \sim p_\theta(y \mid x, y_{<t})$
5:     Update log-probability $q = q + \log p_\theta(y_t \mid x, y_{<t})$
6:     Update sequence $y = (y_1, \dots, y_{t-1}, y_t)$
7: end for
8: Return $y$ and $q$
```

The stochastic gradient for a single sample $\tilde{y}^{(k)}$ with respect to $w$ is the following (Sokolov et al., 2015):

$$s_k^{\text{ER}} = r(\tilde{y}^{(k)}) \left( \phi(x^{(k)}, \tilde{y}^{(k)}) - \mathbb{E}_{p_w(y^{(k)}|x^{(k)})}[\phi(x^{(k)}, y^{(k)})] \right). \tag{33}$$

Intuitively, this gradient compares the sampled feature vector $\phi(x^{(k)}, \tilde{y}^{(k)})$ to the average feature vector $\mathbb{E}_{p_w(y^{(k)}|x^{(k)})}[\phi(x^{(k)}, y^{(k)})]$.[3] In the stochastic update (line 7 in Algorithm 1), a step is taken into the direction of this difference, the more so the higher the reward $r(\tilde{y})$ of the sampled structure.

### Neural Models

For deep neural models (the parameter set $\theta$ contains all weight matrices and bias vectors of the network), the Gibbs distribution is factorized over tokens, since the models are not globally normalized (see Equation 6). The gradient for the bandit expected reward objective with respect to a neural model is the following:

$$\begin{aligned} \nabla_\theta \mathbb{E}_{p(x)p_\theta(y|x)}\left[r(y)\right] &= \mathbb{E}_{p(x)p_\theta(y|x)}\left[r(y) \nabla_\theta \log p_\theta(y \mid x)\right] \\ &= \mathbb{E}_{p(x)p_\theta(y|x)}\left[ r(y) \nabla_\theta \sum_{t=1}^{T} \log p_\theta(y_t \mid x; y_{<t}) \right] \end{aligned} \tag{34}$$

In the case of full-information learning where reference outputs are available, we could evaluate all possible outputs against the reference to obtain an exact estimation of the loss function. However, this is not feasible in our setting since we only receive partial feedback for a single output structure per input. Instead,

[3] For the type of globally normalized linear models over hypergraphs that we consider in our experiments, the expectation over features can be efficiently computed with the Inside-Outside algorithm (Li and Eisner, 2009).

we use stochastic approximation to optimize this loss. The stochastic gradient for this objective based on a single output sample $\tilde{y}^{(k)}$ is computed as follows:

$$s_k^{\text{ER}} = r(\tilde{y}^{(k)}) \frac{\partial \log p_\theta(\tilde{y}^{(k)} \mid x^{(k)})}{\partial \theta}. \tag{35}$$

Intuitively, an update with this stochastic gradient will result in an up-weighting of the sampled structure according to the magnitude of the reward. All tokens in the output structure are reinforced to the same extent. Note that Equation (35) is furthermore an instance of the score function gradient estimator (Fu, 2006) (see Section 2.2.2), with the score function

$$\frac{\partial \log p_\theta(\tilde{y} \mid x^{(k)})}{\partial \theta}. \tag{36}$$

Algorithm 2 describes how to sample structures from an encoder-decoder model. Output tokens are sampled from the multinomial distribution over the target vocabulary conditioned on the history of preceding sampled tokens. It corresponds to the sampling algorithm that is used in MRT presented by Shen et al. (2016) with the difference that it samples single structures, does not assume a reference structure, and additionally returns the sample log probabilities.

**Reward Maximization vs. Loss Minimization**

As hinted above, one can either formalize the objective in Equation (19) as maximization of a reward $r \in [0,1]$ or minimization of a task loss $l \in [0,1]$:

$$\theta \leftarrow \arg\max_\theta \mathbb{E}_{p(x)p_\theta(y|x)}[r(y)] \qquad \text{vs.} \qquad \theta \leftarrow \arg\min_\theta \mathbb{E}_{p(x)p_\theta(y|x)}[l(y)].$$

While the minimization perspective is predominant in the NLP literature (Och, 2003; Smith and Eisner, 2006; Gimpel and Smith, 2010; Yuille and He, 2012; He and Deng, 2012; Shen et al., 2016; Wu et al., 2016), we choose the maximization perspective, first because the signal we obtain from humans or reference translations is usually understood as rewards rather than "punishment", and second because this is the standard in reinforcement learning (cf. Section 2.2.1).

Optimizing $\theta$ is then either done with stochastic gradient ascent

$$\begin{aligned} \theta_{k+1} &= \theta_k + s_k, \text{with} \\ s_k &= r(\tilde{y}) \nabla_\theta \log p_\theta(\tilde{y} \mid x) \end{aligned} \tag{37}$$

or gradient descent

$$\begin{aligned} \theta_{k+1} &= \theta_k - s_k, \text{with} \\ s_k &= l(\tilde{y}) \nabla_\theta \log p_\theta(\tilde{y} \mid x) \end{aligned} \tag{38}$$

where $s_k$ is the stochastic gradient at iteration $k$ for the above objectives for a single sample $\tilde{y}$ from the model.

Both learning procedures are equivalent iff $r(\tilde{y}) = -l(\tilde{y})$. For practical experiments it is important to be aware of two extreme cases: In the first case, the update is equivalent to a fully-supervised update with MLE with the reference iff $r(\tilde{y}) = 1$, i.e., when the sampled structure equals the reference $y$. Conversely, in the second case, when $r(\tilde{y}) = 0$ in that case, then $s_k = 0$, i.e., no update is performed. Especially the latter case has implications for the learning progress, depending on where the reward/loss mass is situated.

For example, one could choose to define $r_y(y') = \text{sBLEU}(y', y)$, which results in an update whenever the sentence-BLEU score is greater than zero, with larger updates the closer it comes to the reference. Setting $l_y(y') = 1 - \text{sBLEU}(y', y)$ might superficially seem to express the same semantics since it produces the same ranking of hypothesis. However, the learner receives an update whenever it does not produce the reference, and it makes larger updates the further it is away from the reference. This means it is rather directed away from bad translations than directed towards bad translations.

In our experiments, we found the former case more effective for neural models and the latter for log-linear models. Baseline control variates allow learners to make updates in both edge cases by expanding the range of the rewards to $[-1, 1]$, as we will further elaborate in Section 3.3.2.

### 3.1.3 Bandit Pairwise Preference Learning

Decomposing complex problems into a series of pairwise comparisons has been shown to be advantageous for human decision making (Thurstone, 1927) and for machine learning (Fürnkranz and Hüllermeier, 2010). When for every pair of outputs the correct preference is known (according to a gold standard), pairwise preference learning can be done in the full information supervised setting (Herbrich et al., 2000; Joachims, 2002; Freund et al., 2003; Cortes et al., 2007; Fürnkranz and Hüllermeier, 2010).

However, this is only feasible if either references are given to induce a complete ranking of outputs, or if experts are available to judge all pairs of possible outputs for each input. Banditizing this pairwise feedback means that we require a preference judgment for only one pair of output structures per input, which is a more realistic scenario. For the example of machine translation, this means that instead of requiring numerical assessments of translation quality from human users, only a relative preference judgement on a pair of translations is elicited.

**Algorithm 3** Bandit Pairwise Preference Learning

```
Input: Sequence of learning rates $\gamma_k$
Output: Optimal parameters $\hat{\theta}$
1: Initialize $\theta_0$
2: for $k = 0, \ldots, K$ do
3:     Observe input structure $x^{(k)}$
4:     Sample a pair of output structures $\langle \tilde{y}_i, \tilde{y}_j \rangle^{(k)} \sim p_\theta(\langle y_i, y_j \rangle \mid x^{(k)})$
5:     Obtain feedback $\Delta(\langle \tilde{y}_i, \tilde{y}_j \rangle^{(k)})$
6:     Compute the stochastic gradient $s_k$ of the PR objective: $\nabla_\theta J^{\mathrm{PR}} = \mathbb{E}[s_k^{\mathrm{PR}}]$
7:     Update the parameters $\theta_{k+1} = \theta_k + \gamma_k s_k$
8: Choose a solution $\hat{\theta}$ from the list $\theta_0, \ldots, \theta_K$
```

This idea can be formalized as an expected reward objective with respect to a conditional distribution of pairs of structured outputs. Let $\mathcal{P}(x) = \{\langle y_i, y_j \rangle \mid y_i, y_j \in \mathcal{Y}(x)\}$ denote the set of output pairs for an input $x$, and let $\Delta(\langle y_i, y_j \rangle) : \mathcal{P}(x) \to [0, 1]$ denote a task reward function that specifies a preference of one output $y_i$ over another output $y_j$.[4]

In the experiments, we induce pairwise feedback from absolute feedback, resulting in either continuous or binary feedback. Continuous pairwise rewards are computed from absolute rewards $r(y_i), r(y_j) \in [0, 1]$ by subtraction

$$\Delta^{\mathrm{CONT}}(\langle y_i, y_j \rangle) = \max(r(y_i) - r(y_j), 0), \tag{39}$$

and binary pairwise rewards by binarizing the comparison

$$\Delta^{\mathrm{BIN}}(\langle y_i, y_j \rangle) = \lceil \Delta^{\mathrm{CONT}}(\langle y_i, y_j \rangle) \rceil. \tag{40}$$

We can now instantiate objective (30) with pairs of output structures, forming the pairwise preference objective (PR):

$$\mathbb{E}_{p(x)p_\theta(\langle y_i, y_j \rangle \mid x)} \left[\Delta(\langle y_i, y_j \rangle)\right] = \sum_x p(x) \sum_{\langle y_i, y_j \rangle \in \mathcal{P}(x)} \Delta(\langle y_i, y_j \rangle)\, p_\theta(\langle y_i, y_j \rangle \mid x). \tag{41}$$

stochastic gradient ascent optimization of this objective leads to Algorithm 3. The objective is non-convex in the use cases in this chapter. Maximizing this objective will assure that high probabilities are assigned to pairs with high reward due to ranking $y_i$ over $y_j$. Stronger assumptions on the learned probability ranking can be made if transitivity and asymmetry of the ordering of feedback structures are assumed.[5]

[4]This can be formalized as a minimization problem as well (cf. Section 3.1.2), where the loss function expresses dispreference or a misranking instead.

[5]See (Busa-Fekete and Hüllermeier, 2014) for an overview of bandit learning from consistent and inconsistent pairwise comparisons.

**Algorithm 4** Sampling Pairs of Structures

```
Input: Model $\theta$, target sequence length limit $T$
Output: Pair of sequences $\langle y, y' \rangle$ and its log-probability $q$
 1: $p = 0$
 2: $y, y', \hat{y} = (\text{START})$
 3: Determine the perturbation position $i \sim \mathcal{U}(1, T)$
 4: for $t \leftarrow 1 \dots T$ do
 5:     Compute greedy output $\hat{y}_t = \arg\max_{y \in V} p_\theta^+(y \mid x, \hat{y}_{<t})$
 6:     if $i = t$ then
 7:         Sample "negative" output token $y'_t \sim p_\theta^-(y \mid x, \hat{y}_{<t})$
 8:         Update log-probability $q = q + \log p_\theta^-(y'_t | x, \hat{y}_{<t})$
 9:     else
10:         Sample "positive" output token $y_t \sim p_\theta^+(y \mid x, \hat{y}_{<t})$
11:         Update log-probability $q = q + \log p_\theta^+(y'_t \mid x, \hat{y}_{<t})$
12:     end if
13:     Update sequence $y = (y_1, \dots, y_{t-1}, y_t)$
14:     Update sequence $y' = (y'_1, \dots, y'_{t-1}, y'_t)$
15:     Update $\hat{y} = (\hat{y}_1, \dots, \hat{y}_{t-1}, \hat{y}_t)$
16: end for
17: Return pair $\langle y, y' \rangle$ and $q$
```

**Log-linear Models**

For the log-linear model, we assume the following factorized Gibbs model:

$$\begin{aligned} p_w(\langle y_i, y_j \rangle \mid x) &= \frac{\exp(w^\top \phi(x, y_i) - w^\top \phi(x, y_j))}{\sum\limits_{\langle y_i, y_j \rangle \in \mathcal{P}(x)} \exp(w^\top \phi(x, y_i) - w^\top \phi(x, y_j))} \\ &= \frac{\exp(w^\top (\phi(x, y_i) - \phi(x, y_j)))}{\sum\limits_{\langle y_i, y_j \rangle \in \mathcal{P}(x)} \exp(w^\top (\phi(x, y_i) - \phi(x, y_j)))} \\ &= p_w(y_i \mid x) \times p_{-w}(y_j \mid x). \end{aligned} \tag{42}$$

The contrasting output $y_j$ is hence sampled from the distribution obtained from negating the model weights. The factorization of this model into the product $p_w(y_i \mid x) p_{-w}(y_j \mid x)$ enables efficient sampling and calculation of expectations. The stochastic gradient $s_k$ for model updates in Algorithm 3 is computed as

$$s_k = \Delta(\langle \tilde{y}_i, \tilde{y}_j \rangle^{(k)}) \, (\phi(x^{(k)}, \langle \tilde{y}_i, \tilde{y}_j \rangle^{(k)}) - \mathbb{E}_{p_w(\langle y_i, y_j \rangle | x^{(k)})}[\phi(x^{(k)}, \langle y_i, y_j \rangle^{(k)})]). \tag{43}$$

### Neural Models

Analogously to the sequence-level sampling for log-linear models, we define the following probabilities for sampling on the token level:

$$p_\theta^+(y_t = w_i \mid x, \hat{y}_{<t}) = \frac{\exp(o_{w_i})}{\sum_{v=1}^{|\mathcal{V}_{trg}|} \exp(o_{w_v})},$$
$$p_\theta^-(y_t = w_j \mid x, \hat{y}_{<t}) = \frac{\exp(-o_{w_j})}{\sum_{v=1}^{|\mathcal{V}_{trg}|} \exp(-o_{w_v})}.$$

The effect of the negation within the softmax is that the two distributions $p_\theta^+$ and $p_\theta^-$ rank the next candidate target words (given the same history, here the greedy output $\hat{y}_{<t}$) in opposite order. Globally normalized models as in the linear case, or LSTM-CRFs (Huang et al., 2015) for the non-linear case, would allow sampling full structures such that the ranking over full structures is reversed. But in the case of locally normalized RNNs we retrieve only locally reversed-rank samples.

Since we want the model to learn to rank $y_i$ over $y_j$, we would have to sample $y_i$ word-by-word from $p_\theta^+$ and $y_j$ from $p_\theta^-$. However, sampling all words of $y_j$ from $p_\theta^-$ leads to translations that are neither fluent nor source-related, so we propose to randomly choose one token of $y_j$ which is sampled from $p_\theta^-$ and sample the remaining words from $p_\theta^+$. We found that this method produces suitable negative samples which are only slightly perturbed and still relatively fluent and source-related. A detailed algorithm is given in Algorithm 4 (page 49).

In the same manner as for linear models, we define

$$p_\theta(\langle y_i, y_j \rangle \mid x) = p_\theta(y_i \mid x) \times p_\theta(y_j \mid x).$$

Due to the token-level sampling scheme and local normalization, however, this is not as clean as in the linear case, as $p_\theta(\langle y_i, y_j \rangle \mid x)$ is no longer a proper probability distribution.

The stochastic gradient for the PR objective (41) is then

$$s_k^{\text{PR}} = \Delta(\langle y_i, y_j \rangle) \times \left( \frac{\partial \log p_\theta(y_i \mid x_k)}{\partial \theta} + \frac{\partial \log p_\theta(y_j \mid x_k)}{\partial \theta} \right). \tag{44}$$

This training procedure resembles well-known approaches for noise contrastive estimation (Gutmann and Hyvärinen, 2010) with negative sampling that are commonly used for neural language modeling (Collobert et al., 2011; Mnih and Teh, 2012; Mikolov et al., 2013a). In these approaches, negative samples are drawn from a non-parametric noise distribution, whereas we draw them from the perturbed model distribution.

### 3.1.4 Bandit Cross-Entropy Minimization

The standard theory of stochastic optimization predicts considerable improvements in convergence speed depending on the functional form of the objective. This motivates the formalization of convex upper bounds on expected normalized gains as presented by Green et al. (2014). Their objective is based on a reward function $\bar{r} : \mathcal{Y} \to [0, 1]$ that is normalized over $n$-best lists where $\bar{r}(y) = \frac{r(y)}{Z_r(x)}$ and $Z_r(x) = \sum_{y \in n\text{-best}(x)} r(y)$. It can be seen as the cross-entropy of model $p_\theta(y \mid x)$ with respect to the "true" distribution $\bar{r}(y)$:

$$\mathbb{E}_{p(x)\bar{r}(y)} \left[ -\log p_\theta(y \mid x) \right] = -\sum_x p(x) \sum_{y \in \mathcal{Y}(x)} \bar{r}(y) \log p_\theta(y \mid x). \tag{45}$$

In this work, we work with unnormalized gain functions, i.e., $r \in [0, 1]$ instead of $\bar{r}$, since the normalization is prohibitive in a bandit setting since it would require to elicit user feedback for each structure in the output space or the $n$-best list. Substituting $\bar{r}$ with $r$, the gradient of objective (48) is:

$$\begin{aligned}\nabla_\theta \mathbb{E}_{p(x)r(y)} \left[ -\log p_\theta(y \mid x) \right] &= -\sum_x p(x) \sum_{y \in \mathcal{Y}(x)} r(y) \nabla_\theta \log p_\theta(y \mid x) \\ &= -\mathbb{E}_{p(x)r(y)} \left[ \nabla_\theta \log p_\theta(y \mid x) \right]. \end{aligned} \tag{46}$$

In order to obtain a stochastic gradient of (46) for stochastic updates, we use importance sampling to sample from distribution $p_s(y \mid x)$ instead of sampling from $r$ (which is infeasible), and normalize $r$ by $\frac{1}{p_s(y|x)}$ to correct the sampling bias:

$$\begin{aligned}-\mathbb{E}_{p(x)r(y)} \left[ \nabla_\theta \log p_\theta(y \mid x) \right] &= -\sum_x p(x) \sum_{y \in \mathcal{Y}(x)} \frac{r(y)}{p_s(y \mid x)} p_s(y \mid x) \nabla_\theta \log p_\theta(y \mid x) \\ &= \mathbb{E}_{p(x)p_s(y|x)} \Big[ -\frac{r(y)}{p_s(y \mid x)} \nabla_\theta \log p_\theta(y \mid x) \Big] \end{aligned} \tag{47}$$

The bias correction is an instance of importance sampling that also serves as a multiplicative control variate (see Section 2.2.3).

The expected reward objective is non-convex even for the linear models presented above, but objective (45) is convex for the linear models. We call this objective *bandit cross-entropy minimization* and refer to it as CE. Since the resulting algorithm closely resembles the ER algorithm (it is identical up to the computation of the stochastic gradient and the sign of the update), we do not give a separate description in pseudo code. Norouzi et al. (2016) proposed a very similar objective for reward-augmented MLE concurrently to our work. Asl alternative to our importance sampling, RAML directly samples from the normalized (and exponentiated) reward distribution. If reward functions can be de-composed into local edit distances, stratified sampling can be used to make sampling from this distribution feasible. For machine translation Norouzi et al. (2016) therefore simplified the reward to Hamming distance and left sampling according to BLEU for future work.

**Log-linear Models**

For a proper probability distribution $\bar{r}(y)$, one can show that objective (45) is a convex upper bound on objective (30) (when working with linear models) by applying Jensen's inequality to the convex negative logarithm function. Not normalizing the reward function sacrifices the upper bound but preserves convexity. This can be seen by rewriting the objective as the sum of a linear and a convex function in $w$:

$$\begin{aligned}\mathbb{E}_{p(x)r(y)}\left[-\log p_w(y \mid x)\right] &= -\sum_x p(x) \sum_{y\in\mathcal{Y}(x)} r(y) \log p_w(y \mid x) \\ &= -\sum_x p(x) \sum_{y\in\mathcal{Y}(x)} r(y) w^\top \phi(x,y) \\ &\quad + \sum_x p(x)(\log \sum_{y\in\mathcal{Y}(x)} \exp(w^\top \phi(x,y)))\, R(x),\end{aligned} \tag{48}$$

where $R(x) = \sum_{y\in\mathcal{Y}(x)} r(y)$ is a constant factor not depending on $w$.

The stochastic gradient $s_k$ of the cross-entropy objective for linear models is

$$s_k = -\frac{r(\tilde{y}^{(k)})}{p_w(\tilde{y}^{(k)} \mid x^{(k)})}\,(\phi(x^{(k)}, \tilde{y}^{(k)}) - \mathbb{E}_{p_w}[\phi(x^{(k)}, y^{(k)})]). \tag{49}$$

It updates the model according to the ratio of gain versus current probability of the sampled structure. A positive ratio expresses a preference of the sampled structure under the gain function compared to the current probability estimate. We compare the sampled feature vector to the average feature vector (just like for ER), and we update towards the sampled feature vector relative to this ratio.

We instantiate the importance sampling distribution $p_s(y \mid x)$ to the current model distribution $p_w(y \mid x)$ in order to present progressively more useful structures to the user.

Strong convexity of objective (48) can be achieved easily by adding an $\ell_2$ regularizer $\frac{\lambda}{2}\|w\|^2$ with constant $\lambda > 0$. The algorithm is then modified to use the following regularized update rule $w_{k+1} = w_k - \gamma_k\,(s_k + \frac{\lambda}{K} w_k)$.

In contrast to Algorithms 1 and 3, each update is affected by the ratio of gain versus current probability, which changes over time and is unreliable when training is started. Especially when model probabilities are low, this increases the variance already present in stochastic optimization. Clipping sampling probabilities (Ionides, 2008) is a simple but effective solution for more stable updates, that means that the probability score in the scaling ratio $\frac{r(\tilde{y}^{(k)})}{p_w(\tilde{y}^{(k)}|x^{(k)})}$ is replaced by its clipped version $\frac{r(\tilde{y}^{(k)})}{\max(p_w(\tilde{y}^{(k)}|x^{(k)}),c)}$, where $c$ is a clipping constant that needs to be tuned.

### Neural Models

The advantage of the cross-entropy minimization objective to have a (strongly) convex optimization problem does not hold for neural models, since their non-linear form turns it into a non-convex optimization problem. Nevertheless, we derive the gradient here for the sake of completeness. The differentiable part of its stochastic gradient is the same as for ER, but it is additionally scaled by a factor dependent on the reward and the probability of the structure:

$$s_k^{\mathrm{CE}} = -\frac{r(\tilde{y}^{(k)})}{p_\theta(\tilde{y}^{(k)} \mid x^{(k)})} \frac{\partial \log p_\theta(\tilde{y}^{(k)} \mid x^{(k)})}{\partial \theta}$$

In practice, model scores for a full translation are often magnitudes lower than log-linear models since they are only locally normalized, and token probabilities are multiplied to form the score for the full sequence (see Equation (6)). This makes the cross-entropy criterion less suitable for neural models, since the vanishing denominator makes the ratio $\frac{r(\tilde{y}^{(k)})}{p_\theta(\tilde{y}^{(k)}|x^{(k)})}$ grow.

Stabilization by clipping can be applied here as well, but in preliminary experiments the clipping constant $c$ had to be set to a high value to prevent explosion of the gradient. This resulted in clipping every single example, which turns the denominator of the ratio into a constant. Effectively, it becomes equivalent to the update of ER (35) up to a scalar factor. We therefore omit CE in the experiments for neural models.

## 3.2 Convergence Analysis

### Iteration Complexity

To analyze convergence, we describe Algorithms ER, PR, and CE as stochastic first-order (SFO) methods in the framework of Ghadimi and Lan (2012). We assume lower bounded, differentiable objective functions $J(w)$ with Lipschitz continuous gradient $\nabla J(w)$ satisfying

$$\|\nabla J(w+w') - \nabla J(w)\| \leq L\|w'\| \quad \forall w, w', \exists L \geq 0. \tag{50}$$

For an iterative process of the form $w_{k+1} = w_k - \gamma_k \, s_k$, the conditions to be met are 1) unbiasedness of the gradient estimate

$$\mathbb{E}[s_k] = \nabla J(w_k), \quad \forall k \geq 0, \tag{51}$$

and 2) boundedness of the variance of the stochastic gradient

$$\mathbb{E}[||s_k - \nabla J(w_k)||^2] \leq \sigma^2, \quad \forall k \geq 0. \tag{52}$$

Condition (51) is met for all three algorithms by taking expectations over all sources of randomness, i.e., over random inputs and output structures. Assuming $\nabla \log p_\theta(y \mid x) \leq R$, $r(y) \in [0,1]$ for all $x, y$, and since the ratio $\frac{r(\tilde{y}_k)}{\hat{p}_{\theta_k}(\tilde{y}_k|x_k)}$ is bounded (with a fixed maximum length for neural models), the variance in condition (52) is bounded.

Convergence speed can be quantified in terms of the number of iterations needed to reach an accuracy of $\epsilon$ for a gradient-based criterion $\mathbb{E}[\|\nabla J(\theta_k)\|^2] \leq \epsilon$.

For stochastic optimization of *non-convex* objectives, the iteration complexity with respect to this criterion is analyzed as $\mathcal{O}(1/\epsilon^2)$ in (Ghadimi and Lan, 2012). There are other constants besides $\epsilon$ in the iteration complexity that play a role in practice—as we will show in the experiments—such as the Lipschitz constant $L$ and the variance $\sigma^2$: $\mathcal{O}(\frac{L}{\epsilon} + \frac{L\sigma^2}{\epsilon^2})$. This complexity result applies to linear implementations of ER and PR, and to all neural implementations.

For PR it is worth highlighting that in contrast to gradient-free stochastic zeroth-order (SZO) approaches for pairwise learning, as for example discussed in (Sokolov et al., 2018), its iteration complexity does not depend on the dimensionality of the function, which is beneficial especially with neural models. On the downside, however, we have to resort to approximations for a pairwise probability distribution due to local token-wise sampling and normalization for the neural PR implementation (see Section 3.1.3).

The iteration complexity of stochastic optimization of *(strongly) convex* objectives, here the case for the log-linear models with a single weight vector $w$ for CE, has been analyzed as at best $\mathcal{O}(1/\epsilon)$ for the suboptimality criterion $\mathbb{E}[J(w_k)] - J(w^*) \leq \epsilon$ for decreasing learning rates (Polyak, 1987).[6] Despite this advantage in the best case, convergence in practice may be slowed down by additional variance that might get introduced in the process of making the objective (strongly) convex, such as through sample normalization in Algorithm CE.

**Measuring Numerical Convergence**

For the purpose of obtaining numerical results on convergence speed, we compute estimates of the expected squared gradient norm $\mathbb{E}[\|\nabla_w J(w_k)\|^2]$, the Lipschitz constant $L$ and the variance $\sigma^2$ in which the asymptotic bounds on iteration complexity grow linearly.

We estimate the squared gradient norm by the squared norm of the stochastic gradient $\|s_K\|^2$ at a fixed time horizon $K$. The Lipschitz constant $L$ in

[6]For constant learning rates, Solodov (1998) show even faster convergence in the search phase of strongly-convex stochastic optimization.

equation (50) is estimated by $\max_{i,j} \frac{\|s_i - s_j\|}{\|w_i - w_j\|}$ for 500 pairs $w_i$ and $w_j$ randomly drawn from the weights produced during training. The variance $\sigma^2$ in equation (52) is computed as the empirical variance of the stochastic gradient, taken at regular intervals after each epoch of size $E$, yielding the estimate $\frac{1}{N}\sum_{n=1}^{N}\|s_{nE} - \frac{1}{N}\sum_{k=1}^{K} s_{nE}\|^2$ where $N = \lfloor \frac{K}{E} \rfloor$. All estimates include multiplication of the stochastic gradient with the learning rate. For comparability of results across different algorithms, we use the same $T$ and the same constant learning rates for all algorithms.[7] Section 3.4.1 compares the three algorithms with empirical estimates for gradient norm and variance, and Lipschitz constant for linear SMT models.

## 3.3 Reward Baselines

Due to the stochasticity of the updates, our algorithms might suffer from large variance of the gradients during learning. In this section we explain how a reward baseline, an instance of additive control variates, can be applied to our algorithms and which effects it has on gradient updates.

### 3.3.1 Additive Control Variate

A simple implementation of an additive control variate (see Section 2.2.3) is the subtraction of a baseline reward $b_k = \frac{1}{k}\sum_{j=1}^{k} r(y^{(j)})$, a running average of historic rewards, from the current reward $r(y^{(k)})$, yielding

$$\begin{aligned}\mathbb{E}[\nabla \log p_\theta(y^{(k)} \mid x^{(k)}) r(y^{(k)})] = \mathbb{E}[\nabla \log p_\theta(y^{(k)} \mid x^{(k)})(r(y^{(k)}) - b_k)]\\ + \mathbb{E}[\nabla \log p_\theta(y^{(k)} \mid x^{(k)}) b_k] \qquad (53)\end{aligned}$$

The expectation (over inputs $x$ and outputs $y$) of the control variate $\mathbb{E}[Y]$ is zero in this case, since

$$\begin{aligned}\mathbb{E}[\nabla \log p_\theta(y^{(k)} \mid x^{(k)}) b_k] &= \mathbb{E}[\frac{\nabla p_\theta(y^{(k)} \mid x^{(k)})}{p_\theta(y^{(k)} \mid x^{(k)})} b_k] &(54)\\ &= b_k \int p_\theta(y^{(k)} \mid x^{(k)}) \frac{\nabla p_\theta(y^{(k)} \mid x^{(k)})}{p_\theta(y^{(k)} \mid x^{(k)})}\, dy &(55)\\ &= b_k \nabla \int p_\theta(y^{(k)} \mid x^{(k)})\, dy &(56)\\ &= b_k \nabla 1 = 0. &(57)\end{aligned}$$

[7]Note that the squared gradient norm upper bounds the suboptimality criterion s.t. $\|\nabla J(\theta_k)\|^2 \geq 2\lambda J(\theta_k)] - J(\theta^*)$ for strongly convex functions. Together with the use of constant learning rates this means that we measure convergence to a point near an optimum for strongly convex objectives.

(54) is obtained by applying the log-derivative rule ($\nabla \log f = \frac{\nabla f}{f}$), (55) by replacing the expectation with the integral, and (56) by switching the order of differentiation and integration (Leibniz integral rule).

Alternatively, Ranganath et al. (2014) propose the score function (Equation 36) directly as additive control variate, in which case the $\mathbb{E}[Y] = 0$ as well. We will explore both variants of the additive control variate for the stochastic gradient of ER (35) in our experiments.

### 3.3.2 Relativizing Rewards

Reward baselines do not only have a variance reduction effect, but also an effect on the scaling of rewards. Instead of learning from absolute, positive reward signals $\in [0, 1]$, models now learn from relative reward signals $\in [-1, 1]$. That means that gradient updates can now switch the sign, so the updates can be towards or away from the sample (as opposed to only making steps towards the sample, see Section 3.1.2). This allows the learner to make customized updates relative to the current state of the model.

In order to analyze the effect on the gradient update further, we inspect the computation of the stochastic gradient update for ER and how it behaves in dependence of model score and reward. We simplify the analysis by looking at a single time step in isolation (the bandit learning perspective), like an output for log-linear models, or one token of a sequence for the neural models, and omit time-specific indices for $y$ for better readability.[8]

Recall that model scores are obtained by softmax normalization (7) of the network output vector $\mathbf{o}$ that contains a score for every output $k \in \mathcal{V}_{trg}$, or a Gibbs distribution with a linear model (31). The gradient update is composed of the score function (36) and the reward, see (35). Applying the chain rule to the score function (detailed derivation in Appendix A.1), we obtain

$$\begin{aligned}
\nabla \log p_\theta(y = j \mid x) &= \nabla_\theta \log \left[\operatorname{softmax}(\mathbf{o})\right]_j && (58)\\
&= \sum_k \left[ ([[k = j]] - \frac{\exp(o_k)}{\sum_{i=1}^{|\mathcal{V}_{trg}|} \exp(o_i)}) \frac{\partial o_k}{\partial \theta} \right] \\
&= (1 - p_\theta(y = j \mid x)) \frac{\partial o_j}{\partial \theta} - \sum_{k \neq j} \left[ p_\theta(y = k \mid x) \frac{\partial o_k}{\partial \theta} \right]. && (59)
\end{aligned}$$

The derivative of the model output with respect to the model parameters $\frac{\partial o_k}{\partial \theta}$ is dependent on the exact model definition, since $o_k$ is either computed by a neural

[8]For neural models, the score function is decomposed into a sum of token-level scoring functions due to the factorization of probabilities (6), which are all multiplied with the same sequence-level reward for the gradient update.

network, or simply the dot product between a weight vector and a feature vector for log-linear models.[9]

With the stochastic update for the sampled output $j$ the model parameters are updated towards the direction of the gradient (reward maximization), scaled by the learning rate $\gamma$ (see Section 3.1.2):

$$\theta' = \theta + \gamma \times r(j) \times \sum_k \left[ ([[k = j]] - p_\theta(y = k \mid x)) \frac{\partial o_k}{\partial \theta} \right] \tag{60}$$

For the parameter update, a weighted sum of the derivatives of the network outputs (the un-normalized scores) are scaled by the learning rate $\gamma$, and the reward of the sampled output $r(j)$. The weighting of the sum is determined by model probabilities for each output. The sum of weights for the partial derivatives of the network outputs at a given update step has the largest absolute value when the probability of the sample $j$ is low. Therefore, the potentially largest updates can be achieved when low-probability samples receive high rewards.[10] On the other hand, when the sampled output is very likely ($p_\theta(y = k \mid x) \approx 1$), the parameters are hardly updated (regardless of the value of the reward), due to vanishing factors in the sum. This observation explains the necessity of exploration (especially for warm-start models that already have peaked output distributions close to convergence), such that low-probability outputs with high rewards can be discovered.

As briefly discussed in Section 3.1.2, learning is also hindered when receiving low rewards $r(y) \approx 0$ (e.g. in cold-start learning or with very sparse rewards), since they scale down the gradient update. The baseline can remedy this problem, because it modifies the scale of the reward depending on the learning context. Subtracting the baseline reward from the current reward can lead to either a large negative update if previous samples received a much higher reward ($b_k >> r(y^{(k)})$), or a large positive update if previous samples received a much lower reward ($b_k << r(y^{(k)})$).

## 3.4 Domain Adaptation Experiments

After having defined three algorithms for banditized machine translation, we now evaluate them in practice. On the task of domain adaptation we measure for both SMT (Section 3.4.1 and NMT (Section 3.4.2) how well pre-trained out-of-domain models can be adapted to a new domain with weak bandit feedback, and how many iterations they require to reach the best performance.

[9]For log-linear models $\frac{\partial o_k}{\partial w} = \phi(x, k)$.

[10]The actual gradient norm depends on the values of the partial derivatives. However, these do not depend on the decision which output was chosen as a sample for the current step. This decision affects only the scalars.

### 3.4.1 Experiments for SMT

In this section we first describe the setup of the experiments for SMT, and then discuss the results for the three algorithms presented in Section 3.1 that include 1) the held-out set performance after training with bandit feedback, 2) the number of iterations until convergence, and 3) the measurements for numerical convergence analysis.

#### Experimental Design

Our experiments follow an online learning protocol where on each of a sequence of rounds, an output structure is randomly sampled, and feedback to it is used to update the model (Shalev-Shwartz, 2012). We simulate bandit feedback by evaluating the reward function $r$ against gold standard structures which are never revealed to the learner (Agarwal et al., 2014). Training is started from an pre-trained model that was only trained on out-of-domain data and has to get adapted to the new domain solely with bandit feedback.

Following the standard practice of early stopping by performance evaluation on a development set, we compute convergence speed as the number of iterations needed to find the point of optimal performance before overfitting on the development set. The convergence criterion is thus based on the respective task reward function $r(\hat{y}_{w_t}(x))$ under maximum-a-posteriori prediction $\hat{y}_w(x) = \arg\max_{y \in \mathcal{Y}(x)} p_w(y \mid x)$. This lets us compare convergence across different objectives, and is justified by the standard practice of performing online-to-batch conversion by early stopping on a development set (Littlestone, 1989), or by tolerant training to avoid overfitting (Solodov, 1998).

As a further measure for comparability of convergence speeds across algorithms, we employ small constant learning rates in all experiments. The use of constant learning rates for ER and PR is justified by the analysis by Ghadimi and Lan (2012). For CE, the use of constant learning rates effectively compares convergence speed towards an area in close vicinity of a local minimum in the search phase of the algorithm (Bottou, 2004).

The development data are also used for meta-parameter search. Optimal configurations are listed in Appendix B.1 in Table 37. Final testing was done by computing BLEU scores on a further unseen test set using the model found by online-to-batch conversion. For bandit-type algorithms, final results are averaged over three runs with different random seeds, but the same seeds across algorithms. For statistical significance testing of results against baselines we use Approximate Randomization testing (Noreen, 1989).

| | **Full Supervision** | | **bandit feedback** | | |
|---|---|---|---|---|---|
| | **out-of-domain** | **in-domain** | **ER** | **PR** | **CE** |
| $n$-best, dense | 25.88 | 28.41 | $26.89_{\pm 0.03}$ | $27.45_{\pm 0.04}$ | $27.63_{\pm 0.05}$ |
| h-graph, sparse | 26.51 | 28.31 | $26.67_{\pm 0.008}$ | $27.33_{\pm 0.05}$ | $27.13_{\pm 0.1}$ |

Table 2: Test set evaluation in BLEU for full information lower and upper bounds and partial information bandit learners on the in-domain NC data (ER: expected reward maximization, PR: pairwise ranking, CE: cross-entropy minimization).

### Data & Model

We perform domain adaptation from Europarl (EP) to News Commentary (NC) domains using the French-to-English portion of the data provided by Koehn and Schroeder (2007) for the WMT news translation task in 2007.[11] One difference of our experiment compared to (Sokolov et al., 2015) is our use of the synchronous context-free grammar decoder `cdec` (Dyer et al., 2010) instead of the phrase-based `Moses` decoder. Furthermore, in addition to bandit learning for re-ranking on unique 5,000-best lists (referred to as "$n$-best"), we perform ranking on hypergraphs (referred to as "h-graph") with re-decoding after each update. Sampling and computation of expectations on the hypergraph uses the Inside-Outside algorithm over the expectation semiring (Li and Eisner, 2009).

The $n$-best model uses a standard set of 15 dense features (6 lexicalized reordering features, two (out-of- and in-domain) language models, 5 translation model features, distortion and word penalty). The h-graph model uses additional lexicalized sparse features: rule-id features, rule source and target bigram features, and rule shape features. The out-of-domain h-graph model contain 214,642 active features, the upper-bound in-domain model 133,531 active features.

For all machine translation experiments we tokenize, lowercase and align words using `cdec` tools, train 4-gram in-domain and out-of-domain language models (on the English sides of EP and in-domain NC), for all tuning in the full-information setting we used `cdec`'s lattice MERT (Och, 2003). The out-of-domain baseline machine translation models were trained on 1.6M parallel EP data and tuned with `cdec`'s implementation of MIRA (Chiang, 2012) on out-of-domain Europarl `dev2006` dev set. The full-information in-domain machine translation models was trained on the same Europarl data and tuned on news in-domain sets.

Learning under bandit feedback starts at the learned weights of the out-of-domain median models. It uses the parallel in-domain data (see Table 5 for the sizes) to simulate bandit feedback, by evaluating the sampled translation against the reference using as loss function a smoothed per-sentence $1 - \text{BLEU}$ (zero

[11] `http://www.statmt.org/wmt07/shared-task.html`

$n$-gram counts being replaced with $\epsilon = 0.01$). For pairwise preference learning we use binary feedback resulting from the comparison of the BLEU scores of the sampled translations.

To speed up training for hypergraph re-decoding, the training instances were reduced to those with at most 60 words (38,350 sentences). Training is distributed across 38 shards using multitask-based feature selection for sparse models (Simianer et al., 2012), where after each epoch of distributed training, the top 10k features across all shards are selected, and all other features are set to zero. The meta-parameters were adjusted on the in-domain dev sets. The final results are obtained on separate in-domain test sets by averaging three independent runs for the optimal validation set meta-parameters.

### Results

**Held-out set performance.** We report BLEU scores on the in-domain test set in Table 2. The gap between out-of-domain and in-domain models defines the range of possible improvements (+2.53 BLEU for $n$-best and +1.8 for h-graph) for bandit learning. For bandit learners we report the average results over three runs with different random seeds. The pairwise feedback simulation, binary or continuous, is treated as a hyperparameter. Binary feedback performed better for both models, so we report only these results. The improvements over the out-of-domain baseline are statistically significant for all bandit learners. Thus we can confirm and generalize the results of Sokolov et al. (2015) to different decoders and new objectives, and with re-decoding instead of $n$-best re-ranking. The ER objective is outperformed by CE and PR for both decoders.

| | **ER** | **PR$^{\text{BIN}}$** | **CE** |
|---|---|---|---|
| $n$-best, dense | 3.8M | 1.2M | 1.2M |
| h-graph, sparse | 370k | 115k | 281k |

Table 3: Number of iterations until stopping criterion on development data.

**Convergence speed.** Early stopping by task performance on the development led to the selection of algorithm PR at a number of iterations that is by a factor of 2-4 smaller compared to Algorithms ER and CE for the h-graph model, as reported in Table 3. For the $n$-best reranking model PR and CE both take the same number of iterations, approx. three times fewer than ER. $n$-best rescoring yields slightly better results than re-decoding with sparse models.

**Numerical convergence results.** Estimates of $\mathbb{E}[\|\nabla_w J(w_k)\|^2]$, $L$ and $\sigma^2$ for three runs of each algorithm with different random seeds for the sparse

| **Algorithm** | $\|s_K\|^2$ | $L$ | $\sigma^2$ |
|---|---|---|---|
| CE | $3.04_{\pm 0.02}$ | $0.54_{\pm 0.3}$ | $35\ _{\pm 6}$ |
| ER | $0.02_{\pm 0.03}$ | $1.63_{\pm 0.67}$ | $3.13 \times 10^{-4}{}_{\pm 3.60 \times 10^{-6}}$ |
| PR$^{\text{BIN}}$ | $2.88 \times 10^{-4}{}_{\pm 3.40 \times 10^{-6}}$ | $0.08_{\pm 0.01}$ | $3.79 \times 10^{-5}{}_{\pm 9.50 \times 10^{-8}}$ |
| PR$^{\text{CONT}}$ | $1.03 \times 10^{-8}{}_{\pm 2.91 \times 10^{-10}}$ | $0.10_{\pm 5.70 \times 10^{-3}}$ | $1.78 \times 10^{-7}{}_{\pm 1.45 \times 10^{-10}}$ |

Table 4: Estimates of squared gradient norm $\|s_K\|^2$, Lipschitz constant $L$ and variance $\sigma^2$ of stochastic gradient (including multiplication with learning rate) for fixed time horizon $K$ and constant learning rates $\gamma = 1.0 \times 10^{-6}$ for sparse SMT. $K = 767,000$ for all models. The clipping and regularization parameters for CE are set as in Table 37. Results are averaged over three runs of each algorithm, with standard deviation shown in subscripts.

h-graph re-decoding model are listed in Table 4 (page 61). At time horizon $K$, the estimated squared gradient norm for Algorithm PR is several orders of magnitude smaller than for Algorithms ER and CE. Furthermore, the estimated Lipschitz constant $L$ and the estimated variance $\sigma^2$ are smallest for Algorithm PR. Since the iteration complexity increases linearly with respect to these terms, smaller constants $L$ and $\sigma^2$ and a smaller value of the estimate $\mathbb{E}[\|\nabla_\theta J(w_k)\|^2]$ at the same number of iterations indicates fastest convergence for Algorithm PR. This theoretically motivated result is consistent with the practical convergence criterion of early stopping on development data: Algorithm PR which yields the smallest squared gradient norm at time horizon $K$ also needs the smallest number of iterations to achieve optimal performance on the development set. Note that for comparability across algorithms, the same constant learning rates were used in all runs. However, we obtained similar relations between algorithms by using the meta-parameter settings chosen on development data as shown in Table 2. Furthermore, the above tendencies hold for both settings of the meta-parameter BIN or CONT of Algorithm PR.

**Summary**

We applied the objectives and algorithms for structured prediction from bandit feedback presented in Section 3.1 to a machine translation tasks with log-linear models and compared them in terms of convergence speed and test set performance. Test set performance was increased by up to 1.75 BLEU points by using weak bandit feedback only to adapt an out-of-domain model to new in-domain data. Our experimental results showed a consistent advantage of convergence speed for bandit pairwise preference learning. In light of the standard stochastic approximation analysis, which predicts a convergence advantage for strongly convex objectives over convex or non-convex objectives for log-linear models, this result is surprising. However, the result can be explained by considering important empirical factors such as the variance of stochastic updates.

### 3.4.2 Experiments for NMT

In the following, we present an experimental evaluation of the learning objectives presented above (Section 3.1) on neural machine translation domain adaptation (Luong and Manning, 2015; Servan et al., 2016; Freitag and Al-Onaizan, 2016). For this task we build on the evaluation of Freitag and Al-Onaizan (2016), who investigate strategies to find the best of both worlds: models that adapt well to the new domain without deteriorating on the old domain. We compare how the presented neural bandit learning objectives perform in comparison to linear models, then discuss the handling of unknown words and eventually investigate the impact of techniques for variance reduction introduced in Section 2.2.3.

#### Setup

**Data.** We perform domain adaptation from Europarl (EP) to News Commentary (NC) for translations from French to English as for the linear models, and additionally from EP to TED talks (TED). Table 5 (page 62) provides details about the data sets. The data is pre-processed in the same way as for the SMT experiments (see Section 3.4.1). The challenge for the neural bandit learner is to adapt from the EP domain to NC or TED with weak feedback only.

| **Domain** | **Version** | **Train** | **Dev** | **Test** |
|---|---|---|---|---|
| Europarl (EP) | v.5 | 1.6M | 2k | 2k |
| News Commentary (NC) | WMT07 | 40k | 1k | 2k |
| TED | TED2013 | 153k | 2k | 2k |

Table 5: Number of parallel sentences for training, development and test sets for French-to-English domain adaptation.

**Model.** We choose a standard recurrent encoder-decoder architecture with single-layer GRUs with 800 hidden units, a word embedding size of 300 and tanh activations. The encoder consists of a bidirectional RNN, where the hidden states of backward and forward RNN are concatenated. The decoder uses the attention mechanism proposed by Bahdanau et al. (2015).[12] Source and target

[12]We do not use beam search nor ensembling, although we are aware that higher performance is almost guaranteed with these techniques. Our goal is to show relative differences between different models, so a simple setup is sufficient for the purpose of our experiments. At the

vocabularies contain the 30k most frequent words of the respective parts of the training corpus. We limit the maximum sentence length to 50. Dropout (Srivastava et al., 2014a) with a probability of 0.5 is applied to the network in several places: on the embedded inputs, before the output layer, and on the initial state of the decoder RNN. The gradient is clipped when its norms exceeds 1.0 to prevent exploding gradients and stabilize learning (Pascanu et al., 2013a). All models are implemented and trained with the sequence learning framework `Neural Monkey` (Libovický et al., 2016; Bojar et al., 2016b). They are trained with a minibatch size of 20, fitting onto single 8GB GPU machines. The training dataset is shuffled before each epoch.

**Baselines.** The out-of-domain baseline is trained on the EP training set with standard MLE. For both NC and TED domains, we train two full-information in-domain baselines: The first in-domain baseline is trained on the relatively small in-domain training data. The second in-domain baseline starts from the out-of-domain model and is further trained on the in-domain data. All baselines are trained with MLE and Adam (Kingma and Ba, 2015) ($\alpha = 1 \times 10^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$) until their performance stops increasing on respective held-out validation sets. The gap between the performance of the out-of-domain model and the in-domain models defines the range of possible improvements for bandit learning. All models are evaluated with Neural Monkey's `mteval`. For statistical significance tests we used Approximate Randomization testing (Noreen, 1989).

**Bandit learning hyperparameters.** Bandit learning starts with the parameters of the out-of-domain baseline. The bandit models are expected to improve over the out-of-domain baseline by receiving feedback from the new domain, but at most to reach the in-domain baseline since the feedback is weak. The models are trained with Adam on in-domain data for at most 20 epochs. Adam's step-size parameter $\alpha$ was tuned on the validation set and was found to perform best when set to $1 \times 10^{-5}$ for non-pairwise, $1 \times 10^{-6}$ for pairwise objectives on NC, $1 \times 10^{-7}$ for pairwise objectives on TED. The best model parameters, selected with early stopping on the in-domain validation set, are evaluated on the held-out in-domain test set. In the spirit of (Freitag and Al-Onaizan, 2016) they are additionally evaluated on the out-of-domain test set to investigate how much knowledge of the old domain the models lose while adapting to the new domain. Bandit learning experiments are repeated two times, with different random seeds, and mean BLEU scores with standard deviation are reported.

time of the experiments, neither beam search nor ensembling were part of the underlying NMT framework.

**Clipping for CE.** For CE we had to apply a threshold clipping (Ionides, 2008) to the model probabilities with $\tau = 1 \times 10^{-30}$ and introduce a constant $c = 1 \times 10^{30}$ to prevent numerical instabilities. However, in preliminary results we noticed that nearly all sample probabilities were clipped. This reduces it to the same update as for ER up to a scalar factor, which can be seen as part of the learning rate. We concluded that unless model probabilities are treated differently (e.g. normalized over a subset of samples as in MRT), the cross-entropy objective is not suited for neural models in the form that we presented it for log-linear models. As a result, we do not include it in the experiments.

**Feedback simulation.** Weak feedback is simulated from the target side of the parallel corpus, but references are never revealed to the learner. For the SMT models we used smoothed sBLEU for simulating the weak feedback for generated translations from the comparison with reference translations. Here, we use GLEU instead, which Wu et al. (2016) introduced for learning from sentence-level reward signals correlating well with corpus BLEU. This metric is similar to BLEU, but does not have a brevity penalty and considers the recall of matching $n$-grams. It is defined as the minimum of recall and precision over the total $n$-grams up to a certain $n$. For our experiments $r(\tilde{y}) = \text{GLEU}(\tilde{y}, y)$, where $\tilde{y}$ is a sample translation and $y$ is the reference translation.

**Unknown words.** One drawback of NMT models is their limitation to a fixed source- and target vocabulary. In a domain adaptation task, this limitation has a critical impact to the translation quality. The larger the distance between old and new domain, the more words in the new domain are unknown to the models trained on the old domain. We consider two strategies for this problem for our experiments: Unknown word (UNK) replacement (Jean et al., 2015b) and Byte-Pair Encoding (BPE) (Sennrich et al., 2016c).

**UNK replacement.** Jean et al. (2015b) and Luong et al. (2015b) replace generated UNK tokens with aligned source words or their lexical translations in a post-processing step. Freitag and Al-Onaizan (2016) and Hashimoto et al. (2016) demonstrated that this technique is beneficial for NMT domain adaptation. We use `fast_align` to generate lexical translations on the EP training data. When an UNK token is generated, we look up the attention weights and find the source token that receives most attention in this step. If possible, we replace the UNK token by its lexical translation. If it is not included in the lexical translations, it is replaced by the source token. The main benefit of this technique is that it handles unknown named entities well by passing them through from source to target. However, since it is a non-differentiable post-processing step, the NMT model cannot directly be trained for this behavior.

**BPE.** Sennrich et al. (2016c) introduce BPE for word segmentation to build translation models on sub-word units. Rare words are decomposed into subword units, while the most frequent words remain single vocabulary items. We train sub-word level NMT with BPE. We apply 29,800 merge operations to obtain a vocabulary of 29,908 sub-words (`subword_nmt v1`).[13] The procedure for training these models is exactly the same as for the word-based models. The advantage of this method is that the model is in principle able to generate any word composing it from sub-word units. However, training sequences become longer and candidate translations are sampled on a sub-word level, which introduces the risk of sampling nonsense words.

**Control variates.** We implement the average baseline control variate as defined in Equation (53), which results in keeping an running average over previous rewards. Intuitively, absolute GLEU feedback is turned into relative feedback that reflects the current state of the model (see Section 3.3.2). In addition, we implement the score function control variate (Ranganath et al., 2014) with a running estimate for $c^*$:

$$c_k^* = \frac{1}{k} \sum_{j=1}^{k} \frac{\mathrm{Cov}(s_j, \nabla \log p_\theta(y^{(j)} \mid x^{(j)}))}{\mathrm{Var}(s_j)}. \tag{61}$$

[13] `https://github.com/rsennrich/subword-nmt`

**Results**

| Model | Train | Iter. | EP | NC | Train | Iter. | EP | TED |
|---|---|---|---|---|---|---|---|---|
| MLE | EP | 12.3M | 31.44 | 26.98 | EP | 12.3M | 31.44 | 23.48 |
| +UNK | | | 31.82 | 28.00 | | | 31.82 | 24.59 |
| +BPE | | 12.0M | 31.81 | 27.20 | | 12.0M | 31.81 | 24.35 |
| MLE | NC | 978k | 13.67 | 22.32 | TED | 2.2M | 14.16 | 32.71 |
| +UNK | | | 13.83 | 22.56 | | | 15.15 | 33.16 |
| +BPE | | 1.0M | 14.09 | 23.01 | | 3.0M | 14.18 | 32.81 |
| MLE | EP→NC | 160k | 26.66 | 31.91 | EP→TED | 460k | 23.88 | 33.65 |
| +UNK | | | 27.19 | 33.19 | | | 24.64 | 35.57 |
| +BPE | | 160k | 27.14 | 33.31 | | 2.2M | 23.39 | 36.23 |

Table 6: Fully-supervised out-of-domain and in-domain NMT baselines results (BLEU) on in- and out-of-domain test sets. The EP model is only trained on EP, the NC model only on NC, the TED model only on TED data. The EP→NC is first trained on EP, then fine-tuned on NC. The EP→TED is first trained on EP, then fine-tuned on TED.

**Baselines.** The NMT out-of-domain baselines, reported in the top section of Table 6, perform comparable to the linear baselines on NC, but the in-domain EP→NC baselines (middle section of Table 6) outperform the linear baseline by more than 3 BLEU points. Continuing training of a pre-trained out-of-domain model on a small amount of in domain data is thus very effective, whilst the performance of the models solely trained on small in-domain data is highly dependent on the size of this training data set. The in-domain dataset for TED is four larger than the NC training set, so the in-domain baselines perform better. This effect was previously observed by Luong et al. (2015a) and Freitag and Al-Onaizan (2016).

**Bandit learning.** The NMT bandit models that optimize the ER objective yield generally a much higher improvement over the out-of-domain models than the corresponding linear models: As shown in Table 7 (page 67), we find improvements of between 2.33 and 2.89 BLEU points on the NC domain, and between 4.18 and 5.18 BLEU points on the TED domain. In contrast, the linear models with sparse features and hypergraph re-decoding achieved a maximum improvement of 0.82 BLEU points on NC. Optimization of the PR objective shows improvements of up to 1.79 BLEU points on NC (compared to 0.6 BLEU

|  | Training: NC |  |  |  | Training: TED |  |  |  |
|---|---|---|---|---|---|---|---|---|
| **Model** | **Iter.** | **EP** | **NC** | **Diff.** | **Iter.** | **EP** | **TED** | **Diff.** |
| ER | 317k | $30.36_{\pm 0.20}$ | $29.34_{\pm 0.29}$ | 2.36 | 976k | $29.34_{\pm 0.42}$ | $27.66_{\pm 0.03}$ | 4.18 |
| +UNK* | 317k | $30.73_{\pm 0.20}$ | $30.33_{\pm 0.42}$ | 2.33 | 976k | $29.68_{\pm 0.29}$ | $29.44_{\pm 0.06}$ | 4.85 |
| +UNK** | 349k | $30.67_{\pm 0.04}$ | $30.45_{\pm 0.27}$ | 2.45 | 1.1M | $29.62_{\pm 0.15}$ | $29.77_{\pm 0.15}$ | 5.18 |
| +UNK**+SF | 713k | $29.97_{\pm 0.09}$ | $30.61_{\pm 0.05}$ | 2.61 | 658k | $30.18_{\pm 0.15}$ | $29.12_{\pm 0.10}$ | 4.53 |
| +UNK**+BL | 531k | $30.19_{\pm 0.37}$ | $31.47_{\pm 0.09}$ | 3.47 | 644k | $29.91_{\pm 0.03}$ | $30.44_{\pm 0.13}$ | 5.85 |
| +BPE | 543k | $30.09_{\pm 0.31}$ | $30.09_{\pm 0.01}$ | 2.89 | 831k | $30.03_{\pm 0.43}$ | $28.54_{\pm 0.04}$ | 4.18 |
| +BPE+SF | 375k | $30.46_{\pm 0.10}$ | $30.20_{\pm 0.11}$ | 3.00 | 590k | $30.32_{\pm 0.26}$ | $28.51_{\pm 0.18}$ | 4.16 |
| +BPE+BL | 755k | $29.88_{\pm 0.07}$ | $31.28_{\pm 0.24}$ | **4.08** | 742k | $29.84_{\pm 0.61}$ | $30.24_{\pm 0.46}$ | **5.89** |
| PR$^{\text{BIN}}$+UNK** | 22k | $30.76_{\pm 0.03}$ | $29.40_{\pm 0.02}$ | 1.40 | 14k | $31.84_{\pm 0.01}$ | $24.85_{\pm 0.00}$ | 0.26 |
| +BPE | 14k | $31.02_{\pm 0.09}$ | $28.92_{\pm 0.03}$ | 1.72 | 69k | $31.77_{\pm 0.01}$ | $24.55_{\pm 0.01}$ | 0.20 |
| PR$^{\text{CONT}}$+UNK** | 12k | $30.81_{\pm 0.02}$ | $29.43_{\pm 0.02}$ | 1.43 | 9k | $31.85_{\pm 0.02}$ | $24.85_{\pm 0.01}$ | 0.26 |
| +BPE | 8k | $30.91_{\pm 0.01}$ | $28.99_{\pm 0.00}$ | 1.79 | 55k | $31.79_{\pm 0.02}$ | $24.59_{\pm 0.01}$ | 0.24 |

Table 7: Bandit NMT results (BLEU) on EP, NC and TED test sets for training on NC and TED bandit feedback. UNK* models involve UNK replacement only during testing, UNK** include UNK replacement already during training. For PR, either binary (BIN) or continuous feedback (CONT) was used. Control variates: average reward baseline (BL) and score function (SF). Results are averaged over two independent runs and standard deviation is given in subscripts. Improvements over out-of-domain models are given in the Diff.-columns.

points for linear models), but no significant improvement on TED. The biggest impact of relative feedback is the considerable speedup of training speed of 1 to 2 orders of magnitude compared to ER.

**Conservative domain adaptation.** A beneficial side-effect of NMT learning from weak feedback is that the knowledge from the out-domain training is not simply "overwritten". This happens to full-information in-domain tuning where more than 4 BLEU points are lost in an evaluation on the out-domain data. On the contrary, the bandit learning models still achieve high results on the original domain. This is useful for conservative domain adaptation, where the performance of the models in the old domain is still relevant.

**Unknown words.** Unknown word treatment leads to consistent improvements over word-based models for all baselines and bandit learning models (+UNK and +BPE rows in Table 7). We observe that the models with UNK replacement benefit from passing through source tokens, and only marginally from lexical translations. Bandit learning models take particular advantage of UNK replacement when it is already included during training (+UNK**). The sub-word (+BPE) models achieve the overall highest improvement over the baselines.

**Control variates.** Applying the score function (+SF) control variate to ER optimization does not largely change learning speed or BLEU results. However, the baseline control variate (+BL) leads to improvements of around 1 BLEU over the ER optimization without variance reduction on both domains. This empirically confirms the analytic results from Section 3.3.2 that the baseline has an additional beneficial effect thanks to relativized rewards. They also take longer until they reach the early stopping point, due to prolonged learning even with high quality rewards (see Section 3.3.2).

**Summary**

We showed how to build neural models for structured prediction under bandit feedback using recurrent neural networks with attention. We transferred the algorithms previously used with log-linear models to train NMT models under numerical feedback to single output structures or under preference rankings over pairs of structures. In our experimental evaluation on the task of neural machine translation domain adaptation, we found relative improvements of up to 5.89 BLEU points over out-of-domain seed models, outperforming also linear bandit models. Furthermore, included average reward and score function baselines as control variates for improved training speed and generalization.

## 3.5 WMT Shared Task Evaluation

At the First Conference of Machine Translation 2017 (WMT), we organized a shared task on bandit learning for machine translation in collaboration with Amazon. In contrast to the other shared tasks that create supervised learning problems with given training, development and test translations, this shared task posed a reinforcement learning problem: Given source sentences and rewards for proposed translations, how fast and how well can models adapt to a new domain? This problem is very similar to the above described simulation, but it was slightly more challenging in that 1) participants did not have access to development or test data, 2) online quality beyond online-to-batch held-out evaluation was also of interest, 3) the domain was Amazon product titles, i.e., less fluent than parliament discussions or news text, and 4) every training sample occurred only once. This created the ideal testbed to compare the adaptability of our statistical and neural models, and also to external participants. In this section we will discuss evaluation metrics and the comparison of participating systems; more details can be found in (Sokolov et al., 2017).

### 3.5.1 Task Setup

In the shared task, user feedback is simulated by a service hosted on Amazon Web Services. Participants submit translations and receive feedback on translation quality one by one, i.e. the next source sentence can only be received after the translation to the previous sentence was sent off. Furthermore, feedback for only one translation per source can be obtained, in a true online bandit learning fashion. The task is to use this feedback to adapt an out-of-domain MT model, pre-trained on mostly news texts, to a new domain (e-commerce), for the translation direction of German (de) to English (en). The reference translations are only used to compute the rewards, but are never revealed to the learner, neither at training nor at test time.

#### Data

For training initial MT systems, out-of-domain parallel data was restricted to German-to-English parts of EP v7, NC v12, CommonCrawl and Rapid data from the WMT 2017 constrained News Translation task[14]. Monolingual English data from the constrained shared task was allowed as well. Tuning of the out-of-domain systems had to be done on the `newstest2016-deen` set. The in-domain parallel data for online bandit learning was taken from the e-commerce domain: The corpus was provided by Amazon and had been sampled from a large real-world collection of post-edited translations of actual product descriptions. The post-edits were not always literal, sometimes adding or deleting a considerable number of tokens and resulting in low feedback BLEU scores for submitted literal translations.

#### Evaluation Metrics

In the case of MT personalization both online quality, i.e., during the learning and exploration phase, as well as offline quality, i.e., after adaptation with exploitation only, are of interest. In order to capture both, we evaluate models with the following metrics:

1. **Online cumulative reward**: This metric measures the cumulative sum $C = \sum_{k=1}^{K} r(y^{(k)})$ of the per-sentence BLEU score $r(y)$ for translation $y$ against the number of iterations. This metric has been used in reinforcement learning competitions (Dimitrakakis et al., 2014). For systems with the same design, this metric favors those that do a good job at balancing exploration and exploitation to achieve high scores over the full data sequence. Unlike in these competitions, where environments (i.e., action spaces and context features) were fixed, in our task the environment is heterogeneous due

[14] `statmt.org/wmt17/translation-task.html`

to the use of different underlying MT architectures. Thus, systems that start out with a well-performing pretrained out-of-domain model have an advantage over systems that might improve more over worse starting points. Even systems that do not perform online learning at all can achieve high cumulative rewards.

2. **Online regret:** In order to overcome the problems of the cumulative reward metric, we can use the regret metric from bandit learning $R = \frac{1}{K}\sum_{k=1}^{K}(r(y_*^{(k)}) - r(y^{(k)}))$ (see Section 2.2). Systems are penalized when predicting translation $y$ instead of the optimal translation $y_*$. Plotting a running average of regret against the number of iterations allows separating the gains due to the MT architecture from the gains due to the learning algorithm: Systems that learn will decrease regret, systems that do not learn will not. In our task, we use as oracle system a model that is fine-tuned on in-domain references.

3. **Relative held-out reward**: A further way to separate out the learning ability of systems from the contribution of the underlying MT architecture is to apply the standard corpus-BLEU score and/or an average of the per-sentence BLEU score $r$ on a held-out set at regular intervals during training. Plotting these scores against the number of iterations, or alternatively, subtracting the performance of the starting point at each evaluation, allows to discern systems that adapt to a new domain from systems that are good from the beginning and can achieve high cumulative rewards without learning. We performed this evaluation by embedding a small fixed held-out set in the beginning (showing the performance of the initial out-of-domain model), and again at regular intervals including the very end of the learning sequence.

### 3.5.2 Systems

**Baselines.** As baseline systems, we used static SMT and NMT models that were trained on out-of-domain data, but did not perform online learning on in-domain data. Oracle systems that were trained in batch on in-domain data were used to compute regret.

- **SMT-static:** A SMT system was built with `cdec` (Dyer et al., 2010), similar to the model in Section 3.4.1 but limited to dense features.

- **WMT16-static:** The Nematus (Sennrich et al., 2017) NMT system that achieved state-of-the-art news translation quality in 2016 (Bojar et al., 2016c) was trained on more data than allowed for the task (Sennrich et al., 2016a), but was nevertheless employed to represent the best one could possibly do on general domain data.

- **BNMT-static:** We train a byte-pair-segmented Neural Monkey (Helcl and Libovický, 2017) system on the shared task parallel data using a BPE vocabulary from 30k merge operations on all tokens and all single characters of the training data, including the UNK token. If unknown characters occur, they are copied from source to target with the UNK replacement strategy of Jean et al. (2015b) (see Section 3.4.2).

- **Oracles:** To simulate full-information systems (oracles) for regret calculation, we train an SMT and an NMT system on the in-domain data that other learning systems accessed only through the numerical feedback. The SMT oracle system was trained on combined in-domain and out-of-domain data, while the NMT oracle system continued training from the converged out-of-domain system on the in-domain data with the same BPE vocabulary.

**Participating Systems.** We summarize the submissions grouped by model and training objective. For detailed hyperparameter descriptions we refer the reader to (Sokolov et al., 2017).

- **Online bandit first-order SMT**: Online bandit learners based on SMT were following the previously presented ER algorithm to adapting an SMT model from weak user feedback (see Section 3.4.1). Average reward baselines were used as control variates (CV) for all SMT online learners.

- **Online bandit zeroth-order SMT**: We implement stochastic zeroth-order (SZO) optimization for online bandit learning (Flaxman et al., 2005). In a nutshell, on each step of the SZO algorithm, the model parameters $w$ are perturbed with an additive standard Gaussian noise $\epsilon$, and the Viterbi translation is sent to the service.

- **Online bandit first-order NMT**: Online bandit learners based on SMT were similar to the models presented in Section 3.4.2 and trained with the ER algorithm. However, since the models are now evaluated online, we would like the model to gradually stop exploring, in order to still achieve high cumulative per-sentence reward and not only final held-out test performance in an exploitation-only evaluation. To achieve such a behavior, the temperature of the softmax over the outputs of the last layer of the network is annealed (Rose, 1998). The annealing schedule for this temperature $T$ is defined as $T_k = 0.99^{\max(k-k_{\text{START}},0)}$, i.e. decreases from iteration $k_{\text{START}}$ on. The same decay is applied to the learning rate, such that $\gamma_k = \gamma_{k-1} \cdot T_k$. As in the experiments before, we compare word-based NMT (WNMT) with a lexical translation model for unknown words with BPE-based models (BNMT).

- **UMD NMT domain adaptation**: The UMD team's systems were based on an neural recurrent model with subwords. They follow a domain adaptation approach of Axelrod et al. (2011) to select training data after receiving in-domain source-side data and selecting the most similar out-of-domain data from the WMT 2016 training set for re-training.

- **UMD NMT reinforce**: Another type of models submitted by UMD uses reinforcement learning techniques to learn from feedback and improve the update of the translation model to optimize the reward, based on Bahdanau et al. (2017) and Ranzato et al. (2016). Details are reported in (Sharaf et al., 2017).

- **LIMSI SMT UCB1**: The team from LIMSI tried to adapt a seed `Moses` system trained on out-domain data to a new, unknown domain relying on two components, each of which addresses one of the challenges raised by the shared task: i) estimate the parameters of a MT system without knowing the reference translation and in a 'one-shot' way (each source sentence can only be translated once); ii) discover the specificities of the target domain 'on-the-fly' as no information about it is available. Details are provided in (Wisniewski, 2017).

### 3.5.3 Results

**Cumulative Reward.** Table 8 (page 73) shows the evaluation results under the cumulative rewards metric. The best performance (of the non-oracle systems) is obtained by the UMD-domain adaptation system which is pre-adapted to the domain and static during training. It is followed closely by the online bandit learner BNMT-ER which optimizes the ER objective online. It outperforms the BNMT-static baseline and the SMT models.

**Online Regret.** The evolution of the online regret plotted against the log-scaled number of iterations during training is shown in Figure 4 (page 75). Most of the learning happens during the first 100,000 iterations, however, online learning systems optimizing structured ER objectives or based on reinforcement learning eventually converge to the same result: BNMT-ER or UMD-reinforce2 get close to the regret of the static UMD-domain adaptation.

**Relative Held-out Reward.** Figures 5, 6a and 6b (page 75 and 76) show the evolution of corpus- and sentence-BLEU on the held-out set that has been embedded in the development and the training sequences.[15] While under corpus-BLEU, static systems always outperform online learners on the held-out embedded set,

[15]In the plots, algorithms and models are named "EL" instead of "ER".

| | Model | Cumulative Reward |
|---|---|---|
| | 'translate' by copying source | 64,481.8 |
| SMT | SMT-oracle | 499,578.0 |
| | SMT-static | 229,621.7 |
| | SMT-ER-CV-ADADELTA | 214,398.8 |
| | SMT-ER-CV-ADAM | 225,535.3 |
| | SMT-SZO-CV-ADAM | 208,464.7 |
| NMT | BNMT-oracle | 780,580.4 |
| | BNMT-static | 222,066.0 |
| | WMT16-static | 139,668.1 |
| | BNMT-ER-CV | 212,703.2 |
| | BNMT-ER | 237,663.0 |
| | WNMT-ER | 115,098.0 |
| | UMD-domain-adaptation | 248,333.2 |

Table 8: Cumulative rewards over the full training sequence. Only completely finished submission are shown.

online learning systems such as BNMT-ER can catch up under corpus-BLEU during development, and under a sentence-BLEU evaluation during training. The curves for corpus- and average sentence-BLEU (Figures 6a and 6b on page 76) show different dynamics, with the corpus-BLEU sometimes decreasing whereas the sentence-BLEU curve continues to increase. This is due to the mismatch between training rewards and evaluation rewards in the case of corpus-BLEU.

**Summary**

In the novel shared task on bandit learning for machine translation we compared the adaptability and learning speed of a set of static, statistical and neural translation models on domain adaptation from the news domain to e-commerce data. Despite challenges like noisy rewards and an unusual domain, we found promising results for both linear and non-linear online learners that could outperform their static SMT and NMT baselines. This validates the empirical successes for the ER-based neural learners that we reported on simulations (Section 3.4.2) on a more realistic task.

## 3.6 Conclusion

In this chapter we started from the definition of online bandit structured prediction and the adaptation of expected loss/reward objectives from supervised to bandit learning ("banditization"). We presented three algorithms and their instantiation for both statistical and neural machine translation models, discussed their strengths and weaknesses and applied them to a domain adaptation task, where the adaptation, or personalization, is guided by a weak reward signal. We introduced and analyzed the effects of control variates and found that they improve variance and generalization. Both statistical and neural models yielded significant improvements on a set of evaluation tasks for English-to-French translations and also performed well in an online evaluation on the challenging product title domain in a shared task with translations from German to English.

From the perspective of real-world interactive applications, bandit pairwise preference learning is perhaps the most attractive algorithm of the three since it only requires comparative judgments for learning. In our experiments we found that relative feedback resulted in improved empirical convergence speed for SMT, but for NMT the decomposition of the structure into a sequence of tokens prohibits a simple transfer of the pairwise objective. Chapter 5 will further investigate the potential of pairwise preference learning for NMT and the suitability for human ratings.

Despite the success of the experiments also revealed that these reinforcement algorithms with exponential action spaces need large amounts of training data to adapt to the reward signal, and careful tuning of learning and annealing rates. The shared task evaluation highlighted the difficulties of balancing online quality and offline evaluations: exploration during learning is a potential loss of online quality (a potential loss of customers if deployed in production), while reaching best offline (held-out) quality requires it. In the next two chapters, we will therefore move to offline learning, where there is no model-based exploration in interaction with a user and feedback collection and model training are completely separated.

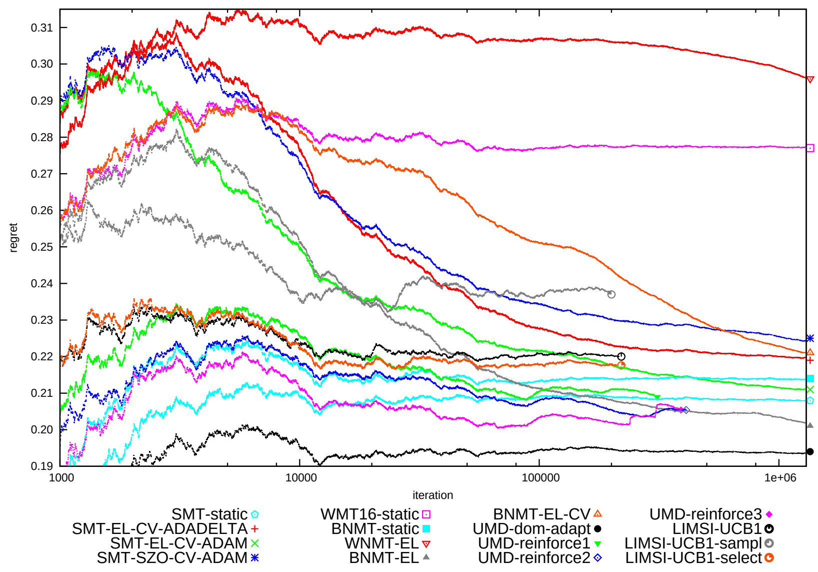


Figure 4: Evolution of regret plotted against log-scaled number of iterations during training. The steeper is the decrease of a curve, the better learning capability has the corresponding algorithm. Algorithms labeled "EL" correspond to our "ER". Plot made by Artem Sokolov.

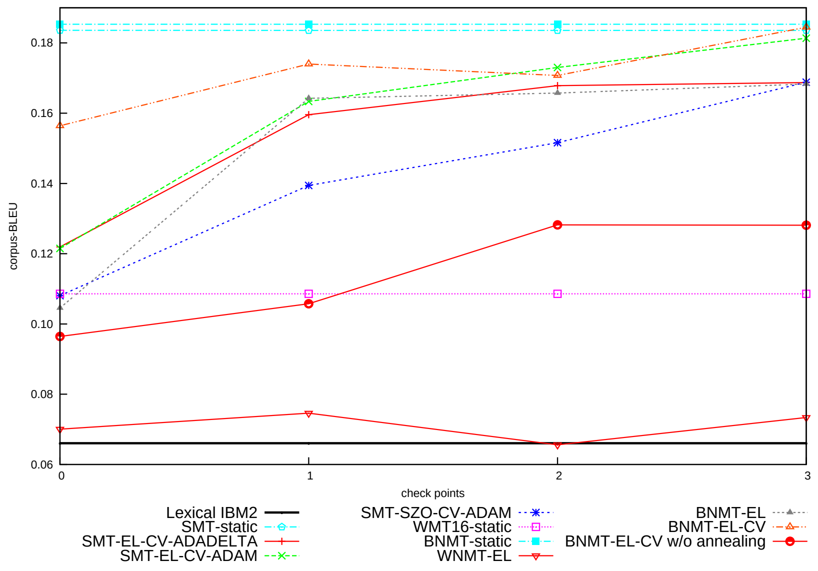
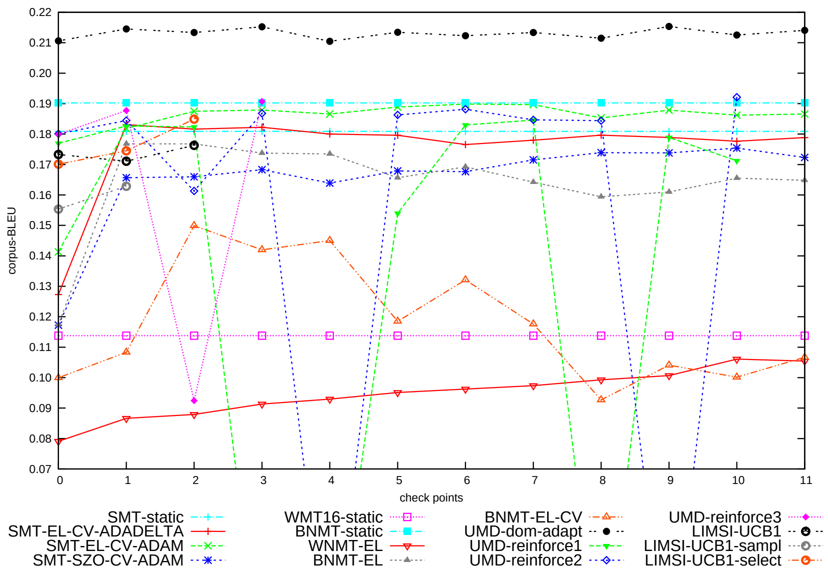
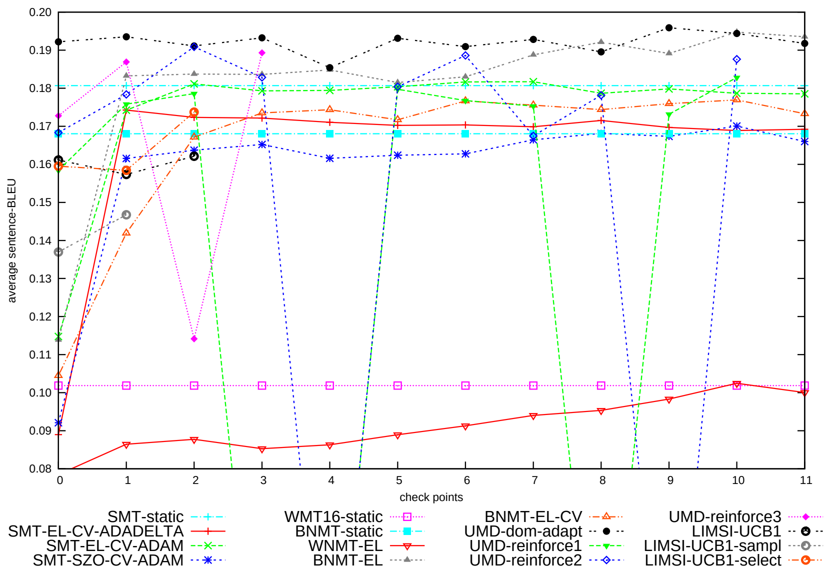


Figure 5: Evolution of corpus BLEU scores during development for configuration selected for the training phase of the competition. Each check point is comprised of the same 700 sentences spaced at a regular intervals of 12,400 sentences starting from the beginning of the development sequence. Algorithms labeled "EL" correspond to our "ER". Plot made by Artem Sokolov.



(a) corpus-BLEU



(b) sentence-BLEU

Figure 6: The evolution of corpus- and sentence-BLEU scores during training for all participant and baselines. Each check point is comprised of the same 4,000 sentences spaced at a regular intervals of 113,634 sentences starting from the beginning of the training sequence. Algorithms labeled "EL" correspond to our "ER". Plots made by Artem Sokolov.

Part II

# Offline Learning with Human Bandit Feedback

# Chapter 4

# Learning from E-commerce User Feedback

In commercial scenarios of neural machine translation (NMT), the one-best translation of a text is shown to multiple users who can reinforce high-quality (or penalize low-quality) translations by explicit feedback (e.g., on a Likert scale) or implicit feedback (e.e., by clicking on a translated page). In such settings this type of feedback can be easily collected in large amounts. Likert-scale ratings are for example implemented by eBay or Facebook (see Figure 1 on page 9). While bandit feedback in the form of user clicks on displayed ads is the standard learning signal for response prediction in online advertising (Bottou et al., 2013), bandit learning for machine translation has so far been restricted to simulation experiments (Sokolov et al., 2016b; Lawrence et al., 2017b; Nguyen et al., 2017; Kreutzer et al., 2017; Bahdanau et al., 2017), as discussed in the previous chapter.

The goal of this chapter is to show that cheap and abundant real-world human bandit feedback can be exploited for machine learning in NMT. We analyze and utilize human reinforcements that have been collected from users of the eBay e-commerce platform. We find that explicit user judgments in form of five-star ratings are not reliable and do not lead to downstream BLEU improvements in this scenario. In contrast, we find that implicit task-based feedback that has been gathered in a cross-lingual search task can be used successfully to improve task-specific metrics and BLEU.

Another crucial difference of our work to previous research is the fact that we assume a counterfactual learning scenario where human feedback has been given to a historic system different from the target system. Learning is done offline from logged data, which is desirable in commercial settings where system updates need to be tested before deployment and the risk of showing inferior translations to users needs to be avoided. Our offline learning algorithms range from a simple bandit-to-supervised conversion (i.e., using translations with good feedback for supervised tuning) to transferring the counterfactual learning techniques presented by Lawrence et al. (2017b) from SMT to NMT models.

A naive bandit-to-supervised conversion proved to be hard to beat, despite theoretical indications of poor generalization for exploration-free learning from logged data (Langford et al., 2008; Strehl et al., 2010). However, we can improve over this with a task-specific reward scoring function, resulting in significant improvements in both BLEU and in task-specific metrics.

**Contributions.** The contributions of this chapter are the following

1. We present the first study on learning from real-user bandit feedback for machine translation.
2. The feedback is collected in production, for which we transfer previous online learning approaches to an offline learning scenario.
3. We adapt two objectives for SMT to offline bandit learning for NMT from explicit user feedback and evaluate them on simulated and real data.
4. We develop a novel technique to elicit implicit feedback from user queries and clicks in a cross-lingual information retrieval pipeline.
5. We also discover the simple but strong bandit-to-supervised conversion that should serve as baseline in any future work concerning learning from bandit feedback.

**Publications.** This work is the result of a collaboration with eBay and was published in (Kreutzer et al., 2018a). The author of this thesis conducted the experiments with the collected feedback and in simulation, developed the objectives and analyzed the models and feedback, while eBay provided the feedback and in-domain data, the baseline models, and the expert annotations.

**Outline.** We describe how to collect explicit and implicit user feedback for the eBay platform in Section 4.1. Section 4.2 introduces algorithms for learning from this feedback. In the experiments reported in Section 4.3 we evaluate how well an NMT system can be improved with the help of the feedback collection. Section 4.4 concludes this chapter with a summary of the findings.

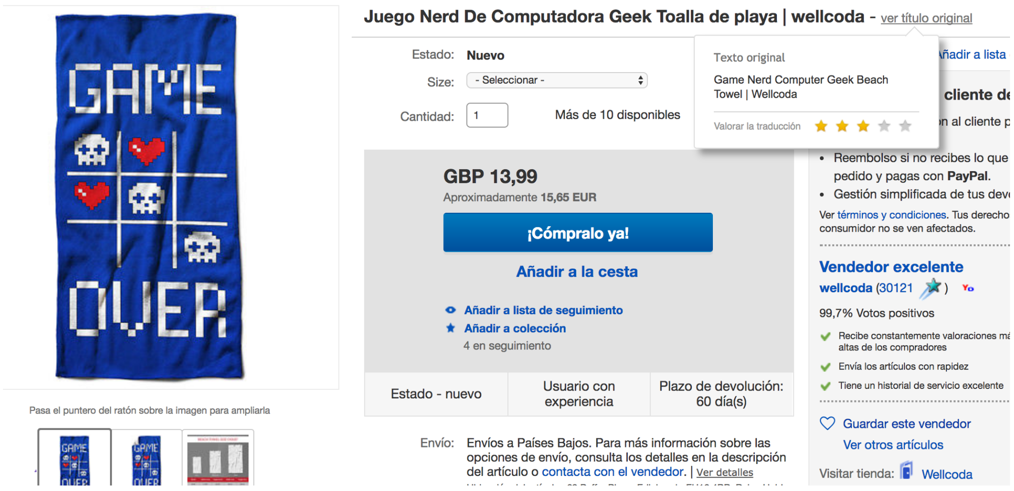
## 4.1 User Feedback

### 4.1.1 Explicit Feedback via Star Ratings

#### Feedback Collection

eBay used to collect user feedback for their translations through a 5-star rating interface, pictured in Figure 7 (page 80).[1] When users visited product pages with translated titles, they could inspect the source by hovering with the mouse over the title. Five stars are shown with the instruction to 'rate this translation' as for example shown for the product in Figure 7. A log is built from the collected star ratings together with the original title and the machine translation.

[1]At the time of the study (Spring 2018); now this feature is disabled.


Juego Nerd De Computadora Geek Toalla de playa | wellcoda - ver título original

Estado: Nuevo

Size: - Seleccionar -

Cantidad: 1 Más de 10 disponibles

Texto original

Game Nerd Computer Geek Beach Towel | Wellcoda

Valorar la traducción ★★★☆☆

GBP 13,99

Aproximadamente 15,65 EUR

¡Cómpralo ya!

Añadir a la cesta

Añadir a lista de seguimiento

Añadir a colección

4 en seguimiento

Estado - nuevo | Usuario con experiencia | Plazo de devolución: 60 día(s)

Envío: Envíos a Países Bajos. Para más información sobre las opciones de envío, consulta los detalles en la descripción del artículo o contacta con el vendedor. | Ver detalles

Pasa el puntero del ratón sobre la imagen para ampliarla

• Reembolso si no recibes lo que pedido y pagas con PayPal.

• Gestión simplificada de tus dev

Ver términos y condiciones. Tus derecho consumidor no se ven afectados.

Vendedor excelente

wellcoda (30121 )

99,7% Votos positivos

✓ Recibe constantemente valoraciones má altas de los compradores

✓ Envía los artículos con rapidez

✓ Tiene un historial de servicio excelente

Guardar este vendedor

Ver otros artículos

Visitar tienda: Wellcoda

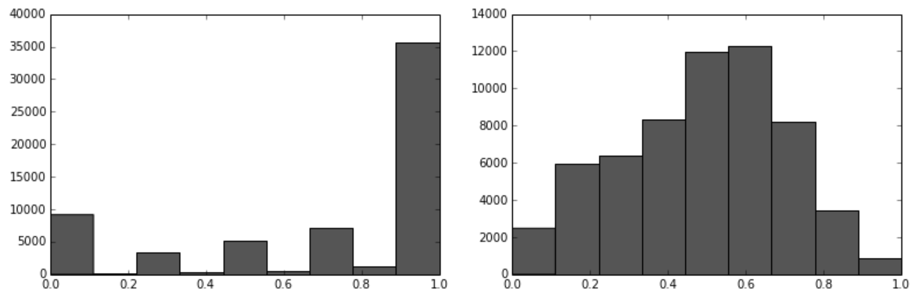
Figure 7: Screenshot of the 5-star rating interface embedded on a product page on `www.ebay.es` for a product translated from English to Spanish.

For the experiments in this paper, we focus on translations from English to Spanish. The user star rating data set contains 69,412 rated product titles with 148k individual ratings. Since 34% of the titles were rated more than once, the ratings for each title are averaged. Figure 8a (page 81) depicts a histogram of the averaged ratings, linearly scaled to $[0, 1]$. We observe a tendency towards high ratings, with half of the titles rated with five stars.

### Reliability and Validity of Star Ratings

The user ratings were available only in aggregated form, so that we could not filter or normalize by rater (see Section 5.2.1). To investigate the reliability and validity of these ratings, we employed three bilingual annotators ('experts') to independently re-evaluate and give five-star ratings for a balanced subset of 1000 product title translations. The annotators were presented the source title and the machine translation, together with instructions to rate the translation with stars. The inter-annotator agreement between experts is relatively low with Fleiss' $\kappa = 0.12$ (Fleiss, 1971). Furthermore, there is no correlation of the averaged 'expert' ratings and the averaged user star ratings (Spearman's $\rho = -0.05$). This dis-agreement might be caused by the difficulty of the domain in general, and the ill-definedness of the translation tasks when the source is not a well-formed sentence.

However, when we ask another three annotators to indicate whether they agree or disagree with a balanced subset of 2,000 user ratings, the majority agrees with 42.3% of the ratings. In this binary meta-judgment task, the inter-



(a) Histogram of user star ratings. (b) Histogram of sBLEU scores.

Figure 8: Histograms of star ratings and simulated sBLEU ratings on the training data for the reward estimator. The original ratings on a five-star scale are averaged per title and linearly scaled to $[0, 1]$. The sBLEU scores are obtained for translations of the out-of-domain baseline for simulation experiments.

annotator agreement between experts is moderate with $\kappa = 0.45$. We observe a strong tendency of the expert annotators to agree with high user ratings and to disagree with low user ratings. Two examples of user ratings, expert ratings and expert judgment are given in Table 9 (page 82). In the first example, all raters agree that the translation is good, but in the second example, there is a strong disagreement between users and experts.

This analysis shows that it is generally not easy for non-professional users of the e-commerce platform, and even for expert annotators, to give star ratings of translations in the domain of user-generated product titles with high reliability. User ratings of product titles translations have a low validity, i.e., we do not know whether their response actually expresses translation quality. We cannot control the influence of other factors on their judgment, e.g., the displayed image, the product itself, or the users' general satisfaction with the e-commerce transaction or their source language understanding. The user judgment might also be given with an adversarial purpose. Furthermore, we do not have control over the quality of sources,[2] nor can we discern to which degree a user rating reflects fluency or adequacy of the translation.

### 4.1.2 Task-Based Implicit Feedback

#### Feedback Collection

Another form to collect human reinforcement signals via the used e-commerce platform is to embed the feedback collection into a cross-lingual information

[2]Most titles consist of a sequence of keywords rather than a fluent sentence. See (Calixto et al., 2017) for a fluency analysis of product titles.

| | |
|---|---|
| **Src** | Universal 4in1 Dual USB Car Charger Adapter Voltage DC 5V 3.1A Tester For iPhone |
| **Hyp** | Coche Cargador Adaptador De Voltaje Probador De Corriente Continua 5V 3.1A para iPhone |
| **Rating** | *Users*: 4.56; *Experts*: 4.33 / Correct |
| **Src** | BEAN BUSH THREE COLOURS: YELLOW BERGGOLD, PURPLE KING AND GREEN TOP CROP |
| **Hyp** | Bean Bush tres colores: Amarillo Berggold, púrpura y verde Top Crop King |
| **Rating** | *Users*: 1.0; *Experts:* 4.66 / Incorrect |

Table 9: Two examples for five-star user and expert ratings of the machine translation (hyp) for the English sources (src), and expert judgments on the user ratings ("Is the user rating correct or incorrect?"). Expert and user ratings are averaged for all raters. The expert judgment is obtained by majority voting.

retrieval task. The product title translation system is part of the following pipeline in the search interaction of a user with the e-commerce system: When the user enters a query in Spanish, it is first translated to English, then a search engine retrieves a list of matching products, and their titles are translated to Spanish and displayed to the user. As soon as the user clicks on one of the translated titles, the original query, the translated query, the source product title and its translation are stored in a log.

From this collection we filter the cases where (a) the original query and the translated query are the same, or (b) more than 90% of the words from the query translation are not contained in the retrieved source title. In this way, we attempt to reduce the propagation of errors in query translation and search. This leaves us with a data set of 164,065 tuples of Spanish queries, English product titles and their Spanish translations (15% of the original collection). Note that this data set is more than twice the size of explicit feedback data set. An example is given in Table 10 (page 83). This approach has similarities to back-translation (Sennrich et al., 2016b) as exploited in dual learning (He et al., 2016) where translations get reinforced to the extent that they reconstruct the original source.

### Inducing Rewards

The advantage of embedding feedback collection into a search task is that we can assume that users who formulate a search query have a genuine intent of finding products that fit their need, and are also likely to be satisfied with product title translations that match their query, i.e., contain terms from query in the user's

| | |
|---|---|
| **Query (es)** | candado bicicleta |
| **Query (en)** | bicycle lock |
| **Title (en)** | New Bicycle Vibration Code Moped Lock Bike Cycling Security Alarm Sound Lock |
| **Title (es)** | Nuevo código de vibración Bicicleta Ciclomotor alarma de seguridad de bloqueo Bicicleta Ciclismo Cerradura De Sonido |
| **Recall** | 0.5 |

Table 10: Example of query and product title translation. 'candado' is translated to 'lock' in the query, but then translated back to 'cerradura' in the title. The recall metric would prefer a title translation with 'candado', as it was specified by the user.

own language. We exploit this assumption to measure the quality of a product title translation by requiring a user to click on the translation when it is displayed as a result of the search, and quantifying the quality of the clicked translation by the extent it matches the query that led the user to the product. For this purpose, we define a word-based, recall-focused matching function match$(w, q)$ that evaluates whether a query $q = (q_1, \dots, q_Q)$ contains the word $w$ of the translation:

$$\text{match}(w, q) = \begin{cases} 1, & \text{if}\, w \in q \\ 0, & \text{otherwise.} \end{cases} \tag{62}$$

A sequence-level reward for a sentence $y$ of length $T$ and a query of length $Q$ is ratio between matching words and query length:

$$\text{recall}(\mathbf{y}, q) = \frac{1}{Q} \sum_{t=1}^{T} \text{match}(y_t, q). \tag{63}$$

## 4.2 Learning from User Feedback

### 4.2.1 On- vs. Off-Policy Learning

While the simulation experiments in the previous chapter were based on online availability of feedback, where the learner could query the teacher/human at any time during the learning process, this is not an attractive solution for production environments. As discussed in the Introduction, the feasible solution for MT in production is to separate the feedback collection step from the model update step. Feedback for translations of the deployed system is collected over some period of time, and copies or variants of the deployed model (or other models

**Algorithm 5** Bandit Structured Prediction with Offline Feedback

```
Input: Logging system with parameters $\psi$, learning system with parameters $\theta$
1: Initialize log $L = \{\}$
2: for $k = 0, \ldots, K$ do
3:     Observe input structure $x$
4:     Predict output structure $\hat{y} \approx \arg\max p_{\psi}(y \mid x)$
5:     Obtain reward $r(\hat{y})$
6:     Store interaction $L = L \cup (x, \hat{y}, r(\hat{y}))$
7: Learn parameters $\theta$ from $L$
```

that have the potential to get deployed) can be trained and tuned on the feedback log in the meantime, until they have reached a performance that exceeds the currently deployed model, at which point they can replace it. This ensures that the deployed model is the best available model at all times and is not harmed by integrating the feedback. But it also results in a mismatch between logging and learning model, which requires the shift from on-policy learning (in simulations) to off-policy (or *counterfactual*) learning (Bottou et al., 2013). It comes with the additional challenge of maintaining means of exploration, which are essential for bandit and reinforcement as we showed in Section 3.3.2.

Algorithm 5 describes the basic procedure in counterfactual bandit structured prediction. In contrast to ER (Algorithm 1) presented in Chapter 3, sampling is replaced with deterministic logging (Line 4), the feedback is stored (Line 6), and there is the distinction between logging and learning model ($\psi$ vs $\theta$). If the logging process was stochastic, one could use the sampling probability under the historic model for inverse propensity scoring (Bottou et al., 2013). In the deterministic case, however, no exploration is allowed during sampling, so alternative solutions for implicit exploration (Lawrence et al., 2017b) have to be found, such as rating online-sampled translations with reward estimatorsin lieu of user feedback.

### 4.2.2 Reward Functions

In reinforcement and bandit learning, rewards received from the environment are used as supervision signals for learning. In our experiments, we investigate several options (Table 11 on page 85) to obtain a reward function $r : \mathcal{Y} \rightarrow [0, 1]$ from logged human bandit feedback:

1. **Direct user reward**: Explicit feedback, e.g., in the form of star ratings, can directly be used as a reward signal by treating the reward function as a black box. Since human feedback is usually only available for one translation per input, learning from direct user rewards requires the use of bandit learning algorithms. In our setup, human bandit feedback has been

| | **Function** | **Availability** | **Source** |
|---|---|---|---|
| Direct user reward | black box | one per input | human |
| Reward scoring function | glass box | unlimited | inferred |
| Estimated reward | glass box | unlimited | model |

Table 11: Comparison of reward functions used in the experiments.

collected for translations of a logging MT system different from the target system to be optimized. This restricts the learning setup to offline learning from logged bandit feedback. Evaluation at test time using the same reward function is infeasible since it would require interactive feedback from the same users under the same circumstances.

2. **Reward scoring function**: A possibility to use human bandit feedback to obtain rewards for more than a single translation per input is to score translations against a logged reference or a logged query, i.e., inferred from stronger supervision. The first option requires a bandit-to-supervised conversion of data where high-quality logged translations are used as references for BLEU or other MT quality metrics (see Section 3.4). The second option uses logged queries to obtain a matching score as in Equation 63. Reward scoring functions can be evaluated for all translations of a given input, thus they can be superior to bandit learning which is bound to a single feedback signal per input. Also, evaluation with respect to reward scores is feasible.

3. **Estimated reward**: Another option to extend bandit feedback to all translations is to learn a parametric model of rewards, e.g., by optimizing a regression objective. The reward function is known, but the model parameters need to be trained based on a history of direct user rewards or by evaluations of a reward scoring function. Estimated rewards share the advantages of reward scoring functions in training and evaluation, but bring the disadvantage of possibly being inaccurate. Most importantly, they will allow the model to learn from implicit exploration in exploration-free logging scenarios.

### 4.2.3 Training Objectives

**Maximum likelihood estimation.** Most commonly, NMT models are trained with MLE (Equation 15) on a given parallel corpus of source and target sequences. It requires reference translations and is agnostic to rewards (see Section 2.1.3). However, in a bandit-to-supervised conversion, translations can be filtered by their rewards to create a corpus of pseudo-references for MLE training. We apply this scenario to simulated bandit feedback and to explicit and implicit human feedback data in our experiments.

**Expected reward maximization.** When rewards can be retrieved for sampled translations during learning, the online bandit structured prediction framework from Chapter 3 can be applied for NMT. The ER objective (Equation 30) maximizes the expectation of a reward over all source and target sequences, and does in principle not require references. Just like in Chapter 3, we optimize ER using smoothed sBLEU in our simulation experiments, comparing a sampled translation $\tilde{y} \sim p_\theta(y \mid x)$ to a given reference translation.

**Sequence-level minimum risk training.** When rewards can be obtained for several translations per input instead of only for one as in the bandit setup, e.g. when the reward function is known (e.g. sBLEU), *Minimum Risk Training* (MRT) (Shen et al., 2016) can be applied to optimize NMT for reward maximization on a given parallel corpus of source and target sequences $D = \{(x^{(s)}, y^{(s)})\}_{s=1}^{S}$:

$$J^{\mathrm{MRT}}(\theta) = \sum_{s=1}^{S} \sum_{\tilde{y} \in \mathcal{S}(x^{(s)})} q_\theta^\alpha(\tilde{y} \mid x^{(s)})\, r_y(\tilde{y}), \tag{64}$$

where sample probabilities are re-normalized over a subset of translation samples $\mathcal{S}(x) \subset \mathcal{Y}(x)$:

$$q_\theta^\alpha(\tilde{y} \mid x) = \frac{p_\theta(\tilde{y} \mid x)^\alpha}{\sum_{y' \in \mathcal{S}(x)} p_\theta(y' \mid x)^\alpha}. \tag{65}$$

The hyper-parameter $\alpha$ controls the sharpness of $q$ (Shen et al., 2016). The re-normalization can be seen as a multiplicative control variate (Section 2.2.3).

**Token-level minimum risk training.** With sequence-level rewards, such as sBLEU, all words of a translation are reinforced to the same extent and are treated as if they contributed equally to the translation quality (uniform credit assignment). A word- or token-based reward function, such as the match with a given query (Equation 62), allows tokens to receive individual weights. The following modification of the sequence-level MRT objective (Equation 64) accounts for token-level rewards:

$$J^{\text{T-MRT}}(\theta) = \sum_{s=1}^{S} \sum_{\tilde{y} \in \mathcal{S}(x^{(s)})} \prod_{t=1}^{T} \left[ q_\theta^\alpha(\tilde{y}_t \mid x^{(s)}, \tilde{y}_{<t})\, r_y(\tilde{y}_t) \right], \tag{66}$$

where the token-level reward is for example the matching score (Equation 62): $r_y(\tilde{y}_t) = \mathrm{match}(\tilde{y}_t, y)$.

**Combining MLE and MRT.** T-MRT and ER typically require a warm start for large output spaces with sparse reward functions, for example by pre-training with MLE (see Section 2.3). After pre-training, fine-tuning is furthermore stabilized through a linear combination of MLE and T-MRT objective, which we will call “MIX” (Wu et al., 2016):

$$J^{\text{T-MIX}}(\theta) = \lambda \cdot J^{\text{MLE}}(\theta) + J^{\text{T-MRT}}(\theta), \tag{67}$$

or analogously on the sequence-level

$$J^{\text{MIX}}(\theta) = \lambda \cdot J^{\text{MLE}}(\theta) + J^{\text{MRT}}(\theta). \tag{68}$$

In our experiments, these MIX objectives are applied to simulated as well as to explicit and implicit human feedback data.

**Deterministic propensity matching.** Counterfactual learning attempts to improve a target MT system from a log of source sentences, translations produced by a historic MT system, and obtained feedback $L = \{(x^{(h)}, y^{(h)}, r(y^{(h)}))\}_{h=1}^{H}$ (see Section 2.3). Lawrence et al. (2017b) introduced the *Deterministic Propensity Matching* (DPM) objective

$$J^{\text{DPM}}(\theta) = \frac{1}{H} \sum_{h=1}^{H} r(y^{(h)})\, \bar{p}_\theta(y^{(h)} \mid x^{(h)}), \tag{69}$$

where translation probabilities are reweighted over the whole log. This reweighting can be seen as a multiplicative control variate as shown by (Swaminathan and Joachims, 2015b): By dividing the original estimator $\frac{1}{H}\sum_{h=1}^{H} r(y^{(h)})\, p_\theta(y^{(h)} \mid x^{(h)})$ by $\frac{1}{H}\sum_{h=1}^{H} p_\theta(y^{(h)} \mid x^{(h)})$, the variance of the estimator is reduced if they covary (see Section 2.2.3).

For the neural models the reweighting over the whole log is infeasible because it would requires re-decoding of the whole log for every training update. We re-normalize over the current mini-batch $B \subset H, B \ll H$ instead:

$$\bar{p}_\theta(y^{(h)} \mid x^{(h)}) = \frac{p_\theta(y^{(h)} \mid x^{(h)})}{\sum_{b=1}^{B} p_\theta(y^{(b)} \mid x^{(b)})}. \tag{70}$$

We additionally normalize the log probability of a translation $y$ by its length $T$: $p_\theta^{norm}(y \mid x) = \exp\left(\frac{\log p_\theta(y|x)}{T}\right)$. This length normalization stabilizes the updates, since the model probabilities are very small due to the composition of the probability of the sequence as a product of token probabilities (Equation 6). In our experiments, we apply the DPM objective to simulated data and explicit human feedback data.

**Doubly controlled estimation.** Lawrence et al. (2017b) furthermore propose the *Doubly Controlled* objective (DC, Equation 71). In addition to learning from the logged reward for the logging system, the reward for other translations is estimated by a parametrized regression model that is trained on the log $\hat{r}_\theta : \mathcal{Y} \rightarrow [0, 1]$. This objective contains both a multiplicative (probability reweighting) and an additive (reward estimate) control variate, hence the name.[3]

$$J^{\mathrm{DC}}(\theta) = \frac{1}{H} \sum_{h=1}^{H} \Bigg[ \Big( r(y^{(h)}) - \hat{r}_\theta(y^{(h)}) \Big) \cdot \bar{p}_\theta(y^{(h)} \mid x^{(h)}) + \sum_{\tilde{y} \in \mathcal{S}(x^{(h)})} \hat{r}_\theta(\tilde{y})\, p_\theta(\tilde{y} \mid x^{(h)}) \Bigg] \tag{71}$$

As for MRT, the expectation over the full output space is approximated with a subset of $k$ sample translations $\mathcal{S}(x) \subset \mathcal{Y}(x)$. This objective can be seen as a combination of DPM and MRT, where the combination strategy is dependent on the quality of the reward estimator: If it is perfect ($r(y^{(h)}) = \hat{r}_\theta(y^{(h)})$), only the MRT part of the objective is active, i.e., the expected estimated reward maximization. As long as the reward estimator is imperfect, the logged reward is part of the update. Similar to DPM, we apply the DC objective to simulated data and explicit human feedback data in our experiments.

**Average reward baseline.** REINFORCE objectives are known to benefit from variance reduction via the average reward baseline. That means that a running average of historic rewards $b$ is subtracted from the current reward in the gradient updates (Section 2.2.3). In the experiments we apply this control variate not only to the online ER optimization, but also to DPM and DC objectives with offline feedback, since learning still operates on the basis of stochastic mini-batch updates. For DC we replace the subtraction of the estimated reward $\hat{r}_\theta(y^{(h)})$ in the first part of Equation 71 with the subtraction of the average logged reward, and add to the second part the subtraction of the average estimated reward $\bar{\hat{r}}_h = \frac{1}{H} \sum_{h=1}^{H} \frac{1}{k} \sum_{\tilde{y} \in \mathcal{S}(x^{(h)})} \hat{r}_\theta(\tilde{y})$. Despite introducing bias in the gradient estimate, we found this use of the baseline control variates to be effective, especially in combination with suboptimal reward estimators, presumably for its reward-relativizing effect (Section 3.3.2).

## 4.3 Experiments

### 4.3.1 Setup

In our experiments, we first evaluate a range of learning objectives in a simulation experiment both on a public domain and the eBay domain, and then investigate

[3]We find empirically that estimating $\hat{c}$ over the current batch as in objective $\hat{c}$DC in (Lawrence et al., 2017b) does not improve over the simple setting with $c = 1$.

| **Description** | **Number of sentences** |
|---|---|
| Out-of-domain training data | 2,741,087 |
| In-domain training data (for simulations) | 62,162 |
| In-domain development data | 1,619 |
| In-domain test data | 1,000 |
| Product titles with user star ratings | 69,412 |
| … of which are rated 5 stars on average | 40,064 |
| … of which are rated by experts | 1,000 |
| … of which are judged (correct/incorrect) by experts | 2,000 |
| Query-title pairs | 164,065 |
| … of which have *recall* $= 1$ | 61,965 |

Table 12: Data set sizes for collected feedback in number of sentences. The in-domain title translations are only used for simulation experiments.

three approaches to learn from logged human feedback for e-commerce translations: (1) bandit-to-supervised conversion, (2) counterfactual bandit learning for explicit human feedback, and (3) minimum-risk training with task-specific reward scores for implicit human feedback.

Learning from feedback starts from a pre-trained English-to-Spanish NMT model that has not seen in-domain data (i.e. no product title translations). This corresponds to a domain adaptation setting, where product title reference translations are not available, but a general-domain model can be used to gather initial feedback from users.

### Data

We conduct experiments on an English-to-Spanish e-commerce item titles translation task. A small in-domain corpus of in-house e-commerce data (product titles, descriptions, etc.) is available, for training in simulations (comparable in size to the user-rated translations), and for development and testing. The baseline is trained on out-of-domain data selected from available parallel corpora, that is Europarl, TAUS, and OpenSubtitles released by the OPUS project (Tiedemann, 2009). It is sub-sampled according to the similarity to the in-domain data by selecting 25% of the most similar sentence pairs following the invitation approach (Cuong and Sima'an, 2014). The corpus is pre-processed by tokenization and replacement of numbers and product specifications with a placeholder token (e.g., '6S', and '1080p'). Table 12 gives an overview of the type and the size of the data, including the feedback that is collected explicitly (rated product title translations) and implicitly (query-title pairs).

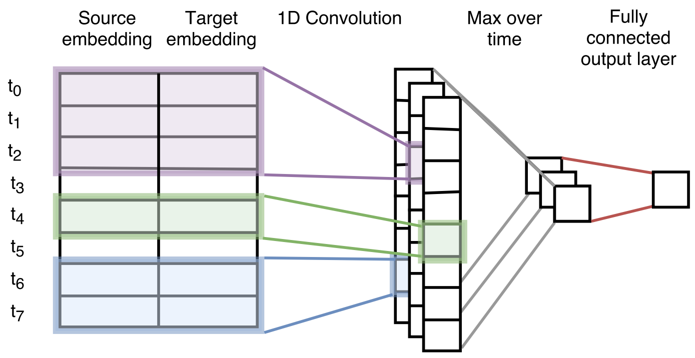


Figure 9: Model architecture for the reward estimator. This example has one filter for each filter size (3: purple, 1: green, 2: blue). Source and target sequences are padded up to a maximum length, here $T_{max} = 8$.

### NMT Model

The NMT baseline model (BL) is a standard subword-based encoder-decoder architecture with attention (Bahdanau et al., 2015), implemented with Python and TensorFlow (Abadi et al., 2016) on top of eBay's in-house NMT toolkit. It has a bi-directional RNN encoder with one layer of 1,000 GRUs, a decoder with 1,000 GRUs, and source and target word embeddings of size 620.

The vocabulary is generated from the out-of-domain training corpus with 40k byte-pair merges (BPE) (Sennrich et al., 2016c) and contains 40,813 source tokens and 41,050 target tokens. The full softmax is approximated by 1,024 samples as proposed by Jean et al. (2015a). Dropout (Gal and Ghahramani, 2016) is applied with probability $p = 0.1$ to the embedding matrices, with $p = 0.2$ to the input and recurrent connections of the RNNs.

The model is trained with MLE on out-of-domain data and the early stopping point is determined on a small in-domain dev set of 1,619 product title translations. A beam of size 12 and length normalization (Wu et al., 2016) are used for beam search decoding. For BLEU score evaluation we use the multi-bleu script of the Moses decoder (Koehn et al., 2007), for TER computation the `tercom` tool (Snover et al., 2006). For NMT models involving random sampling, we report average results and standard deviation (in subscript) for two runs with different random seeds. Detailed hyperparameter settings are given in Appendix B.2.

### Reward Estimator

The model architecture for the reward regressor used in the DC objective is a bilingual extension of the convolutional neural network (CNN) for sentence classification proposed by Kim (2014). Both source and target sequences are padded up to a pre-defined maximum sequence length $T_{max}$, their embeddings are

| Data | MSE | Macro-avg. Distance | Micro-avg. Distance | Pearson's $r$ | Spearman's $\rho$ |
|---|---|---|---|---|---|
| Star ratings | 0.1620 | 0.0065 | 0.3203 | 0.1240 | 0.1026 |
| sBLEU | 0.0096 | 0.0055 | 0.0710 | 0.8816 | 0.8675 |

Table 13: Results for the reward estimators trained and evaluated on human star ratings and simulated sBLEU.

concatenated and further processed by a 1D-Convolution over the time dimension, which is then followed by max-over-time pooling and fed to a fully-connected output layer (Figure 9) that produces a scalar score.

The model is trained to minimize the mean squared error (MSE) on the training portion of the logged feedback data (60,000 for simulated sentence-BLEU feedback, 62,470 for star rating feedback). The word embeddings of the reward estimator are initialized by the word embeddings of the trained baseline NMT system and fine-tuned further together with the other CNN weights. The best parameters are identified by early-stopping on the validation portion of the feedback data (2,162 for the simulation, 6,942 for the star ratings). Detailed hyperparameter settings are reported in Appendix B.3.

### 4.3.2 Reward Estimation Quality

#### Held-Out Set Evaluation

Results for a stand-alone evaluation of the reward estimator on the validation portions of the feedback data are given in Table 13. The estimator for sBLEU models the data much more accurately than the user star rating estimator. The correlation of the estimated star ratings with the ground truth is poor. This is due to large variance and skew of the user ratings.[4]

An estimator trained with MSE typically predicts values around the mean, which is not a suitable strategy for such a skewed distribution of labels, but is successful for the prediction of normal-distributed sBLEU. For further illustration, Figure 8b shows the histogram of sBLEU scores for the simulation experiments with logged feedback on the training set. The distribution of simulated rewards appears to be much closer to a normal distribution than the rewards collected from users that had a strong skew towards positive ratings (compare Figure 8).

[4]Nguyen et al. (2017) analyze the detrimental effects of variance and skew for bandit learning for NMT.

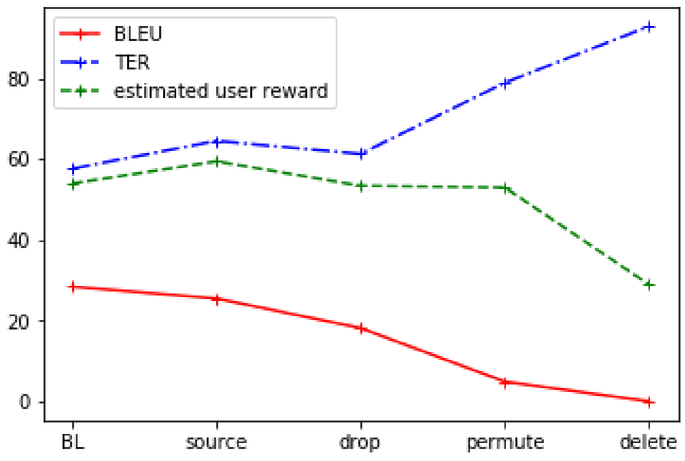


Figure 10: Effect of perturbations on test title translations of the baseline model (BL) measured by different MT evaluation metrics (BLEU, TER, estimated user reward): 'source': replace the translation by its source, 'drop': each word is removed by 20% chance, 'permute': word order is randomly permuted, 'delete': delete complete translation.

### Robustness

The reward estimator is used in the DC objective to judge the quality of sampled translations. In order to get more insights into its behavior, we probe its robustness by perturbing translations and measuring the change in estimated quality. This lets us also compare to non-parametric reward functions like BLEU and TER that are commonly used in simulations. Ideally, the reward estimator should show sensitivity to perturbations that reduce the translation quality.

Figure 10 (page 92) illustrates the effect of a range of perturbations of the BL NMT translations on the estimated user reward in comparison to BLEU and TER. The estimated reward for the baseline translations is 53.93. When the source is equal to the target, the estimated reward rises slightly, just as TER, while BLEU drops ("source" in Figure 10). When a random selection of words from each translation is removed (each word is removed by 20% chance), BLEU and TER drop a bit more. ("drop" in Figure 10). When the order of the words in each translation is randomly permuted, the BLEU is drastically lowered to 4.87, the estimated reward drops only by 0.6, TER rises to 78.87 ("permute"). If the translation is completely deleted, the TER drops, and BLEU is 0. However, the estimated reward is still 28.89 ("delete").

This shows that the estimated reward function is not as sensitive to word omissions and permutations as BLEU and TER, and furthermore encourages source words on the target side. It might be a result of the large skew towards positive ratings in the underlying training data (cf. Figure 8a), and will cause the estimated reward function to be overly optimistic, harming its capability to distinguish good and bad translations.[5]

### 4.3.3 Simulation: Online vs. Offline Feedback

First we test the proposed objectives in simulation in order to find out whether 1) our transfer from statistical to neural models was successful, 2) the objectives work well across domains, and 3) how much loss is expected when going from online to offline feedback.

#### Simulation on Public Data

We use exactly the same data, preprocessing and splits as Lawrence et al. (2017b) to compare with their French-to-English news experiments on counterfactual learning with deterministically logged feedback for SMT. Just like in the experiments in Chapter 3, the baseline model is trained with MLE on Europarl (EP) translations, bandit feedback is then simulated from News Commentary (NC) translations.

The results comparing our neural models to the log-linear ones from Lawrence et al. (2017b) are given in Table 14. The NMT out-of-domain baseline is noticeable stronger than the SMT one. Out-of-domain pre-training and then fine-tuning (EP$\rightarrow$NC) outperforms solely training on in-domain data (NC), and also shows larger improvements for NMT (4.93 BLEU) than SMT (2.81 BLEU).

Looking at the models trained with bandit feedback, we notice that the improvement with online feedback (ER) is relatively small (0.47 BLEU), and DPM with offline feedback does not yield any improvements over the baseline. Thanks to the reward estimator that allows it to explore the output space, DC achieves the highest improvements of 0.65 BLEU.

When greedy decoding is used instead of beam search (like in Chapter 3), the difference in BLEU scores between domains is larger, and there is still an advantage of online over offline feedback (ER:+1.61 BLEU, DC:+1.07 BLEU). Since there is (limited) exploration during decoding in beam search, the gains diminish. This aligns with the relatively larger improvements found for ER in Section 3.4.2, and was also observed by Bahdanau et al. (2017).

#### Simulation on E-commerce Data

Moving to the target domain, we adapt the NMT out-of-domain baseline to the e-commerce domain with the in-domain parallel corpus, where bandit feedback is

[5]In hindsight, it might have been advantageous to train the reward estimator directly as a discriminator (e.g. with a ranking objective or like in generative-adversarial approaches (Wu et al., 2017; Yu et al., 2017b)).

| **Training** | **Model** | **SMT** | **NMT** | |
|---|---|---|---|---|
| | | | beam search | greedy |
| Fully Supervised | EP BL | 25.27 | 27.55 | 26.32 |
| | NC BL | – | 22.35 | 19.63 |
| | EP→NC | 28.08 | 32.48 | 31.04 |
| Bandit Feedback | ER | – | 28.02 | 27.93 |
| | DPM | 26.24 | 27.54 | 26.36 |
| | DC | 26.33 | 28.20 | 27.39 |

Table 14: BLEU scores for simulation models evaluated on the French-to-English NC test set. SMT results are from (Lawrence et al., 2017b) ("–" if not reported). The BL models are trained with MLE on either EP or NC. EP→NC is fine-tuned on NC after being pre-trained on EP. ER, DPM, and DC are trained with simulated bandit feedback on NC data.

simulated by evaluating a sampled translation against a reference using smoothed sentence-BLEU (sBLEU). Similar to previous studies on SMT (Lawrence et al., 2017b), this reward is deterministic and does not contain user-dependent noise. Translations for off-policy learning with simulated rewards are generated by beam search decoding (beam width of 5) with the out-of-domain baseline NMT system. Models that involve random sampling are repeated three times and their mean results and standard deviation are reported. Table 15 lists results with beam search decoding.

When fine-tuning the baseline model (OD) on in-domain references (OD→ID MLE), it improves by 3.34 BLEU on the in-domain test set. By tuning it on the same in-domain data for sBLEU with MIX, it gains another 3 BLEU points (OD→ID MIX). When feedback is given to only one translation per input (=online bandit feedback), the model (ER) achieves comparable performance to MLE training with references.

When the feedback is logged offline for one round of deterministic outputs of the baseline model (=offline bandit feedback), we can still find improvements of 1.81 BLEU (DPM). With a reward estimator trained on this log, DC achieves improvements of 3 BLEU. To test the contribution of the feedback in contrast to a simple in-domain training effect, we randomly perturb the pairing of feedback signal and translation, and retrain (DPM-random). This degrades results, confirming the sBLEU feedback to be a useful signal rather than noise.

| **Training** | **Model** | **Test BLEU** | **Test TER** |
|---|---|---|---|
| Fully Supervised | OD BL | 28.38 | 57.58 |
| | OD$\rightarrow$ID MLE | 31.72 | 53.02 |
| | OD$\rightarrow$ID MIX | $34.79_{\pm 0.02}$ | $48.56_{\pm 0.02}$ |
| Bandit Feedback | ER | $31.78_{\pm 0.06}$ | $51.11_{\pm 0.36}$ |
| | DPM | 30.19 | 56.28 |
| | DPM-random | 28.20 | 57.89 |
| | DC | $31.11_{\pm 0.34}$ | $55.05_{\pm 0.02}$ |

Table 15: Results for simulation experiments with NMT evaluated on product titles dev and test set. MLE and MIX assume references, ER learns on online bandit feedback, DPM and DC on offline feedback. DPM-random was trained on the same logged translations, but with randomly permuted feedback. For models with random sampling, mean scores and standard deviation across three runs are reported. OD: out-of-domain, ID: in-domain.

### 4.3.4 Explicit Star Rating Feedback

#### Counterfactual Bandit Learning

As shown in Table 16, counterfactual learning (DPM/DC) on the logged star ratings as direct reward does not yield any improvements over the baseline model in terms of corpus BLEU or TER. A randomization of feedback signals for translations gives the same results (DPM-random), showing that counterfactual learning from logged star ratings has the same effect as learning from noise. This stands in contrast to the simulation results in the same domain with sBLEU feedback (Section 4.3.3).

Evaluating the translations with the reward estimator instead of BLEU or TER, we find an improvement of +1.49 for DC, +0.04 for DPM over the baseline (53.93) (not shown in Table 16). That means that DC "improved" at least, but not in terms of BLEU and TER. However, the reward estimator largely over-estimates the translation quality of translations with major faults (see Section 4.3.2), so these improvements might be misleading. This over-estimation might be the result of missing exploration in the logged translations (Swaminathan and Joachims, 2015a), and was also observed in (Lawrence et al., 2017b). Hence it is not desirable to optimize towards this signal directly.

#### Bandit-to-Supervised Conversion

Since a large portion of the translations was rated with five stars and experts agreed on these ratings, we utilize the user ratings to filter the log to keep only

| **Training** | **Model** | **Test BLEU** | **Test TER** |
|---|---|---|---|
| Fully Supervised | OD BL | 28.38 | 57.58 |
| Bandit Feedback | DPM | 28.19 | 57.80 |
| | DPM-random | 28.19 | 57.64 |
| | DC | $28.41_{\pm 0.85}$ | $64.25_{\pm 1.66}$ |
| Bandit-to-Supervised | MLE (all) | 31.98 | 51.08 |
| | MIX (all) | $34.47_{\pm 0.06}$ | $47.97_{\pm 0.18}$ |
| | MIX (small) | $34.16_{\pm 0.09}$ | $48.12_{\pm 0.33}$ |
| | MIX (stars = 5) | $34.35_{\pm 0.11}$ | $47.99_{\pm 0.13}$ |

Table 16: Results for models trained on *explicit user ratings* evaluated on product titles dev and test set. 'small' indicates a random subset of logged translations of the same size as the filtered log that only contains translations with an average rating of five stars ('stars = 5'). The differences in BLEU between the MIX models are not significant at $p = 0.05$.

five-star rated translations. On this subset, the model can be trained with MLE and MIX using sBLEU against pseudo-references as reward function.

Table 16 (page 96) shows that this filtering strategy leads to large improvements over the baseline, for MLE and even more for MIX, even though the data set size is reduced by 42%. However, around the same improvements can be achieved with a random selection of logged translations of the same size (MIX small, containing 55% five-star ratings).

Using all logged translations for training MIX achieves the best results. This suggests that the model does not benefit from fine-grained feedback, but mostly from being exposed to in-domain translations of the logging system. This effect is similar to training on pseudo-references created by back-translation (Sennrich et al., 2016a,b), and was also observed by Wun et al. (2018) for a Chinese-to-English NMT system that gets adapted with beam search translations of a monolingual source-side in-domain corpus.

### 4.3.5 Task-Based Implicit Feedback

We apply the same filtering technique to the logged implicit feedback by treating translations with recall = 1 as references for training MIX with sBLEU (reduction of the data set by 62%). The results in Table 17 show that large improvements over the baseline can be obtained even without filtering, BLEU and TER scores being comparable to the ones observed for training on explicit user ratings.

| **Training** | **Model** | **Test BLEU** | **Test TER** |
|---|---|---|---|
| Fully Supervised | OD BL | 28.38 | 57.58 |
| Bandit-to-Supervised | MLE (all) | 31.89 | 51.35 |
| | MIX (all) | $34.39_{\pm 0.08}$ | $47.94_{\pm 0.24}$ |
| | MIX (small) | $34.13_{\pm 0.26}$ | $48.27_{\pm 0.60}$ |
| | MIX (recall = 1) | $34.17_{\pm 0.02}$ | $47.72_{\pm 0.26}$ |

Table 17: Results for models trained on implicit task-based feedback data evaluated on product titles dev and test set. 'small' indicates a random subset of logged translations of the same size as the filtered log that only contains translations that contain all the query words ('recall = 1'). The BLEU score of MIX (small) significantly differs from MIX (all) at $p = 0.05$, the score of MIX (recall = 1) does not.

**Task-based Feedback**

The key difference between the implicit feedback collected in the query-title data and the explicit user ratings, is that it can be used to define reward functions like recall or match (Equations 63, 62). For the experiments we train T-MIX, the token-based MRT objective (Equation 66) linearly combined with MLE, on the logged translations accompanying the queries (160k sentences).

To account for user-generated language in the queries and subwords in the MT model, we soften the conditions for a match: if a token is part of a word that is either contained in the query, or has less than edit distance 3 (or less than 30% of its length) to the query, it counts as a match.

Table 18 repeats the best MIX results from Tables 15 (ID MIX), 16 (MIX (all ratings)), and 17 (MIX (all queries)), and evaluates the models with respect to query recall. We also report the query recall for the logged translations and the out-of-domain baseline. These results are compared to T-MIX training on implicit feedback data described in Section 4.1.2. The development portion of the query-title dataset contains 4,065 sentences, the test set 2,000 sentences, which is used for query recall evaluation. TER and BLEU are computed on translations of the title test set.

The T-MIX model shows the largest improvement in query recall (12% points) and BLEU (6 points) over the baseline. It comes very close to the BLEU/TER results of the model trained on in-domain references, but surpasses its recall by far. This is remarkable since the model does not use any human generated references, but trains all its components on logged data of task-based human feedback. Example translations can be found in Appendix C.1.

| **Training** | **Model** | **Test recall** | **Test BLEU** | **Test TER** |
|---|---|---|---|---|
| – | Logged translations | 65.33 | – | – |
| Fully Supervised | OD BL | 45.96 | 28.38 | 57.58 |
| | ID MIX (simulated) | $51.89_{\pm 0.37}$ | $34.79_{\pm 0.02}$ | $48.56_{\pm 0.02}$ |
| Bandit-to-Supervised | MIX (all ratings) | $62.92_{\pm 0.56}$ | $34.47_{\pm 0.06}$ | $47.97_{\pm 0.18}$ |
| | MIX (all queries) | $63.21_{\pm 0.24}$ | $34.39_{\pm 0.08}$ | $47.94_{\pm 0.24}$ |
| Query Matching | T-MIX | $68.12_{\pm 0.27}$ | $34.52_{\pm 0.02}$ | $46.91_{\pm 0.03}$ |

Table 18: Query recall results on the query test set, and BLEU and TER scores on the title test data, comparing MIX models trained on logged translations with the T-MIX model trained via word-based query matching, and the MIX model trained on in-domain model with simulated rewards (ID MIX). The difference in BLEU between the MIX (rating, queries) models and the T-MIX model is not significant at $p = 0.05$, but the difference to the MIX model trained on in-domain references is significant.

## 4.4 Conclusion

In this chapter we presented, compared, and evaluated methods to improve NMT from offline human reinforcement signals to translations of product titles. The signals were logged from user activities of an e-commerce platform and consist of explicit ratings on a five-point Likert scale and implicit task-based feedback collected in a cross-lingual search task.

We found that learning from explicit feedback is successful only in a simulated setting without noise, and fails with human star ratings. However, implicit task-based feedback was used successfully as a reward signal for NMT optimization, leading to improvements both in terms of enforcing individual word translations and in terms of automatic evaluation measures.

With an initial impression of the extent of noise and user-level variation that explicit user ratings entail, we will dive deeper into the effects of reliability of reinforcement learning signals in the next chapter.

# Chapter 5

# Reliability and Learnability of Human Feedback

In this chapter we will show that the reliability of feedback signals is a catalyst for learning from human feedback. As described in the previous chapter, the first deployment of bandit NMT in an e-commerce translation scenario conjectured lacking reliability of user judgments as the reason for disappointing results. We thus raise the question of how human feedback can be gathered in the most reliable way, and which effect reliability will have in downstream tasks.

In order to answer these questions, we measure intra- and inter-annotator agreement for two feedback tasks for bandit NMT, using cardinal feedback (on a 5-point scale) and ordinal feedback (by pairwise preferences) for 800 translations, conducted by 16 and 14 human raters, respectively. Perhaps surprisingly, while relative feedback is often considered easier for humans to provide (Thurstone, 1927), our investigation shows that $\alpha$-reliability (Krippendorff, 2013) for intra- and inter-rater agreement is similar for both tasks, with highest inter-rater reliability for standardized 5-point ratings.

In a next step, we address the issue of machine learnability of human rewards. We use deep learning models to train reward estimators by regression against cardinal feedback, and by fitting a Bradley-Terry model (Bradley and Terry, 1952) to ordinal feedback. Learnability is understood by a slight misuse of the machine learning notion of learnability (Shalev-Shwartz et al., 2010) as the question of how well reward estimates can approximate human rewards.

Estimating human quality ratings is also known as the task of Quality Estimation (QE) (Specia et al., 2010). However, there are crucial differences between sentence-level QEn (sQE) and the reward estimation in our work: sQE usually has more training data, often from more than one machine translation model. Its gold labels are inferred from post-edits (as for example in the yearly WMT shared task[1]), i.e. corrections of the machine translation output, while we learn from weaker bandit feedback.

Our experiments reveal that rank correlation of reward estimates with TER against human references is higher for regression models trained on standardized cardinal rewards than for Bradley-Terry models trained on pairwise preferences. This emphasizes the influence of the reliability of human feedback signals on the quality of reward estimates learned from them.

[1]`https://www.statmt.org/wmt19/qe-task.html`

Lastly, we investigate machine learnability of the overall NMT task, in the sense of Green et al. (2014) who posed the question of how well an MT system can be tuned on post-edits. We use an RL approach for tuning, where a crucial difference of our work to previous work on RL from human rewards (Knox and Stone, 2009; Christiano et al., 2017) is that our RL scenario is not interactive, but rewards are collected in an offline log. RL then can proceed either by off-policy learning using logged single-shot human rewards directly, or by using estimated rewards.

We expect an advantage by estimating rewards first, since the arguably simpler problem is addressed first—learning a reward estimator instead of a full RL task for improving NMT. The reward estimator can then supply unlimited feedback for the MT model. This type of function approximation instead of using rewards directly has been proposed in the RL literature under the name of "actor-critic" methods with a wide range of algorithmic variants (for a literature review see Section 2.2.4). Previous works with critics trained from human feedback lack a systematic investigation of the reliability of the feedback and its impact on the down-stream task.

Our results show that significant improvements can be achieved by training NMT from both estimated and logged human rewards, with best results for integrating a regression-based reward estimator into RL. This completes the argumentation that high reliability influences quality of reward estimates, which in turn affects the quality of the overall NMT task. Altogether, we gather data- and user-supported insights into the reliability of Likert-scale and pairwise ratings, which enables us to both understand and leverage the collected explicit ratings more successfully than in the eBay study presented in the previous chapter. Since the size of our training data is tiny in machine translation proportions, this result points towards a great potential for larger-scale applications of RL from human feedback.

**Contributions.** The contributions of this chapter are the following:

1. We investigate reliability and learnability of human feedback, which have previously been neglected in human reinforcement learning studies.

2. In addition to the previously used Likert-scale rating interface (Chapter 4), we investigate a pairwise preference interface for learning from logged feedback. To our knowledge this is the first work that compares these rating interfaces in a systematic and thorough manner.

3. The empirical results on a German-to-English translation task show that even small-scale five-point feedback can yield down-stream improvements for NMT, given a certain level of reliability and the abstraction through a reward estimator.

4. We highlight the weaknesses of pairwise ratings, which have previously been considered advantageous.

**Publications.** This work was published in (Kreutzer et al., 2018b). Joshua Uyheng contributed the practical setup of the rating interface and the agreement and ablation analysis (Section 5.1) of the collected ratings. The linear mixed effects analysis (Section 5.2.2) was added after the publication. The remaining work was done by the author. The resulting collection of ratings was published as a data set and is available at `http://www.cl.uni-heidelberg.de/statnlpgroup/humanmt/`.

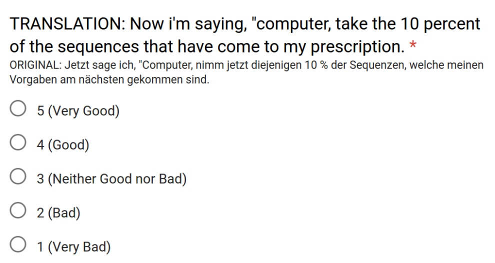
**Outline.** Section 5.1 starts with a description of the setup of the feedback collection study. The reliability of the collected feedback is analyzed in Section 5.2, and the learnability in Section 5.3. Section 5.4 evaluates the feedback and trained reward estimators in how far they can improve the NMT system. The findings of this chapter are finally summarized in Section 5.5.

## 5.1 Human MT Rating Task

### 5.1.1 Data

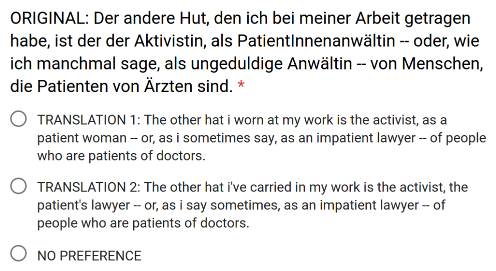
We translate a subset of the TED corpus from German to English with a general-domain and a domain-adapted NMT model (see Section 5.4.2 for details about NMT system and data), post-process the translations (replacing special characters, restoring capitalization) and filter out identical out-of-domain and in-domain translations. In order to compose a homogeneous data set, we first select translations with references of length 20 to 40, then sort the translation pairs by difference in character n-gram F-score (ChrF, $\beta = 3$) (Popović, 2015) and length, and pick the top 400 translation pairs with the highest difference in chrF but lowest difference in length. This yields translation pairs of similar length, but different quality.

The pairs were treated as 800 separate translations for a 5-point rating task. 100 translation pairs were randomly selected for repetition to measure intra-rater reliability. This produced a total of 1,000 individual translations, with 600 occurring once, and 200 occurring twice. The translations were shuffled and separated into five annotation sections, ensuring that a single translation does not occur more than once in each section. For a pairwise rating task, the same procedure was applied to pairs of translations.


TRANSLATION: Now i'm saying, "computer, take the 10 percent of the sequences that have come to my prescription. *
ORIGINAL: Jetzt sage ich, "Computer, nimm jetzt diejenigen 10 % der Sequenzen, welche meinen Vorgaben am nächsten gekommen sind.

○ 5 (Very Good)
○ 4 (Good)
○ 3 (Neither Good nor Bad)
○ 2 (Bad)
○ 1 (Very Bad)

(a) 5-point ratings


ORIGINAL: Der andere Hut, den ich bei meiner Arbeit getragen habe, ist der der Aktivistin, als PatientInnenanwältin -- oder, wie ich manchmal sage, als ungeduldige Anwältin -- von Menschen, die Patienten von Ärzten sind. *

○ TRANSLATION 1: The other hat i worn at my work is the activist, as a patient woman -- or, as i sometimes say, as an impatient lawyer -- of people who are patients of doctors.
○ TRANSLATION 2: The other hat i've carried in my work is the activist, the patient's lawyer -- or, as i say sometimes, as an impatient lawyer -- of people who are patients of doctors.
○ NO PREFERENCE

(b) pairwise ratings

Figure 11: User interfaces for 5-point (a) and pairwise ratings (b).

### 5.1.2 Rating Task

We recruited 14 participants for the pairwise rating task and 16 for the 5-point rating task. The participants were university students with fluent or native language skills in German and English. The above described selected translations (pairs) were presented in a Google form with one annotation section per page. The rating interfaces are shown in Figure 11. Note that no reference translations were presented since the objective is to model a realistic scenario for bandit learning, where references are not available.

Participants for the 5-star rating task were given the following instructions: *"You will be presented with a German statement and a translation of this statement in English. You must assign a rating from 1 (Very Bad) to 5 (Very Good) to each translation."*

Participants for the pairwise task were given the following instructions: *"You will be presented with a German statement and two translations of this statement in English. You must decide which of the two translations you prefer, or whether you have no preference."*

## 5.2 Reliability of Human MT Ratings

The goal of this analysis is to assess the reliability of both types of ratings. We inspect three different aspects of reliability:

1. How well do raters agree in each mode? $\rightarrow$ Reliability is measured in terms of intra- and inter-rater agreement in Section 5.2.1.

2. What is the influence of rater and item variance? $\rightarrow$ We assess this (1) by observing the effect of filtering by rater or item variance, and (2) by a linear mixed effects analysis to compare both rating modes while accounting for rater and item variance in Section 5.2.2.

3. What are the subjective and objective difficulties for each mode? → We capture the rater's experience in a feedback form and discuss the results, together with a quantitative analysis, in Section 5.2.3.

Inter- and intra-rater reliability of the cardinal and ordinal feedback tasks described in Section 5.1.2 can be estimated with Krippendorff's $\alpha$ (Krippendorff, 2013) on interval and ordinal scale, respectively. Table 19 (page 104) shows that differences in inter-rater reliability between the 5-point and pairwise task are small ($\alpha = 0.2308$ vs. $\alpha = 0.2385$. 5-point scores are further normalized by standardization to Z-scores so that individual rater's tendencies towards high or low ratings are corrected. That means that every rating $r_{(i,j)}$ by rater $j$ for example $i$ of $n$ is normalized by subtracting the per-rater-mean:

$$r_{(i,j)}^{norm} = r_{(i,j)} - \frac{1}{n}\sum_{k=1}^{n} r_{(k,j)}. \tag{72}$$

This results in a marked improvement of overall inter-rater reliability for the 5-point task ($\alpha = 0.2820$). A one-way analysis of variance taken over inter-rater reliabilities between pairs of participants suggests statistically significant differences across tasks ($F(2, 328) = 6.399, p < 0.01$), however, a post hoc Tukey's (Larsen and Marx, 2012) honest significance test attributes statistically significant differences solely between the 5-point tasks with and without normalization. When inferring paired ratings from 5-point ratings,[2] the inter-rater agreement slightly improves compared to raw pairwise ratings, but the intra-rater reliability drops. This is consistent with the observation that 5-point ratings have slightly higher inter-rater reliability (when normalized), but tend to have lower intra-reliability. While the absolute agreement scores seem relatively low, they lie within the range reported in literature for MT (Turian et al., 2003; Carl et al., 2011; Lommel et al., 2014; Guzmán et al., 2015) and more generally in text generation evaluation problems (Godwin and Piwek, 2016; Verberne et al., 2018).

### 5.2.1 Inter-rater and Intra-rater Reliability

Intra-rater reliability was on average higher among participants in the pairwise task ($\alpha = 0.5085$) than in the 5-point task ($\alpha = 0.4014$). This suggests that, on average, human raters provide more consistent ratings with themselves in judging a pair translation in contrast versus single translations in isolation. Seeing multiple translations may provide raters with more cues to make a consistent judgment, such as a wider range of lexical translation options (in the

[2] "Re-pairing": If translation A obtained a higher rating on the 5-point scale than translation B, it is assumed to be preferred over B in a pairwise rating setup.

| | **Inter-rater** | **Intra-rater** | |
|---|---|---|---|
| **Type** | $\alpha$ | **Mean** $\alpha$ | **Stdev.** $\alpha$ |
| 5-point | 0.2308 | 0.4014 | 0.1907 |
| + normalized | 0.2820 | | |
| + filtered by rater | 0.5059 | 0.5527 | 0.0470 |
| + filtered by item | 0.3236 | 0.3845 | 0.1545 |
| Pairwise | 0.2385 | 0.5085 | 0.2096 |
| + filtered by rater | 0.3912 | 0.7264 | 0.0533 |
| + filtered by item | 0.3519 | 0.5718 | 0.2591 |
| Re-paired from 5-point | 0.2569 | 0.2800 | 0.0765 |

Table 19: Inter- and intra-reliability measured by Krippendorff's $\alpha$ for 5-point and pairwise ratings of 1,000 translations of which 200 translations are repeated twice. The filtered variants are restricted to either a subset of raters or a subset of items (translations). The "re-paired" pairwise ratings are inferred from pairwise 5-point rating comparisons.

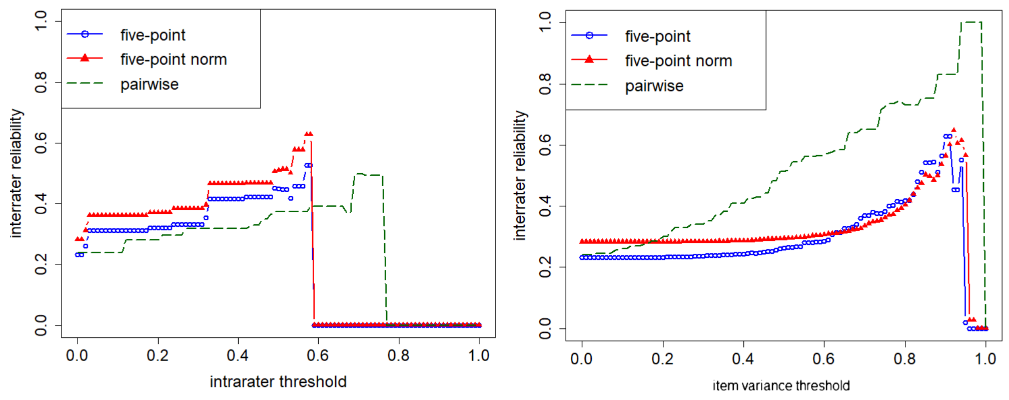
example from Figure 11b "patient woman" or "patient's lawyer" as translation for "PatientInnenanwältin"), and may cause them to inspect the translations more thoroughly in order to spot differences between translations ("as I sometimes say" vs. "as I say sometimes").

However, at the current sample size, a Welch two-sample t-test (Larsen and Marx, 2012) between (unfiltered) 5-point and pairwise intra-rater reliabilities shows no significant difference between the two tasks ($t\,(26.92) = 1.4362$,$p = 0.1625$) and intra-rater normalization does not affect the intra-rater reliability. Thus, it remains difficult to infer whether one task is definitively superior to the other in eliciting more consistent responses. Larger sample sizes, i.e., studies with more raters or translations would be required to draw definite conclusions.

### 5.2.2 Rater and Item Variance

#### Ablation Analysis

The following analysis is based on two assumptions: first, human raters vary in that they do not provide equally good judgments of translation quality, and second, items vary in that some translations may be more difficult to judge than others. This allows to investigate the influence of rater variance and item variance on inter-rater reliability by an ablation analysis where low-quality raters and difficult translations are filtered out. In practice, filtering out unreliable raters, e.g., in (Akkaya et al., 2010) or items with inconsistent ratings, e.g., in (Laws et al., 2011) or (Jamison and Gurevych, 2015), is a common practice for



(a) intra-rater consistency filtering (b) item-variance filtering

Figure 12: Improvements in inter-rater reliability after filtering out raters with lowest intra-rater reliability (a) or items with highest variance (b). Plots created by Joshua Uyheng.

quality control in crowd-sourced annotation tasks on platforms like Amazon Mechanical Turk. For economic reasons it is, however, desirable to retain as many ratings as possible. In particular, when they are intended to train deep models, which are notorious for requiring large amounts of data.

Figure 12a shows a filtering process where human raters with intra-rater $\alpha$ scores lower than a moving threshold are removed from the pool. As the reliability threshold increases from 0 to 1, overall inter-rater reliability is improving. Figure 12b shows a similar filtering process by variance in ratings per item. Item variances are normalized to a scale from 0 to 1 and subtracted from 1 to produce an item variance threshold. As the threshold increases, high-variance items are progressively removed from the pool.

As the plots demonstrate, inter-rater reliability generally increases with consistency and variance filtering. Figure 12a shows how the inter-rater reliability of the 5-point task increases more quickly of the pairwise task when filtering out raters. The reason for that effect is that more participants in the 5-point task had low intra-rater reliability. Pairwise tasks, on the other hand, require higher thresholds before large gains are observed in overall inter-rater reliability, as more participants in the pairwise task had relatively high intra-rater reliability.

In the normalized 5-point task, selecting a threshold of 0.49 for intra-rater reliability retains 8 participants with an inter-rater reliability of 0.5059. For the pairwise task, a threshold of 0.66 leaves 5 participants with an inter-rater reliability of 0.3912, which is the result we include in Table 19 (page 104). Figure 27a in Appendix D.1 shows how this threshold was selected.

The opposite phenomenon is observed in the case of variance filtering. As shown in Figure 12b, the overall inter-rater reliability of the pairwise task quickly overtakes that of the 5-point task. In the pairwise setup, more items can be a source of disagreement among human judges; and especially ambiguous cases, which will be discussed in Section 5.2.3, may result in higher item variance. This problem is not as pronounced in the 5-point task, where judges must simply judge individual translations. It may be surmised that this item variance accounts for why judges in the pairwise task demonstrate higher intra-rater reliability on average than those in the 5-point task, yet the overall inter-rater reliability of the pairwise task is lower.

By selecting a variance threshold (see Figure 27b in Appendix D.1) such that at least 70% of items are retained in the analysis, the improved inter-rater reliabilities were 0.3236 for the 5-point task and 0.3519 for the pairwise task, which is the result we include in Table 19 (page 104).

### Linear Mixed Effects Model

Linear Mixed Effects Models (LMEM) can be used to quantify the effect of item- or rater- and interface-based variance on the preference ratings. More formally, a LMEM models the conditional dependency of a response variable $Y$ on fixed effects $X$ and random effects $Z$. Adopting the matrix form notation from Fox (2002), a LMEM is defined as follows:

$$\mathbf{y}_i = X_i\boldsymbol{\beta} + Z_i\mathbf{b}_i + \boldsymbol{\epsilon}_i \tag{73}$$

$$\mathbf{b}_i \sim \mathcal{N}_q(0, \boldsymbol{\Psi}) \tag{74}$$

$$\boldsymbol{\epsilon}_i \sim \mathcal{N}_{n_i}(0, \sigma^2\boldsymbol{\Lambda}_i) \tag{75}$$

$\mathbf{y}_i$ is a column vector containing measured responses (e.g., HTER) for a range of sentences in of group $i$ (e.g., in-domain translations). The design matrix $X_i$ describes fixed effects in a $n_i \times p$ matrix, where $n_i$ is the number of measurements for this group and $p$ is the number of fixed effects. The $n_i \times q$ matrix $Z_i$ similarly describes $q$ random effects. $\boldsymbol{\epsilon}_i$ is a $n_i$-dimensional vector of residuals. $\boldsymbol{\beta}$ is a column vector of fixed effect coefficients (identical for all groups). $\mathbf{b}_i$ is a $q$-dimensional vector of random effect coefficients for group $i$. $\mathbf{b}_i$ and $\boldsymbol{\epsilon}_i$ are modeled as normal multivariate distributions, with co-variance matrices $\boldsymbol{\Psi}$ and $\sigma^2\boldsymbol{\Lambda}_i$, respectively. The co-variance of the random effects is constant for all groups, while the co-variance of the error is specific for one group, and they are both parametrized.

For our collected data, we are interested in a LMEM for preference judgment as response variable, and model the rating interface as fixed effect, and source and rater id as random effects. In this way we can investigate how certain items and raters prefer either of the translation system in each rating mode.

In order to model the rating interface as a fixed effect, both rating schemes have to be made comparable. Therefore, we formulate the preference decision on a continuous scale, with 0 expressing a tie, and 1 and $-1$ the preference of either the in-domain (ID) or out-of-domain (OD) translation (like Green et al. (2014)):

$$\mathrm{pref}(r(y_i^{\mathrm{ID}}), r(y_i^{\mathrm{OD}})) = \begin{cases} 1 & \text{if } r(y_i^{\mathrm{ID}}) > r(y_i^{\mathrm{OD}}) \\ -1 & \text{if } r(y_i^{\mathrm{ID}}) < r(y_i^{\mathrm{OD}}) \\ 0 & \text{otherwise.} \end{cases} \tag{76}$$

The preference decision comparing $y_i^{\mathrm{ID}}$ and $y_i^{\mathrm{OD}}$ is either directly expressed in the pairwise ratings, or inferred from the 5-point ratings (see "re-pairing" in Section 5.2.1). Random effects are items and raters, for which we assume individual intercepts, with global random slopes for the items.[3] The model is trained with restricted maximum likelihood estimation with the `lme4` R library (Bates et al., 2015):

$$\texttt{pref} \sim \texttt{mode} + (1 + \texttt{mode} \mid \texttt{source\_id}) + (1 \mid \texttt{rater\_id}),$$

where `mode` is a binary variable expressing the two rating modes (5-point and pairwise preferences), and `pref` the above described preference of either out-of-domain or in-domain model or a tie.[4]

The fitted model estimates the variance by item to be much larger than the variance across raters (0.038 vs 0.002). The global intercept is at 0.03, that means that translations from the in-domain model are slightly preferred. In the pairwise mode the preference of the in-domain model's translations is significantly lower than in the absolute rating mode (but still higher than the out-of-domain model) (b=-0.40, $p < 0.05$).[5]

Inspecting the individual coefficients, we can find examples that have a strong tendency to be preferred when translated with either the in-domain system or the out-of-domain system: The first two examples in Table 23 (page 119) have an intercept of around 0.4, hence a strong preference for the in-domain translation (important differences underlined, e.g. the translation of "wegblasen" in the first example); while the third and fourth examples have an intercept of around -0.6, hence a strong preference for the out-of-domain translation (for example because of the translation of "normalerweise" or "der Bach").

Individual slopes tell us how much the rating mode affects the preference for individual items. For the first two examples in Table 24 (page 120), the pairwise preference mode had a strong influence on preferring the in-domain translation, since the out-of-domain translation only contains one part of the source, which might be acceptable (the part that is translated is adequate) when judged in isolation, but not when directly compared to a translation that contains both

[3]We do not include random slopes for raters since there is no overlap of raters across modes.
[4]For a practical tutorial on fitting LMEMs we refer the reader to Winter (2013).
[5]Significance tested with the `lmerTest` library.

parts. There are other cases, for example the last two sentences in Table 24, where the mode did not have an influence on the preference, probably because the translations were both similarly bad (e.g. the translation of the name "Frau Drucker" to "woman printer").

Inspecting the intercepts per rater, we find that four raters in the pairwise task on average preferred the out-of-domain translations, while everyone else had a preference for in-domain translations. Interestingly, the rater that had the strongest preference for out-of-domain translations (intercept of -0.04) also had the lowest intra-rater reliability ($\alpha$=0.10). A potential avenue for future work would be to investigate if filtering according to rater intercepts would result in the same beneficial effect for overall reliability that was observed when filtering by variance in the ablation analysis above.

### 5.2.3 Qualitative Analysis

On completion of the rating task, we asked the participants for a subjective judgment of difficulty on a scale from 1 (very difficult) to 10 (very easy). On average, the pairwise rating task (mean 5.69) was perceived slightly easier than the 5-point rating task (mean 4.8). The participants also had to state which aspects of the tasks they found difficult: They reported that the biggest challenge for 5-point ratings was the weighing of different error types and the rating of long sentences with very few, but essential errors. For pairwise ratings, it was most difficult to distinguish between similar, or similarly bad translations. They reported difficulties with decisions for translations with ungrammatical or incomprehensible sources for both modes.

Comparing items with high and low agreement across raters allows us to draw conclusions about objective difficulty. We assume that high inter-rater agreement indicates an ease of judgment, while difficulties in judgment are manifested in low agreement. Table 42 in Appendix C.2 lists low- and high-variance items for 5-star ratings, Table 43 for pairwise ratings. From the annotations in the tables, the reader may get an impression which translations are "easier" to judge than others.

For 5-point ratings, difficulties arise with ungrammatical sources and omissions, whereas obvious mistakes in the target, such as over-literal translations, make judgment easier. Preference judgments tend to be harder when both translations contain errors and are similar. When there is a tie, the pairwise rating framework does not allow to indicate whether both translations are of high or low quality. Since there is no normalization strategy for pairwise ratings, individual biases or rating schemes can hence have a larger negative impact on the inter-rater agreement.

## 5.3 Learnability of a Reward Estimator

After having analyzed the intrinsic quality of the ratings with measures of reliability and item and rater variance, we now assess how well we can predict the ratings from each feedback mode in order to compare their learnability. Learnability plays an important role for downstream RL applications where the reward estimator helps the learning model to explore outputs that were not originally rated (cf. Section 4.3.2 for the experiments with estimators for eBay ratings).

### 5.3.1 Learning a Reward Estimator

The numbers of ratings that can be obtained directly from human raters in a reasonable amount of time is insignificant compared to the millions of sentences used for standard NMT training. By learning a reward estimator on the collection of human ratings, we seek to generalize to unseen translations.

The reward estimator should not require time-consuming feature extraction so it can be deployed in direct interaction with a learning NMT system, estimating rewards on the fly. Most importantly, it should generalize well so it can guide the NMT towards good local optima. Therefore we opt for a neural model that can be fitted to the collected ratings in an end-to-end fashion with stochastic gradient descent and back-propagation.

State-of-the-art models for sentence-level Quality Estimation such as (Martins et al., 2017; Kim et al., 2017; Kepler et al., 2019) are less suitable for the direct use in this task since they rely on linguistic input features, stacked architectures, post-edit or word-level supervision, or learned ensembles. Similar to approaches for generative adversarial NMT (Yu et al., 2017b; Wu et al., 2017) we prefer a convolutional architecture based on word embeddings for reward estimation.[6]

#### Learning from Cardinal Feedback

The inputs to the reward estimation model are sources $x$ and their (model-generated or reference) translations $y$. Given cardinal judgments for these translations, a regression model with parameters $\psi$ is trained to minimize the mean squared error (MSE) between $K$ judgments (human-generated or simulated) $r$ and the predictions by the model $\hat{r}_\psi$:

$$J^{\mathrm{MSE}}(\psi) = \frac{1}{K}\sum_{k=1}^{K}(r(y^{(k)}) - \hat{r}_\psi(y^{(k)}))^2. \tag{77}$$

[6]Contextualized multilingual embeddings from BERT (Devlin et al., 2019) (which appeared after the publication of these experiments) also have a large potential for this task (Kim et al., 2019), which we explore in Chapter 6.

In simulation experiments, where all translations can be compared to existing references, $r$ may be computed by sentence-BLEU (sBLEU) to obtain scores between 0 and 1. For our human 5-point judgments, we first normalize the judgments per rater as described in Section 5.2, then average the judgments across raters and finally scale them linearly to the interval $[0.0, 1.0]$.

**Learning from Pairwise Preference Feedback**

When pairwise preferences are given instead of cardinal judgments, the Bradley-Terry model (Bradley and Terry, 1952) allows us to train a regressor nevertheless. Following Christiano et al. (2017), let $\hat{P}_\psi[y_i \succ y_j]$ be the probability that any translation $y_i$ is preferred over any other translation $y_j$ by the reward estimator:

$$\hat{P}_\psi[y_i \succ y_j] = \frac{\exp(\hat{r}_\psi y_i)}{\exp(\hat{r}_\psi(y_i)) + \exp(\hat{r}_\psi(y_j)}). \tag{78}$$

Let $Q[y_i \succ y_j]$ be the probability that translation $y_i$ is preferred over translation $y_j$ by a gold standard, e.g. the human raters or in comparison to a reference translation. With this supervision signal we formulate a pairwise (PW) training loss for the reward estimation model with parameters $\psi$:

$$\begin{aligned} J^{\mathrm{PW}}(\psi) = -\frac{1}{K}\sum_{k=1}^{K} & Q[y_i^{(k)} \succ y_j^{(k)}] \log \hat{P}_\psi[y_i^{(k)} \succ y_j^{(k)}] \\ & + Q[y_j^{(k)} \succ y_i^{(k)}] \log \hat{P}_\psi[y_j^{(k)} \succ y_i^{(k)}]. \end{aligned} \tag{79}$$

This objective maximizes the likelihood of the correct preference while minimizing the likelihood of the incorrect preference.

When obtaining preference judgments directly from raters, $Q[y_i \succ y_j]$ is simply the empirical frequency of $y_i$ being preferred over $y_j$ by the human raters.

$$Q_{\text{human}}[y_i \succ y_j] = \frac{\text{Count}(y_i \succ y_j)}{R}, \tag{80}$$

where $\text{Count}(y_i \succ y_j)$ counts how many of a total of $R$ raters preferred $y_i$ over $y_j$ in the collected log.

For simulation experiments—where we lack a genuine supervision for preferences—we compute $Q$ comparing the sBLEU scores for both translations, i.e. translation preferences are modeled according to their difference in sBLEU:

$$Q_{\text{simulated}}[y_i \succ y_j] = \frac{\exp(\text{sBLEU}(y_i))}{\exp(\text{sBLEU}(y_i)) + \exp(\text{sBLEU}(y_j))}. \tag{81}$$

### 5.3.2 Experiments

#### Data Augmentation

The collected ratings for 800 translations (Section 5.1.1) are used to train neural regression models and pairwise preference models. In addition, we train models on simulated rewards (sBLEU) for a comparison with arguably "clean" feedback for the same set of translations.

In order to augment the small collection of human-rated translations, we leverage the available out-of-domain bitext as auxiliary training data: 10k randomly chosen source sentences of WMT (out-of-domain) are translated by the out-of-domain model. Translations from nine beam search ranks are compared to their references to compute sBLEU rewards. This auxiliary data provides 90k out-of-domain training samples with sBLEU reward.

For pairwise rewards, sBLEU scores for two translations for the same source are compared. Each mini-batch during training is sampled from the auxiliary data with probability $p_{aux}$, from the original training data with probability $1 - p_{aux}$. Adding this auxiliary data as a regularization through multi-task learning prevents the model from overfitting to the small set of human ratings, and makes sure it is still aligned with BLEU evaluation in the end. This was a problem in the experiments on eBay user ratings, where the learned estimator behaved very differently than BLEU (Section 4.3.2). $p_{aux} = 0.8$ worked best in our experiments.

#### Model Architecture

We build a convolutional reward estimation, similar to the one presented in Section 4.3.1: Input source and target sequence are split into the BPE subwords used for NMT training, padded up to a maximum length of 100 tokens, and represented as 500-dimensional subword embeddings. Subword embeddings are pre-trained on the WMT bitext with `word2vec` (Mikolov et al., 2013a), normalized to unit length and held constant during further training. Additional 10-dimensional BPE-feature embeddings are appended to the subword embeddings, where a binary indicator encodes whether each subword contains the subword prefix marker "@@". BPE-prefix features are useful information for the model since bad translations can arise from invalid compositions of subwords.

The embeddings are then fed to a source-side and a target-side bidirectional LSTM (biLSTM) (Hochreiter and Schmidhuber, 1997), respectively. The biLSTM outputs are concatenated for each time step and fed to a 1-D convolutional layer with 50 filters each for filter sizes from 2 to 15. The convolution is followed by max-over-time pooling, producing 700 input features for a fully-connected output layer with leaky ReLU (Maas et al., 2013) activation function. Dropout (Srivastava et al., 2014b) with $p = 0.5$ is applied before the final layer.

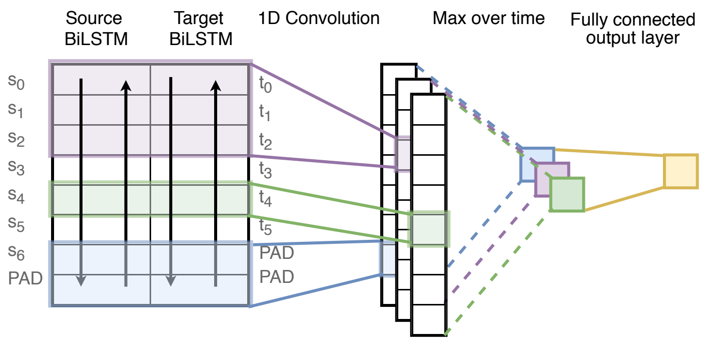


Figure 13: Neural architecture for the reward estimator: Source and target sentences are padded and embedded, read by a biLSTM, then concatenated and fed into convolutional, pooling and fully connected layers.

This architecture, depicted in Figure 13 can be seen as a biLSTM-enhanced bilingual extension to the convolutional model for sentence classification proposed by Kim (2014) and is slightly more advanced than the reward estimator used in Section 4.3.1 by using the biLSTM before the convolution, which supplies the CNN with contextualized embeddings, rather than static ones. In general, this architecture has the advantage of not requiring any feature extraction but still modeling n-gram features on an abstract level.

The quality of the reward estimation models is tested by measuring Spearman's rank correlation $\rho$ against TER on a held-out test set of 1,314 translations, following the standard in sentence-level quality estimation evaluations.[7] Hyper-parameters are tuned on another 1,200 TED translations.

## Results

Table 20 (page 113) reports the results of reward estimators trained on simulated and human rewards. When trained from cardinal rewards, the model of simulated scores performs slightly better than the model of human ratings. This advantage is lost when moving to preference judgments, which might be explained by the fact that the softmax over sBLEUs (Equation 81) with respect to a single reference is just not as expressive as the preference probabilities obtained from several raters.

Intuitively, the information contained in one preference rating is used to infer the absolute quality of two translations, as opposed to only one in the Likert-scale rating task. Hence, it is not surprising that the estimators based on the pairwise ratings are weaker than the five-point ratings, but the gap between

[7]In sQE it is rather HTER than TER since the sentence-level quality score is estimated from post-edits of the system outputs.

| **Model** | **Feedback** | $\rho$ |
|---|---|---|
| MSE | simulated | -0.2571 |
| PW | simulated | -0.1307 |
| MSE | human | -0.2193 |
| PW | human | -0.1310 |
| MSE | human, filtered by rater | -0.2341 |
| PW | human, filtered by item | -0.1255 |

Table 20: Spearman's rank correlation $\rho$ between estimated rewards and TER for models trained with simulated rewards (sBLEU) and human rewards (five-point and pairwise ratings, also filtered subsets).

those is rather large. Filtering by participants (see Section 5.2.1) improves the correlation further for cardinal rewards, but slightly decreases it for preference judgments.

The overall correlation scores are relatively low—in particular for the PW models—which we suspect is due to overfitting to the small set of training data. Figure 14 (page 114) compares histograms of estimated (test split) and collected rewards (train split) for both simulated and human ratings. One can see that the Bradley-Terry model (Equation 78) struggles to model fine-grained differences. From these experiments we conclude that when it comes to estimating translation quality regressors, cardinal human judgments exhibit better learnability than pairwise preference judgments.

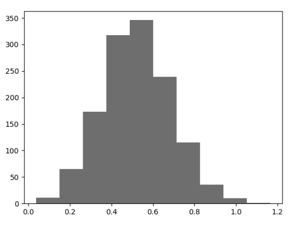
## 5.4 Reinforcing MT with Direct and Estimated Rewards

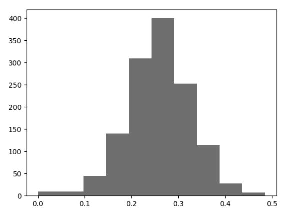
### 5.4.1 Training Objectives

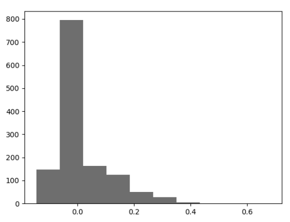
**Online estimated or simulated rewards.** Deploying NMT in a reinforcement learning scenario, the goal is to maximize the expectation of a reward $r$ over all source and target sequences (Equation 30). We extend it further to the multi-sample approximation by Wu et al. (2016):

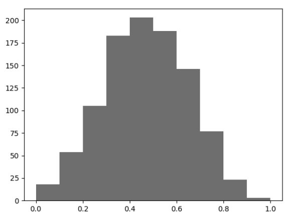
$$\begin{aligned} J^{\mathrm{RL}}(\theta) =& \mathbb{E}_{p(x)p_\theta^\tau(y|x)}\left[r(y)\right] \\ \approx& \mathbb{E}_{p(x)}\left[\sum_{y' \in \mathcal{S}} p_\theta^\tau(y' \mid x) r(y')\right] \end{aligned} \tag{82}$$

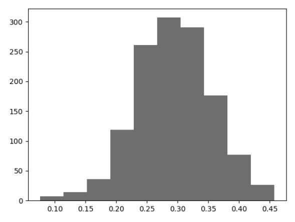
The expectation over outputs is approximated with $k$ samples from a subset of translations $S \subset \mathcal{Y}(x)$ drawn from the model distribution, but the set of samples


(a) TER scores (test)


(b) MSE: sBLEU (test)


(c) PW: sBLEU (test)


(d) Avg. user ratings (train)


(e) MSE: user ratings (test)

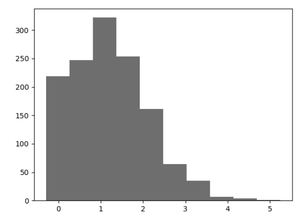

(f) PW: user ratings (test)

Figure 14: Comparison of estimated simulated and user ratings. “MSE”: trained with the MSE objective (77), “PW”: (79).

$\mathcal{S}$ is much smaller than the actual output space to keep the gradient computation feasibly fast, just as in MRT (Equation 64). In contrast to MRT there is no re-normalization over the sample probabilities. When $k = 1$, this is equivalent to the expected reward objective from Equation (30) for bandit feedback.

However, when feedback for more than one output per input is obtainable (e.g. when working in simulations, or with paid annotators instead of users), $k$ can be increased accordingly. In this work, we have non-bandit learning opportunities, when the reward $r$ either comes from a reward estimation model (estimated direct reward) or is computed with respect to a reference in simulation (simulated direct reward).

In order to counteract high variance in the gradient updates, the running average of rewards is subtracted from $r$ for learning (see Section 3.3). Adding a temperature hyper-parameter $\tau \in (0.0, \infty]$ to the softmax (Equation 7) over the model output vector $\mathbf{o}$ allows us to control the sharpness of the sampling distribution, i.e., the amount of exploration during training:

$$p_\theta^\tau(y \mid x) = \text{softmax}(\mathbf{o}/\tau). \tag{83}$$

With temperature $\tau < 1$, the model's entropy decreases and it samples closer to the one-best output. Exploration should be kept low to prevent the NMT to produce samples that lie far outside the training domain of the reward estimator.

**Offline direct rewards.** When rewards can not be obtained for samples from a learning system, but were collected for a static deterministic system (e.g. the eBay use case presented in Chapter 4), we are in an off-policy learning scenario. The challenge is to improve the MT system from a log $L = \{(x^{(h)}, y^{(h)}, r(y^{(h)}))\}_{h=1}^{H}$ of rewarded translations. We can hence directly apply the DPM objective from Equation (69). In contrast to the RL objective, only logged translations are reinforced, i.e. there is no exploration in learning. Also there is no exclusion of translations with low ratings: They will also get reinforced, but with a smaller reward.

### 5.4.2 Experiments

**Data.** We use the WMT 2017 data[8] for training a general domain model for translations from German to English. The training data contains 5.9M sentence pairs, the development data 2,999 sentences (WMT 2016 test set) and the test data 3,004 sentences. For in-domain data, we choose the translations of TED talks[9] as used in IWSLT evaluation campaigns. The training data contains 153k, the development data 6,969, and the test data 6,750 parallel sentences.

**Architecture.** Our NMT model is a standard encoder-decoder architecture with attention (Bahdanau et al., 2015) (see Section 2.1.3). We implemented the RL and DPM objectives in Neural Monkey (Helcl and Libovický, 2017).[10] The NMT has a bidirectional encoder and a single-layer decoder with 1,024 GRUs each, and subword embeddings of size 500 for 30k shared byte-pair merges (Sennrich et al., 2016c). Maximum input and output sequence length are set to 60. For model selection we use greedy decoding, for test set evaluation beam search with a beam of width ten. We sample $k = 5$ translations for RL models and set the softmax temperature $\tau = 0.5$. Training hyperparameters are given in Appendix Section B.4.

**Evaluation.** Trained models are evaluated with respect to BLEU (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2011) using `MultEval` (Clark et al., 2011) and BEER (Stanojević and Sima'an, 2014) to cover a diverse set of automatic measures for translation quality (see Section 2.1.1).[11] We test for statistical significance with Approximate Randomization (Noreen, 1989).

[8]Pre-processed data available at `http://www.statmt.org/wmt17/translation-task.html`.

[9]Pre-processing and data splits as described in `https://github.com/rizar/actor-critic-public/tree/master/exp/ted`.

[10]The code is available in the Neural Monkey fork `https://github.com/juliakreutzer/bandit-neuralmonkey/tree/acl2018`.

[11]Tendencies of improvement turn out to be consistent across metrics, so we only discuss BLEU in the text.

| | WMT | | | TED | | |
|---|---|---|---|---|---|---|
| **Model** | **BLEU** | **METEOR** | **BEER** | **BLEU** | **METEOR** | **BEER** |
| WMT | 27.2 | 31.8 | 60.08 | 27.0 | 30.7 | 59.48 |
| TED | 26.3 | 31.3 | 59.49 | 34.3 | 34.6 | 64.94 |

Table 21: Results on test data for in- and out-of-domain *fully-supervised* models. Both are trained with MLE, the TED model is obtained by fine-tuning the WMT model on TED data.

**Baselines.** The out-of-domain model is trained with MLE on WMT. The task is now to improve the generalization of this model to the TED domain. Table 21 compares the out-of-domain baseline with domain-adapted models that were further trained on TED in a fully-supervised manner supervised fine-tuning as introduced by Freitag and Al-Onaizan (2016); Luong and Manning (2015). The supervised domain-adapted model serves as an upper bound for domain adaptation with human rewards: if we had references, we could improve up to 7 BLEU.

**RL from simulated rewards.** First we simulate "clean" and deterministic rewards by comparing sample translations to references using GLEU (Wu et al., 2016) for RL, and smoothed sBLEU for estimated rewards and DPM.[12] Table 22 (page 117) lists the results for this simulation experiment in rows 2–5 (S). If unlimited clean feedback was given (RL with direct simulated rewards), improvements of over 5 BLEU can be achieved. When limiting the amount of feedback to a log of 800 translations, the improvements over the baseline are only marginal (DPM). When replacing the direct reward by the simulated reward estimators from Section 5.3, i.e. having unlimited amounts of approximately clean rewards, however, improvements of 1.2 BLEU for MSE estimators (RL+MSE) and 0.8 BLEU for pairwise estimators (RL+PW) are found. This suggests that the reward estimation model helps to tackle the challenge of generalization over a small set of ratings.

**RL from human rewards.** Knowing what to expect in an ideal setting with clean feedback, we now move to the experiments with human feedback. DPM is trained with the logged normalized, averaged and re-scaled human reward. RL is trained with the direct reward provided by the reward estimators trained on human rewards from Section 5.3. Table 22 shows the results for training with human rewards in rows 6–8: The improvements for DPM are very similar to

[12]We use GLEU for RL because of the recommendation by Wu et al. (2016), but sBLEU since it was used in previous works on counterfactual learning for MT (Lawrence et al., 2017b).

| Model | Rewards | | BLEU | METEOR | BEER |
|---|---|---|---|---|---|
| Baseline | - | - | 27.0 | 30.7 | 59.48 |
| RL | D | S | $32.5^{\star}_{\pm 0.01}$ | $33.7^{\star}_{\pm 0.01}$ | $63.47^{\star}_{\pm 0.10}$ |
| DPM | D | S | $27.5^{\star}$ | $30.9^{\star}$ | $59.62^{\star}$ |
| RL+MSE | E | S | $28.2^{\star}_{\pm 0.09}$ | $31.6^{\star}_{\pm 0.04}$ | $60.23^{\star}_{\pm 0.14}$ |
| RL+PW | E | S | $27.8^{\star}_{\pm 0.01}$ | $31.2^{\star}_{\pm 0.01}$ | $59.83^{\star}_{\pm 0.04}$ |
| DPM | D | H | $27.5^{\star}$ | $30.9^{\star}$ | $59.72^{\star}$ |
| RL+MSE | E | H | $28.1^{\star}_{\pm 0.01}$ | $31.5^{\star}_{\pm 0.01}$ | $60.21^{\star}_{\pm 0.12}$ |
| RL+PW | E | H | $27.8^{\star}_{\pm 0.09}$ | $31.3^{\star}_{\pm 0.09}$ | $59.88^{\star}_{\pm 0.23}$ |
| RL+MSE | E | F | $28.1^{\star}_{\pm 0.20}$ | $31.6^{\star}_{\pm 0.10}$ | $60.29^{\star}_{\pm 0.13}$ |

Table 22: Results on TED test data for training with *estimated* (E) and *direct* (D) rewards from *simulation* (S), *humans* (H) and *filtered* (F) human ratings. Significant ($p \leq 0.05$) differences to the baseline are marked with $^{\star}$. For RL experiments we show three runs with different random seeds, mean and standard deviation in subscript.

DPM with simulated rewards, both suffering from overfitting. For RL we observe that the MSE-based reward estimator (RL+MSE) leads to significantly higher improvements than the pairwise reward estimator (RL+PW)—the same trend as for simulated ratings. Finally, the improvement of 1.1 BLEU over the baseline demonstrates that NMT can be improved with only a small number of human rewards. Learning from estimated filtered 5-point ratings does not significantly improve these results, since the improvement of the reward estimator is only marginal (see Section 5.3).

## 5.5 Conclusion

In this chapter, we sought to answer the question of how cardinal and ordinal feedback differ in terms of reliability, learnability and effectiveness for RL training of NMT. Our rating study, comparing 5-point and preference ratings, showed that their reliability is comparable, whilst cardinal ratings are easier to learn and to generalize from, and also more suitable for RL in our experiments. Through the user study, the ablation study, and variance analysis we now understand better why pairwise ratings for structured outputs like machine translation are difficult. We learned that translations on a similar level of quality but with different types of errors are hard to rank, and that the reliability of the ratings largely depends on the selection of pairs of translations.

Furthermore, our experiments show that improvements of over 1 BLEU are achievable by learning reward estimators from a data set that is tiny in comparison to standard training corpora for NMT and in comparison to the previous study on large-scale eBay user feedback (see Chapter 4). In contrast to the less successful attempt to learn from explicit feedback in the eBay study, we now proved that with more control over users and items and less resulting noise in the feedback collection, RL methods can successfully be employed in off-policy learning of NMT models even with direct human feedback.

While we found evidence that more reliable feedback can lead to higher down-stream improvements, the question about the general relation of reliability and down-stream improvements remains open. Just like us, Aziz et al. (2013) observed quality scores useful for a down-stream task despite low reliability. Even if ratings were reliable, they might not be accurate and could still mislead the learning system. It is yet to be evaluated how much uncertainty and variance in translation quality judgments is actually inherent in the task definition, and how the models can account for it.

| | |
|---|---|
| **Src** | Man muss das Geräusch des Winds machen, sie wegblasen und den Rest des Buchs lesen. |
| **OD** | You must make the noise of the wind, go away and read the rest of the book. |
| **ID** | You have to make the sound of the wind, blow it away, and read the rest of the book. |
| **Src** | Es ist ebenso einfacher wenn man es in Eukaryonten wie wir es sind tut: man nimmt einfach den Nukleus raus und gibt einen anderen hinein, und das ist genau das was Sie alle über das Klonen gehört haben. |
| **OD** | It's just as easy if you are in eukarypes like we are doing it: You simply take the nucleus out and there is another way, and that is exactly what you all heard about the cloning. |
| **ID** | It's also easier if you do it in eukarytes like we are: You just take the nucleus out and you put in another one, and that's exactly what you all heard about cloning. |
| **Src** | Ich habe die letzten paar Jahre damit verbracht, mich in Situationen zu platzieren, die normalerweise sehr schwierig und gleichzeitig irgendwie gefährlich sind. |
| **OD** | I spent the last couple of years trying to place me in situations that usually are very difficult and at the same time dangerous. |
| **ID** | I've spent the last few years placing myself in situations that are normally very difficult and at the same time dangerous. |
| **Src** | Ich denke, dass der Bach begeistert gewesen wäre, wenn er die klanglichen Möglichkeiten der heutigen modernen Klaviere gehabt hätte. |
| **OD** | I think that the bach would have been enthusiastic if he had had the sound possibilities of today's modern piano. |
| **ID** | I think the stream would have been thrilled if it had had the sound of today's modern piano. |

Table 23: Examples for in- (ID) and out-of-domain (OD) translations for selected source sentences (selection by LMEM intercept). For the first two, the in-domain translations were preferred by the raters, while for the latter ones they preferred out-of-domain translations.

| | |
|---|---|
| **Src** | Ich habe einen Plan dafür. Ich möchte mehr Menschen überzeugen, einschließlich Ihnen allen, mehr Zeit mit dem Spielen größerer und tollerer Spiele zu verbringen. |
| **OD** | I have a plan for it. |
| **ID** | I have a plan for that. I want to convince more people, including all of you, spend more time playing bigger and more fun games. |
| **Src** | Dann ist der Umfang der semantischen Querverbindung und die Fülle, die sich daraus ergibt, wirklich enorm. Es ist ein klassischer Netzwerkeffekt. |
| **OD** | Then the scope of semantic cross-connections and the abundance that result from it really enormous. |
| **ID** | Then the scope of semantic cross-connection and the abundance that comes out of it is really enormous. It's a classic network effect. |
| **Src** | Ich residierte an einem Lehrkrankenhaus hier in Toronto als Frau Drucker zu mir in die Notfallabteilung des Krankenhauses gebracht wurde. |
| **OD** | I resisted at a hospital here in toronto as a woman printer to me in the emergency department of the hospital. |
| **ID** | I was residing at a teacher hospital here in toronto as a woman printer to me in the emergency department of the hospital. |
| **Src** | Also nicht mehr sparen, verzichten, vermeiden, reduzieren, kein Schuldmanagement, sondern intelligente Verschwendung. |
| **OD** | So no more save, forego, avoid, reduce, no debt management, but intelligent waste. |
| **ID** | So not saving, forgiving, avoiding, reducing, not guilt management, but smart waste. |

Table 24: Examples for in- and out-of-domain translations for selected source sentences (selection by LMEM slope). For the first two, the rating mode had a large influence to choose the in-domain examples, while for the latter it did not have an influence.

# Chapter 6

# Learning from Error Corrections and Markings

In previous chapters we evaluated approaches for learning from sequence-level feedback, for example the simulated feedback in the form of sBLEU scores (Section 3.4.2), or the star ratings from the eBay platform (Section 4.1.1) or from pairwise preferences and five-point ratings in our own annotation study (Section 5.1). We argued previously that this weak feedback is easier to obtain than full corrections or translations from scratch in the sense that it requires less expertise (collecting it from users, i.e., translation consumers, instead of experts, i.e., translation producers), less time and effort. Yet this advantage has not been quantified—how much less costly is weak feedback compared to full feedback? This chapter precisely quantifies the differences in time and effort for weak supervision vs. full supervision provided by human annotators.

User studies on machine learnability from machine translation post-edits, together with thorough performance analyses with mixed effects models, have been presented by Green et al. (2014); Bentivogli et al. (2016); Karimova et al. (2018).[1] Albeit showcasing the potential of improving NMT through human corrections of machine-generated outputs, these works do not consider "weaker" annotation modes like error markings. Unlike previous experiments with weak feedback on the sequence level, we move to weak feedback on the token level, motivated by the promising results from token-level implicit feedback on the eBay domain (Section 4.1.2) and related work on token-level weights for NMT (Marie and Max, 2015; Domingo et al., 2017; Petrushkov et al., 2018; Yan et al., 2018; Lam et al., 2019; Jehl et al., 2019). In human-in-the-loop learning this can be realized by asking the human to mark errors in machine outputs, for example erroneous words or phrases in a machine translation.

Our approach takes the middle ground between supervised learning from error corrections as in post-editing[2] (or from translations created from scratch) and reinforcement learning from sequence-level bandit feedback (including self-supervised learning with uniform weights for all tokens). In contrast to sequence-level feedback, error markings offer precise token-level credit/blame assignment

[1]User studies on the process and effort of machine translation post-editing are too numerous to list — a comprehensive overview is given in (Koponen, 2016).

[2]We use the terms "error corrections" and "post-edits" interchangeably for the task of machine translation.

(discussed in Section 2.3.1), are thus more interpretable thanks to the grounding in individual tokens (which might be beneficial for filtering or data cleaning), and still have comparatively low annotation cost. Our experiments show that this type of feedback can lead to an effective fine-grained discriminative signal for machine translation.

Prior work closest to ours is that of Marie and Max (2015); Domingo et al. (2017); Petrushkov et al. (2018); Yan et al. (2018), however, these works were conducted by simulating error markings with an heuristic matching of machine translations against independently created human reference translations. The question of the practical feasibility of machine learning from error markings, and the influence of noise and variance on the quality of the annotations, is left open.

To answer these questions, we conduct a user study on the collection and analysis of error corrections and error markings from junior professional translators (we call them "semi-professional"), which is the first to investigate this type of feedback outside simulations. Moreover, we leverage the markings as training signal for fine-tuning neural machine translation systems. We find that error markings require significantly less effort (in terms of keystroke mouse action ratio and time) and result in a lower correction rate (ratio of words marked as incorrect or corrected in a post-edit) and are chosen more frequently than corrections when users are given the choice between annotation modes. Despite lower inter-annotator agreement for error markings than error corrections, fine-tuning of neural machine translation has been conducted successfully from either signal. Furthermore, we show how to learn automatic error correction and marking models, and achieve best results by fine-tuning a Transformer-based error correction models on small amounts of human error corrections.

**Contributions.** The core contributions of this chapter are the following:

1. We conduct a first study of learning from markings, i.e., error highlights, that were collected from human annotators. This interaction mode has previously only been investigated in simulation. However, we can confirm its success in practice.

2. We quantify the reliability, characteristics and suitability of markings for adapting a machine translation system in comparison to post-editing, the prevalent mode of feedback collection in practice.

3. In contrast to the weak feedback collected in the previous real-world studies, this type of feedback naturally passes the task of credit or blame assignment to the human, instead of leaving it to the machine to learn, and at the same gives humans fewer degrees of freedom than in corrective feedback. We show that this type of feedback is several magnitudes cheaper to collect than

post-edits but still allows to improve the underlying machine translation system.

4. The collected data includes detailed logging records and may serve for future investigations of human effort in post-editing or error marking.

**Publications.** The study reported in this chapter is published in (Kreutzer et al., 2020). The second author of the paper contributed the implementation of the annotation interface, the assignment of annotation tasks to annotators, the computation of reliability scores, and the implementation of the automatic marking module. All NMT experiments, the implementation of the objectives and the training of the automatic corrector module, and the annotation analysis were conducted by the author of this thesis.

**Outline.** We start with a description and analysis of the user study in Section 6.1. Section 6.2 subsequently presents training objectives for improving NMT with corrections and error markings. These objectives are evaluated in Section 6.3. Section 6.4 concludes with a summary of the findings of this chapter.

## 6.1 Annotation Study

The goal of the annotation study is to compare the novel error marking mode to the widely adopted machine translation post-editing mode. We are interested in finding an interaction scenario that costs little time and effort, but still allows to teach the machine how to improve its translations. In this section we present the setup, measure and compare the observed amount of effort and time that went into these annotations, and discuss the reliability and adoption of the new marking mode. Machine learnability is discussed in Section 6.2.

### 6.1.1 Setup

**Participants.** We recruited ten participants that described themselves as native German speakers and having either a C1 or C2 level in English, as measured by the Common European Framework of Reference levels. Eight participants were students studying translation or interpretation and two participants were students studying computational linguistics. All participants were paid 100€ for their participation in the study, which was done online, and limited to a maximum of six hours, and it took them 2–4.5 hours excluding breaks to complete. They agreed to the usage of the recorded data for research purposes.

**Interface.** The annotation interface has three modes: (1) markings, (2) corrections, and (3) the user-choice mode, where annotators first choose between (1) and (2) before submitting their annotation. In any case, annotators are presented the source sentence, the target sentence and an instruction to either mark or correct (aka post-edit) the translation or choose an editing mode. They also had the option to pause and resume the session. No document-level context was presented, i.e., translations were judged in isolation, but in consecutive order like they appeared in the original documents. They received detailed instructions (see Appendix E.1) on how to proceed with the annotation. Each annotator worked on 300 sentences, 100 for each mode, and an extra 15 sentences for intra-annotator agreement measures that were repeated after each mode. After the completion of the annotation task they answered a survey about the preferred mode, the perceived editing/marking speed, user-choice policies, and suggestions for improvement.

**Data.** We selected a subset of 30 TED talks to create the three data sets from the IWSLT17 machine translation training corpus for English to German.[3] The talks were filtered by the following criteria: containing only a single speaker, no music/singing, low intra-line final-sentence punctuation (indicating bad segmentation), length between 80 and 149 sentences. One additional short talk was selected for testing the inter- and intra-annotator reliability. We filtered out those sentences where model hypothesis and references were equal, in order to save annotation effort where it is clearly not needed, and also removed the last line from every talk (usually "thank you"). For each talk, one topic of a set of keywords provided by TED was selected.

### 6.1.2 Analysis

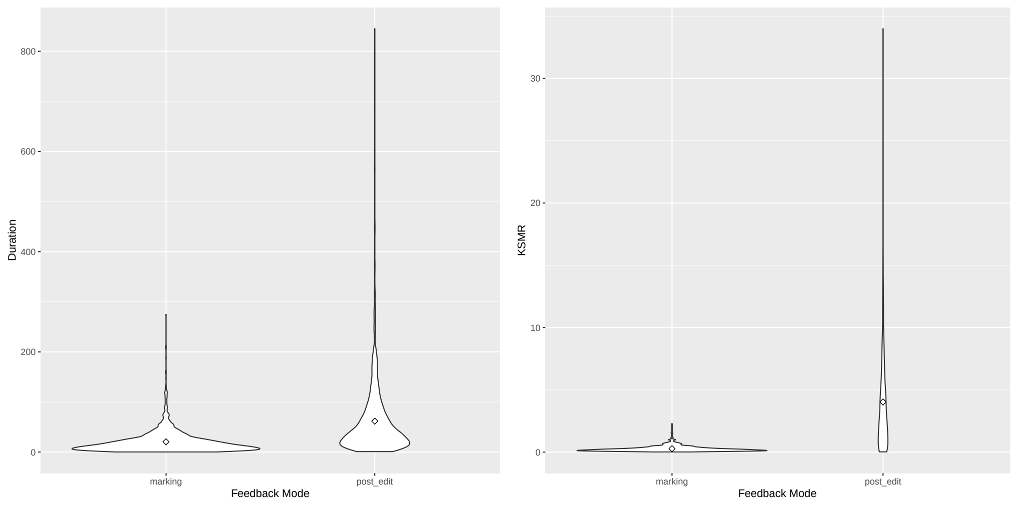
In the following analysis we compare markings and corrections with the focus on three aspects:

1. How much effort and time do markings or corrections take?
2. How do they differ in terms of quality?
3. Do they express the same thing? And which of them is preferable?

#### Effort and Time

Correcting one translated sentence took on average approximately 5 times longer than marking errors, and required 42 more actions, i.e., clicks and keystrokes. That is 0.6 actions per character for post-edits, while only 0.03 actions per

[3]`https://sites.google.com/site/iwsltevaluation2017/`



(a) Edit duration per sentence (seconds). (b) Editing effort per sentence (KSMR).

Figure 15: Duration and effort per sentence for both feedback modes: error markings (left) and error corrections (post-edits, right). Means are marked with diamonds.

character for markings. This measurement aligns with the unanimous subjective impression of the participants that they were faster in marking mode. Figures 15a and 15b compare the edit duration per sentence and the Keystroke Mouse Action Ratio (KSMR) across modes, respectively. There is much more variation in time and effort needed for post-editing, and mean values are higher than for marking.

To investigate the sources of variance affecting time and effort, we train a LMEM for KSMR and total edit duration (excluding breaks) as respective response variables, and with the editing mode (correcting vs. marking) as fixed effect. For both response variables, we model users, talks and target lengths[4] as random effects, here e.g. the one for KSMR:

$$\texttt{KSMR} \sim \texttt{mode} + (1 \mid \texttt{user\_id}) + (1 \mid \texttt{talk\_id}) + (1 \mid \texttt{trg\_length}) \tag{84}$$

We use the implementation in the R package `lmer4` (Bates et al., 2015) and fit the models with residualized maximum likelihood. Inspecting the intercepts of the fitted models, we confirm that KSMR is significantly ($p = 0.01$) higher for post edits than for markings (+3.76 on average). The variance due to the user (0.69) is larger than due to the talk (0.54) and the length (0.05)[5]. Longer

[4]Target lengths measured by number of characters were binned into two groups at the limit of 176 characters.

[5]Note that KSMR is already normalized by reference length, hence the small effect of target

sentences have a slightly higher KSMR than shorter ones. When modeling the topics as random effects (rather than the talks), the highest KSMR (judging by individual intercepts) was obtained for physics and biodiversity and the lowest for language and diseases. This might be explained by e.g. the MT training data or the raters expertise.

Analyzing the LMEM for editing duration, we find that post-editing takes on average 42s longer than marking, which is significant at $p = 0.01$. The variance due to the target length is the largest, followed by the one due to the talk and the one due to the user is smallest. Long sentences have a six time higher editing duration on average than shorter ones. When modeling by topic instead of talk, the longest editing was done for topics like physics and evolution, shortest for diseases and health.

### Annotation Quality

The error corrections increased the quality by 2.1 points in BLEU and 1 point in TER (measured against the reference). While this indicates a general improvement, it has to be taken with a grain of salt, since the error corrections are heavily biased by the structure, word choice etc. by the machine translation, which might not necessarily agree with the translations, while still being accurate.

**Quality of error corrections.** We therefore manually inspect the post-edits to get insights into the differences between post-edits and references. Table 25 (page 127) provides a set of examples[6] with their analysis in the caption. Besides the effect of "literalness" (Koponen, 2016), we observe three major problems:

1. *Over-editing*: Editors edited translations even though they are adequate and fluent.
2. *"Telephone game" effect*: Semantic mistakes introduced by the MT system flow into the post-edit and remain uncorrected, when more obvious corrections are needed elsewhere in the sentence. Guerberof Arenas (2008) explain this by the lack of attention to detail of post-editors when they are presented with fluent and superficially fine looking translations.
3. *Missing information*: Since editors only observe a portion of the complete context, i.e., they do not see the video recording of the speaker or the full transcript of the talk, they are not able to convey as much information as the reference translations.

length. In a LMEM for the raw action count (clicks+key strokes), this effect had a larger impact.

[6]Selected because of their differences to references.

| | |
|---|---|
| **Source** | I am a nomadic artist. |
| **Hypothesis** | Ich bin ein nomadischer Künstler. |
| **Correction** | Ich bin ein nomadischer Künstler. |
| **Reference** | Ich wurde zu einer nomadischen Künstlerin. |
| **Source** | I look at the chemistry of the ocean today. |
| **Hypothesis** | Ich betrachte heute die Chemie des Ozeans. |
| **Correction** | Ich erforsche täglich die Chemie der Meere. |
| **Reference** | Ich untersuche die Chemie der Meere der Gegenwart. |
| **Source** | There's even a software called cadnano that allow ... |
| **Hypothesis** | Es gibt sogar eine Software namens Caboano, die ... |
| **Correction** | Es gibt sogar eine Software namens Caboano, die ... |
| **Reference** | Es gibt sogar eine Software namens "cadnano", ... |
| **Source** | It was a thick forest. |
| **Hypothesis** | Es war ein dicker Wald. |
| **Correction** | Es handelte sich um einen dichten Wald. |
| **Reference** | Auf der Insel war dichter Wald. |

Table 25: Examples of post-editing of model hypotheses to illustrate differences between reference translations and corrections/post-edits. Example 1: The gender in the German translation could not be inferred from the context, since speaker information is unavailable to the post-editor. Example 2: "today" is interpreted as adverb by the NMT, this interpretation is kept in the correction ("telephone game" effect). Example 3: Another case of the "telephone game" effect: the name of the software is changed by the NMT, and not corrected by the annotators. Example 4: Over-editing, and more information in the reference translation than in the source.

**Quality of error markings.** Markings, in contrast, are less prone to over-editing, since they have fewer degrees of freedom. They are equally exposed to problem (3) of missing context, and another limitation is added: Word omissions and word order problems cannot be annotated. Table 26 (page 128 gives a set of examples that illustrate these problems. While annotators were most likely not aware of problems (1) and (2), they might have sensed that information was missing, as well as the additional limitations of markings. Furthermore, we see that the simulation of markings from references as used in previous work (Petrushkov et al., 2018; Marie and Max, 2015) can be overly harsh for the generated target translations, e.g., marking "Hazara-Bevölkerung" as incorrect, even though it is a valid translation of "Hazara population".

| | |
|---|---|
| **Source** | Each year, it sends up a new generation of shoots. |
| **Annotated Marking** | Jedes Jahr sendet es eine neue Generation von Shoots. |
| **Simulated Marking** | Jedes Jahr sendet es eine neue Generation von Shoots. |
| **Reference** | Jedes Jahr wachsen neue Triebe. |
| **Source** | He killed 63 percent of the Hazara population. |
| **Annotated Marking** | Er starb 63 Prozent der Bevölkerung Hazara. |
| **Simulated Marking** | Er starb 63 Prozent der Bevölkerung Hazara. |
| **Reference** | Er tötete 63% der Hazara-Bevölkerung. |
| **Source** | They would ordinarily support fish and other wildlife. |
| **Annotated Marking** | Sie würden Fisch und andere wild lebende Tiere unterstützen. |
| **Simulated Marking** | Sie würden Fisch und andere wild lebende Tiere unterstützen. |
| **Reference** | Normalerweise würden sie Fisch und andere Wildtiere ernähren. |

Table 26: Examples of markings to illustrate differences between human markings and simulated markings. Marked parts are underlined. Example 1: "es" not clear from context, less literal reference translation. Example 2: Word omission (preposition after "Bevölkerung") or incorrect word order is not possible to mark. Example 3: Word order differs between MT and references, word omission ("ordinarily") not marked.

**Inter-agreement for corrections and markings.** In addition to the absolute quality of the annotations, we are interested in measuring their reliability: Do annotators agree on which parts of a translation to mark or edit? While there are many possible valid translations, and hence many ways to annotate one given translation, it has been shown that learnability profits from annotations with less conflicting information (Chapter 5). In order to quantify agreement for both modes on the same scale, we reduce both annotations to sentence-level quality judgments. For markings it is the ratio of words that were marked as incorrect in a sentence, and for corrections the ratio of words in the translation that was actually edited. If the hypothesis was perfect, no markings nor edits would be required, and if it was completely wrong, all of it had to be marked or edited. After this reduction, we measure agreement with Krippendorff's $\alpha$ (Krippendorff, 2013). Table 27 shows that corrections have a consistently higher reliability. However, this reliability metric does not capture whether annotators

| Mode | Intra-Rater $\alpha$ | | Inter-Rater $\alpha$ |
|---|---|---|---|
| | Mean | Std. | |
| Marking | 0.522 | 0.284 | 0.201 |
| Correction | 0.820 | 0.171 | 0.542 |
| User-Chosen | 0.775 | 0.179 | 0.473 |

Table 27: Intra- and Inter-rater agreement for markings, corrections and the user-chosen combination of both, calculated by Krippendorff's $\alpha$.

agreed on their edit actions, but only on the portion on the translation that need editing, which omits insertions. The overall editing ratio is also much higher for post-edits than for markings, so there is naturally less variance (see Section 6.1.2). While these agreement scores seem low, they are—just like the ones reported in Section 5.2.1—within the expected range (see also the discussion on the agreement of error annotation by Lommel et al. (2014)).

### Direct Comparison of Marking

**Empirical preference.** In the user-choice mode, where annotators can choose for each sentence whether they would like to mark or correct it, markings were chosen much more frequently than post-edits (61.9%). Annotators did not agree on the preferred choice of mode for the repeated sentences ($\alpha = -0.008$), which indicates that there is no obvious policy when one of the modes would be advantageous over the other. In the post-annotation questionnaire, however, 60% of the participants said they generally preferred post-edits over markings, despite markings being faster and hence resulting in a higher hourly pay.

**Subjective preference.** To better understand the differences in modes, we asked them about their policies in the user-choice mode. The most commonly described policy is to decide economically based on error types and frequency: choose post-edits when insertions or re-ordering is needed, and markings preferably for translations with word errors (less effort than doing a lookup or replacement). One person preferred post-edits for short translations, markings for longer ones, another three generally preferred markings for any sentence, while one person generally preferred post-edits. Where annotators found the interface to need improvements was (1) in the presentation of inter-sentential context, (2) in the display of overall progress and (3) an option to edit previously edited sentences. Specifically for the marking mode they requested an option to mark missing parts or areas for re-ordering. While (2) and (3) indicate directions for improvement of the annotation interface, the last request points to the need of an "extended" marking mode with a feature to mark omissions or re-ordering.

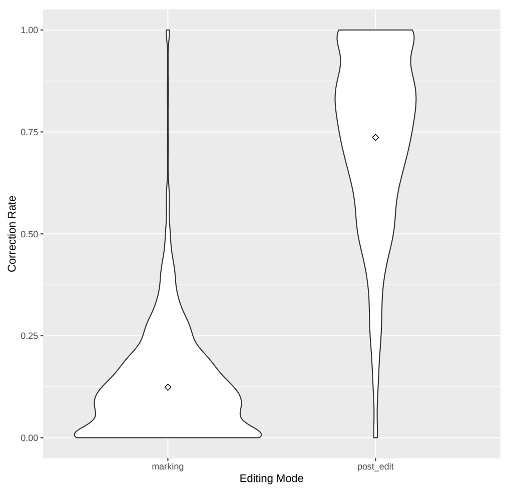


Figure 16: Correction rate by annotation mode. The correction rate describes the ratio of words in the translation that were marked as incorrect (in marking mode) or edited (in correction/post-editing mode). Means are indicated with diamonds.

**Inferred translation quality judgment.** In both modes we can interpret the amounts of edits/markings as a signal of how content the annotator was with the machine translation, i.e., a quality estimate. If a translation contains many errors, it should receive a high number of markings, and analogously, a high number of edits. Hence, we measure which portion of the translations is considered incorrect (marked, deleted or replaced) in both modes, to find out whether the choice of annotation interface has an impact on what is considered correct or not. Table 28 compares how many tokens of the translations were considered incorrect in each mode. In total, annotators find more than twice as many token corrections in the post-edit mode than in the marking mode.[7] In the user-choice mode, the individual tendency for the two modes continues, with a slight increase in corrections in the post-edit mode, which aligns with the policies reported in the previous section.

This is partially caused by the reduced degrees of freedom in marking mode, but also underlines the general trend towards over-editing when in post-edit mode. Figure 16 illustrates, that, if markings and post-edits were used to compute a quality score (correction rate), translations would be judged poorer in post-editing mode. It also holds for whole sentences, where 273 (26.20%) were left un-edited in marking mode, compared to 3 (0.29%) in post-editing mode.

[7]The automatically assessed translation quality for the baseline model does not differ drastically between the portions selected per mode.

| Mode | #incorrect | #correct | Correction Rate |
|---|---|---|---|
| Markings | 2,197 | 15,400 | 12.49% |
| PE | 4,652 | 13,193 | 26.07% |
| User-choice (total) | 3,520 | 14,417 | 19.62% |
| ... Markings | 1,578 | 9,932 | 13.71% |
| ... PE | 1,942 | 4,485 | 30.22% |

Table 28: Correction statistics: count of incorrect and correct tokens in translations according to annotation mode. The correction rate expresses the percentage of tokens in the translations the raters found worthy to correct.

## 6.2 Adapting MT with Error Corrections and Markings

The hypotheses presented to the annotators were generated by a neural machine translation model. The annotations can then be used to improve the underlying model, with the annotators serving as teachers in an interactive human-in-the-loop machine learning scenario. This section describes the fine-tuning objectives to adapt the NMT to human corrections and markings.

### 6.2.1 Objectives

**Learning from Error Corrections.** The standard supervised learning mode in interactive machine translation assumes a fully corrected output $y^*$ for an input $x$ that is treated similar to a gold standard reference translation (Turchi et al., 2017). Model adaptation can be performed by maximizing the likelihood of the user-provided corrections where

$$J(\theta)^{\text{CORR}} = \sum_{x,y^*} \sum_{t=1}^{T} \log p_\theta(y^*_t \mid x; y^*_{<t}), \tag{85}$$

using stochastic gradient descent techniques.

**Learning from Error Markings.** A weaker feedback mode is to let a human teacher mark the correct parts of the machine-generated output $\hat{y}$ (Marie and Max, 2015; Petrushkov et al., 2018; Domingo et al., 2017). As a consequence every token in the output receives a reward $\delta^m_t$, either $\delta^+_t$ if marked as correct, or $\delta^-_t$ otherwise. Petrushkov et al. (2018) proposed a model with $\delta^+_t = 1$ and $\delta^-_t = 0$, but this weighting schemes leads to the ignorance of incorrect outputs in the gradient. Instead, we find it beneficial to penalize incorrect tokens, with e.g. $\delta^-_t = -0.5$, and reward correct tokens $\delta^+_t = 0.5$, which aligns with the findings

| Domain | Train | Dev | Test |
|---|---|---|---|
| WMT17 | 5,919,142 | 2,169 | 3,004 |
| IWSLT17 | 206,112 | 2,385 | 1,138 |
| Selected Talks | 1035 corr / 1042 mark | | 1,043 |

Table 29: Data sizes (en-de). From the IWLST17 training data we use 3,120 sentences contained in a selection of talks.

from Lam et al. (2019). The objective of the learning system is to maximize the likelihood of the correct parts of the output and penalize the incorrect ones:

$$J(\theta)^{\text{MARK}} = \sum_{x,\hat{y}} \sum_{t=1}^{T} \delta_t^m \log p_\theta(\hat{y}_t \mid x; \hat{y}_{<t}). \tag{86}$$

Note the similarity to the token-level expected matching objective (62) that was used in the eBay experiments with online feedback induced from the log of queries (Section 4.1.2). Here we are limited to feedback for one logged output each, so the objective looks rather like a weighted MLE loss than a reinforcement learning objective with token-level rewards. If this token-based feedback was obtained online from humans (or from a reward estimator), expected reward training could be applied here.

## 6.3 Experiments

### 6.3.1 Setup

**NMT Model and Data.** The goal is to adapt a general-domain NMT model (WMT17) to the new domain of TED talks from IWSLT17 (see Table 29) with either corrections or markings. For the general-domain NMT system, we use the pre-trained 4-layer LSTM encoder-decoder Joey NMT WMT17 model for translations from English to German (Kreutzer et al., 2019). The model is trained on a joint vocabulary with 30k subwords (Sennrich et al., 2016c). With the help of the human annotations we adapt this model to the domain of TED talk transcripts by continuing learning on the annotated data. Hyperparameters including learning rate schedule, dropout and batch size for this fine-tuning step are tuned on the IWSLT17 dev set. For the marking mode, the weights $\delta^+$ and $\delta^-$ may be tuned in addition (see Appendix B.5 for an ablation). As test data, we use the split of the selected talks that was annotated in the user-mode, since the purpose of this split was the evaluation of user preference. There is no overlap in the three data splits, but they share topics so that we can both measure local adaptation and draw comparisons between modes.

| System | TER $\downarrow$ | BLEU $\uparrow$ | METEOR $\uparrow$ |
|---|---|---|---|
| WMT baseline | 58.6 | 23.9 | 42.7 |
| **Error Corrections** | | | |
| Full | 57.4$^\star$ | 24.6$^\star$ | 44.7$^\star$ |
| Small | 57.9$^\star$ | 24.1 | 44.2$^\star$ |
| **Error Markings** | | | |
| 0/1 | 57.5$^\star$ | 24.4$^\star$ | 44.0$^\star$ |
| -0.5/0.5 | 57.4$^\star$ | 24.6$^\star$ | 44.2$^\star$ |
| random | 58.1$^\star$ | 24.1 | 43.5$^\star$ |
| **Quality Judgments** | | | |
| from corrections | 57.4$^\star$ | 24.6$^\star$ | 44.7$^\star$ |
| from markings | 57.6$^\star$ | 24.5$^\star$ | 43.8$^\star$ |
| **Automatic Error Corrections** | | | |
| BERT | 56.6$^\star$ | 26.4$^\star$ | 45.1$^\star$ |

Table 30: Results on the test set with feedback collected from humans. Decoding with beam search of width 5 and length penalty of 1. Significant ($p <= 0.05$) improvements over the baseline are marked with $^\star$. Full error corrections and error markings only significantly differ in terms of METEOR. Small: smaller subset of error corrections that consumed the same time as error markings. Random: Marking a random selection of tokens per sentence. Quality judgments are inferred from either feedback mode. Automatic error corrections are obtained from an error correction model trained on human error corrections.

**Evaluation.** The models are evaluated automatically with TER (Snover et al., 2006), BLEU (Papineni et al., 2002) and METEOR (Lavie and Denkowski, 2009) against references translations.[8]

### 6.3.2 Results

**Corrections, Markings and Quality Judgments.** Table 30 compares the models after fine-tuning with corrections and markings with the original WMT out-of-domain model (with exception of the last row which is discussed in Section 6.3.2). The "small" model trained with error corrections is trained on one fifth of the data, which is comparable to the effort it takes to collect the error markings. Both error corrections and markings can be reduced to sentence-

[8]Computed with `MultEval v0.5.1` (Clark et al., 2011) on tokenized outputs.

| **System** | $>$ BL | $=$ BL | $<$ BL |
|---|---|---|---|
| Markings | **43.0%** | 21.0% | 36.4% |
| Corrections | **49.1%** | 16.1% | 34.7% |

Table 31: Human preferences for 300 comparisons between baseline (BL) translations and the NMT system fine-tuned on error markings and corrections. $>$: better than the baseline, $<$ worse than the baseline.

level quality judgments, where all tokens receive the same weight in Eq. 86: $\delta = \frac{\#marked}{hyptokens}$ or $\delta = \frac{\#corrected}{hyptokens}$. In addition, we compare the markings against a random choice of marked tokens per sentence.[9] We find (1) that both models trained on corrections and markings improve significantly over the baseline ("Error Corrections Full", "Error Markings 0/1"), (2) that tuning the weights for (in)correct tokens matters for learning from markings ("Error Markings -0.5/0.5"), (3) that they perform better than random markings ("Error Markings random"), and (4) a uniform treatment of weights through the reduction of quality judgments ("Quality Judgments") results in no loss in comparison to error corrections, and a small loss for markings. We suspect that the small margin between corrections and markings is due to the fact that the models are evaluated against reference translations. Post-edits—which are further away from model outputs than markings, and can also also be further away from references—will be rated more poorly than deserved (and at least METEOR captures some of it).

**Human Evaluation.** It is infeasible to collect markings or corrections for all our systems for a more appropriate comparison than to references. Nevertheless, we conduct a small human evaluation study. Three bilingual raters receive 120 translations of the test set ($\sim$10%) and the corresponding source sentences for each mode and judge whether the translation is better, as good as, or worse than the baseline: 64% of the translations obtained from learning from error markings are judged as good or better than the baseline, compared to 65.2% for the translations obtained from learning from error corrections. Table 31 shows the detailed proportions excluding identical translations.

**Effort vs. Translation Quality.** Figure 17 illustrates the relation between the total time spent on annotations and the resulting translation quality for corrections and markings trained on a selection of subsets of the full annotated data: The overall trend shows that both modes benefit from more training data, with more variance for the marking mode, but also a steeper descent. The point

[9]Each token is marked with probability $p_{mark} = 0.5$.

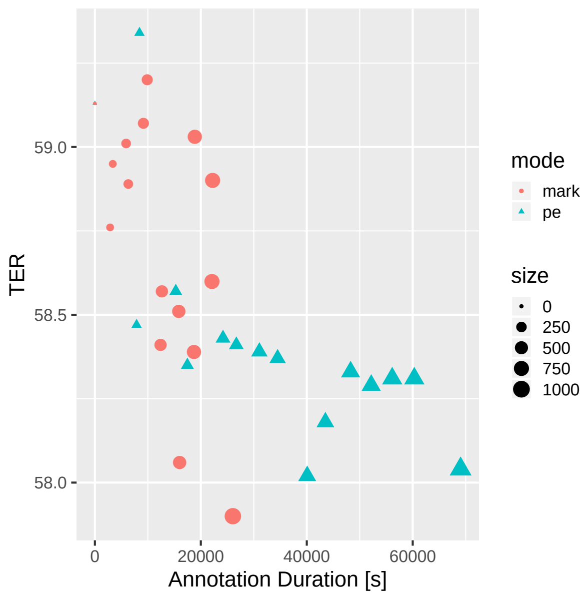


Figure 17: Improvement in TER with training data of varying size: lower is better. Scores are collected across two runs with a random selection of $k \in [125, 250, 375, 500, 625, 750, 875]$ data points each for training.

from where on markings the more efficient choice lies at approximately 20,000s in terms of cumulative annotation time ($\approx$ 5.5h).

**Simulated Adaptation** To put the observed improvements better into perspective, we train systems on the same data splits but with (a) references or (b) model hypotheses simulating corrections and markings.[10] Table 32 shows that training from references, despite the small data size and the same training objective and hyperparameters, is more successful than from the collected corrections (Table 30), which supports our hypothesis of the qualitative differences of corrections and references. Furthermore, treating the model hypotheses as references, the NMT makes surprisingly high improvements, at least on the split (row "hyp mark") that was used for corrections as well. These models are still 0.4 TER / 0.5 BLEU / 0.5 METEOR behind the custom-weighted tokens in learning from markings, which is confirming that the credit assignment by the annotator actually helped the NMT model. Simulating markings from references (as done in previous work (Petrushkov et al., 2018)) or corrections[11] results in

[10]Treating the model output as correction is equivalent to a uniform weighting for all tokens $\delta = 1$.

[11]Error markings were simulated by marking the words that would have to get edited according to minimum edit distance between model hypothesis and reference or correction.

| **System** | **TER** $\downarrow$ | **BLEU** $\uparrow$ | **METEOR** $\uparrow$ |
|---|---|---|---|
| WMT baseline | 58.6 | 23.9 | 42.7 |
| **No Corrections / Markings** | | | |
| hyp corr | 57.4$^\star$ | 24.6$^\star$ | 43.9$^\star$ |
| hyp mark | 57.8$^\star$ | 24.1 | 43.7$^\star$ |
| **Simulated Error Corrections** | | | |
| ref corr | 56.1$^\star$ | 25.2$^\star$ | 44.3$^\star$ |
| ref mark | 55.9$^\star$ | 25.2$^\star$ | 44.4$^\star$ |
| **Simulated Error Markings** | | | |
| sim ref | 57.5$^\star$ | 24.3$^\star$ | 43.5$^\star$ |
| sim corr | 57.7$^\star$ | 24.3$^\star$ | 43.7$^\star$ |

Table 32: Results on the test set after fine-tuning on the splits used for markings (*mark*) and corrections (*corr*) with either out-of-domain hypotheses (*hyp*) or reference translations (*ref*) as simulated corrections, and markings simulated (*sim*) from either the references or the corrections. Scores are averaged across 3 runs. Significant ($p \leq 0.05$) improvements over the baseline are marked with $^\star$.

slightly lower scores than the actual collected markings, which illustrates the weaknesses of this simulation technique (see the examples in Table 26).

### Automatic Corrections and Markings

In our previous experiments (simulation experiments of Chapter 4 and real-world experiments in Chapter 5) reward estimators, i.e. regressions models trained on human judgments, were one of the key ingredients to generalization when working with logged data. Analogously, we train an automatic marker and an automatic corrector model on the human feedback.

Both models are based on pre-trained multilingual BERT (Devlin et al., 2019; Wolf et al., 2019), where the auto-marker adds two feed-forward layers on top of the contextual representations and makes binary predictions, and APE has a Transformer decoder with an output layer over the BERT vocabulary using the model proposed by Correia and Martins (2019) implemented with OpenNMT (Klein et al., 2018). The pre-trained embeddings are fine-tuned on the collection of corrections and markings described above. Training and architecture details are provided in Appendix B.6.

The last line in Table 30 (page 133) reports the result from applying the automatic corrector to the MT output, which improves the translation quality furthest, benefiting from the large training resources distilled into multilingual BERT presentations (Pires et al., 2019). The automatic marker trained on the collected markings achieved an overall 99% accuracy on the training set and 87% accuracy on the validation set, but performed poorly for the class of incorrect tokens. Using its predictions to replace human markings, it fails to improve the MT output, presumably because of the low precision for incorrect tokens.

**LMEM: Sentence-level Quality**

We fit an LMEM (see Section 5.2.2) for sentence-level quality scores of the baseline, and three runs each for the NMT systems fine-tuned on markings and post-edits respectively, and inspect the influence of the system as a fixed effect, and sentence id, topic and source length as random effects.

$$\texttt{TER} \sim \texttt{system} + (1 \mid \texttt{talk\_id/sent\_id}) + (1 \mid \texttt{topic}) + (1 \mid \texttt{src\_length})$$

The fixed effect is significant at $p = 0.05$, i.e., the quality scores of the three systems differ significantly under this model. The global intercept lies at 64.73, the one for marking 1.23 below, and the one for post-editing 0.96 below. The variance in TER is for the largest part explained by the sentence, then the talk, the source length, and the least by the topic.

## 6.4 Conclusion

This chapter presented the first user study on the annotation process and the machine learnability of human error markings of translation outputs. This annotation mode has so far been given less attention than error corrections or quality judgments, and has until now only been investigated in simulation studies.

We found that both according to automatic evaluation metrics and by human evaluation, fine-tuning of NMT models achieved comparable gains by learning from error corrections and markings, however, error markings required several orders of magnitude less human annotation effort. Furthermore, we showed how to learn automatic error correction and marking models, with best results obtained by fine-tuning a Transformer-based error correction model on small amounts of human error corrections. If the automatic error marking model had higher quality, other techniques of adapting NMT systems with learned error markings could be applied, e.g., priming automatic error corrections with automatic markings as proposed for human markings by Grangier and Auli (2018).

In addition, it would be interesting to investigate online adaptation from error markings to test how fast the models adapt and if they succeed in proposing better alternatives after having received negative feedback, similar to what (Lam et al., 2019) proposed for interactive-predictive MT.

Part III

# Learning to Learn in Interaction

# Chapter 7

# Self-Regulated Supervision for Interactive MT

In the three preceding chapters we studied interfaces for gathering feedback signals from humans in various forms: relative and absolute, explicit and implicit, for complete sequences and for single tokens. We observed that weak feedback, in any of these forms—given a certain level of quality—can help to customize and adapt NMT systems. However, supervised learning is often more effective than reinforcement learning, even with little or noisy data (as observed for example in Section 4.3 with the success of the bandit-to-supervised method). Self-training might also bring some benefits (for free, in terms of human feedback costs), as reported for example by Wu et al. (2018a) who leverage "forward" translations for reinforcement learning.

We pursue the goal of making interactive learning as efficient as possible, e.g., by minimizing feedback costs or effort, increasing sample efficiency, or speeding up convergence. So if supervised training and self-training provide more efficient model updates they should not be ignored in our interactive learning. To this aim, the work in this chapter develops a more holistic view on interactive machine learning that integrates fully supervised, weakly supervised and self-training into a joint meta-learning objective. This new algorithm is tested in simulation with online interactions (like in Chapter 3) and can be seen as a pilot to more efficient interactive learning in real-world applications. The question regarding the choice of feedback modes or interfaces that emerged in the preceding part of the thesis, is now shifted from the interface designer to a meta-learning policy, which we call the *regulator*.

The concept of self-regulation has been studied in educational research (Hattie and Timperley, 2007; Hattie and Donoghue, 2016), psychology (Zimmerman and Schunk, 1989; Panadero, 2017), and psychiatry (Nigg, 2017), and was identified as central to successful human learning. "Self-regulated students" can be characterized as "becoming like teachers", in that they have a repertoire of strategies to self-assess and self-manage their learning process, and they know when to seek help and which kind of help to seek. While there is a vast literature on machine learning approaches to meta-learning (Schmidhuber et al., 1996), learning-to-learn (Thrun and Pratt, 1998), or never-ending learning (Mitchell et al., 2015), the aspect of learning when to ask for which kind of feedback has so far been neglected in this field.

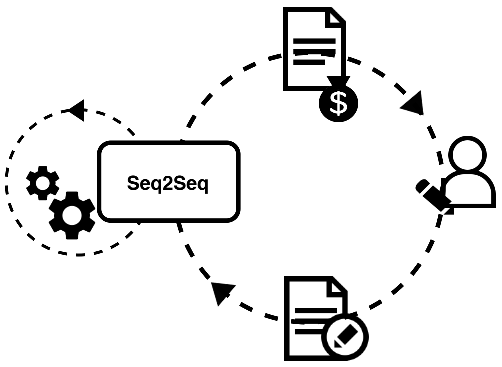


Figure 18: Diagram of human-in-the-loop self-regulated learning.

We propose a machine learning algorithm that uses self-regulation in order to balance the cost and effect of learning from different types of feedback. This is particularly relevant for human-in-the-loop machine learning, where human supervision is costly. The consideration of feedback cost connects our work to active learning, for example, reinforcement learning approaches to learn dynamic active learning strategies (Fang et al., 2017), or to learn a curriculum to order noisy examples (Kumar et al., 2019), or imitation learning approaches for selecting batches of data to be labeled Liu et al. (2018). However, the action space of these approaches is restricted to the decision whether or not to select particular data points for fully-supervised labeling (not any other labeling options), and they are designed for a fixed budget. As we will show, our self-regulation strategy outperforms active learning based on uncertainty sampling (Settles and Craven, 2008; Peris and Casacuberta, 2018), and our reinforcement learner is rewarded in such a way that it will produce the best system as early as possible.

The self-regulation module automatically learns which kind of feedback to apply when in training—full supervision by teacher demonstration or correction, weak supervision in the form of positive or negative rewards for student predictions, or a self-supervision signal generated by the student. Figure 18 illustrates this learning scenario. The learner, in our case a sequence-to-sequence (Seq2Seq) learner, aims to solve a certain task with the help of a human teacher. For every input it receives for training, it can ask the teacher for feedback to its own output, or supervise itself by training on its own output, or skip learning on the input example altogether. The self-regulator's policy for choosing feedback types is guided by their cost and by the performance gain achieved by learning from a particular type of feedback.

The incorporation of a query's cost into reinforcement learning has been addressed, for example, in the framework of active reinforcement learning (Krueger et al., 2016). The central question in active reinforcement learning is to quantify the long-term value of reward information, however, assuming a fixed cost for each action and every round. Our framework is considerably more complicated by the changing costs for each feedback type on each round. Previous work on human reinforcement learning (see Section 2.2.1) have completely ignored the costs that are incurred when eliciting rewards from humans, nor do any of these approaches consider multiple feedback modes.

For the application of machine translation adaptation, feedback in the form of corrections (Turchi et al., 2017), error markings (Domingo et al., 2017), or translation quality judgments (Lam et al., 2018) or the combination of both (Lam et al., 2019), has been successfully integrated in simulation experiments into interactive-predictive machine translation. However, these works do not consider automatic learning of a policy for the optimal choice of feedback. We apply the self-regulation algorithm to interactive machine translation where an NMT system functions as a student which receives feedback simulated from a human reference translation or supervises itself. The intended real-world application is a machine translation personalization scenario where the goal of the human is to teach the NMT system to adapt to in-domain data with the best trade-off between feedback cost and performance gain. It can be transferred to other interactive sequence-to-sequence learning tasks such as conversational AI for question-answering, geographical navigation, or information retrieval.

Our analysis of different configurations of self-regulation yields the following insights: Perhaps unsurprisingly, the self-regulator learns to balance all types of feedback instead of relying only on the strongest or cheapest option. This is an advantage over active learning strategies that only consider the choice between no supervision and full supervision. Interestingly, though, we find that the self-regulator learns to trade off exploration and exploitation similar to a context-free $\epsilon$-greedy strategy that optimizes $\epsilon$ for fastest learning progress. Lastly, we show that the learned regulator is robust in a cold-start transfer to new domains, and even shows improvements over fully supervised learning on domains such as literary books where reference translations provide less effective learning signals.

**Contributions.** This chapter contributes the following:

1. In the preceding chapters the question emerged, which supervision mode, and consequently user interface, is best to be used for which case (level of noise, quality of outputs, reliability and availability of raters). Instead of making this decision manually according to best guesses, it is now passed to a regulator module which meta-learns to choose supervision modes.

2. The combination of supervision modes into one joint objective allows to consider not only their effectiveness but also their costs as optimization target.

3. The empirical evaluation on NMT domain adaptation in interaction with a simulated human teacher shows that learning to supervise is an attractive alternative to standard stream-based active learning or any of the supervision modes in isolation.

**Publications.** This work was published in (Kreutzer and Riezler, 2019); all experiments and analyses were conducted by the author.

**Outline.** Section 7.1 starts with an overview of learning objectives under various levels of supervision (Section 7.1.1), subsumed in the meta-algorithm for self-regulation (Section 7.1.2). Its effectiveness is evaluated in Section 7.2, which covers empirical benchmark results and comparisons against active learning (Section 7.2.2), and also the analysis of the learned self-regulation strategies (Section 7.2.2). Section 7.3 closes this chapter with a summary of the findings.

## 7.1 Self-Regulated Interactive Learning

We model the aspect of self-regulated learning that concerns the ability to decide which type of feedback to query from a teacher (or oneself). This decision should be optimized for most efficient learning, and it should be made depending on the context. In our human-in-the-loop machine learning formulation, we focus on two contextual aspects that can be measured precisely: quality and cost. The self-regulation task is to optimally balance human effort and output quality.

We model self-regulation as an active reinforcement learning problem with dynamic costs, where in each state, the regulator has to choose an action, here a feedback type, and pay a cost. The learner receives feedback of the chosen type from the human to improve its prediction. Based on the effectiveness of this learning update (defined in Eq. 97), the regulator's actions are reinforced or penalized, so that it improves its choice for future inputs.

In the following, we first compare training objectives for a Seq2Seq learner from various types of feedback (Section 7.1.1), then introduce the self-regulator module (Section 7.1.2), and finally combine both in the self-regulation algorithm.

### 7.1.1 Seq2Seq Learning with Various Levels of Supervision

Seq2Seq models can be trained with various levels of supervision. Here, we first consider learning from the two options we investigated in the previous chapter, error corrections and error markings. We further include learning by self-supervision, where the model produces the targets itself. These three modes do not only involve different levels of human interaction, but also require different objectives and gradient updates, which we will present in the following.

**Learning from corrections (Full).** The Seq2Seq learner models the probability $p(y \mid x)$ of an output sequence $y$ given an input sequence $x$ (see Section 2.1.3). Its parameters $\theta$ can be trained with cross-entropy minimization (equivalent to maximizing the likelihood of the data $\mathcal{D}$ under the current model, see Equation 15), if it receives full supervision, i.e., a fully corrected output $y^*$:

$$J^{\text{FULL}}(\theta) = \frac{1}{|\mathcal{D}|} \sum_{(x,y^*)\in\mathcal{D}} -\log p_\theta(y^* \mid x). \tag{87}$$

The stochastic gradient of this objective is

$$g_\theta^{\text{FULL}}(x, y^*) = -\nabla_\theta \log p_\theta(y^* \mid x), \tag{88}$$

constituting an unbiased estimate of the gradient

$$\nabla_\theta J^{\text{FULL}} = \mathbb{E}_{(x,y^*)\sim\mathcal{D}} \left[ g_\theta^{\text{FULL}}(x, y^*) \right]. \tag{89}$$

A local minimum can be found by performing stochastic gradient descent on $g_\theta^{\text{FULL}}(x, y^*)$. This training objective is the standard in supervised learning when training with human-generated references or for online adaptation to post-edits (Turchi et al., 2017) and equivalent to the one used in Chapter 6 when learning from offline corrections (Equation (85)).

**Learning from error markings (Weak).** Petrushkov et al. (2018) presented chunk-based binary feedback as a low-cost alternative to full corrections, and we confirmed with our user study in Chapter 6 that time and effort are much lower for error markings, but that the baseline MT model could still be improved by it. In this scenario the human teacher marks the correct parts of the machine-generated output $\hat{y}$. As a consequence, every token in the output receives a reward $\delta_t$, here either $\delta_t = 1$ if marked as correct, or $\delta_t = 0$ otherwise.[1] The objective of the learner is to maximize the likelihood of the correct parts of the output, or equivalently, to minimize

$$J^{\text{WEAK}}(\theta) = \tfrac{1}{|\mathcal{D}|} \textstyle\sum_{(x,\hat{y})\in\mathcal{D}} \sum_{t=1}^{T} -\delta_t \log p_\theta(\hat{y}_t \mid x; \hat{y}_{<t}) \tag{90}$$

where the stochastic gradient is

$$g_\theta^{\text{WEAK}}(x, \hat{y}) = -\sum_{t=1}^{T} \delta_t \cdot \nabla_\theta \log p_\theta(\hat{y}_t \mid x; y_{<t}), \tag{91}$$

which constitutes an unbiased estimate of the gradient

$$\nabla_\theta J^{\text{WEAK}} = \mathbb{E}_{(x,\hat{y})\sim\mathcal{D}} \left[ g_\theta^{\text{WEAK}}(x, \hat{y}) \right]. \tag{92}$$

[1]We abstain from tuning $\delta$ here, but the experiments in the previous Chapter in Section 6.3 showed that there might be a slight benefit doing so.

The tokens $\hat{y}_t$ that receive $\delta_t = 1$ are part of the correct output $y^*$, so the model receives a hint of what a corrected output should look like. Although the likelihood of the incorrect parts of the sequence does not weigh into the sum, they are contained in the context of the correct parts (in $y_{<t}$). This objective is equivalent to the one used to learn from offline corrections in Chapter 6 (Equation (86)).

**Self-supervision (Self).** Instead of querying the teacher for feedback, the learner can also choose to learn from its own output, effectively supervising itself (Schwenk, 2008). The simplest option is to treat the learner's output as if it was correct, but that quickly leads to overconfidence and degeneration (see Section 6.3 for empirical results). Clark et al. (2018) proposed a cross-view training method: The learner's original prediction is used as a target for a weaker model that shares parameters with the original model. The advantage of this artificial weakening is that it forces the representations to be more robust, and thereby helps generalization, just like standard dropout (Srivastava et al., 2014a). We adopt this strategy by first producing a target sequence $\hat{y}$ with beam search and then weaken the decoder through attention dropout with probability $p_{att}$. The objective is to minimize the negative likelihood of the original target under the weakened model

$$J^{\textsc{Self}}(\theta) = \frac{1}{|\mathcal{D}|} \sum_{(x,\hat{y}) \in \mathcal{D}} - \log p_\theta^{p_{att}}(\hat{y} \mid x), \tag{93}$$

where the stochastic gradient is

$$g_\theta^{\textsc{Self}}(x, \hat{y}) = -\nabla_\theta \log p_\theta^{p_{att}}(\hat{y} \mid x), \tag{94}$$

$$\tag{95}$$

an unbiased estimate of the gradient

$$\nabla_\theta J^{\textsc{Self}} = \mathbb{E}_{(x,\hat{y}) \sim \mathcal{D}} \left[ g_\theta^{\textsc{Self}}(x, \hat{y}) \right]. \tag{96}$$

**Joint formulation.** For self-regulated learning, we also consider a fourth option (None): the option to ignore the current input. Figure 19 summarizes the stochastic gradients for all cases of feedback choices $s$ with a joint formulation. In practice, Seq2Seq learning shows greater stability for mini-batch updates than online updates on single training samples. Mini-batch self-regulated learning can be achieved by accumulating stochastic gradients for a mini-batch of size $\mathcal{B}$ before updating $\theta$ with an average of these stochastic gradients, which we denote as $g_\theta^{s_{[1:\mathcal{B}]}}(x_{[1:\mathcal{B}]}, y_{[1:\mathcal{B}]}) = \frac{1}{\mathcal{B}} \sum_{i=1}^{\mathcal{B}} g_\theta^{s_i}(x_i, y_i)$.

$$
\begin{aligned}
g_\theta^s(x,y) &= -\sum_{t=1}^{T} f_t \cdot \nabla_\theta \log p_\theta^{drop}(y_t \mid x_t; y_{<t}), \\
\text{with } y &= \begin{cases} y^* & \text{if } s = \textsc{Full} \\ \hat{y} & \text{otherwise,} \end{cases} \\
drop &= \begin{cases} p_{\text{att}} & \text{if } s = \textsc{Self} \\ 0 & \text{otherwise,} \end{cases} \\
\text{and } f_t &= \begin{cases} 1 & \text{if } s \in \{\textsc{Full}, \textsc{Self}\} \\ \delta_t & \text{if } s = \textsc{Weak} \\ 0 & \text{if } s = \textsc{None} \end{cases}
\end{aligned}
$$

Figure 19: Stochastic gradients for Seq2Seq depending on feedback type $s$.

### 7.1.2 Learning to Self-Regulate

The regulator is another neural model $q_\phi$ that is optimized for the quality-cost trade-off of the Seq2Seq learner. Given an input $x_i$ and the Seq2Seq's hypothesis $\hat{y}_i$, it chooses an action, here a supervision mode $s_i \sim q_\phi(s \mid x_i, \hat{y}_i)$. This choice of feedback determines the update of the Seq2Seq learner (Figure 19).

The regulator is rewarded by the ratio between the cost $c_i$ of obtaining the feedback $s_i$ and the change in quality $\Delta(\theta_i, \theta_{i-1})$ caused by updating the Seq2Seq learner with the feedback:

$$r(s_i, x_i, \theta_i) = \frac{\Delta(\theta_i, \theta_{i-1})}{c_i + \alpha}. \tag{97}$$

$\Delta(\theta_i, \theta_{i-1})$ is measured as the difference in validation score achieved before and after the learner's update (Fang et al., 2017), and $c_i$ as the cost of user edits. Adding a small constant cost $\alpha$ to the actual feedback cost ensures numerical stability. This meta-parameter can be interpreted as representing a basic cost for model updates of any kind.

The objective for the regulator is to maximize the expected reward defined in Equation 97:

$$J^{\textsc{Meta}}(\phi) = \mathbb{E}_{x \sim p(x), s \sim q_\phi(s \mid x, \hat{y})} \left[ r(s, x, \theta) \right]. \tag{98}$$

The full gradient of this objective is estimated by the stochastic gradient for sampled actions (Williams, 1992):

$$g_\phi^{\textsc{Meta}}(x, \hat{y}, s) = r(s, x, \theta) \nabla_\phi \log q_\phi(s \mid x, \hat{y}). \tag{99}$$

Note that the reward contains the immediate improvement after one update of the Seq2Seq learner and not the overall performance in hindsight. This is

**Algorithm 6** Self-Regulated Interactive Seq2Seq

```
Input: Initial Seq2Seq $\theta_0$, regulator $\phi_0$, mini-batch size $\mathcal{B}$
 1: $j \leftarrow 0$
 2: while inputs and human available do
 3:     $j \leftarrow j + 1$
 4:     for $i \leftarrow 1$ to $\mathcal{B}$ do
 5:         Observe input $x_i$, Seq2Seq output $\hat{y}_i$
 6:         Choose feedback: $s_i \sim q_\phi(s \mid x_i, \hat{y}_i)$
 7:         Obtain feedback $f_i$ of type $s_i$ at cost $c_i$
 8:     Update $\theta$ with $g_\theta^{s_{[1:\mathcal{B}]}}(x_{[1:\mathcal{B}]}, \hat{y}_{[1:\mathcal{B}]})$
 9:     Measure improvement $\Delta(\theta_j, \theta_{j-1})$
10:     Update $\phi$ with $g_\phi^{\text{META}}(x_{[1:\mathcal{B}]}, \hat{y}_{[1:\mathcal{B}]}, s_{[1:\mathcal{B}]})$
```

an important distinction to classic expected reward objectives in reinforcement learning since it biases the regulator towards actions that have an immediate effect. This is desirable in the case of interaction with a human.

However, since Seq2Seq learning requires updates and evaluations based on mini-batches, the regulator update also needs to be based on mini-batches of predictions, leading to the following specification of Equation (99) for a mini-batch $j$:

$$\begin{aligned} & g_\phi^{\text{META}}(x_{[1:\mathcal{B}]}, \hat{y}_{[1:\mathcal{B}]}, s_{[1:\mathcal{B}]}) \\ & = \frac{1}{\mathcal{B}} \sum_{i=1}^{\mathcal{B}} g_\phi^{\text{META}}(x_i, \hat{y}_i, s_i) \\ & = \Delta(\theta_j, \theta_{j-1}) \frac{1}{\mathcal{B}} \sum_{i=1}^{\mathcal{B}} \frac{\nabla_\phi \log q_\phi(s_i \mid x_i, \hat{y}_i)}{c_i + \alpha}. \end{aligned} \tag{100}$$

While mini-batch updates are required for stable Seq2Seq learning, they hinder the regulator from assigning credit for model improvement to individual elements within the mini-batch.

**Algorithm** Algorithm 6 presents the proposed online learning algorithm with model updates accumulated over mini-batches. On arrival of a new input, the regulator predicts a feedback type in line 6. According to this prediction, the environment/user is asked for feedback for the Seq2Seq's prediction at cost $c_i$ (line 7). The Seq2Seq model is updated on the basis of the feedback and mini-batch of stochastic gradients computed as summarized in Figure 19. In order to reinforce the regulator, the Seq2Seq model's improvement (line 9) is assessed, and the parameters of the regulator are updated (line 10, Equation 100). Training ends when the data stream or the provision of feedback ends. The intermediate

Seq2Seq evaluations can be re-used for model selection (early stopping). In practice, these evaluations can either be performed by validation on a held-out set (as in the simulation experiments below) or by human assessment. Figure 20 visualizes the interleaved training procedure for the Seq2Seq learner and regulating meta-learner in interaction with the human teacher.

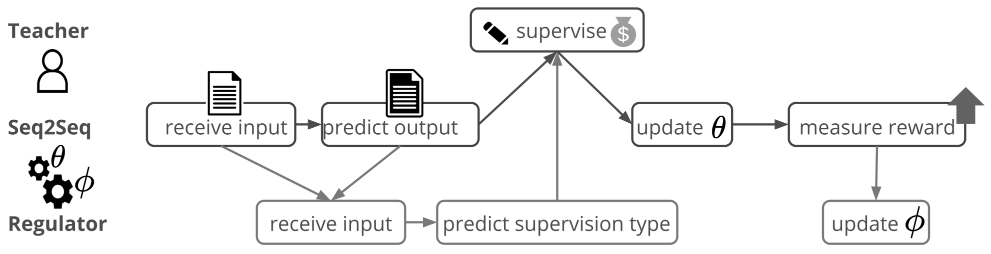


Figure 20: Learning procedure of MT and regulator.

**Practical considerations.** The algorithm does not introduce any additional hyperparameters beyond standard learning rates, architecture design and mini-batch sizes that have to be tuned. As proposed in Petrushkov et al. (2018) or Clark et al. (2018), targets $\hat{y}$ are pre-generated offline with the initial $\theta_0$, which we found crucial for the stability of the learning process. The evaluation step after the Seq2Seq update is an overhead that comes with meta-learning, incurring costs depending on the decoding algorithm and the evaluation strategy. However, Seq2Seq updates can be performed in mini-batches, and the improvement is assessed after a mini-batch of updates, as discussed above.

## 7.2 Experiments

The research question to be answered in our experiments is threefold:

1. Which strategies does the regulator develop?
2. How well does a trained regulator transfer across domains?
3. How do these strategies compare against (active) learning from a single feedback type?

We perform experiments for interactive NMT, where a general-domain NMT model is adapted to a specific domain by learning from the feedback of a human translator. This is a realistic interactive learning scenario where cost-free pre-training on a general domain data is possible, but each feedback generated by the human translator in the personalization step incurs a cost. In our experiment,

| de→en | **WMT** | **IWSLT** | **Books** |
|---|---|---|---|
| Train | 5,889,699 | 206,112 | 46,770 |
| Dev | 2,169 | 2,385 | 2,000 |
| Test | 3,004 | 1,138 | 2,000 |

Table 33: Number of sentences for parallel corpora used for pre-training (WMT), regulator training (IWSLT) and domain transfer evalution (Books).

we use human-generated reference translations to simulate the cost of human feedback and to measure the performance gain achieved by model updates.

### 7.2.1 Architectures

**Seq2Seq architecture.** Both the Seq2Seq learner and the regulator are based on LSTMs (Hochreiter and Schmidhuber, 1997). The Seq2Seq has four bi-directional encoder and four decoder layers with 1,024 units each, embedding layers of size 512. It uses Luong et al. (2015a)'s input feeding and output layer, and global attention with a single feed forward layer (Bahdanau et al., 2015).

**Regulator architecture.** The regulator consists of LSTMs on two levels: Inspired by Siamese Networks (Bromley et al., 1994), a bi-directional LSTM encoder of size 512 separately reads in both the current input sequence and the beam search hypothesis generated by the Seq2Seq. The last state of encoded source and hypothesis sequence and the previous output distribution are concatenated to form the input to a higher-level regulator LSTM of size 256. This LSTM updates its internal state and predicts a score for every feedback type for every input in the mini-batch. The feedback for each input is chosen by sampling from the distribution obtained by softmax normalization of these scores. The embeddings of the regulator are initialized by the Seq2Seq's source embeddings and further tuned during training. The model is implemented in the Joey NMT framework (Kreutzer et al., 2019) based on PyTorch (Paszke et al., 2017).

**Data.** We use three parallel corpora for German-to-English translation: a general-domain data set from the WMT2017 translation shared task for Seq2Seq pre-training, TED talks from the IWSLT2017 evaluation campaign for training the regulator with simulated feedback, and the Books corpus from the OPUS collection (Tiedemann, 2012) for testing the regulator on another domain. The joint vocabulary for Seq2Seq and the regulator consists of 32k BPE sub-words (Sennrich et al., 2016c) trained on WMT. The WMT data is obtained from the

WMT 2017 shared task website[2] and pre-processed as described in Hieber et al. (2017). The pre-processing pipeline is used for IWSLT and Books data as well. IWSLT2017 is obtained from the evaluation campaign website.[3] For validation on WMT, we use the `newstest2015` data, for IWSLT `tst2014+tst2015`, for testing on WMT `newstest2017` and `tst2017` for IWSLT. Since there is no standard split for the Books corpus, we randomly select 2k sentences for validation and testing each. Table 33 gives an overview of the size of the three resources.

**Training.** The Seq2Seq model is first trained on WMT with Adam (Kingma and Ba, 2015) on mini-batches of size 64, an initial learning rate $1 \times 10^{-4}$ that is halved when the loss does not decrease for three validation rounds. Training ends when the validation score does not increase any further (scoring 29.08 BLEU on the WMT test set). This model is then adapted to IWSLT with self-regulated training for one epoch, with online human feedback simulated from reference translations. The mini-batch size is reduced to 32 for self-regulated training to reduce the credit assignment problem for the regulator. The constant cost $\alpha$ (Eq. 97) is set to 1.[4] When multiple runs are reported, the same set of random seeds is used for all models to control the order of the inputs. The best run is evaluated on the Books domain for testing generalization.

**Simulation of cost and performance.** In our experiments, human feedback and its cost, and the performance gain achieved by model updates, is simulated by using human reference translations. Inspired by the keystroke mouse-action ratio (KSMR) (Barrachina et al., 2009), a common metric for measuring human effort in interactive machine translation, we define feedback cost as the sum of costs incurred by character edits and clicks, similar to Peris and Casacuberta (2018). The cost of a full correction (Full) is the number of character edits between model output and reference, simulating the cost of a human typing.[5] Error markings (Weak) are simulated by comparing the hypothesis to the reference and marking the longest common sub-strings as correct, as proposed by Petrushkov et al. (2018). As an extension to Petrushkov et al. (2018) we mark multiple common sub-strings as correct if all of them have the longest length. The cost is defined as the number of marked words, assuming an interface that allows markings by clicking on words. For self-training (Self) and skipping training instances we naively assume zero cost, thus limiting the measurement of cost to the effort of the human teacher, and neglecting the effort on the learner's side. Table 34 illustrates the costs per feedback type on a randomly selected set of examples.

[2]`http://www.statmt.org/wmt17/translation-task.html`
[3]`https://sites.google.com/site/iwsltevaluation2017/`
[4]Values $\neq 1$ distort the rewards for self-training.
[5]As computed by the Python library `difflib`.

| Mode | Cost | Input / Output | |
|---|---|---|---|
| | | $x$ | Sie greift in ihre Geldbörse und gibt ihm einen Zwanziger. |
| SELF | 0 | $\hat{y}$ | It attacks their wallets and gives him a twist. |
| | | $y^*$ | She reaches into her purse and hands him a 20. |
| WEAK | 9 | $x$ | Und als ihr Vater sie sah und sah, wer sie geworden ist, in ihrem vollen Mädchen-Sein, schlang er seine Arme um sie und brach in Tränen aus. |
| | | $\hat{y}$ | And when her father saw them and saw who became them, in their full girl's, he swallowed his arms around them and broke out in tears. |
| | | $y^*$ | When her father saw her and saw who she had become, in her full girl self, he threw his arms around her and broke down crying. |
| FULL | 59 | $x$ | Und durch diese zwei Eigenschaften war es mir möglich, die Bilder zu erschaffen, die Sie jetzt sehen. |
| | | $\hat{y}$ | And th~~rough~~ th~~e~~se two ~~features,~~ I was able to create the images you now ~~see~~. |
| | | $y^*$ | And it was with those two properties that I was able to create the images that you're seeing right now. |

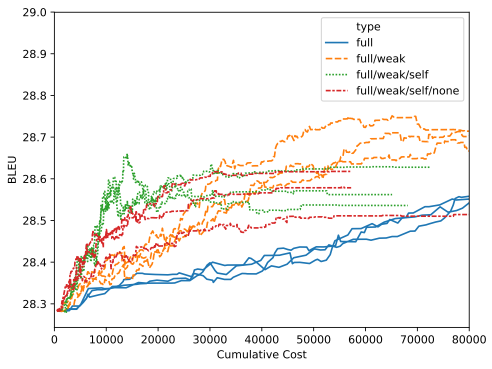
Table 34: Examples from the IWSLT17 training set, cost and feedback decisions made by a regulator. For weak feedback, marked parts are underlined. For full feedback, the corrections are marked by underlining the parts of the reference that got inserted and the parts of the hypothesis that got deleted.

**Evaluation.** We measure the model improvement by evaluating the held-out set translation quality of the learned model at various time steps with corpus BLEU (cased `SacreBLEU` (Post, 2018)) and measure the accumulated costs. The best model is considered the one that delivers the highest quality at the lowest cost. This trade-off is important to bear in mind since it differs from the standard evaluation of machine translation models, where the overall best-scoring model, regardless of the supervision cost, is considered best. Finally, we evaluate the strategy learned by the regulator on an unseen domain, where the regulator decides which type of feedback the learner gets, but is not updated itself.

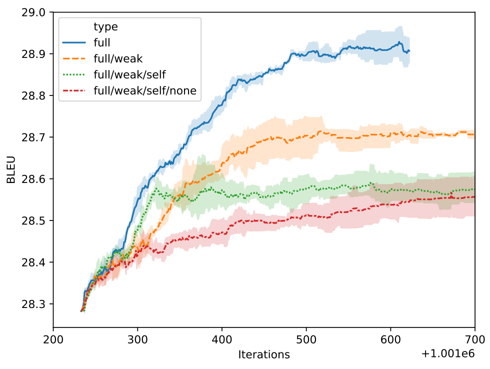
### 7.2.2 Results

We compare learning from one type of feedback in isolation against regulators with the following set of actions:

1. *Reg2*: FULL, WEAK



(a) BLEU over cumulative costs.



(b) BLEU over time.

Figure 21: Regulation variants evaluated in terms of BLEU over time (a) and cumulative costs (b). Iteration counts start from the iteration count of the baseline model. One iteration on IWSLT equals training on one mini-batch of 32 instances. The BLEU score is computed on the tokenized validation set with greedy decoding. In (b) the lines correspond to the means over three runs, the shaded area depicts the estimated 95% confidence interval.

2. *Reg3*: Full, Weak, Self
3. *Reg4*: Full, Weak, Self, None

## Cost vs. Quality

Figure 21a compares the improvement in corpus BLEU (Papineni et al., 2002) (corresponding to results in Translation Error Rate (TER, computed by `pyTER`[6]) (Snover et al., 2006)) of regulation variants and full feedback over cumulative costs of up to 80k character edits and time. Using only full feedback (blue) as in standard supervised learning or learning from post-edits, the overall highest improvement can be reached (visible only after the cutoff of 80k edits, see Figure 21b). However, it comes at a very high cost (417k characters in total to reach +0.6 BLEU).

The regulated variants offer a much cheaper improvement, at least until a cumulative cost between 80k (*Reg4*) and 120k (*Reg2*), depending on the feedback options available. The regulators do not reach the quality of the full model since their choice of feedback is oriented towards costs and immediate improvements. By finding a trade-off between feedback types for immediate improvements, the regulators sacrifice long-term improvement.

[6]`https://pypi.org/project/pyter/`

| Model | IWSLT dev | | IWSLT test | |
|---|---|---|---|---|
| | BLEU↑ | Cost↓ | BLEU↑ | TER↓ |
| *Baseline* | 28.28 | - | 24.84 | 62.42 |
| *Full* | $28.93_{\pm 0.02}$ | 417k | $25.60_{\pm 0.02}$ | $61.86_{\pm 0.03}$ |
| *Weak* | $28.65_{\pm 0.01}$ | 32k | $25.10_{\pm 0.09}$ | $62.12_{\pm 0.12}$ |
| *Self* | $28.58_{\pm 0.02}$ | - | $25.33_{\pm 0.06}$ | $61.96_{\pm 0.05}$ |
| *Reg4* | $28.57_{\pm 0.04}$ | 68k | $25.23_{\pm 0.05}$ | $62.02_{\pm 0.12}$ |
| *Reg3* | $28.61_{\pm 0.03}$ | 18k | $25.23_{\pm 0.09}$ | $62.07_{\pm 0.06}$ |
| *Reg2* | $28.66_{\pm 0.06}$ | 88k | $25.27_{\pm 0.09}$ | $61.91_{\pm 0.06}$ |

Table 35: Evaluation of models at early stopping points. Results for three random seeds on IWSLT are averaged, reporting the standard deviation in the subscript. The translation of the dev set is obtained by greedy decoding (as during validation) and of the test set with beam search of width five. The costs are measured in character edits and clicks, as described in Section 7.2.

Comparing regulators, *Reg2* (orange) reaches the overall highest improvement over the baseline model, but until the cumulative cost of around 35k character edits, *Reg3* (green) offers faster improvement at a lower cost since it has an additional, cheaper feedback option. Adding the option to skip examples (*Reg4*, red) does not give a benefit.

Table 35 reports the offline held-out set evaluations for the early stopping points selected on the dev set for all feedback modes. All models notably improve over the baseline, using only full feedback leads to the overall best model on IWSLT (+0.6 BLEU / -0.6 TER), but costs a massive amounts of edits (417k characters).

Self-regulating models still achieve improvements of 0.4-0.5 BLEU/TER with costs reduced up to a factor of 23 in comparison to the full feedback model. *Reg3* also notably reduces the effort compared to weak feedback only. The reduction in cost is enabled by the use of cheaper feedback, here markings and self-training, which in isolation are successful as well. Self-training works surprisingly well, which makes it attractive for cheap but effective unsupervised domain adaptation.

It has to be noted that both weak and self-supervision worked only well when targets were pre-computed with the baseline model and held fixed during training. We suspect that the strong reward signal ($f_t = 1$) for non-reference outputs leads otherwise to undesired local overfitting effects that a learner with online-generated targets cannot recover from.

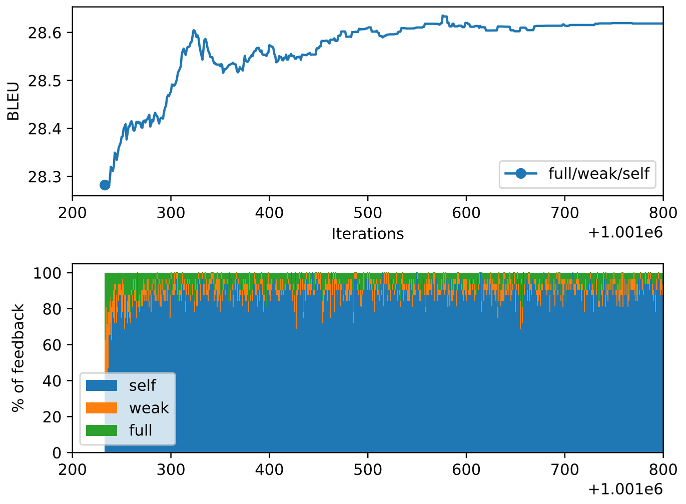
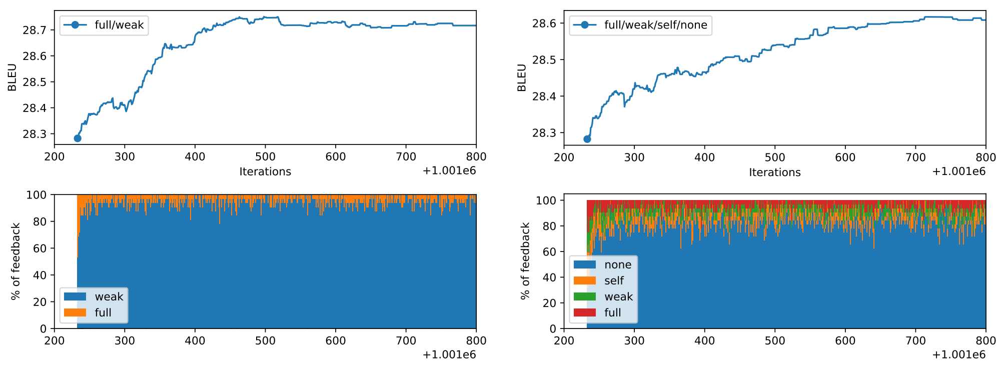


Figure 22: *Reg3* actions chosen over time, and the corresponding BLEU score, depicted for each iteration. Counting of iterations starts at the previous iteration count of the baseline model.

Figure 22 (page 154) shows which actions *Reg3* chooses over time when trained on IWSLT. Most often it chooses to do self-training on the current input. The choice of feedback within one batch varies only slightly during training, with the exception of an initial exploration phase within the first 100 iterations. In general, we observe that all regulators are highly sensitive to balancing cost and performance, and mostly prefer the cheapest option (e.g., *Reg4* by choosing mostly NONE) since they are penalized heavily for choosing (or exploring) expensive options (see Eq. 97). Figures 23a (page 155) and 23b (page 155) additionally show the ratio of feedback types for self-regulation during training with *Reg2* and *Reg4* respectively.

A further research question is whether and how the self-regulation module takes the input or output context into account. We therefore compare its decisions to a context-free $\epsilon$-greedy strategy. The $\epsilon$-greedy algorithm is a successful algorithm for multi-armed bandits (Watkins, 1989). In our case, the arms are the four feedback types. They are chosen based on their reward statistics, here the average empirical reward per feedback type $Q_i(s) = \frac{1}{N_i(s)} \sum_{0,\dots,i} r(s_i)$, where $N$ is the number of previous interaction rounds. With probability $1 - \epsilon$, the algorithm selects the feedback type with the highest empirical reward (exploitation), otherwise picks one of the remaining arms at random (exploration).

In contrast to the neural regulator model, $\epsilon$-greedy decides solely on the



(a) *Reg2* (b) *Reg4*

Figure 23: Feedback chosen by *Reg2* and *Reg4* on IWSLT.

basis of the reward statistics and has no internal contextual state representation. The comparison of *Reg3* with $\epsilon$-greedy for a range of values for $\epsilon$ in Figure 24 (page 156) shows that the learned regulator behaves indeed very similar to an $\epsilon$-greedy strategy with $\epsilon = 0.25$. $\epsilon$-greedy variants with higher amounts of exploration show a slower increase in BLEU, while those with more exploitation show an initial steep increase that flattens out, leading to overall lower BLEU scores. The regulator has hence found the best trade-off, which is an advantage over the $\epsilon$-greedy algorithm where $\epsilon$ requires dedicated tuning.

Considering the $\epsilon$-greedy-like strategy of the regulator and the strong role of the cost factor shown in Figure 22 (page 154), the regulator module does not appear to choose individual actions based e.g., on the difficulty of inputs, but rather composes mini-batches with a feedback ratio according to the feedback type's statistics. This confirms the observations of Peris and Casacuberta (2018), who find that the subset of instances selected for labeling is secondary—it is rather the mixing ratio of feedback types that matters. This finding is also consistent with the mini-batch update regime that forces the regulator to take a higher-level perspective and optimize the expected improvement at the granularity of (mini-batch) updates rather than at the input level.

### Domain Transfer

After training on IWSLT, we evaluate the regulators on the Books domain: Can they choose the best actions for an efficient learning progress without receiving feedback on the new domain?

We evaluate the best run of each regulator type (i.e., $\phi$ trained on IWSLT), with the Seq2Seq model reset to the WMT baseline. The regulator is not further adapted to the Books domain, but decides on the feedback types for training

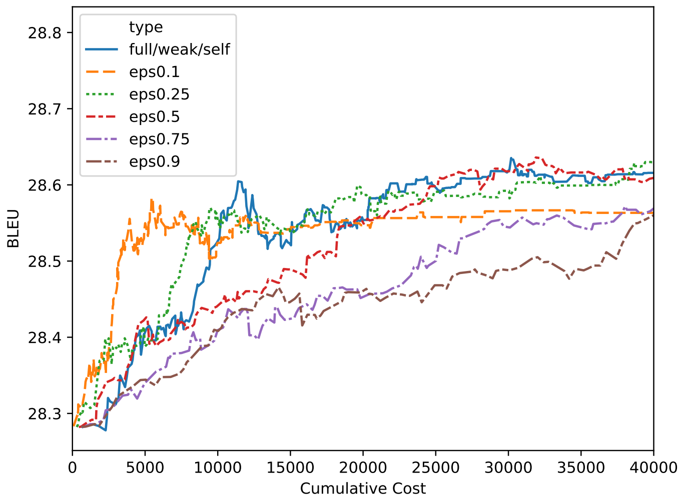


Figure 24: BLEU and cumulative costs on IWSLT for *Reg3* and $\epsilon$-greedy with $\epsilon \in [0.1, 0.25, 0.5, 0.75, 0.9]$.

the Seq2Seq model for a single epoch on the Books data. Figure 25 (page 157) visualizes the regulated training process of the Seq2Seq model. As before, *Reg3* performs best, outperforming weak, full and self-supervision (reaching 14.75 BLEU, not depicted since zero cost).

Learning from full feedback improves much later in training and reaches 14.53 BLEU.[7] One explanation is that the reference translations in the Books corpus are less literal than the ones for IWSLT, such that a weak feedback signal allows the learner to learn more efficiently than from full corrections.

Table 36 reports results for test set evaluation on the Books domain of the best model from the IWSLT domain each. The best result in terms of BLEU and TER is achieved by the *Reg2* model, even outperforming the model with full feedback. As observed for the IWSLT domain (cf. Section 7.2.2), self-training is very effective, but is outperformed by the *Reg2* model and roughly on par with the *Reg3* model.

### Comparison to Active Learning

A classic active learning strategy is to sample a subset of the input data for full labeling based on the uncertainty of the model predictions (Settles and Craven, 2008). The size of this subset, i.e. the amount of human labeling effort, has to be known and determined before learning.

[7]With multiple epochs it would improve further, but we want to avoid showing the human the same inputs multiple times.

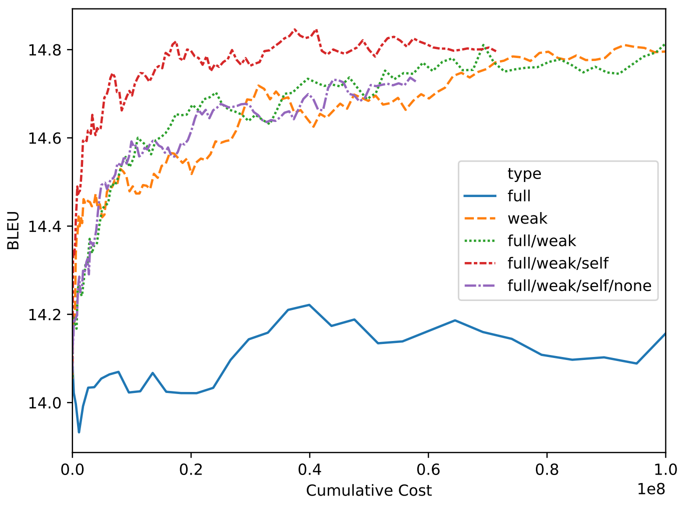


Figure 25: Domain transfer of regulators trained on IWSLT to the Books domain in comparison to full and weak feedback only.

Here, we use a fixed ratio $\gamma$, that determines which percentage of samples per mini-batch are fully-labeled. Figures 26a and 26b (page 159) compare the self-regulators on the Books domain ($\gamma \in [0.1, 0.3, 0.5, 0.7, 0.9]$) over cumulative costs and time with models that learn from a fixed ratio of fully-labeled instances in every batch. These are chosen according to the model's uncertainty, here measured by the average token entropy of the model's best-scoring beam search hypothesis.

The regulated models with a mix of feedback types clearly outperform the active learning strategies, both in terms of cost-efficient learning (Figure 26a on page 159) as well as in overall quality. We conclude that mixing feedback types, especially in the case where full feedback is less reliable, offers large improvements over standard stream-based active learning strategies.

### 7.2.3 Prospects for Field Studies

Our experiments were designed as a pilot study to test the possibilities of self-regulated learning in simulation. In order to advance to field studies where human users interact with Seq2Seq models, several design choices have to be adapted with caution.

Firstly, we simulate both feedback cost and quality improvement by measuring distances to static reference outputs. The experimental design in a field study has to account for a variation of feedback strength and cost, and performance assessments across time, across sentences, and across human users (Settles et al.,

| Model | Books test | | |
|---|---|---|---|
| | **BLEU↑** | **TER↓** | **Cost↓** |
| *Baseline* | 14.19 | 79.81 | - |
| FULL | 14.87 | 79.12 | 1B |
| WEAK | 14.74 | 78.14 | 93M |
| SELF | 14.73 | 78.86 | - |
| *Reg4* | 14.80 | 79.02 | 57M |
| *Reg3* | 14.80 | 78.70 | 41M |
| *Reg2* | 15.00 | 78.21 | 142M |

Table 36: Evaluation of models at early stopping points on the Books test set (beam search with width five).

2008). One desideratum for field studies is thus to analyze this variation by analyzing the experimental results in a mixed effects model that accounts for variability across sentences, users, and annotation sessions (Baayen et al., 2008; Karimova et al., 2018), similar to the LMEMs in Sections 5.2.2 and 6.1.2.

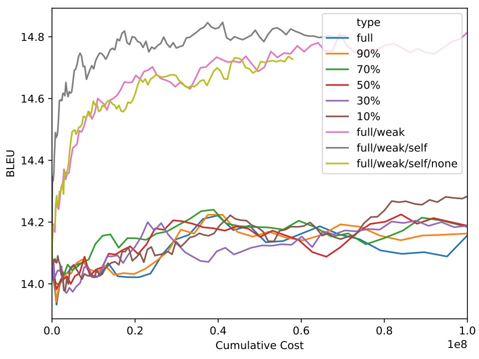
Secondly, our simulation of costs considers only the effort of the human teacher, not the machine learner. The strong preference for the cheapest feedback option might be a result of overestimating the cost of human post-editing and underestimating the cost of self-training. Thus, a model for field studies where data is limited, like the case presented in Chapter 5, might greatly benefit from learned estimates of feedback cost and quality improvement.

## 7.3 Conclusion

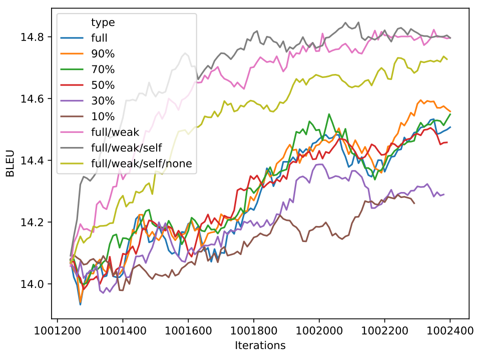
In this chapter we proposed a cost-aware algorithm for interactive sequence-to-sequence learning, with a self-regulation module at its core that learns which type of feedback to query from a human teacher. The empirical study on interactive NMT with simulated human feedback showed that this self-regulated model finds more cost-efficient solutions than models learning from a single feedback type and uncertainty-based active learning models, also under domain shift.

While this setup abstracts away from certain confounding variables to be expected in real-life interactive machine learning, it should be seen as a pilot experiment focused on our central research questions under an exact and noise-free computation of feedback cost and performance gain. The proposed framework can naturally be expanded to integrate more feedback modes suitable for the interaction with humans, e.g., pairwise comparisons or output rankings.

Future research directions will involve the development of reinforcement



(a) BLEU over cumulative costs



(b) BLEU over time

Figure 26: Learned self-regulation strategies in comparison to uncertainty-based active learning with a fixed percentage $\gamma$ of full feedback on the Books domain: development of validation BLEU against costs (a) and time (b). Counting of iterations for (b) starts at the previous iteration count of the baseline model.

learning model with multi-dimensional rewards, and modeling explicit credit assignment for improving the capabilities of the regulator to make context-sensitive decisions in mini-batch learning.

# Chapter 8

# Thesis Conclusion

## 8.1 Summary

In this thesis, we introduced and evaluated policy learning algorithms for human-in-the-loop NMT in various learning scenarios.

Chapter 3 focused on finding sample-efficient and well-generalizing algorithms learning from *online feedback in simulation*, with neural and statistical models with *absolute and relative* judgments that are simulated by comparing proposed translations to reference translations. The algorithms are built on the REINFORCE algorithm (Williams, 1992), adapted for bandit feedback for structured outputs. The resulting models were evaluated on domain adaptation tasks for English-German and English-French translations from Parliament discussions to news, TED talks and e-commerce products. The evaluation focused on generalization performance, online regret, and convergence speed. Pairwise preference learning was found to be more efficient for statistical models, but expected reward training from absolute judgments yielded better results for neural models. We identified control variates such as reward baselines, large amounts of exploration, and clean feedback as the ingredients to successful learning.

Since online feedback and a lot of exploration is challenging to obtain in practice (production systems for example cannot effort to lose users due to exploration of worse alternatives) Chapter 4 addressed the *counterfactual learning scenario*: on the basis of a set of *user-rated logged translations*, learn to generate translations that, if deployed in the same way, would have achieved even higher ratings. The translations (English-to-Spanish) for these experiments were collected through deterministic logging of product titles on the Spanish eBay website. Ratings were either (1) *directly* collected from users who express their judgment of product title translation quality by giving one to five stars, or (2) *indirectly* inferred by comparing Spanish user queries to Spanish translations for (successfully) retrieved product titles.

Algorithms for counterfactual off-policy learning of statistical translation models were adapted to neural models and their quality first showed success in simulations on the same domain, but failed to improve over baseline models with the collection of direct eBay user ratings. In the second use case, where rewards for reinforcement can be inferred for multiple translations per input (since only indirectly computed), minimum-risk training approaches yielded the best results. Just as in Chapter 3, control variates (additive and now also multiplicative) proved themselves very effective, and newly introduced neural reward estimators had a stabilizing effect in this challenging domain. In addition, we discovered the effectiveness of the simple bandit-to-supervised technique that demonstrated that self-training should be considered as a strong baseline.

Puzzled by the large mismatch between the success of the same algorithms in simulation and in practice in the eBay use case, we deepened the investigations in the reliability of *absolute vs relative* machine translation quality ratings and their influence on down-stream off-policy learning. In a small-scale annotation study, we collected five-point and pairwise preference ratings for TED talk translations from German to English from semi-professional translators. These ratings, were used to (1) quantify and analyze *reliability* in terms of inter- and intra-rater agreement per rating type, then (2) train neural reward estimation models (regression models for five-point rations and Bradley-Terry models for pairwise ratings), and finally (3) to adapt an NMT model by reinforcing sampled translations with the trained reward estimators.

While the pairwise preference interface performed only slightly weaker in terms of reliability, it showed clearer disadvantages compared to the absolute ratings further down-stream. Comparing this reward-estimator-led learning with simulated and human ratings to the algorithms proposed in Chapter 4, we learned that the reward estimators help crucially to reduce noise and guide the learning MT model through reinforcement.

One of the incentives for developing algorithms for bandit structured prediction was the hope to lower feedback costs in comparison to full supervision. While previous chapters solely investigated weak supervision, Chapter 6 eventually quantified the differences in feedback cost and effectiveness between learning from *error corrections* (full supervision) and *error markings* (weak supervision). In an annotation study with semi-professional translators we analyzed their annotation actions, agreement, and the overall effort, (dis-)advantages of both feedback modes, and the learning opportunities for the underlying NMT model.

The novel interaction mode with error markings, that has previously only been used in simulations, shifts the task of credit/blame assignment to the human—as opposed to sentence-level feedback, where the machine is left to solve it. The study showed that this mode requires several magnitudes fewer effort and therefore lower costs than classic post-editing, but still leads to significant improvements in a domain adaptation task from news to TED translations from English to German. Moreover, it might constitute the sweet spot between post-edits that usually come with high cost and human over-editing, and sentence-level ratings (as in Chapters 4 and 5) that lack specificity.

The final chapter provided a more abstract and global perspective on the question which types of supervision are the most efficient in terms of feedback cost and model improvement for training NMT models. In human learning, the idea which kind of feedback should be requested from a teacher, is one aspect of self-regulated learning. Instead of committing to one feedback type (e.g. five-star ratings or corrections), the model, equipped with a *meta-learning regulator* module, learns to choose supervision types balancing costs vs. effectiveness of resulting gradient updates. In the experiments, we added self-training and fully-supervised learning to our previously investigated weakly-supervised learning mode, in order to give the model the freedom to choose between each of them based on the current context, the learning experience, and the individual supervision costs. Translation models for English-to-German were adapted to TED talks and a books domain, with the regulator deciding on their supervision and being trained through reinforcement with the ratio of model improvement to supervision costs.

We showed in the evaluations that the regulator was able to find cheaper feedback modes than fully-supervised or stream-based active learning by combining the given supervision modes in batch updates. The trained regulator was shown to operate similar to a $\epsilon$-greedy algorithm tuned for $\epsilon$, due to the problem of assigning credit for model improvement to individual samples contained in a batch update.

In summary, this thesis investigated online policy learning for bandit NMT in simulation, off-policy learning with direct and indirect e-commerce user feedback, delivered insights into reliability and learnability of absolute and pairwise, token-level and sequence-level judgments and corrections, and proposed a novel meta-learning algorithm for self-regulation that smoothly integrates and balances fully, weakly and self-supervised interactive policy learning.

## 8.2 Limitations and Future Directions

Besides the positive outcomes and successful algorithms, this thesis also uncovered the following limitations of the approach of reinforcement learning for NMT, which simultaneously point to promising future directions of research.

**Superiority of supervised learning.** We found a number of techniques that allowed us to turn the reinforcement learning problem into a problem that can be addressed with supervised training objectives, for example the bandit-to-supervised conversion in Section 4.3, which was a strong baseline for model adaptation that did not require any human feedback, and the self-training option which was largely preferred by the regulator in Section 7.2.2. The token-level reinforced objective from Section 6.2 was also discovered in the context of supervised domain adaptation by instance weighting (Yan et al., 2018). It can be considered safer in practice to collect feedback, use it to assess the overall quality of the system, and only then use parts of the rated data to carefully curate new training or test instances for supervised re-training, than to directly trust the feedback for online or offline updates. These less sophisticated techniques of exploiting feedback, might seem more attractive on the risk-benefit scale for industrial adoption than reinforcement learning, and do not require any change of the standard training procedures.

**Dependence on the reward distribution.** Relative scaled rewards that are dependent on the reward history consistently outperformed absolute, history-agnostic rewards (see Chapter 3). The neural models were sensitive to the choice between training with -sBLEU vs 1-BLEU (see Section 3.1.2), which differ in the scale of the updates. And if the model's quality is too weak to begin with, learning completely fails (e.g., in the case of starting from scratch). These cases illustrate how dependent the neural models are on "well-distributed" rewards, that come in not too sparse and with the right scale. In Chapter 4 we also observed the effect of noise: if it is dominating the reward signal, our models failed to improve. However, it is unlikely to find noise-free rewards in the wild, and, as translation quality estimation is in most of the cases only vaguely defined and context-dependent (illustrated in the study on human-chosen rewards in Chapter 6), future methods should aim to acknowledge and model this uncertainty in the reward signal.

**Evaluation.** In expected reward training, the reward should express the desired evaluation metric that one seeks to maximize. In practice, however, we treated the rewards obtained from humans rather as an auxiliary reward and mainly seek to improve automatic metrics like BLEU or METEOR. And if they did not align (see Chapter 4), we failed to find any improvement. The reasons for this training-evaluation mismatch are scalability and human effort: Ideally, each model trained with human rewards would also be tested in an evaluation with humans, under the same conditions and with the same interface as during training. This is infeasible if multiple models need to get evaluated or tuned, and due to the variance in the human reward function, one would have to collect a large amount of rewards to obtain reliable measurements (and probably pre-process, filter or normalize them). An online evaluation procedure like in the shared task evaluation (Section 3.5) would be a better solution, but rarely satisfies the production deployment criteria for MT industry, where the standard procedure is to regularly evaluate against benchmarks to ensure that newly deployed models actually improve over the previous ones—not as in ad placement, where online evaluation metrics like click-through-rate are widely adopted (Joachims, 2002). Finding evaluation interfaces that scale and at the same time satisfy production requirements and ideally also allow replication, would greatly facilitate the adoption of reward-trained and evaluated NLP models.

**On- vs. off-policy learning.** Off-policy learning has the advantage over online learning of allowing for pre-processing (e.g., filter by variance, normalize by user) before the actual learning process. Similarly, hyperparameters over multiple training runs can be tuned. However, it suffers from limited exploration and hence immediate online improvement, which might desirable in use cases with expert users (Wuebker and Simianer, 2019). In our experiments with online policy updates (Chapters 3 and 7), we discarded the information about the interaction after one update with the gathered reward. This might be a missed opportunity to collect a log like in off-policy learning. This log could be used to make stabilizing batch updates to the policy (much like a replay memory (Mnih et al., 2015)), and for reward estimator training and tuning, that do not involve any new feedback requests. Future work should hence consider interleaved online and offline updates to improve both sample efficiency and implementation in production environments.

# Bibliography

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*. Savannah, GA, USA.

Alekh Agarwal, Ofer Dekel, and Liu Xiao. 2010. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *The 23rd Annual Conference on Learning Theory (COLT)*. Haifa, Israel.

Alekh Agarwal, Daniel Hsu, Satyen Kale, John Langford, Lihong Li, and Robert E. Schapire. 2014. Taming the monster: A fast and simple algorithm for contextual bandits. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*. Beijing, China.

Cem Akkaya, Alexander Conrad, Janyce Wiebe, and Rada Mihalcea. 2010. Amazon mechanical turk for subjectivity word sense disambiguation. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. Los Angeles, CA, USA.

Dilip Arumugam, Jun Ki Lee, Sophie Saskin, and Michael L Littman. 2019. Deep reinforcement learning from policy-dependent human feedback. *arXiv preprint arXiv:1902.04257* .

Peter Auer. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research* 3(Nov):397–422.

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. 2002. The nonstochastic multiarmed bandit problem. *SIAM J. on Computing* 32(1):48–77.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on*

*Empirical Methods in Natural Language Processing (EMNLP)*. Edinburgh, Scotland.

Wilker Aziz, Ruslan Mitkov, and Lucia Specia. 2013. Ranking machine translation systems via post-editing. In *International Conference on Text, Speech and Dialogue*. pages 410–418.

R Harald Baayen, Douglas J Davidson, and Douglas M Bates. 2008. Mixed-effects modeling with crossed random effects for subjects and items. *Journal of memory and language* 59(4):390–412.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. An actor-critic algorithm for sequence prediction. In *Proceedings of the International Conference on Learning Representations (ICLR)*. Toulon, France.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations (ICLR)*. San Diego, California.

Sergio Barrachina, Oliver Bender, Francisco Casacuberta, Jorge Civera, Elsa Cubel, Shahram Khadivi, Antonio Lagarda, Hermann Ney, Jesús Tomás, Enrique Vidal, and Juan-Miguel Vilar. 2009. Statistical approaches to computer-assisted translation. *Computational Linguistics* 35(1).

Douglas Bates, Martin Mächler, Ben Bolker, and Steve Walker. 2015. Fitting linear mixed-effects models using lme4. *Journal of Statistical Software* 67(1):1–48.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research* 3(Feb):1137–1155.

Luisa Bentivogli, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. 2016. Neural versus phrase-based machine translation quality: a case study. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Austin, Texas.

Nicola Bertoldi, Patrick Simianer, Mauro Cettolo, Katharina Wäschle, Marcello Federico, and Stefan Riezler. 2014. Online adaptation to post-edits for phrase-based statistical machine translation. *Machine Translation* 29:309–339.

Dimitri P. Bertsekas and John N. Tsitsiklis. 1996. *Neuro-Dynamic Programming*. Athena Scientific.

Ondřej Bojar, Yvette Graham, Amir Kamran, and Miloš Stanojević. 2016a. Results of the WMT16 metrics shared task. In *Proceedings of the First Conference on Machine Translation (WMT)*. Berlin, Germany.

Ondřej Bojar, Roman Sudarikov, Tom Kocmi, Jindřich Helcl, and Ondřej Cıfka. 2016b. UFAL submissions to the IWSLT 2016 MT track. In *Proceedings of the 13rd International Workshop on Spoken Language Translation (IWSLT)*. Seattle, WA.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016c. Findings of the 2016 conference on machine translation. In *WMT*. Berlin, Germany.

Léon Bottou. 2004. Stochastic learning. In Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch, editors, *Advanced Lectures on Machine Learning*, Springer, Berlin, Heidelberg, pages 146–168.

Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X. Charles, D. Max Chickering, Elon Portugaly, Dipanakar Ray, Patrice Simard, and Ed Snelson. 2013. Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research* 14:3207–3260.

Ralph Allan Bradley and Milton E. Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika* 39(3-4):324–345.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a "siamese" time delay neural network. In *Advances In Neural Information Processing Systems (NeurIPS)*. Denver, CO, USA.

Peter Brown, John Cocke, S Della Pietra, V Della Pietra, Frederick Jelinek, Robert Mercer, and Paul Roossin. 1988. A statistical approach to language translation. In *Proceedings of the 12th Conference on Computational Linguistics (COLING)*. Budapest, Hungary.

Peter Brown, Stephen Della Pietra, Vincent Pietra, and Robert Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics* 19:263–311.

Sébastian Bubeck and Nicolò Cesa-Bianchi. 2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning* 5(1):1–122.

Róbert Busa-Fekete and Eyke Hüllermeier. 2014. A survey of preference-based online learning with bandit algorithms. In *International Conference on Algorithmic Learning Theory (ALT)*. Bled, Slovenia, pages 18–39.

Iacer Calixto, Daniel Stein, Evgeny Matusov, Pintu Lohar, Sheila Castilho, and Andy Way. 2017. Using images to improve machine-translating e-commerce product listings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. Valencia, Spain.

Michael Carl, Barbara Dragsted, Jakob Elming, Daniel Hardt, and Arnt Lykke Jakobsen. 2011. The process of post-editing: a pilot study. *Copenhagen Studies in Language* 41:131–142.

Asuncion Castano and Francisco Casacuberta. 1997. A connectionist approach to machine translation. In *Fifth European Conference on Speech Communication and Technology (EUROSPEECH)*. Rhodes, Greece.

Carlos Celemin, Javier Ruiz-del Solar, and Jens Kober. 2019. A fast hybrid reinforcement learning framework with human corrective feedback. *Autonomous Robots* 43(5):1173–1186.

Nicolò Cesa-Bianchi and Gábor Lugosi. 2012. Combinatorial bandits. *Journal of Computer and System Sciences* 78:1401–1422.

Olivier Chapelle, Eren Masnavoglu, and Romer Rosales. 2014. Simple and scalable response prediction for display advertising. *ACM Trans. on Intelligent Systems and Technology* 5(4).

Boxing Chen and Colin Cherry. 2014. A systematic comparison of smoothing techniques for sentence-level BLEU. In *Proceedings of the Ninth Workshop on Statistical Machine Translation (WMT)*. Baltimore, Maryland, USA.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*. Melbourne, Australia.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*. Ann Arbor, MI, USA.

David Chiang. 2012. Hope and fear for discriminative training of statistical translation models. *Journal of Machine Learning Research* 12:1159–1187.

Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar.

Leshem Choshen, Lior Fox, Zohar Aizenbud, and Omri Abend. 2019. On the weaknesses of reinforcement learning for neural machine translation. *arXiv preprint arXiv:1907.01752* .

Paul F. Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems (NeurIPS)*. Long Beach, CA, USA.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *eprint arXiv:1412.3555* .

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*. Portland, OR, USA.

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Brussels, Belgium.

Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12:2461–2505.

Gonçalo M. Correia and André F. T. Martins. 2019. A simple and effective approach to automatic post-editing with transfer learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*. Florence, Italy.

Corinna Cortes, Mehryar Mohri, and Asish Rastogi. 2007. Magnitude-preserving ranking algorithms. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*. Corvallis, OR, USA.

Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research* 3(Jan):951–991.

Hoang Cuong and Khalil Sima'an. 2014. Latent domain translation models in mix-of-domains haystack. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics*. Dublin, Ireland.

Varsha Dani, Thomas P. Hayes, and Sham M. Kakade. 2007. The price of bandit information for online optimization. In *Advances In Neural Information Processing Systems (NeurIPS)*. Vancouver, Canada.

Michael Denkowski, Chris Dyer, and Alon Lavie. 2014. Learning from post-editing: Online model adaptation for statistical machine translation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. Gothenburg, Sweden.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the Sixth Workshop on Statistical Machine Translation (WMT)*. Edinburgh, Scotland.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*. Minneapolis, Minnesota.

Christos Dimitrakakis, Guangliang Li, and Nikolaos Tziortziotis. 2014. The reinforcement learning competition 2014. *AI Magazine* 35(3):61–65.

Miguel Domingo, Álvaro Peris, and Francisco Casacuberta. 2017. Segment-based interactive-predictive machine translation. *Machine Translation* 31(4):163–185.

Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2018. Banditsum: Extractive summarization as a contextual bandit. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Brussels, Belgium.

John C. Duchi, Michael I. Jordan, Martin J. Wainwright, and Andre Wibisono. 2015. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Translactions on Information Theory* 61(5):2788–2806.

Audrey Durand, Charis Achilleos, Demetris Iacovides, Katerina Strati, Georgios D Mitsis, and Joelle Pineau. 2018. Contextual bandits for adapting treatment in a mouse model of de novo carcinogenesis. In *Machine Learning for Healthcare Conference*. pages 67–82.

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (ACL Demo)*. Uppsala, Sweden.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science* 14(2):179–211.

Meng Fang, Yuan Li, and Trevor Cohn. 2017. Learning how to active learn: A deep reinforcement learning approach. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Copenhagen, Denmark.

Abraham D. Flaxman, Adam Tauman Kalai, and H. Brendan McMahan. 2005. Online convex optimization in the bandit setting: gradient descent without a gradient. In *SODA*. Philadelphia, PA, USA.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin* 76(5):378.

John Fox. 2002. Linear mixed models. *Appendix to An R and S-PLUS Companion to Applied Regression* .

Markus Freitag and Yaser Al-Onaizan. 2016. Fast domain adaptation for neural machine translation. *arXiv preprint arXiv:1612.06897* .

Yoav Freund, Ray Iyer, Robert E. Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research* 4:933–969.

Michael C. Fu. 2006. Gradient estimation. In S.G. Henderson and B.L. Nelson, editors, *Handbook in Operations Research and Management Science*, Elsevier, volume 13, pages 575–616.

Johannes Fürnkranz and Eyke Hüllermeier. 2010. Preference learning and ranking by pairwise comparison. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, Springer.

Y Gal and Z Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*. New York City, NY.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*. Vancouver, Canada.

Saeed Ghadimi and Guanghui Lan. 2012. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM J. on Optimization* 4(23):2342–2368.

Kevin Gimpel and Noah A. Smith. 2010. Softmax-margin training for structured log-linear models. Technical Report CMU-LTI-10-008, Carnegie Mellon University, Pittsburgh, PA.

Keith Godwin and Paul Piwek. 2016. Collecting reliable human judgements on machine-generated language: The case of the QG-STEC data. In *Proceedings of the 9th International Natural Language Generation conference (INLG)*. Edinburgh, UK.

Yoav Goldberg. 2016. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research* 57:345–420.

David Grangier and Michael Auli. 2018. QuickEdit: Editing text & translations by crossing words out. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, LA, USA.

Spence Green, Sida I. Wang, Jason Chuang, Jeffrey Heer, Sebastian Schuster, and Christopher D. Manning. 2014. Human effort and machine learnability in computer aided translation. In *Proceedings the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar.

Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. 2017. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*. Marina Bay Sands, Singapur.

Ana Guerberof Arenas. 2008. Productivity and quality in the post-editing of outputs from translation memories and machine translation. *Localisation Focus The International Journal of Localisation* 7(1):11–21.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*. Sardinia, Italy.

Francisco Guzmán, Ahmed Abdelali, Irina Temnikova, Hassan Sajjad, and Stephan Vogel. 2015. How do humans evaluate machine translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation (WMT)*. Lisbon, Portugal.

Kazuma Hashimoto, Akiko Eriguchi, and Yoshimasa Tsuruoka. 2016. Domain adaptation and attention-based unknown word replacement in chinese-to-japanese neural machine translation. In *COLING Workshop on Asian Translation*. Osaka, Japan.

John Hattie and Gregory M. Donoghue. 2016. Learning strategies: a synthesis and conceptual model. *NPJ Science of Learning* 1:16013–16013.

John Hattie and Helen Timperley. 2007. The power of feedback. *American Educational Research Association* 77(1):81–112.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tieyan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances In Neural Information Processing Systems (NeurIPS)*. Barcelona, Spain.

Xiaodong He and Li Deng. 2012. Maximum expected BLEU training of phrase and lexicon translation models. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*. Jeju Island, Korea.

Jindřich Helcl and Jindřich Libovický. 2017. Neural Monkey: An Open-source Tool for Sequence Learning. *The Prague Bulletin of Mathematical Linguistics* 107:5–17.

Ralf Herbrich, Thore Graepel, and Klaus Obermayer. 2000. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*, MIT Press Cambridge, Cambridge, MA, pages 115–132.

Felix Hieber, Tobias Domhan, Michael Denkowski, David Vilar, Artem Sokolov, Ann Clifton, and Matt Post. 2017. Sockeye: A toolkit for neural machine translation. *arXiv preprint arXiv:1712.05690* .

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv eprint arXiv:1508.01991* .

W John Hutchins. 2000. *Early years in machine translation: memoirs and biographies of pioneers*, volume 97. John Benjamins Publishing.

Edward L. Ionides. 2008. Truncated importance sampling. *J. of Comp. and Graph. Stat.* 17(2):295–311.

Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. 2017. Learning a neural semantic parser from user feedback. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*. Vancouver, Canada.

Emily Jamison and Iryna Gurevych. 2015. Noise or additional information? leveraging crowdsource annotation item agreement for natural language tasks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Lisbon, Portugal.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015a. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*. Beijing, China.

Sébastien Jean, Orhan Firat, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015b. Montreal neural machine translation systems for WMT'15. In *WMT*. Lisbon, Portugal.

Laura Jehl, Carolin Lawrence, and Stefan Riezler. 2019. Learning neural sequence-to-sequence models from weak feedback with bipolar ramp loss. *Transactions of the Association for Computational Linguistics* 7:233–248.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD*. New York, NY.

Sham M Kakade. 2002. A natural policy gradient. In *Advances in Neural Information Processing Systems (NeurIPS*. Vancouver, Canada, pages 1531–1538.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Seattle, WA, USA.

Sariya Karimova, Patrick Simianer, and Stefan Riezler. 2018. A user-study on online adaptation of neural machine translation to human post-edits. *Machine Translation* 32(4):309–324.

Fabio Kepler, Jonay Trénous, Marcos Treviso, Miguel Vera, António Góis, M. Amin Farajian, António V. Lopes, and André F. T. Martins. 2019. Unbabel's participation in the WMT19 translation quality estimation shared task. In *Proceedings of the Fourth Conference on Machine Translation (WMT)*. Florence, Italy.

Hyun Kim, Jong-Hyeok Lee, and Seung-Hoon Na. 2017. Predictor-estimator using multilevel task learning with stack propagation for neural quality estimation. In *Proceedings of the Second Conference on Machine Translation (WMT)*. Copenhagen, Denmark.

Hyun Kim, Joon-Ho Lim, Hyun-Ki Kim, and Seung-Hoon Na. 2019. QE BERT: Bilingual BERT using multi-task learning for neural quality estimation. In *Proceedings of the Fourth Conference on Machine Translation (WMT)*. Florence, Italy.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*. San Diego, CA, USA.

Guillaume Klein, Yoon Kim, Yuntian Deng, Vincent Nguyen, Jean Senellart, and Alexander Rush. 2018. OpenNMT: Neural machine translation toolkit. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (AMTA)*. Boston, MA.

Kevin Knight. 1999. Decoding complexity in word-replacement translation models. *Computational Linguistics* 25(4):607–615.

W. Bradley Knox and Peter Stone. 2009. Interactively shaping agents via human reinforcement: The TAMER framework. In *Proceedings of the International Conference on Knowledge Capture (K-CAP)*. Redondo Beach, CA, USA.

Philipp Koehn. 2009. *Statistical machine translation*. Cambridge University Press.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*. Prague, Czech Republic.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*. Edmonton, Canada.

Philipp Koehn and Josh Schroeder. 2007. Experiments in domain adaptation for statistical machine translation. In *WMT*. Prague, Czech Republic.

Augustine Kong. 1992. A note on importance sampling using standardized weights. *University of Chicago, Dept. of Statistics, Tech. Rep* 348.

Maarit Koponen. 2016. *Machine Translation Post-Editing and Effort. Empirical Studies on the Post-Editing Process*. Ph.D. thesis, University of Helsinki.

Julia Kreutzer, Jasmijn Bastings, and Stefan Riezler. 2019. Joey NMT: A minimalist NMT toolkit for novices. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*. Hong Kong, China.

Julia Kreutzer, Nathaniel Berger, and Stefan Riezler. 2020. Correct me if you can: Learning from error corrections and markings. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation (EAMT)*. Virtual.

Julia Kreutzer, Shahram Khadivi, Evgeny Matusov, and Stefan Riezler. 2018a. Can neural machine translation be improved with user feedback? In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Industry Track (NAACL-HLT)*. New Orleans, LA, USA.

Julia Kreutzer and Stefan Riezler. 2019. Self-regulated interactive sequence-to-sequence learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*. Florence, Italy.

Julia Kreutzer, Artem Sokolov, and Stefan Riezler. 2017. Bandit structured prediction for neural sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*. Vancouver, Canada.

Julia Kreutzer, Joshua Uyheng, and Stefan Riezler. 2018b. Reliability and learnability of human bandit feedback for sequence-to-sequence reinforcement learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*. Melbourne, Australia.

Klaus Krippendorff. 2013. *Content Analysis. An Introduction to Its Methodology*. Sage, third edition.

David Krueger, Jan Leike, Owain Evans, and John Salvatier. 2016. Active reinforcement learning: Observing rewards at a cost. In *Advances In Neural Information Processing Systems (NeurIPS)*. Barcelona, Spain.

Gaurav Kumar, George Foster, Colin Cherry, and Maxim Krikun. 2019. Reinforcement learning based curriculum optimization for neural machine translation. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. Minneapolis, MN, USA.

Tsz Kin Lam, Julia Kreutzer, and Stefan Riezler. 2018. A reinforcement learning approach to interactive-predictive neural machine translation. In *Proceedings of the 21st Annual Conference of the European Association for Machine Translation (EAMT)*. Alicante, Spain.

Tsz Kin Lam, Shigehiko Schamoni, and Stefan Riezler. 2019. Interactive-predictive neural machine translation through reinforcement and imitation. In *Proceedings of the Machine Translation Summit MTSUMMIT XVII*.

John Langford, Alexander Strehl, and Jennifer Wortman. 2008. Exploration scavenging. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*. Helsinki, Finland.

John Langford and Tong Zhang. 2007. The epoch-greedy algorithm for contextual multi-armed bandits. In *Advances In Neural Information Processing Systems (NeurIPS)*. Vancouver, Canada.

Richard Larsen and Morris Marx. 2012. *An Introduction to Mathematical Statistics and Its Applications*. Prentice Hall, fifth edition.

Alon Lavie and Michael J. Denkowski. 2009. The meteor metric for automatic evaluation of machine translation. *Machine Translation* 23(2-3):105–115.

Carolin Lawrence, Pratik Gajane, and Stefan Riezler. 2017a. Counterfactual learning for machine translation: Degeneracies and solutions. In *Proceedings of the NeurIPS WhatIF Workshop*. Long Beach, CA.

Carolin Lawrence and Stefan Riezler. 2018. Improving a Neural Semantic Parser by Counterfactual Learning from Human Bandit Feedback. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*. Melbourne, Australia.

Carolin Lawrence, Artem Sokolov, and Stefan Riezler. 2017b. Counterfactual learning from bandit feedback under deterministic logging: A case study in statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Copenhagen, Denmark.

Florian Laws, Christian Scheible, and Hinrich Schütze. 2011. Active learning with Amazon mechanical turk. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Edinburgh, Scotland.

Hoa T. Le, Christophe Cerisara, and Alexandre Denis. 2017. Do convolutional networks need to be deep for text classification? *arXiv preprint arXiv:1707.04108*
.

Anton Leuski, Chin-Yew Lin, and Eduard Hovy. 2003. ineats: interactive multi-document summarization. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL)*. Sapporo, Japan.

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Austin, TX, USA.

Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. 2017. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Copenhagen, Denmark.

Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *WWW*. Raleigh, NC.

Zhifei Li and Jason Eisner. 2009. First-and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Edinburgh, UK.

Chen Liang, Jonathan Berant, Quoc Le, Kenneth D. Forbus, and Ni Lao. 2017. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*. Vancouver, Canada.

Jindřich Libovickỳ, Jindřich Helcl, Marek Tlustỳ, Pavel Pecina, and Ondřej Bojar. 2016. CUNI system for WMT16 automatic post-editing and multimodal translation tasks. In *WMT*. Berlin, Germany.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* .

Zachary C Lipton, John Berkowitz, and Charles Elkan. 2015. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019* .

Nick Littlestone. 1989. From on-line to batch learning. In *Proceedings of the 25th Annual Conference on Learning Theory (COLT)*. Santa Cruz, CA.

Ming Liu, Wray Buntine, and Gholamreza Haffari. 2018. Learning to actively learn neural machine translation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL)*. Brussels, Belgium.

Shixia Liu, Michelle X Zhou, Shimei Pan, Weihong Qian, Weijia Cai, and Xiaoxiao Lian. 2009. Interactive, topic-based visual text summarization and analysis. In *Proceedings of the 18th ACM conference on Information and knowledge management*. pages 543–552.

Arle Lommel, Maja Popovic, and Aljoscha Burchardt. 2014. Assessing inter-annotator agreement for translation error annotation. In *MTE: Workshop on Automatic and Manual Metrics for Operational Translation Evaluation*.

Adam Lopez. 2008. Statistical machine translation. *ACM ACM Computing Surveys* 40.

Minh-Thang Luong and Christopher D. Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*. Da Nang, Vietnam.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015a. Effective approaches to attention-based neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Lisbon, Portugal, pages 1412–1421.

Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba. 2015b. Addressing the rare word problem in neural machine translation. In *Proceedings of the 53th Annual Meeting of the Association for Computational Linguistics ACL*. Beijing, China.

Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*. Atlanta, GA, USA.

James MacGlashan, Mark K. Ho, Robert Loftin, Bei Peng, Guan Wang, David L. Roberts, Matthew E. Taylor, and Michael L. Littman. 2017. Interactive learning from policy-dependent human feedback. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*. Sydney, Australia.

Benjamin Marie and Aurélien Max. 2015. Touch-based pre-post-editing of machine translation output. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Lisbon, Portugal.

André Martins, Marcin Junczys-Dowmunt, Fabio Kepler, Ramón Astudillo, Chris Hokamp, and Roman Grundkiewicz. 2017. Pushing the limits of translation quality estimation. *Transactions of the Association for Computational Linguistics (TACL)* 5:205–218.

Jeff Michels, Ashutosh Saxena, and Andrew Y. Ng. 2005. High speed obstacle avoidance using monocular vision and reinforcement learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances In Neural Information Processing Systems (NeurIPS)*. Lake Tahoe, CA.

T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling. 2015. Never-ending learning. In *Proceedings of the 29th Conference on Artificial Intelligence (AAAI)*. Austin, TX, USA.

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*. Edinburgh, Scotland.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning (ICML)*. New York City, NY, USA, pages 1928–1937.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. In *NeurIPS Deep Learning Workshop*. Lake Tahoe, NV, USA.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518:529–533.

Michael C Mozer. 1995. A focused backpropagation algorithm for temporal. *Backpropagation: Theory, architectures, and applications* 137.

Ramon P Neco and Mikel L Forcada. 1997. Asynchronous translations with recurrent neural nets. In *Proceedings of International Conference on Neural Networks (ICNN)*. volume 4, pages 2535–2540.

Barry L Nelson. 1987. On control variate estimators. *Computers & Operations Research* 14(3):219–225.

Khanh Nguyen, Hal Daumé, and Jordan Boyd-Graber. 2017. Reinforcement learning for bandit neural machine translation with simulated feedback. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Copenhagen, Denmark.

Joel T. Nigg. 2017. Annual research review: On the relations among self-regulation, self-control, executive functioning, effortful control, cognitive control, impulsivity, risk-taking, and inhibition for developmental psychopathology. *Journal of Child Psychology and Psychiatry* 58(4):361–383.

Eric W. Noreen. 1989. *Computer Intensive Methods for Testing Hypotheses. An Introduction*. Wiley, New York.

Mohammad Norouzi, Samy Bengio, zhifeng Chen, Navdeep Jaitly, Mike Schuster, Yonghui Wu, and Dale Schuurmans. 2016. Reward augmented maximum likelihood for neural structured prediction. In *Advances In Neural Information Processing Systems (NeurIPS)*. Barcelona, Spain.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*. Edmonton, Canada.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*. Philadelphia, PA, USA.

Ernesto Panadero. 2017. A review of self-regulated learning: Six models and four directions of research. *Frontiers in Psychology* 8(422):1–28.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL)*. Philadelphia, PA, USA.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013a. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*. Atlanta, GA.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013b. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*. Atlanta, GA, USA.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NeurIPS Autodiff Workshop*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304* .

Álvaro Peris and Francisco Casacuberta. 2018. Active learning for interactive neural machine translation of data streams. In *Proceedings of the 22nd Conference on Computational Natural Language Learning (CONLL)*. Brussels, Belgium.

Pavel Petrushkov, Shahram Khadivi, and Evgeny Matusov. 2018. Learning from chunk-based feedback in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*. Melbourne, Australia.

Patrick M. Pilarski, Michael R. Dawson, Thomas Degris, Farbod Fahimi, Jason P. Carey, and Richard S. Sutton. 2011. Online human training of a myoelectric prosthesis controller via actor-critic reinforcement learning. In *Proceedings of the IEEE International Conference on Rehabilitation Robotics*. Zürich, Switzerland.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*. Florence, Italy.

Boris T. Polyak. 1964. Some methods of speeding up the convergence of iteration methods. *USSR Comp. Math. and Math. Phys.* 4(5):1–17.

Boris T. Polyak. 1987. *Introduction to Optimization*. Optimization Software, Inc., New York.

Maja Popović. 2015. chrf: character n-gram f-score for automatic mt evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation (WMT)*. Lisbon, Portugal.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation (WMT)*. Brussels, Belgium.

S Avinesh PV, Benjamin Hättasch, Orkan Özyurt, Carsten Binnig, and Christian M Meyer. 2018. Sherlock: A system for interactive summarization of large text collections. *Proceedings of the VLDB Endowment* 11(12):1902–1905.

Rajesh Ranganath, Sean Gerrish, and David M. Blei. 2014. Black box variational inference. In *AISTATS*. Reykjavik, Iceland.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *Proceedings of the International Conference on Learning Representation (ICLR)*. San Juan, Puerto Rico.

Kenneth Rose. 1998. Deterministic annealing for clustering, compression, classification, regression and related optimization problems. *IEEE* 86(11).

Sheldon M. Ross. 2013. *Simulation*. Elsevier, fifth edition.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *Nature* 323(6088):533–536.

Germán Sanchis-Trilles, Vicent Alabau, Christian Buck, Michael Carl, Francisco Casacuberta, Mercedes García-Martínez, Ulrich Germann, Jesús González-Rubio, Robin L. Hill, Philipp Koehn, Luis A. Leiva, Bartolomé Mesa-Lao, Daniel Ortiz-Martínez, Herve Saint-Amand, Chara Tsoukala, and Enrique Vidal. 2014. Interactive translation prediction versus conventional post-editing in practice: a study with the casmacat workbench. *Machine Translation* 28(3):217–235.

Jürgen Schmidhuber, Jieyu Zhao, and Marco Wiering. 1996. Simple principles of metalaerning. Technical Report 69 96, IDSIA, Lugano, Switzerland.

Holger Schwenk. 2008. Investigations on large-scale lightly-supervised training for statistical machine translation. In *Proceedings of the 5th International Workshop on Spoken Language Translation (IWSLT)*.

Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nadejde. 2017. Nematus: a toolkit for neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. Valencia, Spain.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Edinburgh neural machine translation systems for wmt 16. In *Proceedings of the First Conference on Machine Translation (WMT)*. Berlin, Germany.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*. Berlin, Germany.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016c. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*. Berlin, Germany.

Christophe Servan, Josep Crego, and Jean Senellart. 2016. Domain specialization: a post-training domain adaptation for neural machine translation. *eprint arXiv:1612.06141* .

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Honolulu, Hawaii.

Burr Settles, Mark Craven, and Lewis Friedland. 2008. Active learning with real annotation costs. In *Proceedings of the NeurIPS Workshop on Cost-Sensitive Learning*. Vancouver, Canada.

Shai Shalev-Shwartz. 2012. Online learning and online convex optimization. *Foundations and Trends in Machine Learning* 4(2):107–194.

Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. 2010. Learnability, stability and uniform convergence. *Journal of Machine Learning Research* 11:2635–2670.

Ori Shapira, Hadar Ronen, Meni Adler, Yael Amsterdamer, Judit Bar-Ilan, and Ido Dagan. 2017. Interactive abstractive summarization for event news tweets. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP): System Demonstrations*. Copenhagen, Denmark.

Amr Sharaf and Hal Daumé III. 2017. Structured prediction via learning to search under bandit feedback. In *Workshop on Structured Prediction for NLP*. Copenhagen, Denmark.

Amr Sharaf, Shi Feng, Khanh Nguyen, Kiante Brantley, and Hal Daumé III. 2017. The umd neural machine translation systems at wmt17 bandit learning task. In *Proceedings of the Second Conference on Machine Translation (WMT)*. Copenhagen, Denmark.

Shiqi Shen, Yong Cheng, Zongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*. Berlin, Germany.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. 2016. Mastering the game of go with deep neural networks and tree search. *Nature* 529:484–489.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In *International Conference on Machine Learning (ICML)*. Beijing, China, pages 387–395.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *Nature* 550(7676):354–359.

Patrick Simianer, Sariya Karimova, and Stefan Riezler. 2016. A post-editing interface for immediate adaptation in statistical machine translation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*. Osaka, Japan.

Patrick Simianer, Stefan Riezler, and Chris Dyer. 2012. Joint feature selection in distributed stochastic learning for large-scale discriminative training in SMT. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*. Jeju Island, Korea.

David A. Smith and Jason Eisner. 2006. Minimum risk annealing for training log-linear models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL)*. Sydney, Australia.

Noah A. Smith. 2011. *Linguistic Structure Prediction*. Morgan and Claypool.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the Conference of the Association for Machine Translation in the Americas (AMTA)*. Cambridge, MA, pages 223–231.

Artem Sokolov, Julian Hitschler, and Stefan Riezler. 2018. Sparse stochastic zeroth-order optimization with an application to bandit structured prediction. *arxiv preprint arxiv:1806.04458* .

Artem Sokolov, Julia Kreutzer, Christopher Lo, and Stefan Riezler. 2016a. Learning structured predictors from bandit feedback for interactive NLP. In

*Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany.

Artem Sokolov, Julia Kreutzer, Stefan Riezler, and Christopher Lo. 2016b. Stochastic structured prediction under bandit feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*. Barcelona, Spain.

Artem Sokolov, Julia Kreutzer, Kellen Sunderland, Pavel Danchenko, Witold Szymaniak, Hagen Fürstenau, and Stefan Riezler. 2017. A shared task on bandit learning for machine translation. In *Proceedings of the Second Conference on Machine Translation (WMT)*. Copenhagen, Denmark.

Artem Sokolov, Stefan Riezler, and Tanguy Urvoy. 2015. Bandit structured prediction for learning from user feedback in statistical machine translation. In *MT Summit XV*. Miami, FL.

Mikhail V. Solodov. 1998. Incremental gradient algorithms with stepsizes bounded away from zero. *Computational Optimization and Applications* 11:23–35.

Lucia Specia, Dhwaj Raj, and Marco Turchi. 2010. Machine translation evaluation versus quality estimation. *Machine translation* 24(1):39–50.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014a. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15(1):1929–1958.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014b. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15:1929–1958.

Miloš Stanojević and Khalil Sima'an. 2014. Fitting sentence level translation evaluation with many dense features. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar.

Alexander L. Strehl, John Langford, Lihong Li, and Sham M. Kakade. 2010. Learning from logged implicit exploration data. In *Advances in Neural Information Processing Sytems (NeurIPS)*. Vancouver, Canada.

Ilya Sutskever, James Martens, George E. Dahl, and Geoffrey E. Hinton. 2013. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*. Atlanta, GA.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances In Neural Information Processing Systems (NeurIPS)*. Montreal, Canada.

Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement Learning. An Introduction*. The MIT Press.

Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 2000a. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processings Systems (NeurIPS)*. Vancouver, Canada.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000b. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems (NeurIPS)*. Denver, Colorado.

Adith Swaminathan and Thorsten Joachims. 2015a. Counterfactual risk minimization: Learning from logged bandit feedback. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. Lille, France.

Adith Swaminathan and Thorsten Joachims. 2015b. The self-normalized estimator for counterfactual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*. Montreal, Canada.

Csaba Szepesvári. 2009. *Algorithms for Reinforcement Learning*. Morgan & Claypool.

William R Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25(3/4):285–294.

Sebastian Thrun and Lorien Pratt, editors. 1998. *Learning to Learn*. Kluwer, Dortrecht, MA, USA.

Louis Leon Thurstone. 1927. A law of comparative judgement. *Psychological Review* 34:278–286.

Jörg Tiedemann. 2009. News from opus-a collection of multilingual parallel corpora with tools and interfaces. In *Recent advances in natural language processing*. volume 5, pages 237–248.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC)*. Istanbul, Turkey.

Marco Turchi, Matteo Negri, M Amin Farajian, and Marcello Federico. 2017. Continuous learning from human post-edits for neural machine translation. *The Prague Bulletin of Mathematical Linguistics* 108(1):233–244.

Joseph P Turian, Luke Shea, and I Dan Melamed. 2003. Evaluation of machine translation and its evaluation. *Proceedings of MT Summit, 2003* pages 386–393.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*. Long Beach, CA, USA.

Suzan Verberne, Emiel Krahmer, Iris Hendrickx, Sander Wubben, and Antal van den Bosch. 2018. Creating a reference data set for the summarization of discussion forum threads. *Language Resources and Evaluation* 52(2):461–483.

Garrett Warnell, Nicholas Waytowich, Vernon Lawhern, and Peter Stone. 2018. Deep tamer: Interactive agent shaping in high-dimensional state spaces. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*.

Katharina Wäschle and Stefan Riezler. 2012. Structural and topical dimensions in multi-task patent translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Prague, Czech Republic.

Christopher Watkins. 1989. Learning from delayed rewards. *PhD thesis, Cambridge University* .

Tsung-Hsien Wen, Milica Gašić, Dongho Kim, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Prague, Czech Republic.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8:229–256.

James R Wilson. 1984. Variance reduction techniques for digital simulation. *American Journal of Mathematical and Management Sciences* 4(3-4):277–312.

Bodo Winter. 2013. Linear models and linear mixed effects models in R with linguistic applications. *arXiv preprint arXiv:1308.5499* .

Guillaume Wisniewski. 2017. Limsi submission for wmt'17 shared task on bandit learning. In *Proceedings of the Second Conference on Machine Translation (WMT)*. Copenhagen, Denmark.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771* .

Lijun Wu, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018a. A study of reinforcement learning for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Brussels, Belgium.

Lijun Wu, Yingce Xia, Fei Tian, Li Zhao, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018b. Adversarial neural machine translation. In *Asian Conference on Machine Learning (ACML)*. Beijing, China.

Lijun Wu, Yingce Xia, Li Zhao, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2017. Adversarial neural machine translation. *arXiv preprint arXiv:1704.06933* .

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* .

Joern Wuebker, Spence Green, and John DeNero. 2015. Hierarchical incremental adaptation for statistical machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Lisbon, Portugal.

Joern Wuebker and Patrick DeNero John Simianer. 2019. Compact personalized models for neural machine translation. In *Proceedings of the 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. Minneapolis, MN, USA.

Lijun Wun, Fei Tian, Yingce Xia, Yang Fan, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018. Learning to teach with dynamic loss functions. In *Advances In Neural Information Processing Systems (NeurIPS)*. Montreal, Canada.

Shen Yan, Leonard Dahlmann, Pavel Petrushkov, Sanjika Hewavitharana, and Shahram Khadivi. 2018. Word-based domain adaptation for neural machine translation. *Proceedings of the 15th International Workshop on Spoken Language Translation (IWSLT)* .

Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. 2018. Improving neural machine translation with conditional sequence generative adversarial nets. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*. New Orleans, LA, USA.

Ziyu Yao, Xiujun Li, Jianfeng Gao, Brian Sadler, and Huan Sun. 2019. Interactive semantic parsing for if-then recipes via hierarchical reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. Honolulu, HI, USA.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017a. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*. San Francisco, CA, USA.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017b. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*. San Francisco, CA, USA.

Yisong Yue and Thorsten Joachims. 2009. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*. Montreal, Canada.

Alan Yuille and Xuming He. 2012. Probabilistic models of vision and max-margin methods. *Frontiers of Electrical and Electronic Engineering* 7(1):94–106.

Barry J. Zimmerman and Dale H. Schunk, editors. 1989. *Self-Regulated Learning and Academic Achievement*. Springer, New York, NY, USA.

# Acknowledgments

I am very grateful for the support I received during my PhD from many sides.

First of all, I would like to thank my advisor Stefan Riezler for introducing me to research, machine translation and machine learning, for giving me lots of freedom and support to go on internships/summer schools/conferences, and guiding me through the PhD jungle.

In addition, I would like to thank my internship supervisors André, Artem, Shahram, Evgeny, George and Colin and their colleagues. They inspired me and helped me discover new (work)places and angles to research.

I was very lucky to work with awesome colleagues in Heidelberg—Artem, Carolin, Ana, Sariya, Hiko, Mayumi, Rafael, Tsz Kin, Patrick, Laura, inter alia. Special thanks to Michi, Nathan and $\phi$lipp (Ehrengast) for making Bürotimes so much fun. And big thanks to Sabrina and Cata for sharing the Coli journey till the very end. I'm glad we had so many years in HD together. Carolin, Cata, Sabrina, Michi, Nathan—thanks for proofreading parts of this thesis.

I am lucky to have met amazing friends who make Feierabend and Wochenende the best time of the week. Alina, Gwen, Laurent, Sophia, Cathy, Sabrina, Cata (triple mention!), Jenny and Daniel—Merci beaucoup! And thanks to the Auftakt crew for the (mostly) beautiful music and game nights. Thanks, Jasmijn, for being a partner in PhD-crime, and for building Joey NMT with me.

And most importantly, I would like to express huge thanks to my family for the decades of support, lots of love and the right amount of craziness.

# Appendix

# Appendix A

# Detailed Gradient Derivatives

## A.1 Derivative of the Score Function

The ER gradient update (Eq. 35) is composed of the score function (36) and the reward. In this section we will further derive the gradient of the score function to analyze the components of it in practice. In the following, $N = |\mathcal{V}_{trg}|$ describes the size of the output vocabulary.

$$\begin{aligned}\nabla \log p_\theta(y = j \mid x) &= \nabla_\theta \log \left[\text{softmax}(\mathbf{o})\right]_j \\ &= \frac{\nabla_\theta \left[\text{softmax}(\mathbf{o})\right]_j}{\left[\text{softmax}(\mathbf{o})\right]_j}\end{aligned} \tag{101}$$

The nominator of (101), the gradient of the softmax-normalized output score, is

$$\begin{aligned}\nabla_\theta \left[\text{softmax}(\mathbf{o})\right]_j &= \frac{\partial \left[\text{softmax}(\mathbf{o})\right]_j}{\partial \mathbf{o}} \times \frac{\partial \mathbf{o}}{\partial \theta} \\ &= \frac{\partial \exp(o_j)/\sum_{i=1}^N \exp(o_i)}{\partial \mathbf{o}} \times \frac{\partial \mathbf{o}}{\partial \theta} \\ &= \sum_k \left[\frac{\partial \exp(o_j)/\sum_{i=1}^N \exp(o_i)}{\partial o_k} \times \frac{\partial o_k}{\partial \theta}\right].\end{aligned} \tag{102}$$

We can then apply the quotient rule for $f(x) = \frac{g(x)}{h(x)}$: $f'(x) = \frac{g'(x)h(x) - h'(x)g(x)}{[h(x)]^2}$ with $g = \exp(o_j)$ and $h = \sum_{i=1}^N \exp(o_i)$ and

$$g' = \frac{\partial g}{\partial o_k} = \exp(o_k) \text{ if} j = k \text{ else } 0, \tag{103}$$

$$\begin{aligned}h' &= \frac{\partial \sum_{i=1}^N \exp(o_i)}{\partial o_k} = \sum_{i=1}^N \frac{\partial \exp(o_i)}{\partial o_k} \\ &= \sum_{i \neq k} \frac{\partial \exp(o_i)}{\partial o_k} + \frac{\partial \exp(o_k)}{\partial o_k} \\ &= \exp(o_k).\end{aligned} \tag{104}$$

Further calculations need to differentiate between $j = k$ and $j \neq k$:

1. $j = k$:

$$\begin{aligned} &\frac{\exp(o_j) \times \sum_{i=1}^N \exp(o_i) - \exp(o_k) \times \exp(o_j)}{\left[\sum_{i=1}^N \exp(o_i)\right]^2} \\ &= \frac{\exp(o_j)}{\sum_{i=1}^N \exp(o_i)} \times \frac{\sum_{i=1}^N \exp(o_i) - \exp(o_k)}{\sum_{i=1}^N \exp(o_i)} \\ &= [\text{softmax}(\mathbf{o})]_j \times (1 - [\text{softmax}(\mathbf{o})]_k) \end{aligned} \tag{105}$$

2. $j \neq k$:

$$\begin{aligned} &\frac{-\exp(o_k) \times \exp(o_j)}{\left[\sum_{i=1}^N \exp(o_i)\right]^2} \\ &= -[\text{softmax}(\mathbf{o})]_k \times [\text{softmax}(\mathbf{o})]_j \end{aligned} \tag{106}$$

Both cases can be summarized as

$$([[k = j]] - [\text{softmax}(\mathbf{o})]_k) \times [\text{softmax}(\mathbf{o})]_j \,. \tag{107}$$

Placing (107) back into (101), we obtain

$$\begin{aligned} \nabla_\theta \log p_\theta(y = j \mid k) &= \frac{\sum_k \left[([[k = j]] - [\text{softmax}(\mathbf{o})]_k) \times [\text{softmax}(\mathbf{o})]_j \times \frac{\partial o_k}{\partial \theta}\right]}{[\text{softmax}(\mathbf{o})]_j} \\ &= \sum_k \left[([[k = j]] - [\text{softmax}(\mathbf{o})]_k) \times \frac{\partial o_k}{\partial \theta}\right]. \end{aligned} \tag{108}$$

# Appendix B

# Hyperparameters

## B.1 SMT Hyperparameters

| | **ER** | **CE** | **PR** |
|---|---|---|---|
| News ($n$-best, dense) | $\gamma_t = 10^{-5}$ | $\gamma_t = 10^{-4.75}$ | $\gamma_t = 10^{-5}$ |
| News (h-graph, sparse) | $\gamma_t = 10^{-5}$ | $\gamma_t = 10^{-4}$ | $\lambda = 10^{-6}, c = 5 \cdot 10^{-3}, \gamma_t = 10^{-6}$ |

Table 37: Hyperparameter settings determined on validation sets for constant learning rate $\gamma_t$, momentum coefficient $\min\{1 - 1/(t/2+2), \mu\}$ (Polyak, 1964; Sutskever et al., 2013), clipping constant $c$ , $\ell_2$ regularization constant $\lambda$. Unspecified parameters are set to zero.

## B.2 NMT Hyperparameters (Ch. 4)

The out-of-domain model in Chapter 4 is trained with mini-batches of size 100 and L2 regularization with weight $1 \times 10^{-7}$, optimized with Adam (Kingma and Ba, 2015) with initial $\alpha = 0.0002$, then decaying $\alpha$ by 0.9 each epoch. The remaining models are trained with constant learning rates and mini-batch size 30, regularization and dropout stays the same. The settings for the other hyperparameters are listed in Table 38. The estimator loss weight is only relevant for DC, where the pre-trained estimator gets further fine-tuned during DC training.

## B.3 Reward Estimator Hyperparameters (Ch. 4)

We find that for reward estimation a shallow CNN architecture with wide filters performs superior to a deeper CNN architecture (Le et al., 2017) and also to a recurrent architecture. Hence, we use one convolutional layer with ReLU activation of $n_f$ filters each for filter sizes from 2 to 15, capturing both local and more global features. For reward estimation on star ratings, $n_f = 100$ and on simulated sBLEU $n_f = 20$ worked best. Dropout with $p = 0.5$ is applied before the output layer for the simulation setting. We set $T_{max} = 60$. The loss of each

| Model | Adam's $\alpha$ | Length-Normalization | MRT $\alpha$ | Sample Size $k$ | MIX $\lambda$ | Estimator Loss Weight |
|---|---|---|---|---|---|---|
| Simulated Feedback | | | | | | |
| MLE | 0.002 | - | - | - | - | - |
| MIX | 0.002 | - | 0.005 | 5 | 0.05 | - |
| ER | $2 \times 10^{-6}$ | - | - | - | - | - |
| DPM | $2 \times 10^{-6}$ | x | - | - | - | - |
| DPM-random | $2 \times 10^{-6}$ | x | - | - | - | - |
| DC | 0.002 | - | - | 5 | - | 1000 |
| Explicit Star Rating Feedback | | | | | | |
| DPM | $2 \times 10^{-6}$ | x | - | - | - | - |
| DPM-random | $2 \times 10^{-6}$ | x | - | - | - | - |
| DC | $2 \times 10^{-6}$ | x | - | 5 | - | 1000 |
| MLE (all) | 0.002 | - | - | - | - | - |
| MIX (all) | 0.002 | - | 0.005 | 5 | 0.05 | - |
| MIX (small) | 0.002 | - | 0.005 | 5 | 0.05 | - |
| MIX (stars=5) | 0.002 | - | 0.005 | 5 | 0.05 | - |
| Implicit Task-Based Feedback | | | | | | |
| MLE (all) | 0.002 | - | - | - | - | - |
| MIX (all) | 0.002 | - | 0.005 | 5 | 0.05 | - |
| MIX (small) | 0.002 | - | 0.005 | 5 | 0.05 | - |
| MIX (recall=1) | 0.002 | - | 0.005 | 5 | 0.05 | - |
| W-MIX | 0.002 | - | 0.005 | 5 | 0.05 | - |

Table 38: Hyperparameter settings for training of the models in Section 4.3.

item in the batch is weighted by inverse frequency of its feedback in the current batch (counted in 10 buckets) to counterbalance skewed feedback distributions. The model is optimized with Adam (Kingma and Ba, 2015) (constant $\alpha = 0.001$ for star ratings, $\alpha = 0.002$ for the simulation) on mini-batches of size 30. Note that the differences in hyper-parameters between both settings are the result of tuning and do not cause the difference in quality of the resulting estimators. We do not evaluate on a separate test set, since their final quality can be measured in how much well they serve as policy evaluators in counterfactual learning.

## B.4 NMT Hyperparameters (Ch. 5)

For the MLE training of the out-of-domain model, we optimize the parameters with Adam ($\alpha = 10^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$) (Kingma and Ba, 2015). For further in-domain tuning (supervised, DPM and RL), $\alpha$ is reduced to $10^{-5}$. To prevent the models from overfitting, dropout with probability 0.2 (Srivastava et al., 2014b) and l2-regularization with weight $10^{-8}$ are applied during training. The gradient is clipped to its norm when its norm exceeds 1.0 (Pascanu et al., 2013b). Early stopping points are determined on the respective development sets. For MLE and DPM models, mini-batches of size 60 are used. For the RL models, we reduce the batch size to 20 to fit $k = 5$ samples for each source into memory. The temperature is furthermore set to $\tau = 0.5$. We found that learning rate and temperature were the most critical hyperparameters and tuned both on the development set.

| $\delta^+$ | $\delta^-$ | TER(1) | TER(2) |
|---|---|---|---|
| 10 | -0.1 | 58.01 | 58.33 |
| 1 | -1 | 58.19 | 58.53 |
| 0.9 | -0.1 | 58.41 | 58.06 |
| 0.8 | -0.2 | 58.43 | 58.02 |
| 0.7 | -0.3 | 58.1 | 57.81 |
| 0.6 | -0.2 | 58.14 | 58.41 |
| 0.5 | -0.5 | 58.16 | 57.9 |
| 0.1 | -0.2 | 58.65 | 58.89 |

Table 39: Impact of marked token weights on translation quality. (1) and (2) indicate the splits of the talks that were not used for training. Note that the TER computation here is not by `MultEval` (reported in the main paper, lowercased) but by `PyTer` (cased).

## B.5 Tuning Token-level Weights

Table 39 documents the results for various settings of $\delta^-$ and $\delta^+$ as evaluated with TER on the two splits it was not trained on.

## B.6 Hyperparameters for Automatic Marking and Corrections

**Automatic corrector.** The OpenNMT-APE implementation by Correia and Martins (2019) was used. It uses multilingual BERT embeddings provided by (Wolf et al., 2019) to initialize a dual-source encoder and a decoder, that are then fine-tuned on the concatenated sources and MT hypotheses as encoder input, and the corrected hypotheses as decoder input. The configuration used in our experiments shares the self-attention between encoder and decoder. The model was trained for 5000 iterations with batches of 512 tokens. For inference, beam search with a beam width of 8 and average length penalty is used.

**Automatic marker.** Our automatic marking model is built on top of BERT (Devlin et al., 2019; Wolf et al., 2019) to take advantage of its contextualized word embeddings. The English source sentence and the German target sentence are concatenated together with a '[SEP]' token between the two sentences and with a '[CLS]' token at the beginning of the sequence. BERT pools sentence level information to the representation of the '[CLS]' token which is then used for downstream classification tasks. In order to classify individual tokens, we use both the individual token embedding and the '[CLS]' representation, combining token specific information and sentence level information. Using both features allows the model to condition the marking probability on not only individual tokens but on sentence semantics as well. The marking head for BERT is a two layer feed-forward neural network with relu activations and sizes of 768. Due to the large class imbalance, words with a ground truth marking of 0, incorrect tokens, receive a weight of 20x in the loss. No fine tuning of the underlying BERT model was performed.

# Appendix C

# Examples

## C.1 eBay Title Translation

| | |
|---|---|
| **Title (en)** | hall linvatec pro2070 powerpro ao drill synthes dhs & dcs attachment / warranty |
| **Ref-0** | hall linvatec pro2070 powerpro ao taladro synthes dhs & dcs accesorio / garantía |
| **Ref-1** | hall linvatec pro2070 powerpro synthes , perforación , accesorio de dhs y dcs , todo original , garantía |
| **OD BL** | hall linvatec pro2070 powerpro ao perforadora synthes dhs & dcs adjuntos / garantía |
| **MIX (all ratings)** | hall linvatec pro2070 powerpro ao perforadora synthes dhs & dcs adjuntos / garantía |
| **MIX (small)** | hall linvatec pro2070 powerpro ao perforadora synthes dhs & dcs adjuntos / garantía |
| **MIX (all queries)** | hall linvatec pro2070 powerpro ao taladro synthes dhs & dcs adjuntos / garantía |
| **T-MIX** | hall linvatec pro2070 powerpro ao taladro synthes dhs & dcs accesorio / garantía |

Table 40: Example for product title translation (en-es) from the test set where T-MIX improved the lexical choice over BL and MIX on in-domain title set and MIX on full query-title set ("perforadora" vs. "taladro" as translation for "drill", "adjuntos" vs. "accesorio" as translation for "attachment").

Table 40 gives an example where T-MIX training improved lexical translation choices. Table 41 lists two examples of T-MIX translations. They are compared to the baseline and logged translations for given queries and product titles, illustrating the specific difficulties of the domain.

## C.2 Human Ratings

| | |
|---|---|
| **Title (en)** | Unicorn Thread 12pcs Makeup Brushes Set Gorgeous Colorful Foundation Brush |
| **Query (es)** | unicorn brushes // makeup brushes // brochas de unicornio // brochas unicornio |
| **Query (en)** | unicorn brushes // makeup brushes |
| **OD BL** | galletas de maquillaje de 12pcs |
| **Logged** | Unicorn Rosca 12 un. Conjunto de Pinceles para Maquillaje Hermosa Colorida Base Cepillo |
| **T-MIX** | unicornio rosca 12pcs brochas maquillaje conjunto precioso colorido fundación cepillo |
| **Title (en)** | 12 × Men Women Plastic Shoe Boxes 33*20*12cm Storage Organisers Clear Large Boxes |
| **Query (es)** | cajas plasticas para zapatos |
| **Query (en)** | plastic shoe boxes |
| **OD BL** | 12 × hombres mujeres zapatos de plástico cajas de almacenamiento 33*20*12cm organizadores de gran tamaño |
| **Logged** | 12 × Zapato De Hombre Mujer De Plástico Cajas Organizadores de almacenamiento 33*20*12cm cajas Grande Claro |
| **T-MIX** | 12 × para hombres zapatos de plástico cajas de plástico 33*20*12cm almacenamiento organizador transparente grandes cajas |

Table 41: Examples for product title translations (en-es) of the logged query test set. In the first example, the T-MIX model improves the translation of "brushes", but also chooses a worse translation for "foundation" ("fundación" vs "base"). In the second example, one of the tricky parts is to translate the sequence of nouns "Men Women Plastic Shoe Boxes" and to disambiguate the relations between them. The BL model translates "shoes of plastic", the Log has "woman of plastic" and the T-MIX model makes it "shoes of plastic" and "boxes of plastic". The T-MIX model learns to use "para" from the query, but omits the translation of "women".

| | |
|---|---|
| **Src** | Diese könnten Kurierdienste sein, oder Techniker zum Beispiel, nur um sicherzustellen, dass der gemeldete AED sich immer noch an seiner Stelle befindet. |
| **Hyp** | These could be courier services, or technicians like, for example, just to make sure that the abalone aed is still in its place. |
| **Rating** | $\sigma = 0.46$, $\varnothing = -0.30$ |
| **Src** | Es muss für mich im Hier und Jetzt stimmig sein, sonst kann ich mein Publikum nicht davon überzeugen, dass das mein Anliegen ist. |
| **Hyp** | It must be for me here and now, otherwise i cannot convince my audience that my concern is. |
| **Rating** | $\sigma = 0.46$, $\varnothing = -0.70$ |
| **Src** | Finden Sie heraus, wie Sie überleben würden. Die meisten unserer Spieler haben die im Spiel gelernten Gewohnheiten beibehalten. |
| **Hyp** | Find out how you would survive. |
| **Rating** | $\sigma = 1.31$, $\varnothing = -0.79$ |
| **Src** | Sie können das googlen, aber es ist keine Infektion des Rachens sondern der oberen Atemwege und verursacht den Verschluss der Atemwege. |
| **Hyp** | You can googlen, but it's not an infection of the rag, but the upper respiratory pathway, and it causes respiratory traction. |
| **Rating** | $\sigma = 1.31$, $\varnothing = -0.52$ |

Table 42: Items with lowest (top) and highest (bottom) deviation in 5-point ratings. Mean normalized rating and standard deviation are reported. Problematic parts of source and target are underlined, namely hallucinated or inadequate target words (#1, #3), over-literal translations (#2), and omissions (#3).

| | |
|---|---|
| **Src** | Zu diesem Zeitpunkt haben wir mehrzellige Gemeinschaften, Gemeinschaften von vielen verschiedlichen Zellentypen, welche zusammen als einzelner Organismus fungieren. |
| **Hyp1** | At this time we have multi-tent communities, communities of many different cell types, which act together as individual organism. |
| **Hyp2** | At this point, we have multicellular communities, communities of many different cell types, which act together as individual organism. |
| **Rating** | $\sigma = 0.0$, $\varnothing = 1.0$ |
| **Src** | Wir durchgehen dieselben Stufen, welche Mehrzellerorganismen durchgemacht haben – Die Abstraktion unserer Methoden, wie wir Daten festhalten, präsentieren, verarbeiten. |
| **Hyp1** | We pass the same steps that have passed through multi-cell organisms to process the abstraction of our methods, how we record data. |
| **Hyp2** | We go through the same steps that multicellular organisms have gone through – the abstraction of our methods of holding data, representing, processing. |
| **Rating** | $\sigma = 0.0$, $\varnothing = 1.0$ |
| **Src** | So in diesen Plänen, wir hatten ungefähr 657 Plänen die den Menschen irgendetwas zwischen zwei bis 59 verschiedenen Fonds anboten. |
| **Hyp1** | So in these plans, we had about 657 plans that offered the people something between two to 59 different funds. |
| **Hyp2** | So in these plans, we had about 657 plans that offered people anything between two to 59 different funds. |
| **Rating** | $\sigma = 0.99$, $\varnothing = 0.14$ |
| **Src** | Wir fingen dann an, über Musik zu sprechen, angefangen von Bach über Beethoven, Brahms, Bruckner und all die anderen Bs, von Bartók bis hin zu Esa-Pekka Salonen. |
| **Hyp1** | We then began to talk about music, starting from bach on Beethoven, Brahms, Bruckner and all the other bs, from Bartók to esa-pekka salons. |
| **Hyp2** | We started talking about music from bach, Beethoven, Brahms, Bruckner and all the other bs, from Bartok to esa-pekka salons. |
| **Rating** | $\sigma = 0.99$, $\varnothing = -0.14$ |

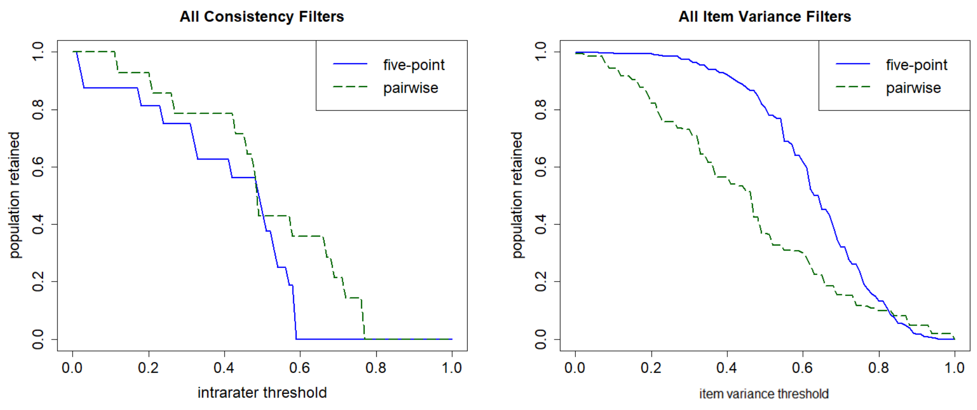
Table 43: Items with lowest (top) and highest (bottom) deviation in pairwise ratings. Preferences of target1: -1, target2: 1, tie: 0. Problematic parts of source and targets are underlined, namely hallucinated or inadequate target words (#1, #2, #3), incorrect target logic (#2), ungrammatical source (#3), capitalization (#4), over-literal translations (#4).

# Appendix D

# Reliability

## D.1 Rater and Item Variance Filtering

Figure 27 shows how the the population diminishes when the intra-rater consistency or item-variance threshold is increased for filtering.



(a) intra-rater consistency filtering (b) item-variance filtering

Figure 27: Filtering: Filtering by intra-rater and item-variance threshold. Plots by Joshua Uyheng.

# Appendix E

# Rating Tasks

## E.1 Error Marking and Correction Annotator Instructions

The annotators received the following instructions:

- You will be shown a source sentence, its translation and an instruction.
- Read the source sentence and the translation.
- Follow the instruction by either marking the incorrect words of the translation by clicking on them or highlighting them, correcting the translation by deleting, inserting and replacing words or parts of words, or choosing between modes (i) and (ii), and then click "submit".
  - In (ii), if you make a mistake and want to start over, you can click on the button "reset".
  - In (i), to highlight, click on the word you would like to start highlighting from, keep the mouse button pushed down, drag the pointer to the word you would like to stop highlighting on, and release the mouse button while over that word.
- If you want to take a short break (get a coffee, etc.), click on "pause" to pause the session. We're measuring time it takes to work on each sentence, so please do not overuse this button (e.g. do not press pause while you're making your decisions), but also do not feel rushed if you feel uncertain about a sentence.
- Instead, if you want to take a longer break, just log out. The website will return you return you to the latest unannotated sentence when you log back in. If you log out in the middle of an annotation, your markings or post-edits will not be saved.
- After completing all sentences (ca. 300), you'll be asked to fill a survey about your experience.

- Important:
  - Please contact us [our email here] immediately if you have any questions or problems with the interface.
  - Please do not use any external dictionaries or translation tools.
  - You might notice that some sentences re-appear, which is desired. Please try to be consistent with repeated sentences.
  - There is no way to return and re-edit previous sentences, so please make sure you're confident with the edits/markings you provided before you click "submit".