UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Improving Neural Sequence-to-Sequence Learning via Data Enhancement

Tsz Kin Lam

This dissertation is submitted for the degree of

Doktor der Philosophie (Dr. phil.)

Institute of Computational Linguistics
Faculty of Modern Languages
Heidelberg University
Germany

Hierbei handelt es sich um eine Heidelberger Dissertation.

First examiner: Prof. Dr. Stefan Riezler
Heidelberg University, Germany
Second examiner: Prof. Dr. Jan Niehues
Karlsruhe Institute of Technology

Date of thesis submission: 12th April 2023
Date of oral examination: 12th June 2023

Declaration

I hereby declare that the thesis has been composed by myself and that the work has not be submitted for any other degree or professional qualification. I confirm that the work submitted is my own, except where work which has formed part of jointly-authored publications has been included. My contribution and those of the other authors to this work have been explicitly indicated in section 1.1. I confirm that appropriate credit has been given within this thesis where reference has been made to the work of others.

Because of the prevalence of ChatGPT in writing, I used it in the earlier phase of my thesis writing, especially in drafting a few chapter summary sections and the conclusion section. Through the interactive dialogues, I knew what English grammar and writing structure, such as combining clauses and phrases, I should revisit and pay attention to. I mainly used two simple prompts: "Improve the below paragraph: ..." and "Explain/elaborate why your improvement is better" for interactions. In short, I used ChatGPT to guide and refine my search on grammatical mistakes when I had no clear ideas or keywords in my mind. However, I benefited more from the blog posts by these companies: 1) Scribbr, 2) Grammarly, and 3) ProWritingAid once I knew the related *keywords*.

Tsz Kin Lam
5th April 2023

Acknowledgements

I would like to express my deepest appreciation to Prof. Dr. Stefan Riezler for his trust and his 3-year funding support. Without him, I would not become an independent researcher and work on a field I am passionate about. Furthermore, I sincerely thank for his encouragement to actively do industrial internships for networking and inspiration that are essential for the thesis completion and personal enrichment. I am deeply indebted to my colleagues Hiko (Shigehiko Schamoni) and Michael Hagmann. Both Hiko and Michael are like my advisors who actively provided feedback in my research ideas. More importantly, they regularly listened to my frustration, cheered me up and taught me tools to navigate through the challenging "paths" in front. Words cannot express my gratitude to Hiko (Yes! Again!) who regularly helped to tackle bureaucratic issues in Germany so that I could focus on my research.

During my PhD, I was fortunate to be an intern at eBay MT team in Aachen, Bing Translator team at Microsoft Redmond, and Amazon MT team in Berlin. I would like to extend my sincere thanks to my mentors, managers and colleagues there, in particular (at eBay) Nicola Ueffing, José G. C. de Souza and Shahram Khadivi, (at Microsoft) Marcin Junczys-Dowmunt and Christian Federmann, (at Amazon) Eva Hasler, Felix Hieber, Ke Tran, Sony Trenous, Tobias Domhan and Prof. Bill Byrne. The presented challenges and their long-term support are indispensable to the completion of this thesis.

I would like to acknowledge my former colleagues in the StatNLP group Heidelberg: Carolin Lawrence, Julia Kreutzer, Julian Hitschler, Laura Jehl and Patrick Simianer. Without their guidance and support, I would not pursue my PhD degree. Lastly, I had the pleasure of working with my current colleagues in the StatNLP group: Mayumi Ohta, Michael Staniek, Nathan Berger and Raphael Schumann. I felt really fortunate to have them laughing and "squeezing the eyebrows" together.

I was financially supported by the German research foundation (DFG) under grant RI-2221/4-1.

Abstract

In recent years, deep learning has revolutionized many areas of life as the driving technology of artificial intelligence. One of the reasons for their success is their use of huge amounts of data and computing resources. However, for many applications, such data are scarce, and straightforward solutions to overcome data scarcity via expert annotation or crowd-sourcing are costly or result in low-quality data. The goal of my thesis is to investigate data enhancement algorithms as automatic and cost-effective alternatives to manual data annotation, with the additional benefit of improved robustness and generalization of models trained on the enhanced data. In particular, we investigate algorithms for data augmentation, data selection, and data correction. Our focus is on neural sequence-to-sequence learning which is a fundamental deep learning technique for a wide range of commercial products such as machine translation and speech recognition, which are essential in breaking language barriers between people from different origins.

In data augmentation, we devise algorithms for reassembling new and effective training data within the given parallel data via segmentation and recombination. This within-corpus augmentation algorithms are simple and effective through possessing three properties: 1) *on-the-fly*, 2) *memory-efficient* and 3) *source-target alignment*. We demonstrate their effectiveness on speech recognition and speech-to-text translation.

In data selection, we aim to remove noisy training data with respect to the targeted data instances. We devise algorithm for selecting pseudo labels based on translation performance in a cascade speech-to-text translation system. In addition, we examine the use of Influence Functions, an attribution technique, on neural machine translation. Influence functions are shown to be useful in classification tasks such as image recognition and toxic speech detection. We analyze its properties, and illustrate the challenges when applying it to neural machine translation.

In data correction, we aim at efficient personalization of a neural machine translation system via human-in-the-loop training. We integrate lightweight feedback such as "keep", "delete" and "substitute" into model training under an active learning based interactive process. In our simulation, we show that such lightweight feedback can produce a competitive machine translation model to that trained with standard cross-entropy loss on the gold-reference translations.

Keywords: speech-to-text translation, speech recognition, neural machine translation, data augmentation, data selection, data correction, data enhancement, neural sequence-to-sequence learning.

Contents

List of Tables

List of Figures

Nomenclature

Acronyms / Abbreviations

ADA Aligned Data Augmentation

Adam Adaptive moment estimation

AED Attention-based encoder-decoder neural network

AL Active learning

ASR Automatic speech recognition

AST/ASTT Automatic speech-to-text translation

BiLSTM Bidirectional long short term memory

BIP-NMT Bandit inteactive-predictive neural machine translation

BLEU bilingual evaluation understudy

CatRandom Concatenation by random

CatSelf Concatenation by self

CatSpeaker Concatenation by speaker

chrF(2) Character n-gram F-score (n-gram = 6 and $\beta = 2$)

CoVoST Speech-to-text translation dataset based on the Common Voice dataset

CT Continued training

CTC Connectionist temporal classification

DNN Deep neural network

EP Europarl corpus for machine translation

Europarl-ST Speech-to-text translation dataset based on the Europarl corpus

FFN Feed forward neural network (or position-wise FFN in Transformer)

FT Fine-tuning

IF Influence functions

IL Imitation learning

IPNMT Interactive-predictive neural machine translation

KD Knowledge distillation

KL divergence Kullback-Leibler divergence

LibriVoxDeEn A German-to-English speech-to-text translation corpus based on the LibriVox platform

LM Language model

LSTM Long short term memory

MT Machine translation

MuST-C Multilingual speech-to-text translation corpus based on English TED talks

NC News commentary dataset

NED-A2C Neural encoder-decoder advantage-actor-critic

NeurS2S Neural sequence-to-sequence

NLP Natural language processing

NMT Neural machine translation

PT Pre-training

RL Reinforcement learning

RNN Recurrent neural network

SeqKD Sequential knowledge distillation

SF Shallow fusion

SGD Stochastic gradient descent optimization algorithm

STR Sample, translate and recombine

TTS Text-to-speech synthesis

VGG Visual geometry group

WER Word Error Rate

Notations

$\boldsymbol{\pi}$ A path of a label sequence in CTC loss

$\mathbf{a}_{i,r:s}$ An sub-sequence composing the rth to sth characters of a sequence $\mathbf{a}_i$.

$\mathbf{x}$ A random variable representing a source sequence

$\mathbf{x}_i$ An instance of a source sequence

$\mathbf{y}$ A random variable representing a target sequence

$\mathbf{y}_i$ An instance of a target sequence

$\mathcal{V}$ Vocabulary used in ASR, NMT or ASTT

$\nabla_X(\mathrm{Y})$ Probing gradient (i.e., a gradient vector) with respect to component X and with source-Y pair as the input. Y is the target input, e.g., hypothesis (HYP), reference (REF) or corrected hypothesis (CorrHYP)

∇_X Gradient vector with respect to a network component X where X, e.g., is source-embedding (srcEmb) or the target-embedding (trgEmb). On top of that, it refers to the gradient vector used in computing the influence functions

π_t A token on timestep t of the path $\boldsymbol{\pi}$ in CTC loss

θ Weights/parameters of a neural network

q_θ A predicted/approximated distribution that is parameterize by θ

Other symbols

<b> Blank symbol in CTC vocabulary

<bos> Begin-of-sentence symbol

<eos> End-of-sentence symbol

<pad> Padding symbol

Chapter 1

Introduction

In recent years, the research and deployment of deep neural network (DNN) have advanced and refined many industries and research areas. Deep neural network reaches state-of-the-art results over conventional methods, and even reach human parity (Silver et al., 2017; Hassan et al., 2018), by leveraging huge amount of data and computing resources, aka Big data. Neural sequence-to-sequence learning (NeurS2S), which is one of those advanced areas, learns the mapping between the source sequence and its target sequence using DNN (Sutskever et al., 2014). NeurS2S learning is a very flexible and powerful algorithm for sequence mapping because it does not impose any restriction on their formats. They can have the same modality, such as in text-to-text translation, or different modalities, such as in image-to-text translation. Their length can be very different, e.g., mapping a sequence of 1000 acoustic frames to a sequence of 8 words. Such properties support NeurS2S to model well a wide range of applications such as machine translation (Cho et al., 2014a; Bahdanau et al., 2015), automatic speech recognition (Chan et al., 2016b), and text-to-speech synthesis (van den Oord et al., 2016).

Despite its wide range of applications, similar to other neural network based method, e.g., image recognition (Krizhevsky et al., 2012), a high-quality NeurS2S model heavily depends on the quantity, quality and diversity of the training data available. An intuitive solution is to increase the amount of training data via expert annotation with quality control, e.g., translation from professional translators with proper translation guidelines and rater agreement. Expert annotation, however, is very costly to obtain in terms of both time and money (Post et al., 2013). In addition, it along may not be sufficient to train a high-quality NeurS2S model which may take million, hundreds of million or even billions number of training instances.

In this thesis, we examine data enhancement techniques to improve NeurS2S

models without costly annotation from human expert. In particular, we focus on these three aspects: 1) data augmentation, 2) data selection and 3) data correction which improve model performance via increasing the effective data size and reducing noisiness in data through selection and refinement. Among the vast amount of applications, we focus on neural machine translation (NMT), automatic speech recognition (ASR), and automatic speech-to-text translation (ASTT) because these NLP and speech applications break the language barrier and thus enhance communication between people from different origins (Wu et al., 2016; Chan et al., 2016b; Chiu et al., 2018; Hassan et al., 2018).

1.1 Contributions

This dissertation makes a number of contributions to the areas of data augmentation, data selection and data correction in NeurS2S. In particular,

- We develop within-corpus data augmentation algorithms for end-to-end ASR and ASTT. This algorithms are simple and effective through possessing three properties: 1) *on-the-fly*, 2) *memory-efficient* and 3) *source-target alignment*. This line of works has been published in the i) Proceedings of 22nd Annual Conference of the International Speech Communication Association (INTERSPEECH 2021), see Lam et al. (2021b), ii) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022), see Lam et al. (2022b), and iii) Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing 2023 (ICASSP 2023), see Lam et al. (2023).

- We develop selection algorithm for pseudo-labels based on down-stream translation performance in a cascade speech-to-text translation system. This work has been published in the Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing 2021 (ICASSP 2021), see Lam et al. (2021b).

- We examine Influence Functions (IF), an attribution technique, for data filtering on NMT. On top of examination, we modify it with contrastive signals and present potential challenges. This work has been published in the Proceedings of the Seventh Conference on Machine Translation (WMT 2022), see Lam et al. (2022a).

- We develop training algorithms which integrate lightweight feedback in an active-learning based interactive setting for more efficient personalization purpose. In a simulation setting, we demonstrate that such lightweight feedback can improve the performance of NMT. This work has been published in the i) Proceedings of the 21st Annual Conference of the European Association for Machine Translation (EAMT 2018), see Lam et al. (2018), and ii) Proceedings of Machine Translation Summit XVII (MT Summit 2019), see Lam et al. (2019).

In these accepted work, the contribution of Tsz Kin Lam is on the construction of project ideas, the implementation of experiments and the drafting of manuscripts. Coauthors contribute via insightful discussions and support in both writing and rebuttal. Some exceptions are

- on the EAMT 2018 paper which Prof. Stefan Riezler initiated to combine bandit sequence-to-sequence learning with interactive machine translation system.
- on the ICASSP 2021 paper which Prof. Stefan Riezler initiated to improve a cascade speech-to-text translation system. Additionally, Shigehiko Schamoni participated parts of the implementation.
- on the WMT 2022 paper which Influence Functions is one of the available projects for the internship.

1.2 Outline of the dissertation

In this dissertation, we discuss the data scarcity issue in NeurS2S and suggest data enhancement as a more cost-effective alternative.

In chapter 2, we review the fundamentals of NeurS2S learning, including its major architectures, loss functions and evaluation metrics.

In chapter 3, we provide background on the data scarcity problem and some existing solutions.

In chapter 4, we present within-corpus data augmentation as our approach to increase the effective training data size. We devise augmentation algorithms for speech-to-text applications such as ASR and ASTT.

In chapter 5, we present instance-specific data selection in handling nuanced model errors and for leveraging end-task feedback. We devise algorithms in improving cascaded ASTT and NMT.

In chapter 6, we present our interactive learning protocols for more efficient personalization of NMT. For cost-saving purpose, we limit our training to simulated human feedback using gold-reference translations.

In conclusion section, we summarize our findings and introduce futures works.

Chapter 2

Neural Sequence-to-Sequence Learning

This chapter provides the fundamentals of neural sequence-to-sequence learning. We review its core components, including attention-based encoder-decoder neural network, loss functions and evaluation metrics. We place a specific emphasis on speech/text-to-text application.

2.1 Architecture: Attention-based Encoder-Decoder

Attention-based encoder-decoder (AED) neural network is a core architecture for neural sequence-to-sequence (NeurS2S) learning. It models the conditional probability $p_\theta(\mathbf{y}|\mathbf{x}_i)$ of a target sequence $\mathbf{y}_i = (y_{i,1}, \dots y_{i,T_y})$ given its source sequence $\mathbf{x}_i = (x_{i,1}, \dots, x_{i,T_x})$

$$p_\theta(\mathbf{y}_i|\mathbf{x}_i) = \prod_{t=1}^{T_y} p_\theta(y_{i,t}|\mathbf{y}_{i,<t}, \mathbf{x}_i) \tag{2.1}$$

where $\mathbf{y}_{i,<t}$ are all tokens in the target sentence prior to the token $y_{i,t}$. The model parameters θ contain all trainable parameters, or called weights, of the network, typically including an encoder, an decoder, an attention layer and an output layer.

The encoder takes in $\mathbf{x}_i$, e.g., a sequence of characters in German, and transform it into (source) contextual representations $h^{enc} = encoder(\mathbf{x}_i)$ which is then processed by the decoder. On the target side, the decoder takes in a target token and combine its information with h^{enc} in an auto-regressive manner. At a decoder step t, the decoder turns that target unit y_t, e.g. an English token, into a representation. An attention mechanism then connects the source context and the target representation up to step t to produce representation h_t^{dec} for the output layer, e.g., a linear layer with softmax activation.

In supervised training, we have access to the entire target sequence which is not available in decoding test inputs. Therefore, architecture specific measures are imposed during training to prevent the decoder from using or attending the future target tokens. In section 2.1.1, we discuss Recurrent Neural Network (RNN) that is a fundamental building block for sequence modeling. In section 2.1.2 and 2.1.3, we discuss Transformer and Conformer architectures which are modifications of RNN and are based entirely on attention mechanism.

Textual Inputs A discrete input sequence, such as a sequence of English characters, has to be transformed into a numerical representation before processed by a neural network. One simplest format is one-hot vector which contains a digit of one in the position representing the character and zero elsewhere. Despite its simplicity, however, one-hot vector has two major problems. Firstly, its dimension increases with the size of vocabulary, resulting in the curse of dimensionality (Bellman, 1957). Secondly, it does not present any semantic meaning between the words. Bengio et al. (2000) proposed *word embedding* to reduce the dimensionality of word representations while capturing the semantics between words.

Word embedding $\mathbf{W}$, a matrix of size $N \times M$, maps N distinct tokens in one-hot vector format to their continuous representation of size $1 \times M$ where $M \ll N$. The embedding $\mathbf{W}$ contains trainable parameters which are adjusted on a huge amount of corpus. Two commonly used training algorithms for word embedding are *Skim-Gram* and *Continuous Bag-of-Words* (Mikolov et al., 2013). In Skip-Gram, the word embedding is trained to predict the surrounding words given a central word. In opposite, Continuous Bag-of-Words predicts the central word given its surroundings. Both algorithms train the parameters by minimizing the negative log-likelihood of the corresponding predictive distributions. These methods are called *static embedding* as the embedding of a given token is the same, irrespective of its context. Contextualized word embedding (Devlin et al., 2019), however, create different embedding in response to the context of the given token.

Both static and contextualized embedding can be learnt on unlabeled corpus. Therefore, they are unsupervised or self-supervised. Such unsupervised nature greatly relieve the data scarcity issue in most NLP and Speech problems by pre-training certain network parameters using unlabeled corpus, followed by fine-tuning on limited labeled data[1]. In NeurS2S learning, however, the embedding is normally trained from

[1]At the time of writing this thesis, un/self-supervised training is one dominant approach for NLP and speech processing. However, these pre-training techniques are computationally very costly. In addition, there are works showing the complementarity of un/self-supervised learning techniques,

scratch with the entire neural network.

Audio Inputs In speech-to-text, the encoder processes an acoustic waveform or a sequence of acoustic representations instead of textual inputs. This eliminates the needs of an encoder embedding[2]. However, the length of the acoustic sequence is usually much longer than the textual sequence, making alignment between speech and text a challenging task. Furthermore, the attention mechanism in Transformer-like architecture has a quadratic computational complexity with respect to the input-sequence length. This further highlights the need of downs-sampling the acoustic-sequence length to save computation time and improve speech-to-text alignment.

Concatenating neighboured frames is one simple form of down-sampling. This can be done on either the input level or on the intermediate representations such as Network-in-network (Sperber et al., 2019a) or pyramidal encoder (Chan et al., 2016a). Another simple approach is the use of convolution layers with a higher stride value or with max-pooling layers (Pino et al., 2019). In our experiments, we use convolution layers with a higher stride for down-sampling so that the down-sampled features can be trained together with other network components.

2.1.1 Recurrent Neural Network

Vanilla Recurrent Encoder-Decoder Network In the vanilla recurrent encoder-decoder network, the encoder receives an input token $x_{i,k}$ and outputs a fixed-length vector $h^{enc}_{i,k} = \text{encoder}(x_{i,k}, h_{i,k-1})$ for each time step k where $h_{i,0}$ can be a vector of zeros. This fixed-length vector, or called the hidden state, represents some high-level abstractions of the source input. The iterative process continues across source tokens, and the last hidden state is passed to initialize the decoder's hidden state, i.e., $h^{dec}_{i,0} = h^{enc}_{i,T_x}$. In the decoder, the hidden state from the previous time step is concatenated with the representation of the current target-token to generate a new hidden state $h^{dec}_{i,t} = \text{decoder}(y_{i,t}, h^{dec}_{i,t-1})$. A linear layer with softmax activation then process $h^{dec}_{i,t}$ to generate a predictive distribution $p(y_{i,t}|\mathbf{y}_{i,<t}, \mathbf{x}_i) = \text{softmax}(\text{linear}(h^{dec}_{i,t}))$ over the possible tokens in the vocabulary $\mathcal{V}$. Such process is iterated across all the target tokens, including the end-of-sentence token <eos>. In case of multi-layer RNN network, $h^{enc/dec}_{i,k/t}$ is passed as the "input-token" representation of the next RNN layer for each timestep.

---

data augmentation and human feedback (Xu et al., 2021a; Ouyang et al., 2022).

[2]Recently, there do have works of using self-supervised learning and quantization to discretize the audio representations, e.g., Lee et al. (2022), but they are beyond the scope of this thesis.

Attention Mechanism (Bahdanau) In the above vanilla recurrent network, h^{enc}_{i,T_x} is the sole source-input's information that the decoder can access during sequence generation. In case of a long source-input, the fixed-size vector h^{enc}_{i,T_x} has to encode more information, resulting in further information loss (Cho et al., 2014b). Bahdanau et al. (2015) introduced *attention*, or called Bahdanau attention, to relieve the bottleneck. In his formulation, a source-context vector $c_{i,t}$ is computed for each target-input $y_{i,t}$ so that the decoder can attend to the relevant parts of the source-input. More specifically, each source-context vector $c_{i,t}$ is the linear combination of *all* encoder hidden states $\{h^{enc}_{i,k}\}_{k=1}^{T_x}$:

$$c_{i,t} = \sum_{k=1}^{T_x} \alpha_{i,tk} h^{enc}_{i,k} \tag{2.2}$$

The weights $\alpha_{i,tk}$ for each $h^{enc}_{i,k}$ shows the relative importance of the source-token[3] $x_{i,k}$ on the prediction of the target-token $y_{i,t}$. Its value is computed by normalizing the *attention score* $e_{i,tk}$ with a softmax function so that $\sum_k \alpha_{i,tk} = 1$:

$$\alpha_{i,tk} = \frac{e_{i,tk}}{\sum_{k'}^{T_x} e_{i,tk'}} \tag{2.3}$$

where $e_{i,tk} = score(h^{dec}_{i,t-1}, h^{enc}_{i,k})$[4] represents the similarity between $h^{dec}_{i,t-1}$ and $h^{enc}_{i,k}$. In Bahdanau attention, the score function is a single-layer multilayer perceptron:

$$score(h^{\text{dec}}_{i,t-1}, h^{\text{enc}}_{i,k}) = \nu_a^\top \tanh(\mathbf{W}_a[h^{\text{dec}}_{i,t-1}; h^{\text{enc}}_{i,k}]) \tag{2.4}$$

where ν_a, $\mathbf{W}_a$ are the weight matrices. The resulting $c_{i,t}$, together with $h^{dec}_{i,t-1}$ and $y_{i,t}$, is processed by the recurrent layer, such as LSTM, to generate $h^{dec}_{i,t}$, which would be processed by the output layer for making prediction.

Attention Mechanism (Luong) Luong attention (Luong et al., 2015) is another commonly used attention in RNN-based framework. It differs from Badhdanau attention in a few aspects. First of all, the score function used for computing $e_{i,tk}$ takes in h^{dec}_t instead of h^{dec}_{t-1}. Secondly, Luong attention explore *dot* and *bilinear* operations

[3] In single-layer Bi-directional RNN encoder, $h^{enc}_{i,k}$ is the concatenation of the forward and backward hidden state at that time step, i.e., $h^{enc}_{i,k} = \text{concat}(\overrightarrow{h_{i,k}}; \overleftarrow{h_{i,k}})$.

[4] Be reminded that the decoder's hidden representation at the **previous** time step is used in Bahdanau attention.

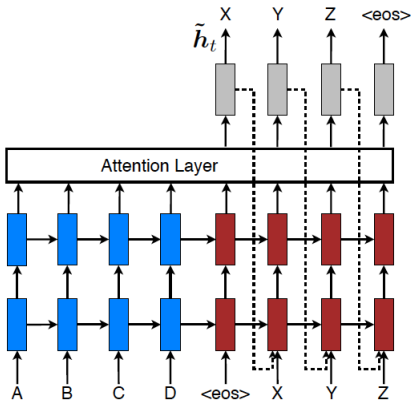

Figure 2.1: A schema showing input feeding mechanism. Source: figure 4 in Luong et al. (2015).

to compute $e_{t,tk}$ in addition to multilayer perceptron:

$$score(h_{i,t}^{dec}, h_{i,k}^{enc}) = \begin{cases} h_{i,t}^{dec\top} h_{i,k}^{enc} & \text{dot} \\ h_{i,t}^{dec\top} \mathbf{W}_a h_{i,k}^{enc} & \text{Bilinear} \\ \nu_a^\top \tanh(\mathbf{W}_a [h_{i,t}^{dec\top}; h_{i,k}^{enc}]) & \text{MLP} \end{cases} \tag{2.5}$$

Thirdly, the resulting $c_{i,t}$ is not used for generating the decoder hidden state $h_{i,t}^{dec}$. Instead, it is combined with $h_{i,t}^{dec}$ through a linear layer and a hyperbolic tangent activation function to generate an *attentional hidden state* $\tilde{h}_{i,t}$ which would be fed to the output layer $\mathbf{W}_o$ for making prediction:

$$\begin{aligned} \tilde{h}_{i,t} &= \tanh(\mathbf{W}_c [c_{i,t}; h_{i,t}^{dec}]) \\ p_\theta(y_{i,t} | \mathbf{y}_{i,<t}, \mathbf{x}_i) &= \text{softmax}(\mathbf{W}_o \tilde{h}_{i,t}) \end{aligned} \tag{2.6}$$

Lastly, $\tilde{h}_{i,t}$ would be concatenated with the next target token $y_{i,t+1}$ as the target input at step $t+1$. This process is called *Input Feeding* which aims to inform the model about past alignment decisions, see Figure (2.1).

Luong attention can be further classified into *global* and *local* attention. The key difference lies in the number of encoder hidden states used in computing $c_{i,t}$. Global attention considers all encoder hidden states, i.e, equation (2.2). In local attention, a context window, such as a Gaussian distribution, is centered around the corresponding encoder hidden states so that only its neighboured encoder hidden states are considered.

2.1.2 Transformer

Recurrent network is a natural choice for modeling sequences because it captures the context of each token through its left-to-right history. In addition, this left-to-right recurrence allows the network to incorporate the positional information. However, such modeling process makes parallel computation over the hidden states impossible, resulting in a substantial increase in computation time for long sequences. Using convolution layers in replace of recurrent layers (Gehring et al., 2017) makes the parallel computation possible, but the number of convolution filters increases according to the input sequence length.

Transformer (Vaswani et al., 2017), which is introduced by researchers at Google, addresses the above deficiencies by replacing convolution layers or recurrent layers with purely attention mechanism(s). Transformer is still belong to the class of AED which has an encoder-decoder structure. In the following, we discuss Transformer for text-to-text translation[5], including information flow and its major components.

Figure 2.2 shows the architectural components of Transformer for text-to-text translation. In encoder, a sequence of source tokens are fed to the word embedding for extracting their embedding representation. The representation is then added element-wisely with its positional encoding, which takes the index of the input sequence as inputs, before feeding into the encoder layers. Each encoder layer contains two major sub-components: 1) self-attention and 2) position-wise feed-forward neural network (FFN); both are built for capturing the contextual dependencies and high-level features. Notably, there is a residual connection which adds sub-component's input to its output for avoiding vanishing gradients. The resulting representation is then proceeded to a layer-normalization layer before passing to the next sub-component or layer. In decoder, similar to the encoder, the target sequence is first transformed to dense representation by passing through the word-embedding layer and the positional encoding layer. In each decoder layer, the representation from the previous layer is first processed by the *(masked) self-attention* layer for capturing the contextual dependencies among the processed (target) sequence. A crucial difference in the decoder is an extra *encoder-decoder attention* layer between the masked self-attention layer and the FFN. This extra layer takes in the output from the last encoder layer and the output from the masked self-attention layer as inputs to model the conditional dependence of target sequence on its source sequence. Notably, the residual connection of the encoder-decoder attention excludes the encoder output. Finally, for prediction,

[5]The case for speech-to-text can be referred to the part of **Audio Inputs** above.

the representation of the last decoder layer is transformed by a linear layer with softmax activation.

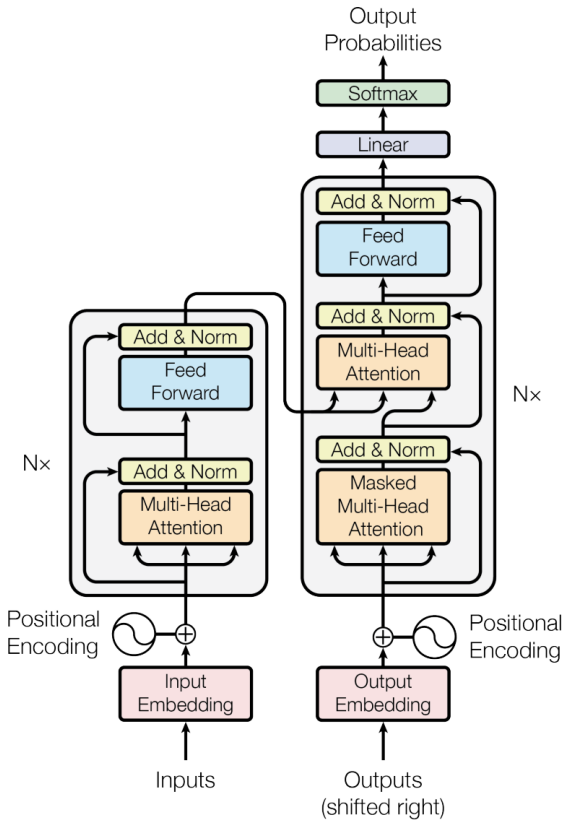

Figure 1: The Transformer - model architecture.

Figure 2.2: Transformer architecture. Source: Figure 1 from Vaswani et al. (2017).

Scaled Dot-Product Attention Transformer captures the context and dependencies between tokens by using scaled dot-product attention:

$$\text{Attention(Q,K,V)} = \text{softmax}(\frac{QK^\top}{\sqrt{d_k}})V \tag{2.7}$$

where Q, K and $V \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$ are query, key and value matrices respectively, and d_{model} is the attention layer's dimension. The scaling factor $d_k \in \mathbb{R}^{1 \times d_{\text{model}}}$ prevents the dot-product from being too extreme so that the gradient of the softmax activation remains not too small, resulting in more stable training.

In encoder, the Q-K-V matrices are identical, and they are the encoder representations from the previous layer. This is known as self-attention which allows each source

token to attend to all other tokens in the sequence.

In decoder, self-attention is also applied to the target input. However, we apply a *mask* to prevent the current and its previous target tokens to attend to the future ones because of the absence of future target tokens during generation. After masked self-attention, there is a decoder-encoder attention which attends the decoder representation to the encoder output, i.e., Q is the decoder representation and both K and V are the encoder output.

Multi-Head Attention The authors find it more beneficial to project the Q, K and V matrices into k-different sub-spaces, or called heads, followed by merging the information. This is called the *Multi-Head Attention*:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots, \text{head}_k)W^O \tag{2.8}$$

$$\text{where} \quad \text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{2.9}$$

where $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_h}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_h}$ and $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ are the projection matrices, and $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$ is a linear layer.

Positional Encoding The positional encoding helps the model to capture the token's order information in the input sequences. It has the same dimension as the input word embedding so that they can be summed before processed by self-attention. The positional encoding in Transformer constitutes sine and cosine function of different frequencies:

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{\text{model}}}) \tag{2.10}$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{\text{model}}}) \tag{2.11}$$

where $i \in [1, \cdots, d_{\text{model}}]$ is the ith-dimension, and pos is the location index, which starts from the left hand side, of the token.

Position-wise Feed-Forward Neural Network In Transformer, the FFN consists of two linear transformations with a ReLU activation in between.

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2.12}$$

where $W_1 \in \mathbb{R}^{d_{\text{model}} \times d_{ff}}$ and $W_2 \in \mathbb{R}^{d_{ff} \times d_{\text{model}}}$ are the weight matrices. Notably, the dimension of the input and the output of $\text{FFN}(x)$ is the same.

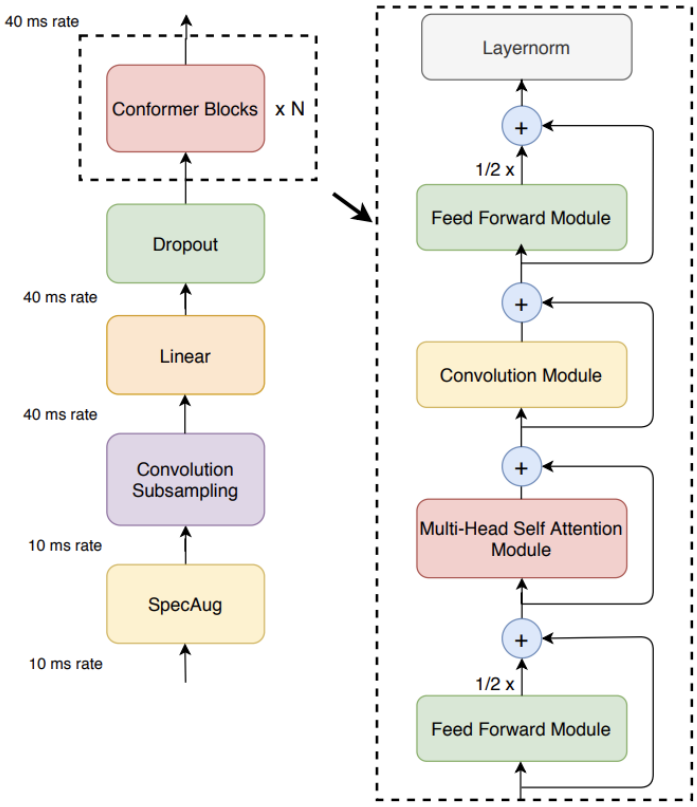

Figure 2.3: Structure of a Conformer encoder. Source: figure 1 in Gulati et al. (2020).

2.1.3 Conformer Encoder for Speech-to-Text

Convolution-augmented Transformer (Gulati et al., 2020), Conformer in short, is another Transformer-based neural network which was developed by Google for speech-to-text. Its advantage over Transformer lies in the N layers of Conformer blocks inside its encoder. Similar to an encoder layer in Transformer, each conformer block contains a self-attention layer, FFN and a layer-normalization layer. In addition, the conformer block contains a convolution module which, together with the self-attention, makes Conformer better at capturing both local and global dependencies of an audio sequence. Empirically, conformer encoder has been shown to perform better than transformer encoder in speech-to-text applications.

Figure 2.3 shows the structure of a Conformer encoder. An input sequence of audio features, such as log Mel filter bank features, is first partially masked by SpecAugment as input regularization. The masked spectrogram is then downsampled in its temporal dimension by convolution sub-sampling layers for easier source-target alignment and also memory efficiency. Inside each conformer block, there are 5

sequential sub-components: 1) feed-forward module, 2) multi-head self-attention module, 3) convolution module, 4) feed-forward module and a 5) layer-normalization layer. Except the layer-normalization layer, the output of the first 4 sub-components is reinforced by its input via residual connection.

Feed-Forward Module Inside the feed-forward module, there are 6 sub-components in the following sequential order: 1) layer-normalization, 2) linear layer, 3) swish activation (Ramachandran et al., 2018), 4) dropout, 5) linear layer and 6) dropout. The first linear layer has an expansion factor of 4 whereas the second linear layer scales the dimension back, resulting in an auto-encoder like architecture. Additionally, there is a (half-step) pre-norm residual unit (Nguyen and Salazar, 2019; Wang et al., 2019) which connects the input to the layer-normalization layer and the output of the last dropout layer. Such pre-norm residual unit has been shown to be better than the post-norm residual unit proposed in the original Transformer.

Notably, there are two such feed-forward modules inside a conformer block. This two modules sandwich the multi-head self-attention module and the conformer module, resulting in a Marcaron-like architecture (Lu et al., 2019). Additionally, each residual connection is only half-step, i.e, to scale down the output of the dropout layer by 0.5 before adding the residual input. We would not dive into its details because it is related to dynamical system and theory of ordinary differential equations which are beyond the scope of this thesis.

Multi-Head Self-Attention Module The multi-head self-attention module is modified based on that in transformer encoder. It has three sub-components that are connected sequentially: 1) layer-normalization, 2) multi-head self-attention and 3) dropout. Similar to other modules, there is a pre-norm residual units connecting the input of the layer-normalization layer to the output of the dropout layer. What further differentiate it with transformer encoder is the use of relative positional encoding (Dai et al., 2019) to better capture long-term dependencies in the input audio sequence.

Convolution Module The convolution module contains multiple advanced techniques for capturing attention of long-short range, such as a gating mechanism, which is formed by pointwise convolution and GLU activation (Dauphin et al., 2017), and a swish activation. There are in total 8 sub-components: 1) layer-normalization, 2) pointwise convolution, 3) GLU activation, 4) 1-D depthwise convolution, 5) batch-normalization, 6) switsh activation, 7) pointwise-convolution and 8) dropout. Addi-

tionally, it also contains a pre-norm residual unit to wire the front layer-normalization and the output of the dropout.

2.2 Training: Loss functions

In supervised learning, we minimize the model loss over its labeled training data. This minimization is usually an iterative process of computing the loss gradient using back-propagation algorithm (Rumelhart et al., 1986), followed by parameters update using gradient descent or its variants, such as SGD (Robbins, 1951) or Adam (Kingma and Ba, 2015).

Typically, the choice of the loss function is related to the target task. In regression, a popular choice is Mean Square Error which minimizes the mean squared distance between the true value and and the model output. Other popular choices are Mean Absolute Error, Huber loss and Quantile loss. In classification or sequence labeling, cross-entropy, hinge loss[6] and their variants are commonly used.

Notably, multiple loss functions can be used together to make the final model even better. Un/Self-supervised pre-training, for an example, initializes the parameters of the targeted model by those trained with other tasks and, most likely, different loss functions. Multi-task learning (Caruana, 1998), on the other hand, leverage multiple loss functions simultaneously, resulting in a shared representation that perform well across the selected tasks.

In the following, we provide an overview of the loss functions used in this thesis.

2.2.1 Cross Entropy Loss and Label-Smoothing

In classification task, given an input x, the model estimates a probability distribution $q_\theta(y|x)$ over a set of labels $\mathcal{V}$ where $y \in \mathcal{V}$. The loss L, or called an error signal, is obtained by computing the discrepancy between its predicted distribution $q_\theta(y|x)$ and

[6]The original formulation of hinge loss is not suitable for gradient descent algorithms because it is not differentiable everywhere.

the true-distribution p via the Kullback–Leibler (KL) divergence:

$$\begin{aligned}\mathrm{KL}(p||q_\theta) &= H(p, q_\theta) - H(p) \\ &= E_p[-\log q_\theta] - E_p[-\log p] \\ &= \sum_{i=1}^{|\mathcal{V}|} -p_i \log q_i + \sum_{i=1}^{|\mathcal{V}|} p_i \log p_i \\ &= \sum_{i=1}^{|\mathcal{V}|} p_i \log \frac{p_i}{q_i} \end{aligned} \tag{2.13}$$

where $H(p, q_\theta)$ is the cross-entropy loss, $H(p)$ is the Shannon entropy, p_i is $p(y_i \in \mathcal{V}|x)$ and q_i is $q_\theta(y_i \in \mathcal{V}|x)$.

The Shannon entropy $H(p)$ can be dropped from the optimization process because it is not dependant on the model parameters θ. Additionally, p is a one-hot vector in classification task. As a result, equation (2.13) is reduced to $-\log q_\theta(y = l_g|x)$, i.e., the negative log-likelihood of the true label l_g on the predicted distribution. Therefore, the below minimization is equivalent:

$$\min_\theta \mathrm{KL}(p||q) \Leftrightarrow \min_\theta H(p, q_\theta) \Leftrightarrow \min_\theta -\log q_\theta(y = l_g|x) \tag{2.14}$$

Reverse KL Divergence Notably, the forward KL divergence $\mathrm{KL}(p||q_\theta)$ is not the same as its backward, i.e., $\mathrm{KL}(q_\theta||p)$ given $p \neq q_\theta$. That is, KL-divergence itself is not a distance function, or called a metric. In forward divergence, the predicted distribution q_θ is likely to be non-zero for x where $p(x) > 0$ because of the penalization. This makes q_θ to spread over those x values. In backward divergence, q_θ can be zeros for x where $p(x) > 0$ since the KL divergence would be zero. Therefore, optimizing backward divergence is known as *zero forcing*.

Label Smoothing One potential issue of optimizing cross-entropy loss is the result of model's over-confidence since all the probability mass is located on the true label. Such over-confidence issue could be alleviated by using a soft target such as label-smoothing (Szegedy et al., 2016; Pereyra et al., 2017). Label-smoothing distributes probability mass ϵ from the ground-truth label to other classes on p, reducing the model's over-confidence on the true label. Specifically, we replace p by $p'(y|x) = \epsilon u(y|x) + (1 - \epsilon)p(y|x)$, resulting in a new loss function:

$$L = -\sum_{i=1} \sum_{y \in C} p'(y|x_i) \log q_\theta(y|x_i) = \sum_{i=1} (1 - \epsilon) H_i(p, q_\theta) + \epsilon H_i(u, q_\theta) \tag{2.15}$$

where $u = \frac{1}{C}$ with C being the number of classes.

2.2.2 Policy Gradient and Control Variates

Another potential issue of cross-entropy loss is that the metric optimized is not the metric we used in evaluation. Policy gradient (Sutton et al., 1999) is an alternative which allow us to directly optimize the expected (future) reward, i.e., the evaluation metric:

$$L_{\text{PG}} = \mathbb{E}_{\hat{\mathbf{y}} \sim p_\theta(\cdot|\mathbf{x}_i)}[r_i(\hat{\mathbf{y}})] \tag{2.16}$$

where $r_i(\hat{\mathbf{y}}_i) = \Delta(\hat{\mathbf{y}}_i, \mathbf{y}_i)$ is a reward, such as BLEU, chrF, or human evaluation, to describe the translation quality of a sampled trajectory $\hat{\mathbf{y}}_i$ (with respect to its input $\mathbf{x}_i$) against its reference $\mathbf{y}_i$.

In spite of the direct optimization, such formulation creates a stochastic function of which the loss derivative depends on samples drawn from the underlying model. One solution to disentangle the model parameters and its stochasticity is the log-gradient trick:

$$\begin{aligned}
\nabla_\theta L_{\text{PG}} &= \nabla_\theta \left(\sum_{\hat{\mathbf{y}}_i \in \mathcal{Y}} p_\theta(\hat{\mathbf{y}}_i|\mathbf{x}_i) r_i(\hat{\mathbf{y}}_i) \right) = \sum_{\hat{\mathbf{y}}_i \in \mathcal{Y}} \nabla_\theta \left(p_\theta(\hat{\mathbf{y}}_i|\mathbf{x}_i) r_i(\hat{\mathbf{y}}_i) \right) \\
&= \sum_{\hat{\mathbf{y}}_i \in \mathcal{Y}} \left(r_i(\hat{\mathbf{y}}_i) \nabla_\theta p_\theta(\hat{\mathbf{y}}_i|\mathbf{x}_i) + \cancel{p_\theta(\hat{\mathbf{y}}_i|\mathbf{x}_i) \nabla_\theta r_i(\hat{\mathbf{y}}_i)} \right) \\
&= \sum_{\hat{\mathbf{y}}_i \in \mathcal{Y}} r_i(\hat{\mathbf{y}}_i) \left(p_\theta(\hat{\mathbf{y}}_i|\mathbf{x}_i) \nabla_\theta \log p_\theta(\hat{\mathbf{y}}_i|\mathbf{x}_i) \right) \\
&= \mathbb{E}_{\hat{\mathbf{y}} \sim p_\theta(\cdot|\mathbf{x}_i)}[r_i(\hat{\mathbf{y}}) \nabla_\theta \log p_\theta(\hat{\mathbf{y}}|\mathbf{x}_i)]
\end{aligned} \tag{2.17}$$

where the sum is over all the possible target sequences $\hat{\mathbf{y}}_i \in \mathcal{Y}$. The log-gradient trick optimizes the expected reward as weighting the log-likelihood of the trajectory $\hat{\mathbf{y}}_i$. In other words, it allows human (or simulated) feedback to directly scale the contribution of model's output $\hat{\mathbf{y}}_i$. Another popular approach is Gumbel Softmax (Jang et al., 2017; Maddison et al., 2017), which is based on the reparameterization trick, but it is beyond the scope of this thesis.

Variance Reduction by Control Variate Despite direct optimization of the evaluation metric, policy gradient requires the expectation (with respect to its predicted distribution) over all possible target sequences given an input sequence $\mathbf{x}_i$. Such expectation is intractable, considering the long sequence length and large vocabulary size in NeurS2S tasks.

Williams (1992) proposed *REINFORCE* algorithm to approximate the intractable expectation by a *single-sample estimator*. In case of mini-batch gradient descent, we include other training instances by averaging their individual losses:

$$\nabla_\theta L_{\text{REINFORCE}} = \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} r_i \nabla_\theta \log p(\hat{\mathbf{y}}_i|\mathbf{x}_i) \tag{2.18}$$

where $\hat{\mathbf{y}}_i$ is a trajectory sampled from the distribution $p_\theta(\cdot|\mathbf{x}_i)$, r_i is $r_i(\hat{\mathbf{y}}_i)$ and $|\mathcal{B}|$ is the mini-batch size. It is worth noting that equation (2.18) resembles cross-entropy loss, except that the reward is not always equal to 1, and the sampled target sequence is used instead of the gold-reference target sequence. However, an accompanied drawback of the *single-sample estimator* is its larger variance[7] that is inversely proportional to the number of samples, which are drawn independently, used in the estimator.

Additive control variate (Evans and Swartz, 2000; Fishman, 2013) is one common approach to reduce the variance of Monte Carlo estimates of integrals, such as the expected gradients in equation (2.17). In additive control variate, a random variable b is subtracted from the original estimator X so that the new estimator $X - b$ is of lower variance:

$$\text{Var}[X - b] = \text{Var}[X] + \text{Var}[b] - 2\text{Cov}[X, b] \tag{2.19}$$

where Var, and Cov are the variance and the covariance respectively. As shown in equation (2.19), the variable X and b should be strongly correlated, i.e., a large Cov[X, b] so that the resulting estimator $X - b$ has lower variance. Meanwhile, the new estimator should be unbiased with respect to the original estimator, i.e, $\mathbb{E}[X] = \mathbb{E}[X - b]$.

In our single-sample estimator, we have X to be $r_i(\hat{\mathbf{y}})\nabla_\theta \log p(\hat{\mathbf{y}}|\mathbf{x}_i)$ and b to be $r_b \nabla_\theta \log p(\hat{\mathbf{y}}|\mathbf{x}_i)$. These two estimators are strongly correlated because of the common gradient term $\nabla_\theta \log p(\hat{\mathbf{y}}|\mathbf{x}_i)$ and a scalar reward r_b having the same sign as $r_i(\hat{\mathbf{y}})$. Furthermore, the expected value of the estimator b is

$$\begin{aligned}
\mathbb{E}_{\hat{\mathbf{y}} \sim p_\theta(\cdot|\mathbf{x}_i)}[r_b \nabla_\theta \log p(\hat{\mathbf{y}}|\mathbf{x}_i)] &= \sum_{\hat{\mathbf{y}}_j \in \mathcal{Y}} \cancel{p} r_b \left(\frac{\nabla_\theta p(\hat{\mathbf{y}}_j|\mathbf{x}_i)}{\cancel{p}} \right) \\
&= r_b \sum \nabla_\theta p(\hat{\mathbf{y}}_j|\mathbf{x}_i) = r_b \nabla_\theta \sum p(\hat{\mathbf{y}}_j|\mathbf{x}_i) \\
&= \cancel{r_b \nabla_\theta 1} = 0
\end{aligned} \tag{2.20}$$

so that this new estimator is unbiased. Notably, equation (2.20) is only valid if r_b is

[7] See Bienaymé's identity in probability theory.

independent of the underlying samples $\hat{\mathbf{y}}_j$[8] so that r_b can be moved outside of the expectation operator, resulting in an unbiased estimator. Combining equation (2.18) and the result of equation (2.20), the final equation would be

$$\nabla_\theta L_{\text{REINFORCE\_BASELINE}} = \frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}(r_i - r_b)\nabla_\theta \log p(\hat{\mathbf{y}}_i|\mathbf{x}_i) \tag{2.21}$$

Because of the importance of r_b, there is a plethora of literature about it, such as 1) constant baseline (Williams, 1992; Kimura et al., 1995, 1997) and 2) actor-critic methods (Barto et al., 1983; Kimura et al., 1998; Konda and Tsitsiklis, 1999, 2003).

Token-level reward We can expand the above equation to its token-level likelihood

$$\nabla_\theta L_{\text{REINFORCE\_BASELINE}} = \frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}(r_i - r_b)\sum_{t=1}^{T_i}\nabla_\theta \log p(\hat{y}_{i,t}|\mathbf{x}_i, \hat{\mathbf{y}}_{i,<t}) \tag{2.22}$$

where $\hat{y}_{i,t}$ is the token in timestep t on the sampled trajectory $\hat{\mathbf{y}}_i$. Notably, the reward r_i is the sentence-level reward so that the same numerical value is distributed over the corresponding token-level likelihood. We can design a reward scheme to sharpen/weaken the contribution of certain target tokens:

$$\nabla_\theta L_{\text{TOKEN\_LEVEL}} = \frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\sum_{t=1}^{T_i}(r_{i,t} - r_b)\nabla_\theta \log p(\hat{y}_{i,t}|\mathbf{x}_i, \hat{\mathbf{y}}_{i,<t}) \tag{2.23}$$

where $r_{i,t}$ is the reward on the token in timestep t on the sampled trajectory $\hat{\mathbf{y}}_i$.

2.2.3 Connectionist Temporal Classification (CTC)

Connectionist Temporal Classification (Graves et al., 2006) is another loss function for learning the alignment between a source-input and its target-input. Unlike AED which attention allows one-to-many (source-to-target) mapping, CTC learns the assignment of one target label to one or multiple timesteps on the source sequence, i.e., many-to-one mapping. Therefore, CTC only applies when the source sequence is longer than or equal to the target sequence, such as in speech recognition. CTC computes the likelihood $p_{\text{CTC}}(\mathbf{y}_i|\mathbf{x}_i)$ by considering all possible paths $\mathcal{B}^{-1}(\mathbf{y}_i)$ between the target

[8]We use the subscript j here instead of i for the target sequences because the target sequences in the entire space $\mathcal{Y}$ is independent of the source sequence $\mathbf{x}_i$.

sequence $\mathbf{y}_i$ and the source sequence $\mathbf{x}_i$:

$$p_{\text{CTC}}(\mathbf{y}_i|\mathbf{x}_i) = \sum_{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{y}_i)} p(\boldsymbol{\pi}|\mathbf{x}_i) \tag{2.24}$$

where $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_{T_x})$ is a path having the same length as $\mathbf{x}_i$ with $\pi_t \in \mathcal{V} \cup$ <b>. The *blank* token <b> represents an emission of a non-target label, e.g., silence or environmental noise in speech. Compared to its target sequence $\mathbf{y}_i$, each path $\boldsymbol{\pi}$ may have <b> tokens inserted and/or have repeated target tokens. The collapse function $\mathcal{B}()$ returns a target sequence $\mathbf{y}_i$ of a path $\boldsymbol{\pi}$ by removing all its <b> tokens and by deduplicating its consecutively repeated tokens. The inverse, i.e., $\mathcal{B}^{-1}()$, represents the set of all corresponding *valid* paths. Given a path $\boldsymbol{\pi}$, it is valid with respect to a target sequence $\mathbf{y}_i$ if it satisfies these properties:

- $\boldsymbol{\pi}$ contains all labels on $\mathbf{y}_i$ in the order presented in $\mathbf{y}_i$. In other words, CTC captures the monotonic alignment between $\mathbf{x}_i$ and $\mathbf{y}_i$.
- Between two repeated tokens on $\mathbf{y}_i$, there must be one or more <b> tokens inserted on $\boldsymbol{\pi}$. Otherwise, the repeated tokens would be deduplicated by the $\mathcal{B}()$, violating the above rule.

For example, in a transcription task of mapping 10 audio frames to a single word "hello", a valid $\boldsymbol{\pi}$ would be '<b> h e l <b> <b> l l o o' whereas '<b> h l e <b> <b> l l o o' and '<b> h e <b> l l l <b> o o' are invalid.

Unlike cross-entropy in AED system, CTC assumes conditional independence between the network outputs. Each sequence probability $p(\boldsymbol{\pi}|\mathbf{x}_i)$ in equation (2.24) is thus factorized into a product of conditional probabilities of π_t given $\mathbf{x}_i$:

$$p(\boldsymbol{\pi}|\mathbf{x}_i) = \prod_{t=1}^{T_x} p(\pi_t|\mathbf{x}_i) \tag{2.25}$$

where $p(\pi_t|\mathbf{x}_i)$ refers to the model's output probability in timestep t.

In training, given a pair of $(\mathbf{x}_i, \mathbf{y}_i)$, we enumerate over all paths in $\mathcal{B}^{-1}(\mathbf{y}_i)$, gather the sequence probability of each path $\boldsymbol{\pi}$ and sum them up to compute $p_{\text{CTC}}(\mathbf{y}_i|\mathbf{x}_i)$, following equation (2.24) and 2.25. The objective function is derived from Maximum Likelihood Estimation on the ground-truth target sequence $\mathbf{y}_i$, or equivalently, we take the negative log-likelihood on $\mathbf{y}_i$:

$$L = -\log p_{\text{CTC}}(\mathbf{y}_i|\mathbf{x}_i) \tag{2.26}$$

A critical issue, however, is the potentially large number of valid paths available in $\mathcal{B}^{-1}(\mathbf{y}_i)$. The sum becomes intractable for long source sequence and large vocabulary, which is common in speech-to-text.

Dynamic programming is adopted to address the above intractability problem by merging paths reaching the same label at the same timestep t. More specifically, it breaks down the sum over paths corresponding a label sequence $\mathbf{y}_i$, i.e., equation (2.24), into an iterative sum over paths corresponding to prefixes of that label sequence. The iterative sum can be efficiently computed by using forward variables $\alpha_t(s)$ and backward variables $\beta_t(s)$ on a *modified* label sequence $\mathbf{y}'$, similar to the forward-backward algorithm in HMM:

$$p_{\text{CTC}}(\mathbf{y}_i|\mathbf{x}_i) = \sum_{s=1}^{|\mathbf{y}'|} \frac{\alpha_t(s)\beta_t(s)}{y^t_{\mathbf{y}'_s}} \tag{2.27}$$

where $y^t_{\mathbf{y}'_s} = p(\pi_t = \mathbf{y}'_s|\mathbf{x}_i)$ refers to the model's output probability at timestep t on the sth token of $\mathbf{y}'$. The modified label sequence $\mathbf{y}'$ has <b> tokens inserted at the beginning and at the end and between every pair of labels so that its length would be $2|\mathbf{y}_i| + 1$. For example, the modified label sequence $\mathbf{y}'$ of a label sequence $\mathbf{y}_i$ = hello would be '<b> h <b> e <b> l <b> l <b> o <b>'. Notably, $\mathbf{y}'$ indicates possible emissions of <b> tokens between the target tokens and is reduced to the label sequence $\mathbf{y}_i$ when it is processed by the collapse function $\mathcal{B}()$.

The forward variable $\alpha_t(s)$, where s runs on $\mathbf{y}'$, describes the model's total probability of all valid paths which reach a prefix $\mathbf{y}'_{1:s}$ of size s at timestep t:

$$\alpha_t(s) = \sum_{\substack{\boldsymbol{\pi} \in N^{T_x}: \\ \mathcal{B}(\boldsymbol{\pi}_{1:t}) = \mathbf{y}'_{1:s}}} \prod_{t'=1}^{t} y^{t'}_{\pi_{t'}} \tag{2.28}$$

with the below valid initialization conditions:

$$\alpha_1(1) = y^1_{<\text{b}>} \tag{2.29}$$

$$\alpha_1(2) = y^1_{\mathbf{y}_1} \tag{2.30}$$

$$\alpha_1(s) = 0, \forall s > 2 \tag{2.31}$$

That is, we only include paths which begin with either a <b> token or the first non-blank token, i.e., $\mathbf{y}_1$, of the target sequence $\mathbf{y}_i$. Given the initial conditions, we

can compute $\alpha_t(s)$ recursively from its successors:

$$\alpha_t(s) = \begin{cases} (\alpha_{t-1}(s) + \alpha_{t-1}(s-1))y^t_{\mathbf{y'}_s} & \text{if } \mathbf{y'}_s = b \text{ or } \mathbf{y'}_{s-2} = \mathbf{y'}_s \\ (\alpha_{t-1}(s) + \alpha_{t-1}(s-1) + \alpha_{t-1}(s-2))y^t_{\mathbf{y'}_s} & \text{otherwise} \end{cases} \tag{2.32}$$

The top condition(s) indicates that a valid transition from step $t-1$ to step t with emission of <b> token ($\mathbf{y'}_s = b$) must come from either a <b> token state or a non-blank token state. Additionally, in case of consecutive non-blank tokens on the original label sequence $\mathbf{y}_i$ (equivalently, $\mathbf{y'}_{s-2} = \mathbf{y'}_s$ on the *modified* label sequence), e.g., 'll' in 'hello', the emission of the second token at timestep t must either come from a state that represents its emission in previous timestep $t-1$ (not the emission of the first consecutive target token but the previous emission of the second target token) or come from the transition of a <b> token state $\alpha_{t-1}(s-1 =< \text{b} >)$. The bottom condition refers to the transition to a state of non-blank and non-repeated token. Such transition is valid only from either a state of its previous emission $\alpha_{t-1}(s)$, a state of a blank token $\alpha_{t-1}(s-1)$ or a state of another non-blank and non-repeated token $\alpha_{t-1}(s-2)$.

Similarly, the backward variable $\beta_t(s)$ defines the total probability of $\mathbf{y'}_{s:|\mathbf{y'}|}$ at time t:

$$\beta_t(s) = \sum_{\substack{\boldsymbol{\pi} \in N^{T_x}: \\ \mathcal{B}(\boldsymbol{\pi}_{t:T_x}) = \mathbf{y'}_{s:|\mathbf{y'}|}}} \prod_{t'=t}^{T_x} y^{t'}_{\pi_{t'}} \tag{2.33}$$

with the below termination conditions:

$$\beta_{T_x}(|\mathbf{y'}|) = y^{T_x}_{<\text{b}>} \tag{2.34}$$

$$\beta_{T_x}(|\mathbf{y'}| - 1) = y^{T_x}_{\mathbf{y}_{i,|\mathbf{y}_i|}} \tag{2.35}$$

$$\beta_{T_x}(s) = 0, \forall s < |\mathbf{y'}| - 1 \tag{2.36}$$

That is, we only include suffix paths which either end with a <b> token or with $\mathbf{y}_{i,|\mathbf{y}_i|}$, i.e., the last non-blank label of $\mathbf{y}_i$. Additionally, we exclude paths which fail to predict the entire target sequence $\mathbf{y}_i$. Given the termination conditions, we can also compute $\beta_t(s)$ recursively:

$$\beta_t(s) = \begin{cases} (\beta_{t+1}(s) + \beta_{t+1}(s+1))y^t_{\mathbf{y'}_s} & \text{if } \mathbf{y'}_s = b \text{ or } \mathbf{y'}_{s+2} = \mathbf{y'}_s \\ (\beta_{t+1}(s) + \beta_{t+1}(s+1) + \beta_{t+1}(s+2))y^t_{\mathbf{y'}_s} & \text{otherwise} \end{cases} \tag{2.37}$$

The top condition(s) indicates that a state of a blank token ($\mathbf{y'}_s = b$) at timestep t can only transit to either a <b> token state, i.e., the $\beta_{t+1}(s)$ term, or a non-blank target token, i.e., the $\beta_{t+1}(s+1)$ term. Additionally, if s is the first of the consecutive tokens on the original label sequence $\mathbf{y}_i$, it can only transit to itself, i.e., the $\beta_{t+1}(s)$ term, or to the <b> token state, i.e., the $\beta_{t+1}(s+1)$ term. The bottom condition indicates that a non-blank and non-repeated token (with respect to $\mathbf{y}$) can transit to itself, i.e., the $\beta_{t+1}(s)$ term, a blank token state, i.e., the $\beta_{t+1}(s+1)$ term, or the next non-blank and non-repeated token, i.e., the $\beta_{t+1}(s+2)$ term.

For a label sequence $\mathbf{y}_i$, the product of $\alpha_t(s)$ and $\beta_y(s)$ at a given s and t is the probability of all the paths $\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{y}_i)$ that go through the symbol s at time t on the modified label sequence $\mathbf{y}'$:

$$
\begin{aligned}
\alpha_t(s)\beta_t(s) &= \left(\sum_{\substack{\boldsymbol{\pi} \in N^{T_x}: \\ \mathcal{B}(\boldsymbol{\pi}_{1:t}) = \mathbf{y'}_{1:s}}} \prod_{t'=1}^{t} y^{t'}_{\pi_{t'}} \right) \left(\sum_{\substack{\boldsymbol{\pi} \in N^{T_x}: \\ \mathcal{B}(\boldsymbol{\pi}_{t:T_x}) = \mathbf{y'}_{s:|\mathbf{y}'|}}} \prod_{t''=t}^{T_x} y^{t''}_{\pi_{t''}} \right) \\
&= \left(\sum_{\substack{\boldsymbol{\pi} \in N^{T_x}: \\ \mathcal{B}\boldsymbol{\pi}_{1:t}) = \mathbf{y'}_{1:s}}} \prod_{t'=1}^{t} y^{t'}_{\pi_{t'}} \right) \left(\sum_{\substack{\boldsymbol{\pi} \in N^{T_x}: \\ \mathcal{B}(\boldsymbol{\pi}_{t:T_x}) = \mathbf{y'}_{s:|\mathbf{y}'|}}} y^{t}_{\pi_t} \prod_{t''=t+1}^{T_x} y^{t''}_{\pi_{t''}} \right) \\
&= \left(\sum_{\substack{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{y}_i): \\ \pi_t = \mathbf{y'}_s}} \prod_{t'=1}^{t} y^{t'}_{\pi_{t'}} y^{t}_{\pi_t} \prod_{t''=t+1}^{T_x} y^{t''}_{\pi_{t''}} \right) = \left(\sum_{\substack{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{y}_i): \\ \pi_t = \mathbf{y'}_s}} y^{t}_{\mathbf{y'}_s} \prod_{t'=1}^{T_x} y^{t'}_{\pi_{t'}} \right) \\
&= y^{t}_{\mathbf{y'}_s} \left(\sum_{\substack{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{y}_i): \\ \pi_t = \mathbf{y'}_s}} \prod_{t'=1}^{T_x} y^{t'}_{\pi_{t'}} \right) = y^{t}_{\mathbf{y'}_s} \left(\sum_{\substack{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{y}_i): \\ \pi_t = \mathbf{y'}_s}} p(\boldsymbol{\pi}|\mathbf{x}_i) \right)
\end{aligned}
$$

Rearranging the above, we obtain the total probability of a label sequence $\mathbf{y}_i$ given $\mathbf{x}_i$ under the constraint of $\pi_t = \mathbf{y'}_s$:

$$
\frac{\alpha_t(s)\beta_t(s)}{y^{t}_{\mathbf{y'}_s}} = \sum_{\substack{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{y}_i): \\ \pi_t = \mathbf{y'}_s}} p(\boldsymbol{\pi}|\mathbf{x}_i) \tag{2.38}
$$

Finally, we can get equation (2.25) by summing equation (2.38) over all s over the modified labelling $\mathbf{y}'$.

2.3 Inference: Greedy Decoding and Beam Search

In inference, the model generates a sequence of probable tokens based on some decoding methods. Each generated token is fed as the target input of the next timestep until the termination criteria is met. There are numerous decoding methods which are typically either sampling-based, such as temperature-based sampling, or search-based, such as beam-search. In this section, we review two search-based methods: greedy decoding and beam-search decoding, which usually perform better than sampling-based methods in NeurS2S learning.

In greedy decoding, we select the target token of the highest probability at each timestep t to form the output sequence $\hat{\mathbf{y}}_i$:

$$\hat{y}_{i,t} = \arg\max_{y \in \mathcal{V}} p_\theta(y|\hat{\mathbf{y}}_{i,<t}, \mathbf{x}_i) \tag{2.39}$$

Greedy decoding is computationally simple but it is sub-optimal. Because of the left-to-right dependency in decoding, the most probable token at a timestep may result in a sequence with lower probability than another sequence generated with a different token at that timestep. Beam search decoding aims to remedy such sub-optimal decision by maintaining multiple sequences, called beams, during its search.

In beam-search, it maintains b number of the most probable sequences, or called b beams, instead of picking the most likely token at each timestep. For each timestep t, each beam is expanded into another b number of sequences by appending each of the top-b most probable target tokens, which results in $b \cdot b$ sequences. The best b sequences are selected for the next timestep according to their sequence log-probability[9]. This process is repeated until all the b hypotheses encounter the <eos> token, and the beam with the highest log-probability is the model output. Notably, when $b = 1$, beam-search is reduced to the greedy decoding.

In vanilla beam-search, short sentences are preferred because each summed log-likelihood term is negative. One heuristic based solution is to normalise the sum of log-probabilities by the number of decoded target tokens during beam selection. When the search is terminated, similarly, we return the beam with the highest average log-probabilities per word. Wu et al. (2016) proposed a better heuristic that modifies the length normalization term $\text{lp}(\hat{\mathbf{y}}_j)$ and includes a coverage penalty term $\text{cp}(\mathbf{x}_i, \hat{\mathbf{y}}_j)$[10]. The modified length normalization, i.e., $\text{lp}(\hat{\mathbf{y}}_j) = \frac{(5+|\hat{\mathbf{y}}_j|)^\alpha}{(5+1)^\alpha}$, includes a hyper-parameter α that control its strength with 0 being no length normalization. The coverage penalty,

[9] It is more numerically stable by summing the log-probabilities than multiplying the probabilities.

[10] We use the subscript j instead of i in $\hat{\mathbf{y}}_j$ because there are multiple such sequences in beam-search.

i.e., $\text{cp}(\mathbf{x}_i, \hat{\mathbf{y}}_j) = \beta \sum_{k=1}^{|\mathbf{x}_i|} \log(\min(\sum_{t=1}^{|\hat{\mathbf{y}}_j|} p_{tk}, 1.0))$, penalizes decoded sequences that use only a few source tokens in generation (the min operator), resulting in outputs that better cover the entire source sequence. It uses the attention probability p_{tk}, such that $\sum_{k=1}^{|\mathbf{x}_i|} p_{tk} = 1$, to measure the degree of coverage and has a hyper-parameter β to control its strength/contribution to the final score function $s(\hat{\mathbf{y}}_j, \mathbf{x}_i)$:

$$s(\hat{\mathbf{y}}_j, \mathbf{x}_i) = \frac{\log p_\theta(\hat{\mathbf{y}}_j|\mathbf{x}_i)}{\text{lp}(\hat{\mathbf{y}}_j)} + \text{cp}(\mathbf{x}_i, \hat{\mathbf{y}}_j) \tag{2.40}$$

2.4 Evaluation

In our experiments, we use automatic evaluation metrics to assess the quality of our model's output. In speech recognition, we use Word Error Rate (WER) which is based on edit-distance between the model's output and its gold-reference transcriptions. In machine translation, we use BLEU (Bilingual Evaluation Understudy) and chrF2 (character n-gram F-score) which measure the similarity between two input strings based on n-gram matching in word-level and character-level respectively.

2.4.1 Word Error Rate

Word Error Rate (WER) is a common metric used in speech recognition. It is based on Levenshtein distance (LevDist) which computes the minimum number of valid operations required to convert one string (StrA) to another (StrB):

$$\text{LevDist(StrA, StrB)} = \text{Insertion} + \text{Deletion} + \text{Substitution} \tag{2.41}$$

The valid operations are 1) insertion, 2) deletion and 3) substitution, and their numbers are computed via dynamic programming. In general, a lower value of LevDist indicates a higher similarity between the input strings. Given the Levenshtein distance between the model's output (StrA) and its gold-reference transcription (StrB), the WER is:

$$\text{WER(StrA, StrB)} = \frac{\text{LevDist(StrA, StrB)}}{\text{Length(StrB)}} \times 100 \tag{2.42}$$

In other words, WER is the Levenshtein distance normalized by the length of the gold-reference transcription. Notably, WER is not a distance metric so its value is affected by the order of inputs.

Because of its positive correlation with LevDist, a lower value of WER usually indicates higher transcription quality. However, WER only captures the lexical error in the model's output, irrespective of the semantics which can be crucial for downstream application such as natural language understanding task (Kim et al., 2022).

2.4.2 BLEU

BLEU (Papineni et al., 2002) is one of the most popular corpus-level metric used in evaluating machine translation. It is based on the n-gram overlap between the model output and its gold-reference translation. More specifically, BLEU score is computed by taking the geometric average of different n-gram precision. As only precision is into account, the score is biased towards short translation. The authors thus introduced a brevity penalty (BP) term to counterbalance such bias resulting in the below formula:

$$\text{BLEU-N} = \text{BP} \times \exp\left(\sum_{i=1}^{N} w_i \log p_i\right) \tag{2.43}$$

$$\text{BP} = \min(1, e^{1-\frac{\text{reference length}}{\text{output length}}}) \tag{2.44}$$

where N, usually taken to be 4, refers to the maximum gram taken into calculation and p_i is the (modified) precision of the i-gram:

$$p_n = \frac{\sum_{C \in \{\text{Candidates}\}} \sum_{\text{n-gram} \in C} \text{Count}_{\text{clip}}(\text{n-gram})}{\sum_{C' \in \{\text{Candidates}\}} \sum_{\text{n-gram'} \in C'} \text{Count}(\text{n-gram}')} \tag{2.45}$$

In general, a higher value of BLEU score indicates better n-gram matching and hence better translation quality. In spite of its popularity, recent studies show poor correlation between BLEU score and human evaluation (Kocmi et al., 2021).

2.4.3 Character n-gram F-score (chrF)

Character-F score (Popović, 2015) is another commonly used n-gram matching based metric in machine translation. Unlike BLEU score, chrF score calculates character n-gram instead of word-level n-gram overlap. In addition to character n-gram precision, it also takes recall into account in order to better compensate the bias toward short translation. The formula for the chrF score is:

$$\text{chrF}\beta = (1+\beta^2)\frac{\text{chrP} \times \text{chrR}}{\beta^2 \text{chrP} + \text{chrR}} \tag{2.46}$$

where β is a weight indicating the importance of recall over precision[11] stand for character n-gram precision and recall averaged arithmetically over all n-grams. In another empirical study (Popović, 2016), chrF2, i.e., n-gram=6 and $\beta = 2$, correlates better with human rankings. Additionally, chrF2 is shown to have higher correlation to human judgement than BLEU score in measuring translation quality.

[11] chrF $\xrightarrow{\beta \to \infty}$ chrR

Chapter 3

Data Scarcity and Annotation

In this chapter, we provide information about the data scarcity issue and review literature related to common augmentation techniques used in sequence-to-sequence learning.

3.1 Human Annotation

Human annotation is the most intuitive way to obtain labelled data for training deep neural network. For example, in machine translation, one can pay human translators to translate a given passage or transcriptions to other languages (Post et al., 2013; Kreutzer et al., 2020). Under the rapid development of AI research and its deployment, human annotation is now a highly profitable business, such as Amazon Mechanical Turk[1] and Scale AI[2].

However, human annotation is not very scalable in terms of monetary cost and time. At the current market rate, the cost[3] of hiring human translator is about 0.1 USD/word or 23 USD/hour. Its cost would be even higher, depending on factors such as domains and language pairs. This calls the need of more automated methods for efficient training signals.

3.2 Pseudo-Labeling

Pseudo-labeling is one major method for automated labeling. It applies pre-trained models to generate labels on unlabeled data, such as applying an ASR on unlabeled

[1]https://www.mturk.com/
[2]https://scale.com/pricing
[3]https://search.proz.com/?sp=pfe/rates (Dated 30th Jan 2023)

Algorithm 1: Pseudo-Labeling for Sequence-To-Sequence Learning
Input: initial model θ_0, unlabeled data U, labeled data L
Output: converged θ^*

1 Train θ_0 on L to get a base model θ_1;
2 $\theta \leftarrow \theta_1$;
3 **repeat**
4 Generate pseudo labels $\hat{\mathbf{y}} \sim p_\theta(\mathbf{y}|\mathbf{x})$ where $\mathbf{x} \sim U$;
5 Select pseudo-labelled subset $\hat{L} \subset \{\mathbf{x}, \hat{\mathbf{y}}\}_{i=1}^{|U|}$;
6 Train a new model θ_2 on $L \cup \hat{L}$;
7 $\theta \leftarrow \theta_2$;
8 **until** θ *converged*;

speech data to generate more paired speech-transcription data. In sequence-to-sequence learning, such method is called *Self-Training* (III, 1965) if the source sequence is unlabeled. In case of unlabeled target sequence, it is called *Back-Translation* (Sennrich et al., 2016a; Edunov et al., 2018a).

Self-Training and Back-Translation are very similar to each other, possibly except the side of the unlabeled sequence. Algorithm 1 shows the major procedures in Self-Training which takes a randomly initialized model θ_0, unlabeled data U and labeled data L as inputs. The initial model θ_0 is first trained on labeled data L (lines 1-2) to obtain a base, or call pre-trained, model θ_1 for labeling. After that, we use the base model θ_1 to generate pseudo-labels $\hat{y}$ for unlabelled data U (line 4), followed by filtering for quality control (line 5). Finally, we train a new θ_2 on the combined data $L \cup \hat{L}$ (lines 6-7). The entire procedure is repeated[4] until termination, e.g, no further improvement on the validation set.

Noisy pseudo-labeling In spite of having more training data, it is puzzling how the targeted model benefits from training on its own predictions. He et al. (2020) attributes the success of self-training to its regularization effect which helps semantically similar inputs to have closed predictions. They find that both beam-search decoding in pseudo-label generation and noise injection during pseudo-label training are crucial. In standard pseudo-labeling, e.g., in algorithm 1, the injected noise is dropout which introduces perturbation in the hidden representation while keeping the inputs fixed. In their introduced *Noisy Pseudo-Labeling*, they examine input perturbation, such as randomly dropped input tokens in NMT, which is shown to perform better, when applied together with dropout. Algorithm 2 shows the steps in noisy pseudo labeling;

[4]When the procedure is repeated more than once, they are sometimes called iterative self-training or iterative back-translation instead.

Algorithm 2: Noisy Pseudo-Labeling for Sequence-To-Sequence Learning
Input: initial model θ_0, unlabeled data U, labeled data $L = \{\mathbf{x_i}, \mathbf{y_i}\}_{i=1}^{l}$, perturbation function $g()$
Output: converged θ^*

1 Train θ_0 on L to get a base model θ_1;
2 $\theta \leftarrow \theta_1$;
3 **repeat**
4 Generate pseudo labels $\hat{\mathbf{y}} \sim p_\theta(\mathbf{y}|\mathbf{x})$ where $\mathbf{x} \sim U$;
5 Select pseudo-labelled subset $\hat{L} \subset \{\mathbf{x}, \hat{\mathbf{y}}\}_{i=1}^{|U|}$;
6 Train a new model θ_2 on $L \cup \hat{L}$ with perturbed input $g(\mathbf{x})$;
7 $\theta \leftarrow \theta_2$;
8 **until** θ *converged*;

it is almost identical to algorithm 1, except using perturbed input when training on $L \cup \hat{L}$ (line 6).

Although the above analysis is limited to self-training, there are similar works conducted for back-translation, showing the effectiveness of noise injection (in hidden space) and the use of beam-search over sampling (Edunov et al., 2018b).

Connection to Sequence-Level Knowledge Distillation (SeqKD) Sequence-Level Knowledge Distillation (Kim and Rush, 2016) is a similar technique which also generates more training data by using a pre-trained model for labeling. However, it aims at model compression (Breiman and Shang, 1996) for better memory footprint without sacrificing much performance. SeqKD is a type of Knowledge Distillation (KD) which boosts the performance of a smaller neural network, called student $q_\theta(\mathbf{x}|\mathbf{y})$, by training it also on data labeled by a pre-trained massive neural network, called teacher $p_\theta(\mathbf{y}_i|\mathbf{x}_i)$. In word-level KD, the objective L is to minimize the cross-entropy loss between the teacher model and the student model on the original pair of training data $(\mathbf{x}_i, \mathbf{y}_i)$

$$\mathcal{L}_{\text{word-KD}} = -\sum_{t=1}^{T_y}\sum_{k=1}^{|\mathcal{V}|} p_\phi(y_t = k|\mathbf{x}_i) \times \log q_\theta(y_t = k|\mathbf{x}_i) \tag{3.1}$$

where $\mathcal{V}$ is the vocabulary set. Word-level KD focuses on transfer in token level by allowing the student model to mimic the token level behaviour of the teacher model. It is rather effective, but it ignores the training signal for modeling the sequence-level distribution. In SeqKD, the student model's target is to mimic the sequence-level

distribution which can be approximated by beam-search because of its intractability:

$$\mathcal{L}_{\text{SeqKD}} = -\sum_{\mathbf{y}_j \in \mathcal{Y}} p_\phi(\mathbf{y}_j|\mathbf{x}_i) \log q_\theta(\mathbf{y}_j|\mathbf{x}_i) \approx -\log q_\theta(\hat{\mathbf{y}}_i|\mathbf{x}_i) \tag{3.2}$$

where $\mathcal{Y}$ represents the space of all possible target sequences and $\hat{\mathbf{y}}_i$ refers to the beam-search output from the teacher model. Normally, both SeqKD and word-level KD can be applied together.

The compression ability can be attributed to the reduction of training data complexity, such as the degree of target tokens dependency (Ren et al., 2020), the number of modes in the output distribution (Gu et al., 2018), the lexical diversity and the degree of word reordering (Xu et al., 2021b), which makes fitting easier for a model of reduced complexity.

Notably, SeqKD is mainly applied on labeled data to create another target sequence for easier student model learning. Its purpose is different from self-training that leverages a large amount of unlabeled data to relieve limited label data scenario, i.e., semi-supervised learning. Furthermore, self-training typically encounters domain mismatch problem because the unlabeled data are mined from various sources and domains.

3.3 Active Learning

Pseudo-labeling is a purely algorithmic approach which generates more labeled data by applying a pre-trained model on vast amount of unlabeled data. Its strength lies in the relatively low cost in producing vast amount of training data. However, its effectiveness is also limited by the quality of the pre-trained model, especially the domain similarity between its training data and the unlabeled data. Related remedies such as unsupervised domain adaptation (Kouw and Loog, 2018) are therefore used together with pseudo-labeling for better performance.

In spite of the aforementioned remedies, human annotation is still perceived as more effective in creating high-quality training data in the target domain. Its weakness, however, lies in the cost of massive production. Is there an approach which can leverage the data quality provided by human annotation while controlling the annotation budget? This calls Active Learning (Cohn et al., 1994).

Active Learning (AL) aims to maximize the benefit of human annotation under a fixed or limited cost budget. Its idea is to select a subset of unlabeled data for human annotation, such that the resulting model would still have competitive performance,

resulting in reduction of annotation cost. Its key element is a scoring function $\phi()$ for data selection, such as confidence estimation (Gandrabur and Foster, 2003; Ueffing et al., 2003; Blatz et al., 2004; Quirk, 2004; Ueffing and Ney, 2007), predictive entropy (Joshi et al., 2009), random sampling (Gangadharaiah et al., 2009; Miura et al., 2016; Zeng et al., 2019), cosine similarity of sentence embedding (Zhang et al., 2018; Hu and Neubig, 2021), round trip translation likelihood (Haffari et al., 2009), or other uncertainty measures, which defines the annotation criteria. In the following, we review a few common selection approaches in active learning for machine translation.

Random sampling It is the simplest data selection strategy in picking an unlabeled subset for annotation. In spite of its simplicity, it is effective since it is an unbiased estimation of the (unlabeled) data distribution. Furthermore, it is computationally simpler.

Confidence sampling The idea of confidence sampling is to estimate the quality of a translation according to either the word-level or sentence-level confidence score. In word-level confidence score (Blatz et al., 2004; Ueffing and Ney, 2007), the confidence $C_w(\mathbf{x}_i, y_{i,t})$ between the source sequence $\mathbf{x}_i$ and a target word $y_{i,t}$ is

$$C_w(\mathbf{x}_i, y_{i,t}) = \max_{0 \le j \le J} p(y_{i,t}|x_{i,j}) \tag{3.3}$$

where $p(y_{i,t}|x_{i,j})$ is the alignment probability of $y_{i,t}$ and $x_{i,j}$ given by an IBM Model 2. In sentence-level, the confidence score $C_{\text{sent}}(\mathbf{x}_i, \mathbf{y}_i)$ between the source sequence $\mathbf{x}_i$ and the target sequence $\mathbf{y}_i$ is

$$C_{\text{sent}}(\mathbf{x}_i, \mathbf{y}_i) = \frac{\{y_{i,t} \in \mathbf{y}_i | C_w(\mathbf{x}_i, y_{i,t}) > \tau_w\}}{|\mathbf{y}_i|} \tag{3.4}$$

where τ_w, a hyper-parameter, is a word confidence threshold.

Cosine similarity between embeddings In this method, its idea is to select unlabeled sequences that are distant from the out-of-domain labeled data. Human annotation on such distant unlabeled data instances should provide richer information gain to the underlying model. The metric used for measuring the distance between two input sequences $\mathbf{x}_i$ and $\mathbf{x}_l$ is based on the cosine similarity between their sentence embeddings[5] $\mathbf{e}_{\mathbf{x}_i}$ and $\mathbf{e}_{\mathbf{x}_l}$ (Zhang et al., 2018) or a modified cosine similarity (Artetxe

[5]The sentence embedding can be obtained by averaging the word representations from a pre-trained model such as mBERT.

and Schwenk, 2019) that also includes the average cosine similarity with their k nearest neighbors

$$\text{dist}(\mathbf{x}_i, \mathbf{x}_l) = \frac{\cos(\mathbf{e}_{\mathbf{x}_i}, \mathbf{e}_{\mathbf{x}_l})}{\sum\limits_{z \in \text{NN}_k(\mathbf{x}_i)} \frac{\cos(\mathbf{e}_{\mathbf{x}_i}, \mathbf{e}_z)}{2k} + \sum\limits_{z \in \text{NN}_k(\mathbf{x}_l)} \frac{\cos(\mathbf{e}_{\mathbf{x}_l}, \mathbf{e}_z)}{2k}} \tag{3.5}$$

Given an unlabeled sequence $\mathbf{x}_i$, ideally, we would iterate equation (3.5) over all labeled sequences $\mathbf{x}_l \subset \mathcal{L}$, but, in practise, we instead iterate over a sampled subset $\mathcal{L}'$ owing to limited computing resources. The final scoring function on $\mathbf{x}_i$ (Hu and Neubig, 2021) is the minimum distance to labeled instances within $\mathcal{L}'$

$$\phi(\mathbf{x}_i) = \min_{\mathbf{x}_l \in \mathcal{L}'} \text{dist}(\mathbf{e}_{\mathbf{x}_i}, \mathbf{e}_{\mathbf{x}_l}) \tag{3.6}$$

In both ASR and MT, there is a plethora of literature about active learning. In spite of comparison made between the baselines and the proposed method, there is no clear winner(s) of which selection method is the best. Furthermore, a majority of works only highlights the cost reduction in terms of the number of annotated instances, irrespective of the instance-specific annotation cost, such as effort in post-editing. Therefore, there is a line of work which combines active learning and interactive-predictve machine translation (Foster et al., 1997; Barrachina et al., 2009) for reduction in both annotation cost and annotation effort (González-Rubio et al., 2011; González-Rubio and Casacuberta, 2014).

3.4 Noise Injection

In addition to labeling, noise or perturbation injection is another common way of increasing the amount of training data or to reduce overfitting. Typically, noise injection is computationally cheaper than labeling because of its independence of the pre-trained models for labeling.

In NMT, an example of simple perturbation is random replacement of tokens on both source and target sides (Fadaee et al., 2017a; Wang et al., 2018b). In ASR, simple perturbation on speech inputs can range from changing its speed (Ko et al., 2015), stretching its temporal component dynamically (Nguyen et al., 2020), or masking out sections (Park et al., 2019). Despite the vast amount of noise injection strategies, typically, they can be applied together either sequentially (Cubuk et al., 2019) or in a multi-task fashion (Sánchez-Cartagena et al., 2021) to create more complex transformation for better regularization effect.

In this section, we review SpecAugment (Park et al., 2019) because it is computa-

tionally simple and effective. It is commonly applied to most speech related tasks and in our works as baselines.

3.4.1 SpecAugment

There are three major components in the original formulation of SpecAugment on a spectrogram with its y-axis being frequency channels and x-axis being the timesteps. They are 1) time warping, 2) frequnecy masking, and 3) time masking.

Time warping Given a log Mel spectrogram with τ timesteps, a random point along the time axis passing through the center of the spectrogram within the time steps $(W, \tau - W)$ is to be warped either to the left or right by a distance w. W is the time warp parameter, a hyper-parameter, which defines the upper bound of a uniform distribution $\sim U(0, W)$ for w to be sampled from. Six anchor points on the boundary are fixed– four corner points and the midpoints of the vertical edges.

Frequency masking In frequency masking, f_0 consecutive mel frequency channels $[f_0, f_0 + f)$ are masked. Both parameters f_0 and f are sampled from their distributions– f follows an uniform distribution $\sim U(0, F)$ bounded by a hyper-parameter F whereas f_0 is sampled from an interval $[0, \nu - f)$. The parameter ν is the number of frequency channels.

Time masking Similar to frequency masking, time masking masks out consecutive frames in the interval of $[t_0, t_0 + t)$. The parameter t is sampled from an uniform distribution from 0 to a hyper-parameter T whereas t_0 is chosen from $[0, \tau - t)$.

In masking, masked areas are filled either by zero or the mean value of the spectrogram. The two formulation are equivalent if the spectrogram is normalized over all its elements. The above three strategies can be used together during training to provide diverse perturbations, Figure 3.1. However, time warping only brings marginal improvement over masking despite being more computationally demanding. Most follow up works, including ours, thus ignore time warping when using SpecAugment.

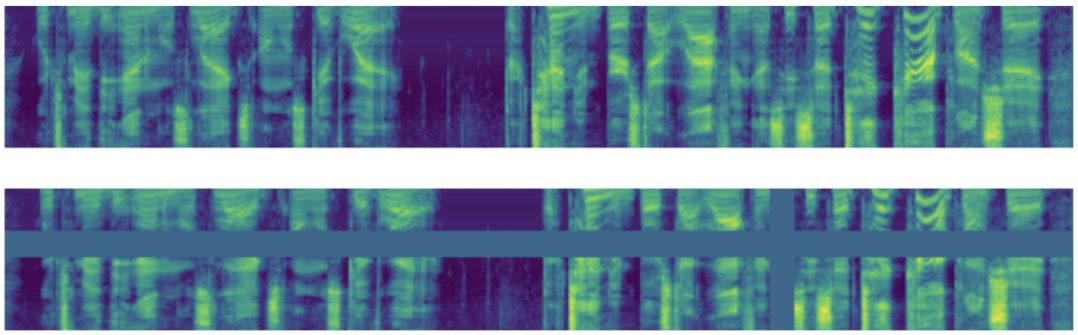

Figure 3.1: Spectrogram before and after SpecAugment. **Top**: Before augmentation. **Bottom**: After augmentation by the three strategies in SpecAugment. Source: figure 2 in (Park et al., 2019).

Chapter 4

Within-corpus Data Augmentation

In this chapter, we present our idea of within-corpus data augmentation for relieving data scarcity issue via increasing the amount of effective training data. To make the augmentation algorithms simple and effective, we define three properties 1) *on-the-fly*, 2) *memory-efficient* and *source-target alignment*, and illustrate their effectiveness on (end-to-end) speech-to-text applications.

Materials in this chapter have been drawn from the following publications: INTERSPEECH 2021 (Lam et al., 2021b), ACL 2022 (Lam et al., 2022b) and ICASSP 2023 (Lam et al., 2023).

4.1 Introduction and Overview

Expert annotation or crowd-sourcing are straightforward approaches for the aforementioned data scarcity issue. These methods include human annotators to create more training data but are costly and time consuming. In addition, it is unclear if the expert annotated data cause substantial distribution shift, e.g., writing style or content domain, to the underlying model.

Data augmentation instead is a purely algorithmic approach. It improves model performance by increasing the amount of training data and by reducing overfitting at the same time. Because of its effectiveness, there is a plethora of works about it over most application areas in deep learning.

Some data augmentation methods are computationally simpler. In computer vision, this approaches apply geometric transformations or add noise to images to make the models more robust (LeCun et al., 1998; Simard et al., 2003). In MT, a straightforward idea is to replace tokens on both source and target texts (Wang et al., 2018b; Fadaee et al., 2017b). In ASR, this computationally simpler methods usually

inject perturbations on the audio inputs, e.g. changing speed (Ko et al., 2015) or dynamic time stretching (Nguyen et al., 2020), or masking out sections (Park et al., 2019). Computationally simpler methods usually augment data *on-the-fly* through injecting simple but usually *unaligned* perturbation[1] while more complex methods cannot be applied in this manner. Since the perturbation is applied on the existing parallel data, this kind of augmentation usually does not cause huge domain shift.

More complex methods usually uses (external) model(s) to generate new source-target training pairs from the unpaired data. Pseudo-labeling (Xu et al., 2020; Chen et al., 2020; Zhang et al., 2020; Xu et al., 2021a), is such a more complex method. In NMT, a notable example is back-translation (Sennrich et al., 2016b) that trains a NMT model in the opposite language direction to label the monolingual data. In ASR, such examples are noisy student training (Park et al., 2020) and synthesizing speech or speech representation via a TTS-like system (Tjandra et al., 2017; Hayashi et al., 2018; Hori et al., 2019; Rosenberg et al., 2019; Wang et al., 2020c; Chen et al., 2021). Pseudo-labeling is a powerful method as it can take the large amount of unpaired data into the training process. It has also been shown for its complementarity with self-supervised learning (Wang et al., 2021b; Liu et al., 2021). However, pseudo-labelling usually encounters a trade-off between the *on-the-fly* property and the *memory-efficient* property. If the augmented data is generated before training, they require extra memory for storage. If the augmented data is created during training, the training process can be substantially slowed down because the data generation usually involves the inference of a deep neural network. In addition, pseudo-labeling can cause substantial domain shift if the given unpaired data is not of the same domain as the parallel data.

Therefore, we propose *within-corpus data augmentation* which created effective training data through segmenting the existing parallel data and then recombination. These algorithms are simple and effective because they are 1) *on-the-fly*, 2) *memory-efficient* and 3) *source-target aligned*. In the below sections, we demonstrate this augmentation idea in ASR and ASTT.

This chapter is organized as follows. In section 4.2, we present Aligned Data Augmentation (ADA) for ASR. Our ADA algorithm injects aligned perturbation at sampled locations of a training data instance. Next, section 4.3, we present STR which is the extension of ADA to ASTT. In section 4.4, we examine concatenation-based augmentation for both ASR and ASTT.

[1]For an example, SpecAugment is *on-the-fly* but not *source-target aligned* because it introduces random masking of negligible computational overhead, irrespective of target-side during training.

4.2 Aligned Data Augmentation (ADA) for ASR

To begin with, we present our ADA algorithm which is able to apply variations on both source audio and target text to generate new training examples for ASR in a computationally efficient manner.

The central component of ADA is an audio dictionary that allows us to apply our strategies to data augmentation on-the-fly, thus increasing variability in contrast to offline augmentation, and improving efficiency by avoiding the necessity to store and load augmented data. On the target text, we either use a masked language model to replace tokens with semantically close variations, or we replace tokens following a random token strategy. In the former case, the augmented text differs not too much from the original text, while in the latter case, the possibly ungrammatical examples force the model to put more focus on the audio input and less focus on the inherent language model. Both approaches successfully increase robustness, i.e. performance on previously unseen examples (Ng et al., 2020). On the source audio, sections are replaced by entries sampled from an audio dictionary. Our audio dictionary is extracted from the training corpus and its functionality is comparable to the aforementioned TTS+speech synthesizer method, Vocal Tract Length Perturbation (Jaitly and Hinton, 2013), or Stochastic Feature Mapping (Cui et al., 2015), however, it requires much less computing power and the audio representation replacements are from real human speech. The resulting audio sequence is thus a combination of the original audio representations and replacements, which is similar to data augmentation techniques for images such as CutMix (Yun et al., 2019).

There are other work which has shown the utility of alignment information for training of speech-to-text systems. Salesky et al. (2019) improved end-to-end speech translation by using phoneme-level alignments. However, they use this information to compress phoneme representations and they do not increase the amount of training data. Nguyen et al. (2020) proposed to create subsequences by truncating source audio and target transcriptions in an aligned manner. In contrast to this, we generate complete previously unseen examples by partial replacements of source audio and target text.

One crucial difference between ADA and other semi-supervised learning methods such as Kahn et al. (2020); Laptev et al. (2020); Rossenbach et al. (2020) is that ADA uses the original dataset, while the other three include additional in-domain unlabelled audio or text to generate augmented data pairs. Furthermore, the on-the-fly data augmentation done in ADA differentiates our technique from complex techniques that

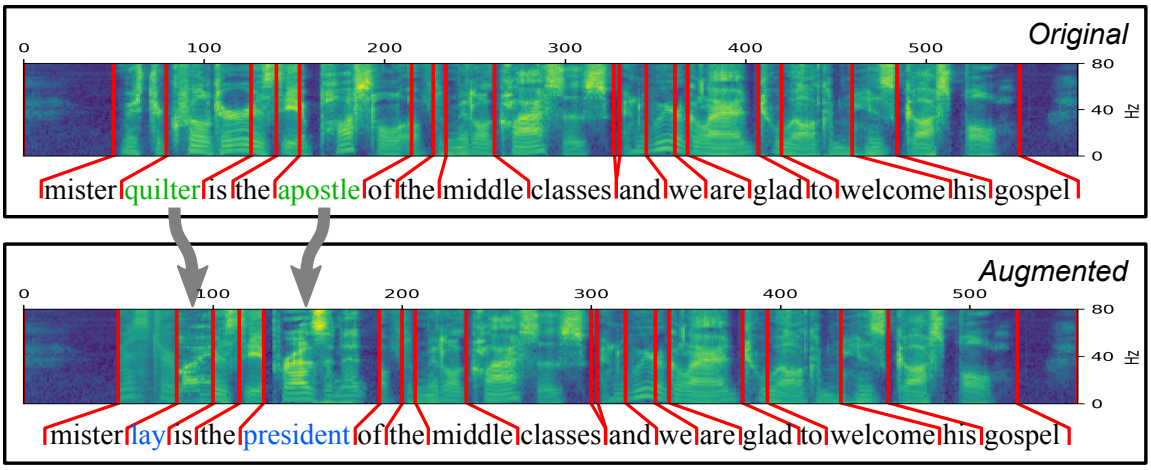

Figure 4.1: Example from the LibriSpeech dataset illustrating aligned data augmentation: In the original audio-text pair, certain tokens (green) are replaced following certain strategies (blue). An audio dictionary created on the training data is then queried to replace the aligned audio representations of the predicted tokens, resulting in an augmented audio-text pair.

require offline preprocessing, e.g. speed perturbations (Ko et al., 2015).

Our experiments on a Transformer speech-to-text architecture show that ADA can be applied on top of SpecAugment, and achieves about 9–23% and 4–15% relative improvements in WER over SpecAugment alone on LibriSpeech 100h and LibriSpeech 960h test datasets, respectively.

4.2.1 Method Description

Figure 4.1 illustrates the ADA process of creating an augmented data pair from an original data pair. Starting with the target side of the original pair, we randomly select tokens for replacement (printed in green). These tokens are then replaced on the augmented target side by candidate tokens (printed in blue) following two alternative strategies: (1) language model guided strategy which we call ADA-LM, and (2) random token strategy which we call ADA-RT. Finally, an audio dictionary is queried for audio representations corresponding to the replaced tokens, resulting in a new augmented data pair. In case the replacement token suggested by the language model has no corresponding entry in the audio dictionary which happens in less than 5% of the replacements, ADA-LM masks out the aligned audio representation. Data pairs augmented in that manner look very similar to the ones generated by SpecAugment, however, we inject additional aligned target side variations. The candidates suggested by the language model are in general semantically close, such as

the replacement of "apostle" with "president". By exchanging the names "quilter" and "lay", as suggested by the language model, ADA-LM receives out-of-domain knowledge, increasing model robustness. The second strategy, ADA-RT, samples a random token and then replaces the corresponding audio representation. This mostly results in ungrammatical sentences, however, it forces the model to adjust the influence of the inherent language model. For each token, our audio dictionary usually keeps multiple audio representations available that differ in tone, speed, or speaker. Such an audio dictionary introduces several aspects useful for ASR training. First, it leads to clean augmented data pairs containing real human speech. Then, it introduces variations of prosody and gender in the audio representations. Finally, it is computationally simple and can be applied on-the-fly.

Language Model Only In a separate experiment, we evaluate the impact of utilizing only a language model to generate new data pairs. In this scenario, we do not mask out the corresponding audio segments and only apply SpecAugment to the audio representations. This effectively creates variations in the transcriptions that do not perfectly match with the audio representations.

Audio Dictionary Only The audio dictionary can also be applied without text token replacement. Here, the source audio representation of a word that is not changed on the target side is replaced by a sample of the same token from the dictionary. Since ASR is a many-to-one problem, audio-text pairs modified in this manner increase the recognition performance. This happens for the word "mister" in Figure 4.1 where the audio representation in the augmented pair is replaced by an entry from the dictionary. In our experiments in Section 4.2.3, we use this source-side-only replacement to evaluate the contribution of the audio dictionary. A source-side-only replacement can also occur rarely in the aligned case if the language model predicts the same token as the original one, e.g. when function words or parts of common named entities are to be replaced by the language model.

4.2.2 Experimental Setup

Architecture: ASR We used the implementation of the Speech-to-Text Transformer by Wang et al. (2020b) from the FAIRSEQ website and added a Connectionist Temporal Classification (CTC) component and the data augmentation code. The model has two convolution layers of stride 2 to down-sample the audio representations by a factor of 4 before the self-attention blocks. There are 12 self-attention layers in

#	model	type	dev-clean	dev-other	test-clean	test-other	time
1	SpecAugment (baseline)	–	10.51 ± 0.04	22.92 ± 0.17	11.50 ± 0.10	23.50 ± 0.11	1.0
2	LanguageModel	target only	[1]9.98 ± 0.13	[1]22.31 ± 0.19	[1]10.88 ± 0.15	[1]22.80 ± 0.15	2.0
3	AudioDict	source only	[1]9.90 ± 0.05	[1,2]21.50 ± 0.10	[1]10.70 ± 0.04	[1,2]22.12 ± 0.09	1.2
4	ADA-LM	aligned	[1,2,3]9.26 ± 0.07	[1,2]21.30 ± 0.11	[1,2,3]10.12 ± 0.06	[1,2,3]21.41 ± 0.08	2.4
5	ADA-RT	aligned	[1,2,3,4]8.54 ± 0.03	[1,2]21.11 ± 0.09	[1,2,3,4]8.80 ± 0.09	[1,2,3]21.32 ± 0.07	1.3

Table 4.1: Average WER on the LibriSpeech 100h dataset over 3 runs with standard deviations (±). SpecAugment with RoBERTa on the target side only (LanguageModel) and with the audio dictionary on the source audio only (AudioDict) already gives consistent relative improvements of 2.7% to 5.4% and 5.8% to 7.0% respectively across all datasets. The language model guided ADA method (ADA-LM) combines RoBERTa and the audio dictionary in an aligned manner and delivers relative improvements of 11.9% to 12.0% on the `clean` datasets, and of 7.1% to 8.9% on the `other` datasets over the baseline. The random token strategy for ADA (ADA-RT) improves even more and gives relative improvements of 18.7% to 23.5% on the `clean` datasets, and of 7.9% to 9.3% on the `other` datasets. Prepended numbers denote statistically significant difference to the model numbered in column "#" at the 1% level.

the encoder which is followed by 6 layers in the decoder. The embedding dimension is 512, and we set the dimension of feed forward networks to 2,048. In order to achieve faster and more stable convergence, we added another output layer after the encoder so that its parameters are shared between the Cross-Entropy loss L_{XENT} with label smoothing of 0.1 and CTC loss L_{CTC} (Watanabe et al., 2017). The final loss is a linear combination of both loss components, $L = \alpha L_{XENT} + (1 - \alpha) L_{CTC}$, where we followed (Karita et al., 2019) and set α to 0.7.

Architecture: ADA Components To obtain token-level alignment information between source audio representation and target text, we used the Montreal Forced Aligner (McAuliffe et al., 2017). The alignment information is initially used to construct the audio dictionary as follows. For each training example in the corpus, the construction process iterates over the tokens and adds the aligned audio representations to a key-value store where the key is the token and the value is a pool of aligned audio representations. This audio dictionary can be further enriched with pre-calculated speed and frequency perturbations, effectively integrating offline methods in online augmentation.

To get the target token predictions during training under the ADA-LM configuration, we queried `roberta.base`, a pre-trained RoBERTa (Liu et al., 2019b) language model downloaded from the FAIRSEQ (Ott et al., 2019) examples repository.[2] We chose the 125M parameter model to trade-off speed and memory consumption. Under

[2] `www.github.com/pytorch/fairseq/tree/master/examples/roberta` (accessed 03/25/2021)

the ADA-RT configuration, tokens are directly sampled from the keys in the audio dictionary in a random manner.

Data Description We used LibriSpeech[3] (Panayotov et al., 2015) in our experiments since there exist many baselines in well-defined small to medium and large scale scenarios. For our small to medium scale experiments, we used the split `train-clean-100` and extracted subword units of size 5,000 using SentencePiece (Kudo and Richardson, 2018). For the large scale experiments, we combined splits `train-clean-100`, `train-clean-360` and `train-other-500` to form the 960h data and extracted 10,000 subword units. We used log Mel-filter banks of 80 dimensions as our acoustic features. We filtered data instances which have more than 3,000 frames or are longer than 80 subword units in the training sets. This results in the removal of 50 samples and 78 samples in `train-clean-100` and `train-960`, respectively. In both scenarios, the same audio dictionary extracted from `train-clean-100` is used.

Training and Inference In training, we used Adam optimizer with a peak learning rate of 2e-3 and warmup of 12,500 steps. We accumulated the gradients for 8 mini-batches before updating with at most 40,000 frames per mini-batch. The learning rate was adjusted according to the inverse square root learning rate schedule.

We also used a static mixture schedule for the type of replacement such that a significant amount of transcriptions remains unchanged. The static mixture schedule of ADA was tuned on the respective LibriSpeech dev sets. Details of our static mixture schedule are listed in Table 4.2. In each mini-batch, there is a fraction of sentences augmented using our aligned method and fraction of sentences augmented using the AudioDict only[4]. The "tokens" columns indicate the amount of token-level replacements applied to these sentences. The final column lists the total of augmented sentences.

In all configurations, we applied SpecAugment at the end with a frequency mask parameter of 30 and a time mask parameter of 40, both with 2 masks along their respective dimension. We used beam search with a beam size of 5 in decoding.

[3] `www.openslr.org/12` (accessed 03/25/2021)

[4] In our side experiments, we find that either aligned strategy alone, irrespective of using LM or random replacement strategies on the target side, already perform better than both AudioDict and LanguageModel. On top of the aligned strategy, we find that applying "AudioDict only" on certain portion of the unaugmented sentences can further improve the performance. We thus put "AudioDict only" together with aligned-LM or aligned-RT to form ADA-LM or ADA-RT.

data set	Aligned Augm. sentences	Aligned Augm. tokens	AudioDict sentences	AudioDict tokens	ADA sentences
100h	50%	20%	15%	20%	65%
960h	30%	20%	21%	15%	51%

Table 4.2: Details of the static mixture schedule of ADA we used in the experiments on LibriSpeech 100h and 960h datasets.

#	model	dev-clean	dev-other	test-clean	test-other	time
1	SpecAugment (baseline)	3.94 ± 0.10	8.47 ± 0.13	4.45 ± 0.10	8.29 ± 0.06	1.0
2	ADA-LM	3.78 ± 0.02	8.25 ± 0.00	[1]4.22 ± 0.07	[1]8.07 ± 0.01	1.8
3	ADA-RT	[1,2]3.50 ± 0.03	[1]8.14 ± 0.05	[1,2]3.75 ± 0.01	[1]7.97 ± 0.04	1.3

Table 4.3: Average WER on the LibriSpeech 960h dataset over two runs with standard deviations (±). Our language model guided aligned ADA method (ADA-LM) is about twice as slow as SpecAugment. ADA-LM gains relative improvements of 4.1% to 5.2% on the `clean` datasets, and of 2.6% to 2.7% on the `other` datasets. ADA-RT gains relative improvements of 11.2% to 15.7% on the `clean` datasets, and of 3.9% on the `other` datasets. Prepended numbers denote statistically significant difference to the model numbered in column "#" at the 5% level determined following approximate randomization test in Riezler and Maxwell (2005).

4.2.3 Results and Analysis

Results on train-clean-100 On this dataset, our ASR models were trained for 200 epochs and checkpoints are averaged over the last 75 epochs for each setting. We report mean and standard deviation of micro word error rate (WER) calculated over 3 runs on the standard data splits, i.e. `dev-clean`, `dev-other`, `test-clean` and `test-other`. We also performed an ablation study to investigate the effect of each proposed augmentation. Table 4.1 summarizes the results.

Our main baseline uses only SpecAugment for data augmentation. Other SpecAugment baselines that were trained on the same data split report comparable results, e.g. baselines in (Lüscher et al., 2019) and (Kahn et al., 2020) are worse than ours while the baselines in (Laptev et al., 2020) are very close to ours.

Both non-aligned source side only (AudioDict) and non-aligned target side only (LanguageModel) augmentations show consistent reductions in WER of about 0.5–0.8 points on the `clean` datasets. On the `other` datasets, however, the source side only AudioDict augmentation gives improvements of about 1.4 points in WER while the target side only LanguageModel augmentation gives lower improvements of 0.6–0.7. In both non-aligned experiments, we applied augmentations to 50% of the sentences

and 20% of their tokens.

In comparison to the plain SpecAugment baseline, the language model guided ADA-LM reduces WER by 1.25 and 1.38 points on the `clean` datasets, and by 1.62 and 2.09 points on the `other` datasets.

Switching to the random token replacement strategy of ADA-RT gives even larger reductions in WER by 1.97 and 2.70 points on the `clean` datasets (relative improvement 18.7% to 23.5%), and by 1.81 and 2.18 points on the `other` datasets (relative improvement 7.9% to 9.3%).

This result is surprising at first as the augmented examples mostly represent ungrammatical sentences. On second thought, such examples effectively force the model to rely less on the inherent language model of the ASR system, and put more weight on the plain audio recognition component as conditioning factor, resulting in increased model performance on unseen examples. This result is confirmed by a side experiment where randomly replacing tokens on the target side only did not in general improve results over the baseline.

Results on train-960 On this large dataset, we train our ASR model for 150 epochs and average checkpoints of the last 20 epochs for evaluation. Mean and standard deviation of WER are calculated over 2 runs. Table 4.3 summarizes our results. In comparison to the SpecAugment baseline, we observe consistent improvements for the ADA-based methods on all datasets.

For the language model guided ADA-LM method, WER is reduced by 0.22 points on the `other` datasets corresponding to a relative improvement of 2.6% and 2.7%, and WER is reduced by 0.26 points and by 0.23 points on the `clean` datasets corresponding to a relative improvement of 4.1% and 5.2% on `dev` and `test`, respectively.

Switching to the random token strategy implemented in ADA-RT gives further improvements. WER is reduced by 0.33 points and by 0.32 points on the `other` datasets which corresponds to a relative improvement of 3.9%, and WER is reduced by 0.44 points and by 0.70 points on the `clean` datasets which corresponds to a relative improvement of 11.2% and 15.7% on `dev` and `test`, respectively.

For the large 960h dataset, comparable numbers to our baselines can be found in Hu et al. (2021). They use a transformer architecture implemented in ESPnet (Watanabe et al., 2018) and report SpecAugment baseline WER scores that are very close to ours.

Training Speed and Model Complementarity We report average per-instance training time normalized by the baseline model's training time in the last column of Tables 4.1 and 4.3. Adding the audio dictionary for augmentation increases training time by a factor of 1.2 compared to SpecAugment alone. Notably, the largest increase in computational effort is introduced by querying the language model. The target side only LanguageModel experiment increases training time by a factor of 2.0, and our language model guided ADA-LM implementation increases training time by a factor of 2.4 for 100h, and 1.8 for 960h. The random token strategy of ADA-RT is thus significantly faster than its language model guided counterpart, resulting in a training time increase factor of only 1.3. This shows that both ADA variants are well-suited for on-the-fly training, where the latter, ADA-RT, is remarkably efficient.

model	`test-clean`	`test-other`
w/o Augmentation	13.74	31.91
SpecAugment (SA) only	11.50 (−2.24)	23.50 (−8.41)
ADA-RT w/o SA	10.95 (−2.79)	30.03 (−1.88)
ADA-RT with SA	8.80 (−4.94)	21.32 (−10.59)

Table 4.4: Numbers in "()" are the differences in WER to the topmost model which was trained w/o any augmentation method. They illustrate that ADA-RT is complementary to SpecAugment.

In a side experiment, we evaluated the complementarity between ADA and SpecAugment and report results in Table 4.4. The numbers in parenthesis nicely illustrate that the contributions of ADA-RT w/o SA and SpecAugment directly add up.

Significance Testing Significance testing across different runs is not straightforward. Pairwise tests (Gillick and Cox, 1989) across different runs of models are problematic as the rejection of the null hypothesis might be based on different initializations and not only on architectural model differences. Bootstrap tests (Bisani and Ney, 2004) might be problematic because of the assumption that the test set is representative of the population distribution, something which is not satisfied if train and test data are from different domains. A non-parametric significance test that only relies on the strategy of stratified shuffling is the permutation test, a.k.a. (approximate) randomization test, dating back to Fisher (1935). For large samples, this test has been shown to be as powerful as related parametric tests (Hoeffding, 1952), and it produces fewer Type-I and Type-II errors than the bootstrap (Noreen, 1989).

Thus, to compare our systems we conduct significance tests as follows. For each

transcription reference, we compress the score results of each system's runs to a single average, effectively reducing score variance per example. This is valid for WER values case because the reference length is constant. We then determine p-values to decide whether our proposed systems are significantly different to their SpecAugment baseline models and we also compare systems against each other. For Librispeech 100h, we conduct in total 10 pairwise comparisons of models, thus we apply a Bonferroni correction to the 1% level and set α to 0.001 (0.01/10). For Librispeech 960h, we conduct 3 pairwise comparisons, thus we set α to 0.0166 (0.05/3) for the 5% level. Statistical significance of model differences is then determined using an approximate randomization test: we set the number of randomly shuffled runs to 1000, and in each run, we exchange the transcriptions' scores of two models with a probability of 0.5 for each example and calculate the test statistic.

On Librispeech 100h, both ADA models as well as the LanguageModel and AudioDict models are significantly different to their SpecAugment baseline models on all four datasets. On the `clean` datasets, the ADA models are also significantly different to all models that don't use alignment information, with the best performing ADA-RT model being significantly different to ADA-LM. On Librispeech 960h, significant differences of ADA-LM to their SpecAugment baseline models were identified on `test-clean` and `test-other`. The ADA-RT method, however, is significantly different to its SpecAugment baseline models on all datasets, and turns out to be also significantly different to ADA-LM on the `clean` datasets.

4.2.4 Summary

We propose a data augmentation method that makes use of alignment information to create effective training examples. An audio dictionary that is extracted from the training set can be queried with low computational overhead and is used to construct previously unseen utterances and speaker combinations. By combining textual token replacements with the audio dictionary in an aligned manner, our model is able to construct unseen examples on-the-fly with acceptable impact on training speed if we use predictions from a language model. In case we employ a random strategy for token replacements, we see even larger improvements with very little impact on training speed. Our aligned methods show significant improvements in WER over methods that don't use alignment information on small to medium and large LibriSpeech datasets.

ADA algorithms posses the characteristics of the proposed within-corpus data augmentation. The alignment information serves as the segmentation step whereas

the replacement methods guided by random sampling or by masked language model serve as the recombination step. As discussed, ADA injects perturbation in an aligned manner, and it reassembles the data during training with rather negligible computational overhead, especially ADA-RT. Hence, it is both *on-the-fly* and *source-target aligned*. Though not mentioned explicitly, the stored audio representations in the audio dictionary are from the original training data. Hence, it is sufficient to store the references rather than the copies, making ADA *memory-efficient*. In the next section, we present STR which is an extension of ADA for speech-to-text translation.

4.3 Sample, Translate and Recombine for ASTT

End-to-end ASTT relies on data that consist only of speech inputs and corresponding translations. Such data are notoriously limited. Data augmentation approaches attempt to compensate the scarcity of such data by generating synthetic data by translating transcripts into foreign languages or by back-translating target-language data via TTS (Pino et al., 2019; Jia et al., 2019), or by performing knowledge distillation (KD) using a translation system trained on gold standard transcripts and reference translations (Inaguma et al., 2021). In this paper, we present a simple, resource conserving approach that does not require TTS and yields improvements complementary to KD.

For training cascade systems, monolingual data for ASR and textual translation data for MT can be used, reducing the problem of scarcity. Cascaded systems, however, suffer from error propagation, which has been addressed by using more complex intermediate representations such as n-best MT outputs or lattices (Bertoldi and Federico, 2005; Beck et al., 2019, *inter alia*) or by modifying training data to incorporate errors from ASR and MT (Ruiz et al., 2015; Lam et al., 2021b). End-to-end systems are unaffected by this kind of error propagation and are able to surpass cascaded systems if trained on sufficient amounts of data (Sperber and Paulik, 2020a).

Our approach transfers an idea on aligned data augmentation that has been presented for ASR (Lam et al., 2021a) to aligned data augmentation in ASTT. Similar to aligned data augmentation for ASR, we utilize forced alignment information to create unseen training pairs in a structured manner. Unlike ASR which alignment is rather monotonic, alignment in translation is more complicated. This presents challenges to the injection of aligned perturbation in ASTT. In order to tackle it, we leverage the linguistic property of the source transcription, e.g. SVO scheme in English, to identify exchangeable textual segments.

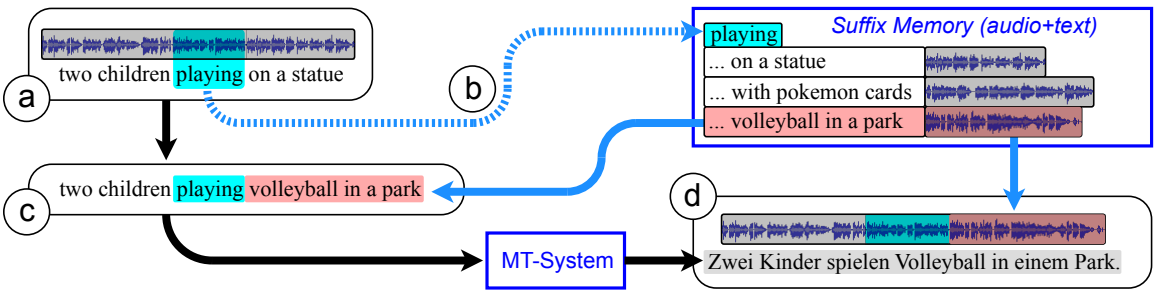

Figure 4.2: An illustration of STR. (a) Select a pivoting token, e.g., "playing". (b) Retrieve suitable text-audio entries from the suffix memory to sample a replacement. (c) Compile a new transcription containing prefix, pivoting token, and replacement suffix. (d) Recombine a new training example by translating the new transcription and concatenating the audio sections.

Our augmentation procedure consists of the following steps: (1) *Sampling* of a replacement suffix of a transcription and its aligned speech representations, guided by linguistic constraints. (2) *Translation* of the transcription containing the new suffix. (3) *Recombination* of audio data containing the new suffix and the generated translation to distill a new training pair. We thus use the acronym STR (Sample, Translate, Recombine) to refer to our method.

In comparison to Pino et al. (2019) and Jia et al. (2019) who used TTS to generate synthetic speech, we create new examples by recombining real human speech. This reduces the problem of overfitting to synthetic data as for example in SkinAugment (McCarthy et al., 2020) where synthetic audio is generated by auto-encoding speaker conversions.

The basic idea of our method is comparable to data augmentation techniques for images such as CutMix (Yun et al., 2019) where images are blended together to form new data examples. However, CutMix selects images randomly, while we recombine phrases in a structured manner.

Our experimental evaluation is conducted for five language pairs on the CoVoST 2 dataset (Wang et al., 2021a) and for two language pairs on the Europarl-ST (Iranzo-Sánchez et al., 2020) dataset. We find considerable improvements for all language pairs on all datasets for our approach on top of KD. Our approach can be seen as an enhancement of Inaguma et al. (2021)'s KD approach since it requires roughly the same computational resources and consistently improves their gains.

4.3.1 Method Description

Our method exploits audio-transcription alignment information to generate previously unseen data pairs for end-to-end ASTT training. By applying a Part-of-Speech (POS) Tagger on a sentence, we identify potential "pivoting tokens" where the token's prefix or suffix, i.e., the preceding or succeeding tokens, can be exchanged between other sentences containing the same token of the same syntactic function. We then sample possible suffixes for that token from a suffix memory containing text and audio suffixes, and concatenate the prefix, verb, and suffix to generate a new transcription. Then, an MT system translates the new transcription, picking up on the idea of knowledge distillation in ASTT (Inaguma et al., 2021). The MT system is trained or fine-tuned on the transcription-translation pairs. Finally, using the previously sampled audio suffix, we concatenate prefix, verb, and suffix audio together with the MT generated translation to recombine a new audio-translation pair for end-to-end ASTT training.

Our augmentation method implements linguistic constraints by making use of the transcription's syntactic structure in combination with alignment information. Effectively, we exploit the strict SVO-scheme of English sentences as we select the verb as our pivoting token. Our method is applicable to other languages, however, it will require more effort to identify exchangeable syntactic structures.

Figure 4.2 illustrates our approach. We start by identifying the pivoting token in a transcription we want to augment, here "playing" in the sentence "two children are *playing* on a statue". Then, we extract the list of possible suffixes following "playing" from the suffix memory and sample a single audio-text suffix, here "volleyball in a park". Together with the original prefix and pivoting token, the textual part of the sampled suffix builds a new augmented transcription. Similarly, together with the audio prefix and token, the audio part of the suffix builds a new augmented audio example. The augmented transcription is then translated by an MT model. The new audio example (i.e., the representation of "two children playing volleyball in a park") and the translation (i.e., the text "Zwei Kinder spielen Volleyball in einem Park") are then recombined to form a new audio-translation pair.

4.3.2 Experimental Setup

Data Description We evaluated our method on two common ASTT datasets, CoVoST 2 (Wang et al., 2021a) and Europarl-ST (Iranzo-Sánchez et al., 2020). CoVoST 2 is a large scale dataset of 430h English audio and 288k sentences for each language in the training set. The training set contains repetitions of the same sen-

tence spoken by different speakers. We used the original data splits generated by the `get_covost_splits.py` script[5] on five languages pairs, namely English-German (En-De), English-Catalan (En-Ca), English-Turkish (En-Tr), English-Welsh (En-Cy) and English-Slovenian (En-Sl), resulting in about 15.5k sentences for each dev and test dataset. Europarl-ST, in contrast, is a small ASTT dataset. It contains debates held in the European Parliament and their translations, thus representing a realistic ASTT scenario imposing very different challenges than the CoVoST 2 dataset. We conducted experiments on the English-German (En-De) and English-French (En-Fr) language pairs. The En-De data contains 89h of audio and 35.5k sentences. The En-Fr data contains 87h of audio and 34.5k sentences. Since Europarl-ST is too small for MT training from scratch, we used 1.6M En-De sentence pairs from Wikipedia following Schwenk et al. (2021) and 3.2M En-Fr sentence pairs from the Common Crawl corpus[6] as additional data.

Data Preprocessing For speech data preprocessing, we extracted log Mel-filter banks of 80 dimensions computed every 10ms with a 25ms window. We normalized the speech features per channel using mean and variance per instance. For all textual data, punctuation was normalized using SACREMOSES.[7] The transcriptions were lowercased with punctuation removed.

For the speech-to-text tasks on CoVoST 2, we employed character-level models due to the availability of pre-trained high quality ASR models. For the speech-to-text tasks on Europarl-ST, we learnt 5,000 subword units for each target language. For the machine translation tasks in knowledge distillation, we learnt a joint subword vocabulary on both source and target for each language pair of size 5,000 for CoVoST 2 and size 40,000 for Europarl-ST including the additional training data. Subword unit creation was always conducted with SENTENCEPIECE (Kudo and Richardson, 2018).

The Montreal Forced Aligner (McAuliffe et al., 2017) was applied without any fine-tuning to extract the acoustic alignments. Thus, the obtained alignments can be error-prone. In very rare cases, the acoustic aligner does not return an alignment at all and we have to discard these examples. In some cases, the obtained alignments by the acoustic aligner are of low quality, i.e., contain alignments to unknown tokens. In such cases, if the number of tokens of the output transcriptions of the acoustic aligner matches the number of tokens in the input transcriptions, we can still use this alignment for data augmentation as alignments in ASR are always strictly parallel. Thus, if we

[5]github.com/facebookresearch/covost, accessed 3/11/2022
[6]www.statmt.org/wmt13/..., accessed 3/11/2022
[7]github.com/alvations/sacremoses, accessed 3/11/2022

Data	Baseline	KD	STR
CoVoST 2	288k	+288k	+255k
Europarl-ST (En-De)	3.25k	+3.25k	+2.78k
Europarl-ST (En-Fr)	3.17k	+3.17k	+2.71k

Table 4.5: Number of examples per configuration.

cannot retrieve suitable alignments, we discard the example. This procedure reduced the amount of augmented data: we discarded approximately 12% of the examples for CoVoST 2, and about 15% of the examples for Europarl-ST.

To extract POS-tags, we used the SPACY[8] toolkit. We selected the verb as our pivoting token and generated the suffix memory as follows: for each verb, we generated a list of audio-text suffix pairs and stored the data in a key-value table. The audio entries contain only references to the original audio segments and our implementation is thus very memory efficient. We only utilized basic off-the-shelf components that are widely available and our suffix memory has a negligible memory footprint.

Table 4.5 summarizes the number of training examples in each experiment.

Model configuration All our implementations were based on FAIRSEQ (Ott et al., 2019; Wang et al., 2020b).[9] In all speech-to-text tasks, we used the Transformer architecture (Vaswani et al., 2017) labelled as "s2t_transformer_s" in FAIRSEQ, which consists of convolutional layers for downsampling the input sequence with a factor of 4 before the self-attention layers. The encoder has 12 layers while the decoder has 6 layers with the dimensions of the self-attention layers set to 256 and the feed-forward network dimension set to 2048.

For the CoVoST 2 MT tasks, we used a smaller Transformer model of 3 layers for both encoder and decoder. The encoder-decoder embeddings and the output layer were shared. For the Europarl-ST MT tasks, we used the Transformer BASE configuration as described in Vaswani et al. (2017).

Training In the CoVoST 2 ASTT experiments, we used the character-level ASR model downloaded from the FAIRSEQ GitHub webpage[10] to initialize the encoder of the ASTT systems. Each ASTT system was then trained for another 50,000 steps. For Europarl-ST, we trained a subword unit ASR system on the English audio-transcription

[8] github.com/explosion/spaCy, accessed 3/11/2022
[9] github.com/statnlp/str/, accessed 3/10/2022
[10] github.com/pytorch/fairseq/..., accessed 3/11/2022

pairs of the En-De data for 25,000 steps. The resulting ASR system was used to initialize both En-De and En-Fr ASTT systems which were trained for another 20,000 steps. Throughout all speech-to-text experiments, we applied gradient accumulation resulting in an effective mini-batch size of 160k frames. We used Adam optimizer with an inverse square root learning rate schedule. We used 10k steps for warmup and a peak learning rate of 2e-3. SpecAugment was applied with a frequency mask parameter of 27 and a time mask parameter of 100, both with 1 mask along their respective dimension. We performed validation and checkpoint saving after every 1,000 updates.

In case of the CoVoST 2 MT task, the Transformer model was pre-trained on in-domain data with 30,000 steps and an effective mini-batch size of 16,000 tokens. For the Europarl-ST dataset, the MT models were pre-trained on a combination of Europarl-ST and the additional training data. The Adam optimizer was used with an inverse square root learning rate schedule again, now with 4k steps for warmup and a peak learning rate of 5e-4. After pre-training, we finetuned the model on the in-domain data with SGD and a constant learning rate of 5e-5.

Inference In the speech-to-text experiments, we averaged the 10 best checkpoints based on the validation loss. For the MT tasks, we averaged the 5 best checkpoints. Throughout all ASTT experiments and MT tasks, we applied beam search with a beam size of 5.

4.3.3 Results and Analysis

Our experiments are focused on the improvements of our proposed method over KD alone on both CoVoST 2 and Europarl-ST datasets. We evaluated the translation results with both BLEU[11] (Papineni et al., 2002) and chrF2[12] (Popović, 2016) using the implementation of SACREBLEU (Post, 2018). Each experiment was repeated 3 times and we report mean and standard deviation.

In addition, we provide discussions of the following: 1) the connection between STR- and MT-performance, 2) how the amount of STR data affects the final performance, and 3) an error analysis with examples and limitations of STR.

[11] nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.0.0
[12] nrefs:1|case:mixed|eff:yes|nc:6|nw:0|space:no|version:2.0.0

model	En-De	En-Ca	En-Tr	En-Cy	En-Sl
Wang et al. (2021a) Bi-AST	16.3	21.8	10.0	23.9	16.0
Baseline	17.22 ± 0.09	23.15 ± 0.10	10.31 ± 0.04	25.46 ± 0.08	15.64 ± 0.04
KD	18.26 ± 0.05	24.48 ± 0.16	11.10 ± 0.03	26.87 ± 0.16	17.21 ± 0.02
STR	18.77 ± 0.04	24.83 ± 0.12	11.62 ± 0.04	27.28 ± 0.11	17.54 ± 0.14
KD+STR	19.06 ± 0.02	25.33 ± 0.06	11.83 ± 0.01	27.73 ± 0.09	17.83 ± 0.09

Table 4.6: Average BLEU on the CoVoST 2 dataset over 3 runs with standard deviations (±). Models KD and KD+STR are significantly different for all language pairs with $p < 0.0002$ using a paired randomization test.

model	En-De	En-Ca	En-Tr	En-Cy	En-Sl
Baseline	42.80 ± 0.08	46.63 ± 0.09	36.77 ± 0.09	49.13 ± 0.05	39.83 ± 0.05
KD	44.13 ± 0.05	48.17 ± 0.12	38.53 ± 0.05	50.67 ± 0.05	41.73 ± 0.05
STR	44.43 ± 0.05	48.60 ± 0.08	39.30 ± 0.08	51.03 ± 0.05	42.17 ± 0.05
KD+STR	45.13 ± 0.05	49.10 ± 0.08	39.70 ± 0.08	51.50 ± 0.00	42.60 ± 0.08

Table 4.7: Average chrF2 on the CoVoST 2 dataset over 3 runs with standard deviations (±). Models KD and KD+STR are significantly different for all language pairs with $p < 0.0002$ using a paired randomization test.

Results on CoVoST 2

Table 4.6 lists BLEU scores on the five considered CoVoST 2 language pairs. Our baseline model is the ASTT system finetuned on the in-domain audio-translation pairs only. Its performance over the selected language pairs is quite diverse with BLEU scores ranging from 10.31 (En-Tr) to 25.46 (En-Cy). Our baseline models are comparable to and often better in terms of BLEU than the bilingual ASTT (Bi-AST) models by Wang et al. (2021a). Training together with translations generated by KD improves the baseline model by a substantial margin of 0.8 to 1.6 BLEU points. Our proposed STR method alone slightly surpasses the KD performance and brings further improvements when the augmented data is combined (KD+STR) with BLEU score increases ranging from 0.62 for En-Sl to 0.86 for En-Cy. In total, we observe BLEU score improvements of 1.5 to 2.3 for KD+STR.

Since BLEU scores are often biased towards short translations, we additionally calculate chrF2 scores. This issue is especially problematic on the CoVoST 2 datasets because of its large number of very short sentences. Our chrF2 results, in Table 4.7, averaged over three runs confirm the improvements we observed throughout our experiments in terms of BLEU. When we look at chrF2, the better performing KD+STR models are always significantly different to the KD models.

Results on Europarl-ST

Table 4.8 lists the BLEU score results of Europarl-ST En-De and En-Fr. Similar to the results on CoVoST 2, the KD models bring substantial improvements over the baseline systems. The gains are 6.02 points for En-De and 6.27 points for En-Fr. We attribute this to the strong machine translation model that is trained on large amounts of additional training data (see the next section for more details). Our proposed STR method alone does not reach the KD performance but the combination KD+STR still delivers remarkable gains over KD, i.e., 1.13 points on En-De and 0.45 points on En-Fr, showing the complementarity of KD and STR. We also evaluate our models using chrF2. The findings are consistent to those evaluated in BLEU– STR performs worse than KD but the combination KD+STR still outperforms KD with extra gain of 0.94 points for En-De and 0.4 points for En-Fr. The number are listed in Table 4.9.

model	En-De	En-Fr
Baseline	14.47 ± 0.16	22.52 ± 0.07
KD	20.49 ± 0.07	28.79 ± 0.14
STR	19.80 ± 0.14	28.01 ± 0.17
KD+STR	21.62 ± 0.12	29.28 ± 0.10

Table 4.8: Average BLEU on the Europarl-ST dataset over 3 runs with standard deviations (±). Models KD and KD+STR are significantly different for En-De with $p < 0.00025$. For En-Fr, we only found two runs to be significantly different with $p < 0.05$.

model	En-De	En-Fr
Baseline	44.90 ± 0.22	48.60 ± 0.14
KD	51.43 ± 0.05	54.97 ± 0.05
STR	50.6 ± 0.0	54.1 ± 0.22
KD+STR	52.37 ± 0.09	55.37 ± 0.09

Table 4.9: Average chrF2 on En-De and En-Fr of Europarl-ST dataset over 3 runs with standard deviations (±). Models KD and KD+STR are significantly different for En-De with $p < 0.0002$ using a paired randomization test. For En-Fr, the models are significantly different with $p < 0.025$.

Connection to MT-Performance

To evaluate the dependency of STR on the MT-performance, we calculate BLEU scores for the MT-systems we use for CoVoST 2 and Europarl-ST data augmentation with

STR and compare them in a cross-lingual manner. We see a noticeable correlation of MT-performance and STR-improvement.

On CoVoST 2, the highest improvement for STR is observed on the En-Cy language pair, which is also the best performing MT-model. The En-Ca language pair's MT-model also performs very well and shows the second highest gain for STR together with En-Sl. See Table 4.10 for more details.

On Europarl-ST, we observe a different behavior. While the MT-model for En-Fr is clearly better than the one for En-De, the gains are larger in the latter case. This might be due to the fact that the En-Fr ST-model already has a relatively high performance after training on KD alone (see Table 4.8). We also hypothesize that adding our STR method to KD is more useful if the sentence structure of source and target languages is very different. In case the alignments between source and target language are relatively parallel, KD already generates very useful examples and our approach can only introduce limited new information on top of that, e.g., by adding speaker variations. See Table 4.11 for the exact BLEU scores and improvements.

model	En-De	En-Ca	En-Tr	En-Cy	En-Sl
MT	30.05	39.66	21.28	43.57	30.32
STR-Δ	+1.84	+2.18	+1.51	+2.27	+2.19

Table 4.10: Machine translation performance measured in BLEU on the CoVoST 2 test set. The second row (STR-Δ) reports the BLEU improvements of KD+STR in comparison to the baseline.

model	En-De	En-Fr
MT	32.16	40.11
STR-Δ	+7.15	+6.76

Table 4.11: Machine translation performance measured in BLEU on the Europarl-ST test set. The second row (STR-Δ) reports BLEU improvements of KD+STR in comparison to the baseline.

Dependence on Amount of STR Data

We conduct an additional experiment on CoVoST 2 to evaluate the dependence of our STR method on the amount of generated training data. In Figure 4.3 we report the test performance on 5 language pairs of a single run (seed=0) after training on 1/3, 2/3, or all STR generated data. For some language pairs, we already observe

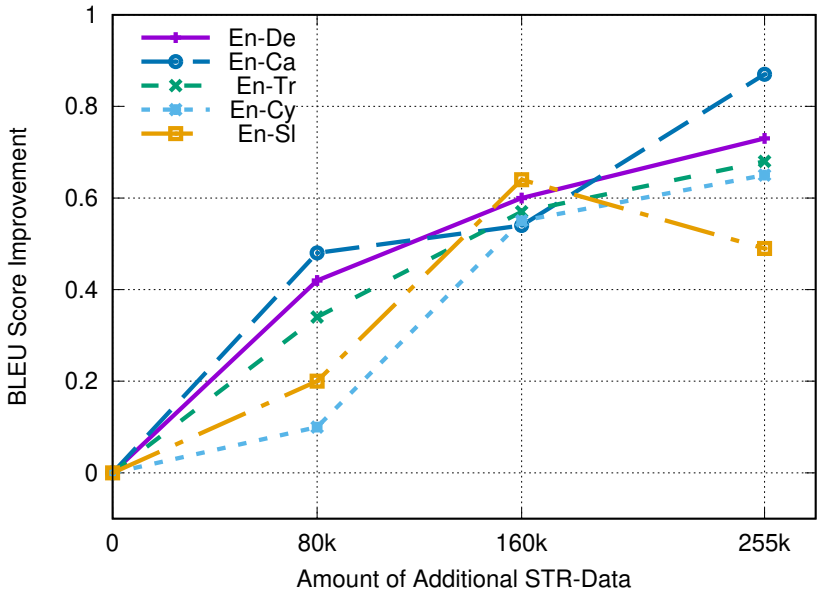

Figure 4.3: BLEU improvements for different amounts of STR augmented data on CoVoST 2 on a single run (seed=0) for 5 language pairs. We evaluate the addition of 0, 80k, 160k, and 255k STR-generated data points to the baseline KD data.

large gains after using 1/3 or 2/3 of the total STR data. Most language pairs will further benefit from more additional data, while one language pair (En-Sl) seems to degrade when moving from 2/3 to all training data on this single run. Summarizing, we observe a trend on all but one language pair that more augmented data improves performance.

Examples and Error Analysis

We also take a look at the quality of our STR-augmented data and list examples in Table 4.12 and Table 4.13 for CoVoST 2 and Europarl-ST, respectively. Rows "src-A" and "src-B" contain the unmodified transcriptions from CoVoST 2 with our pivoting token underlined and segments we recombine in *italics*. The row "augm." shows the STR-augmented example, the row "transl." contains the MT-generated translation. The presented examples are the first 5 data examples taken directly from our augmented data set and are *not* cherry-picked.

Of the first five augmented examples from CoVoST 2 listed in Table 4.12, examples 1, 3, and 5 contain grammatically correct augmented source data (row "augm.") and the latter two are also semantically correct. Example 2 contains a grammatically wrong segment due to the problematic transcription of "src-B": here, the example is already an ungrammatical sentence and this transfers to our augmented example.

Example 4 is also grammatically wrong. In this example, our augmentation method mixes the different senses of the word "directed" and produces a semantically incorrect result. This could be fixed by integrating more context, e.g., "directed through" can be used to disambiguate the different word senses of "directed".

Of the first five augmented examples from Europarl-ST in Table 4.13, examples 1, 3, and 5 are actually grammatically correct. Example 2 is grammatically wrong as our STR method does not respect the different grammatical forms of "pass" in "will pass" and "to pass", mixing up the two objects. Example 4 is also grammatically wrong, and it is again the wrong treatment of different grammatical forms of "do" in "do work" and "to do". These problems could be addressed by putting more effort into the suffix memory construction, e.g., by using n-grams as keys. Examples 3 and 5 demonstrate a property of Europarl-ST that partly explains the lower performance gain we observe for our STR-method here: there are many repetitive formalized sentences, and in these examples our augmentation method only differs by a single word from an already existing data example. Still, such augmented examples can be useful for training due to the speaker variations injected by STR.

We observe common errors in our augmented examples for CoVoST 2 and Europarl-ST that are often connected to the different word senses and syntactical functions of the selected pivoting token. However, even grammatically wrong sentences can sometimes be useful in training as they prevent overfitting on common structures in the data. Furthermore, the speaker variations in the examples that we produce can be helpful even if the augmented examples do not differ much from existing ones. Summarizing the error analysis, our simple STR-method is able to produce examples that are useful even with errors. Investigating more complex methods for better identification of pivoting tokens is a promising direction for future work.

4.3.4 Summary

We proposed STR–an effective data augmentation method for end-to-end speech translation which leverages audio alignments, linguistic properties, and translation. It creates new audio-translation pairs via *sampling* from a memory-efficient suffix memory, *translating* through an MT model and *recombining* original and sampled audio segments with translations. Our method achieves significant improvements over augmentation with KD alone on both large (CoVoST 2) and small scale (Europarl-ST) datasets.

STR possess the characteristics of the within-corpus data augmentation. It uses the

1	src-A	*these data components in turn* serve as the building blocks of data exchanges
	src-B	the governor appoints members of the board each of whom serve *seven years*
	augm.	*these data components in turn serve seven years*
	transl.	Diese Datenkomponenten wiederum servieren sieben Jahre.
2	src-A	*the church* is unrelated to the jewish political movement of zionism
	src-B	both sacks contain a man b is *on the left a on the right*
	augm.	*the church is on the left a on the right*
	transl.	Die Kirche befindet sich rechts auf der linken Seite.
3	src-A	*the following* represents architectures which have been utilized at one point or another
	src-B	monism sees brahma as the ultimate reality while monotheism represents *the personal form brahman*
	augm.	*the following represents the personal form brahman*
	transl.	Die folgende Darstellung repräsentiert die persönliche Form Brahman.
4	src-A	*additionally the pulse output can be* directed through one of three resonator banks
	src-B	he directed *no fewer than thirty seven productions at stratford*
	augm.	*additionally the pulse output can be directed no fewer than thirty seven productions at stratford*
	transl.	Darüber hinaus kann der Pulsausgang nicht weniger als siebenunddreißig Produktionen in Stratford geleitet werden.
5	src-A	*the two* are robbed by a pickpocket who is losing in gambling
	src-B	there are *six large portraits displayed in the senate chamber*
	augm.	*the two are six large portraits displayed in the senate chamber*
	transl.	Die beiden sind sechs große Porträts, die in der Senatskammer ausgestellt sind.

Table 4.12: The first 5 augmented data examples from CoVoST 2 for the En-De language pair. "src-A" and "src-B" are the unmodified transcriptions from CoVoST 2 with our pivoting token underlined and segments we recombine in *italics*. The "augm." row shows the STR-augmented example. The "transl." row contains the MT-generated translation.

audio-transcription alignment for segmentation whereas the recombination is guided by the use of linguistic rule and a machine translation system. Compared to random recombination, such linguistically guided augmentation on the transcription-level ensures better *source-target alignment*. Similar to ADA, the suffix memory contains references rather than copies of the acoustic representations. In addition, translation of the augmented transcriptions can be generated beforehand while the source-side is concatenated during training, making STR both *memory-efficient* and *on-the-fly*.

	src-A	*i would just like to say that there are more amendments in my report because my committee* has been more ambitious in the improvements it wanted to make to the commission proposal
	src-B	economic cooperation has *always been europe s most powerful engine for greater integration and europe has owed its success to this pragmatic approach since 1956*
1	augm.	*i would just like to say that there are more amendments in my report because my committee has always been europe s most powerful engine for greater integration and europe has owed its success to this pragmatic approach since 1956*
	transl.	Je voudrais juste dire qu ' il y a plus de modifications dans mon rapport, car ma commission a toujours été le moteur le plus puissant de l ' Europe pour une plus grande intégration, et l ' Europe doit son succès à cette approche pragmatique depuis 1956.
	src-A	*i would like to thank all my colleagues on the committee who worked with me to put together some really big compromise amendments which we will* pass today
	src-B	the right of every member state to pass *laws as it deems fit as long as it has a democratic majority and that those laws should be recognised by other countries*
2	augm.	*i would like to thank all my colleagues on the committee who worked with me to put together some really big compromise amendments which we will pass laws as it deems fit as long as it has a democratic majority and that those laws should be recognised by other countries*
	transl.	Je tiens à remercier tous mes collègues de la commission qui ont travaillé avec moi pour mettre en place des amendements de compromis vraiment importants, que nous adopterons des lois, tant qu ' elle a une majorité démocratique et que ces lois devraient être reconnues par d ' autres pays.
	src-A	*i would* like all of you to give us a huge majority for this so that when we come to negotiate with the commission and council we will do our very best for europe s consumers
	src-B	i would also like *to thank all the shadow rapporteurs*
3	augm.	*i would like to thank all the shadow rapporteurs*
	transl.	Je tiens à remercier tous les rapporteurs fictifs.
	src-A	*mr president let us hope that the american proposals for purchases of toxic assets* do work because if they do not the contagion will almost certainly spread over here
	src-B	what we really need to do *is empower women*
4	augm.	*mr president let us hope that the american proposals for purchases of toxic assets do is empower women*
	transl.	Monsieur le Président, espérons que les propositions américaines d ' achats d ' actifs toxiques permettent aux femmes.
	src-A	*i would* like assurance from mr jouyet and mr almunia that we really do have our defences in place
	src-B	mr president i would like *to thank the rapporteurs and other shadows for the hard work they have put into producing these reports*
5	augm.	*i would like to thank the rapporteurs and other shadows for the hard work they have put into producing these reports*
	transl.	Je voudrais remercier les rapporteurs et d ' autres ombres pour le travail qu ' ils ont accompli dans la production de ces rapports.

Table 4.13: The first 5 augmented data examples from Europarl-ST for the En-Fr language pair. "src-A" and "src-B" are the unmodified transcriptions from Europarl-ST with our pivoting token underlined and segments we recombine in *italics*. The "augm." row shows the STR-augmented example. The "transl." row contains the MT-generated translation.

4.4 Concatenation-based Augmentation

In the last two sections, we present ADA for ASR and STR for ASTT as examples of our within-corpus data augmentation. In spite of their effectiveness, both ADA and STR require external tools such as an acoustic aligner or/and a part-of-speech tagger for augmentation. This dependence limits the applicability of the proposed approaches to non-English speech data.

In this work, we evaluate the applicability of one of the simplest data augmentation techniques, namely concatenating training instances of the original data to create new training instances, to speech-to-text processing. Our method does not need any additional data or resources, and comes with low computational effort that allows applying the augmentation procedure in-memory and on-the-fly. Our experiments show that already very strong models can be further improved with continued training using a concatenation based data augmentation approach. We further evaluate different strategies for selecting data to concatenate, and find that these strategies can make a difference depending on the size and complexity of the data set. Furthermore, we show that it is important to combine augmented data with the original to prevent degradation during continued training.

Our results are evaluated on the LibriSpeech-960h data, with and without shallow fusion (Toshniwal et al., 2018), i.e., the integration of an external language model (LM) in the decoding step, where our method is able to reduce WER down to 2.55 and 6.27 on `test-clean` and `test-other`, respectively. We also conduct experiments on the ASR part of the CoVoST-2 data set for five languages, namely English, German, Catalan, French and Spanish, and show absolute improvements of up to 0.9 WER points.

4.4.1 Related Work

Pseudo-labeling (Xu et al., 2020; Chen et al., 2020; Zhang et al., 2020) is an effective technique to use external models to generate new source-target training pairs from speech sources without transcriptions or target texts without audio. Examples are noisy student training (Park et al., 2020), consistency training (Tjandra et al., 2017; Hayashi et al., 2018; Hori et al., 2019) and TTS-generated data (Rosenberg et al., 2019; Wang et al., 2020c; Chen et al., 2021). Possible disadvantages of pseudo-labeling are its dependency on the quality of the data, and cost of integrating external models or tools, which is not necessary in our approach.

Other techniques generate new labeled data by assembling information solely from

the existing training data. For example, MixSpeech (Meng et al., 2021) creates a new audio spectrogram by linearly interpolating two spectrograms. Our method creates new data instances by concatenation in the temporal dimension. This is similar to segmenting audio-target sequences into smaller paired units in the temporal dimension, for ASR (Nguyen et al., 2020; Lam et al., 2021b; Ye et al., 2022) and speech-translation (Lam et al., 2022b). These augmentation by segmentation methods require an acoustic aligner, whereas our method does not rely on any external information.

Similar concatenation-based techniques have been applied for special purposes, e.g., random audio concatenation in speech-to-speech translation (Jia et al., 2022), or generating longer inputs for document-level neural machine translation (NMT) (Nguyen et al., 2021). Our work focuses on speech-to-text with the purpose of improving pre-trained models via continued training.

4.4.2 Method Description

Our DA strategy is to concatenate selected training instances in the temporal dimension, i.e., source-source and target-target concatenations. As there is no special separating token introduced by our method, we can make use of pre-trained off-the-shelf models. We evaluate three simple concatenation strategies: (1) CatSelf generates new training instances by repeating the original instance along the temporal dimension. (2) CatSpeaker makes use of speaker information and generates longer audio-text pairs spoken by the same person. (3) CatRandom generates new training instances by randomly concatenating audio-text pairs, spoken by different persons.

Our approach applied data augmentation *on-the-fly*. At the beginning of each epoch, we allow concatenations over the entire training data to get D_{aug}. Then, we combine the original training data and the augmented data to get $D_{\text{orig}} \cup D_{\text{orig}}$, and apply length filtering before generating the training batches. By allowing concatenations over the entire training data instead of over only the current batch, we increase diversity of the augmented data. This concatenated data is then used for *continued training* of pre-trained models or training new models *from scratch*. Figure 4.4 illustrates the procedure.

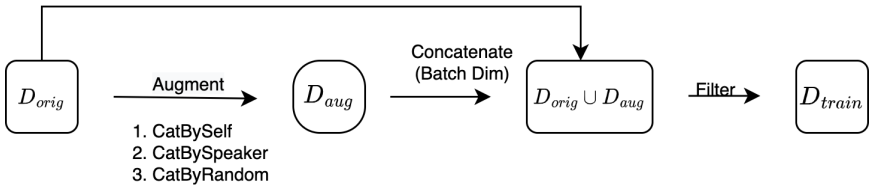

Figure 4.4: Augmentation workflow for the proposed concatenation strategy

4.4.3 Experimental Setup

Datasets and preprocessing

For the ASR tasks, we evaluated our method on LibriSpeech (Panayotov et al., 2015) and the CoVoST-2 (Wang et al., 2020a) ASR dataset. For LibriSpeech, we combined the transcriptions in train960h and the extra 800M-word monolingual text data to train the LM. For CoVoST-2 ASR, we tested on five languages: English (En), German (De), Catalan (Ca), French (Fr) and Spanish (Es). For the automatic speech translation (ASTT) tasks, we evaluated our method on CoVoST-2 and MuST-C for En-De. On both dataset, we used their own transcription-translation training data to train NMT models for KD.

For all speech inputs, we extracted 80-dimensional log Mel-filterbank with 25ms FFT windows and 10ms frame shift. We filtered instances with more than 3k frames. For transcriptions in LibriSpeech, we used the vocabulary file of 10k subword units from the FairSeq GitHub repository[13]. For CoVoST-2 ASR tasks, we lowercased transcriptions and removed punctuation. For each language, we used 5k subword units. For translation tasks, we did not apply preprocessing on the translation data. For NMT and ASTT, the size of subword units were set to 5k and 8k for CoVoST-2 and MuST-C, respectively. All sub-word units were built using SentencePiece (Kudo and Richardson, 2018).

Model Architectures

We used FairSeq for our implementation. For LibriSpeech, we used a pre-trained Transformer-based ASR model labeled *s_transformer_l* downloaded from the FairSeq GitHub repository mentioned above. For shallow fusion, we used a Transformer-based LM of about 24M parameters. It has 6 layers with attention dimension of 512 and with FFN dimension of 2048.

[13] `https://github.com/facebookresearch/fairseq`

For CoVoST-2 ASR & ASTT and MuST-C ASTT, we used a Conformer architecture (Gulati et al., 2020), labeled as *s2t_conformer*, of about 45M parameters. We followed the default configuration, with the exception of using 12 encoder layers and using attention type "attn-type=espnet".

For NMT, we used a transformer of encoder-decoder-layers of size 3 and 6 for CoVoST-2 and MuST-C, respectively. Dimensions of attention and FFN-layer are 256 and 2048, respectively.

Training and Inference

We used Adam optimizer with inverse square root learning rate schedule for all experiments. For all experiments, we used a peak learning rate (lr) of 2e-3, with the exception of LM and NMT training where we used a lr of 5e-4 and of 1e-3, respectively. For pre-training and training from scratch, we adjusted the warm-up steps for different settings. For continued training, we reset the optimizer with 1k warm-up. All speech-to-text experiments used a batch size of 40k×8 frames for training except for MuST-C, where we used 40k×2 and 25k×8 for ASR and ASTT, respectively. SpecAugment (Park et al., 2019) was applied with a frequency mask of 27 and a time mask parameter of 100, with 2 masks along their respective dimension.

For LibriSpeech, we examined our strategies by training the pre-trained ASR model for 50k steps with validation step of 2k. The LM was trained for 200k steps with a batch size of 16k×2 tokens with 4k warm-up steps. For both ASR and LM, decoder-input and output embedding were shared.

For CoVoST-2 ASR, the pre-trained ASR models and the FC cases were trained for 30k steps, validated by every 500 steps. The exception is English which has more data. We thus trained it for 60k steps with a validation step of 1k. All above models used 10k warm-up steps. For continued training, the En-ASR was trained for 20k steps, validated every 1k steps. De-ASR and Fr-ASR were trained for 10k steps whereas Ca-ASR and Es-ASR were trained for 8k steps. These four language pairs were validated every 500 steps.

For MuST-C, both ASR[14] and ASTT used 100k steps in training with 25k in warm-up and every 2k steps in validation. The NMT[15] was trained for 100 epochs with 8k warm-up steps, validated every epoch, and with a batch size of 100 sentences.

[14]For both CoVoST-2 and MuST-C, the En-ASR models used in initialisation were trained on the original data only. In addition, the ASR models were obtained by averaging their 5 best checkpoints on their validation losses.

[15]CoVoST-2 NMT is similar except of a batch size of 16k tokens.

Model	`test-clean`	w/shallow fusion	`test-other`	w/shallow fusion
Pre-trained	3.30	3.13	7.51	6.81
CT orig ∪ CatSelf	3.81	4.24	7.97	7.49
CT orig ∪ CatSpeaker	2.83 ±0.03	2.55 ±0.04	6.87 ±0.03	6.27 ±0.07
CT orig ∪ CatRandom	2.90 ±0.01	2.65 ±0.02	6.93 ±0.06	6.36 ±0.09

Table 4.14: Word Error Rate of *pre-trained* and *continued training (CT)* ASR models on LibriSpeech `test-clean` and `test-other` data sets with and without shallow fusion (SF). The "±" values indicate standard deviation over 3 runs.

For CoVoST-2 ASTT, we initialized the encoder with a pre-trained En-ASR. The ASTT was then trained for 50k steps with 10k warm-up and validated every 1k steps.

We used beam search of size 5 during inference. In shallow fusion, we used the last checkpoint with an interpolation weight of 0.3. For pre-training and training from scratch, we averaged the best 5 checkpoints by validation loss. For continued training, we averaged the *last* 5 checkpoints per validation step to prevent the averaging over pre-training checkpoints. For ASTT, we again averaged over the best 5 checkpoints.

4.4.4 Results and Analysis

LibriSpeech Table 4.14 lists WER of the continued training experiments for each of the proposed concatenation strategies. CatSelf shows the worst performance in all settings and deteriorates even over the baseline model, resulting in WER degradation from 0.46 to 1.11. Both CatSpeaker and CatRandom, on the other hand, show significant improvements over the baseline system, with CatSpeaker performing slightly better than CatRandom throughout the experiments. We conjecture that speaker information is useful for ASR in the audiobooks domain, but the effect is very limited. Compared to the baseline that is trained on the original data only, CatSpeaker shows a reduction of 0.47 WER (14.2% relative) and of 0.64 WER (8.5% relative) on the `test-clean` and `test-other` splits, respectively. Further improvements can be achieved by using shallow fusion in decoding, resulting in 2.55 WER on `test-clean` (18.5% relative reduction) and 6.27 WER on `test-other` (7.9% relative reduction). All improvements over the pre-trained model are significant with $p < 0.005$ according to an approximate randomization test.

Table 4.15 shows an ablation study where training is continued using only augmented data without adding the original data. "CT orig" refers to continued training on the original training data set by the same number of updates as the augmented one. Here, we observed only minimal to no improvements. Continued training on the CatSelf data only shows largely worse performance compared to the baseline. A

Model	`test-clean`	with SF	`test-other`	with SF
Pre-trained	3.30	3.13	7.51	6.81
CT orig	3.26	3.05	7.38	6.82
CT CatSelf	41.29	46.48	54.31	57.80
CT CatSpeaker	2.94	2.55	7.09	6.42
CT CatRandom	2.94	2.64	7.31	6.51

Table 4.15: Ablation experiment: Word Error Rate of *continued training (CT)* using only original or augmented data on LibriSpeech `test-clean` and `test-other` data sets with and without shallow fusion (SF).

detailed inspection of the generated transcriptions reveals the underlying problem: The Transformer-based system resulted in spurious repetitions in the output, consistently producing worse results than the pre-trained methods. Continued training on both CatSpeaker and CatRandom yields similar improvements with and without the inclusion of the original data.

CoVoST-2 Table 4.16 lists the WER of our concatenation strategies with continued training on 5 languages of the CoVoST-2 dataset. We see similar results to the LibriSpeech experiments, where CatSelf results in worse WER than the pre-trained models on all 5 languages. The degradation in WER ranges from 0.54 points for Catalan to 5.59 points for German, where the pre-trained systems is best for Catalan and worst for German. The CatSpeaker and CatRandom strategies yield similar WER improvements for each language. However, there is no consistent trend that might indicate if speaker information is useful or not. Throughout all languages, both CatSpeaker and CatRandom shows improvements over the pre-trained model with the largest WER improvement of 0.92 points (4.4% relative) for German, and the largest relative improvement in WER of 6.2% (0.75 points absolute) for Catalan. At the same time, the improvement on English is rather marginal even in the best case, i.e., 0.13 WER (0.7% relative) for CatRandom. We attribute this to the larger amount of the English training data compared to the other languages. The fact that this observation differs from the ASR improvements on LibriSpeech can be explained by the much simpler sentence complexity of the CoVoST-2 data. All improvements over the pre-trained model are significant with $p < 0.002$ except for English.

In Table 4.17 we repeat our ablation experiment to evaluate the contribution of the augmented data only. Similar to LibriSpeech, continued training using CatSelf data shows the worst performance compared to the baseline. An analysis of the transcriptions again reveals that the models tend to have spurious repetitions in the

Model	test (En)	test (De)	test (Ca)	test (Fr)	test (Es)
Pre-trained	19.76	20.47	13.64	15.41	14.66
CT orig ∪ CatSelf	20.75	26.06	14.18	16.05	15.21
CT orig ∪ CatSpeaker	19.67 ±0.00	19.71 ±0.02	12.79 ±0.18	14.98 ±0.00	14.05 ±0.04
CT orig ∪ CatRandom	19.63 ±0.13	19.55 ±0.04	12.89 ±0.02	15.04 ±0.07	14.13 ±0.05

Table 4.16: Word Error Rate of *pre-trained* and *continued training (CT)* ASR models trained on CoVoST-2 English (En), German (De), Catalan (Ca), French (Fr), and Spanish (Es) languages. The "±" values indicate standard deviation over 3 runs.

Model	test (En)	test (De)	test (Ca)	test (Fr)	test (Es)
Pre-trained	19.76	20.47	13.64	15.41	14.66
CT orig	20.10	20.90	13.98	15.38	15.31
CT CatSelf	117.87	118.20	110.32	114.20	112.15
CT CatSpeaker	27.36	22.34	12.94	15.69	15.22
CT CatRandom	25.46	21.74	13.68	15.69	15.54

Table 4.17: Ablation experiment: Word Error Rate *continued training (CT)* using only original or augmented data on CoVoST-2 English (En), German (De), Catalan (Ca), French (Fr), and Spanish (Es) languages.

output. In all cases except French, continued training using the original data also slightly degrades the model compared to the baseline. This is likely due to overfitting, as the pre-trained models use checkpoint-averaging to improve generalization, which is then reduced by continued training. Unlike the previous results on LibriSpeech, training on augmented data created by CatSpeaker and CatRandom mostly show worse performance over the pre-trained model. A slight improvement can be observed only for Catalan using the CatSpeaker data. We conjecture that the inclusion of the original data is vital for continued training on this dataset.

Training from scratch (CoVoST-2) Finally, we evaluate our concatenation strategies by training the entire ASR model from scratch for each language. Table 4.18 lists the results. For most cases, the improvements obtained by training from scratch are very close to those by continued training. Only for Catalan we observe further WER reduction of 0.86 compared to the continued training. Thus, our method also works for training from scratch if such training resources are available. Alternatively, one can use an off-the-shelf model and improve it via continued training with our method consuming much less computing power.

Model	test (En)	test (De)	test (Ca)	test (Fr)	test (Es)
Pre-trained	19.64 ±0.09	20.40 ±0.07	13.58 ±0.09	15.35 ±0.05	14.79 ±0.18
FS orig ∪ CatSelf	21.59	21.47	13.67	16.47	15.53
FS orig ∪ CatSpeaker	19.65 ±0.04	19.50 ±0.05	12.09 ±0.19	14.87 ±0.11	14.03 ±0.11
FS orig ∪ CatRandom	19.44 ±0.17	19.22 ±0.00	11.96 ±0.12	14.94 ±0.08	14.14 ±0.06

Table 4.18: Word Error Rate of different ASR systems trained *from scratch (FS)* on the ASR part of CoVoST-2 English (En), German (De), Catalan (Ca), French (Fr) and Spanish (Es) languages. The "±" values are standard deviations over 3 runs.

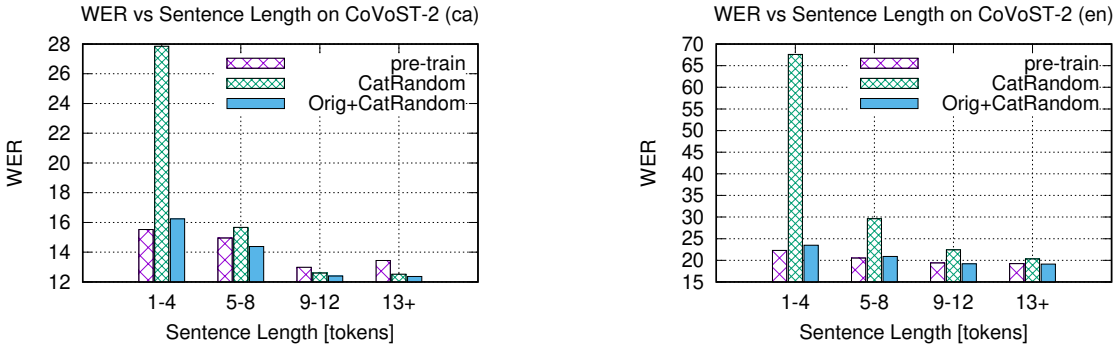

Figure 4.5: WER w.r.t. sentence length on CoVoST-2.

4.4.5 Length Dependent Analysis

We conducted a deeper analysis of the ablation study in Table 4.15 and in Table 4.17 by evaluating test examples based on their length: training using only augmented data leads to a strong increase in WER for short examples, where spurious repetitions of the textual output is the most noticeable problem. By concatenating examples, the data length distribution is shifted to the longer side and changed particularly at the ends; e.g., examples containing only 1 token are completely absent in the augmented data. Furthermore, CoVoST-2 has about 9% test examples of 5 tokens or less, whereas LibriSpeech has only 1.5%. The increase of errors on short examples thus affects the overall WER much more on the CoVoST-2 dataset.

In Figure 4.5 we plot WER on CoVoST-2 w.r.t. sentence length for the pre-trained models, and for continued training on CatRandom and on original+CatRandom data. Including the original data during training effectively reduces this problem as can be seen from Figure 4.5 for Catalan (left) and English (right), and we found a similar behavior for the other three languages.

ASTT (En-De): MuST-C and CoVoST-2 We also evaluate our proposed data augmentation strategies on two En-De speech-to-text translation tasks. Table 4.19

Model	MuST-C tst-COMMON	CoVoST-2 test
orig	52.8 ±0.0	47.65 ±0.05
orig ∪ CatSpeaker	53.55 ±0.05	48.55 ±0.05
orig ∪ CatRandom	53.55 ±0.05	48.45 ±0.05

Table 4.19: chrF2 on MuST-C ASTT and CoVoST-2 ASTT (En-De). The "±" values indicate standard deviations over 2 runs.

lists the chrF2 scores of systems trained with "orig" (original data plus translations generated by knowledge distillation) and trained with combined data created by CatSpeaker or by CatRandom. Both concatenation strategies achieve significant improvements with $p < 0.00025$ both on MuST-C tst-COMMON and on CoVoST-2 test sets using the approximate randomization test implementation of SACREBLEU[16]. The results show that our simple method is also applicable to ASTT where the speech-text alignments are not parallel.

4.4.6 Summary

We propose and evaluate temporal-concatenation as a data augmentation method for improving Transformer and Conformer based speech-to-text models. The method can be applied to improve pre-trained models without requiring extra information or external tools. We evaluate three concatenation strategies for ASR on LibriSpeech and CoVoST-2 data and found that concatenation by random and concatenation by speaker perform similarly and bring significant improvements. Finally, we evaluate our method for ASTT on Must-C and CoVoST-2 and also observed significant improvements.

Similar to ADA and STR, these concatenation-based methods also posses the desired characteristics and are computationally simpler. The original sentence-level information becomes the segmentation method whereas 1) CatByRandom, 2) CatBySpeaker and 3) CatBySelf are the recombination strategies.

[16] nrefs:1|ar:10000|seed:12345|case:mixed|eff:yes|nc:6|nw:0|space:no|version:2.0.0

4.5 Chapter Summary

In this chapter, we present three approaches of within-corpus data augmentation for ASR and ASTT. In ADA for ASR, we use acoustic aligner to segment the parallel data, followed by recombination via aligned replacement. The monotonic alignment between acoustic sequences and their transcriptions ensure source-target alignment after perturbation. On LibriSpeech corpus, we show that the improvement of ADA is complementary to SpecAugment, a simple but effective augmentation method in ASR. In STR for ASTT, we also use an acoustic aligner to segment the acoustic representations and the transcriptions. The recombination step, however, is guided by the linguistic property of the source transcriptions and a machine translation system in order to ensure better source-target alignment. On multiple language pairs of CoVoST-2 and Europarl-ST data, we show that STR brings further improvement even when applied together with SpecAugment and knowledge distillation. In the last method, we resort to use the official sentence-level data instances as the result of the segmentation step and use simple attributes, such as speaker information, as the recombination strategies. In spite of its simplicity, both concatenation-by-speaker and concatenation-by-random methods bring significant improvement over baselines in ASR and ASTT.

Future work may to investigate methods for combining existing data augmentation techniques such as learning a sample-adaptive policy, or to combine our approach with self-training given source audio data. We may also enrich the audio/suffix memory with representations extracted on speed perturbed raw audio waveforms or with n-gram information as additional keys. In addition, we would examine different linguistic properties to guide the recombination process.

Chapter 5

Instance-specific Data Selection

In the previous chapter, we approach the issue of data scarcity from the angle of increasing the amount of effective training data. In this chapter, we discuss our work which attempts the opposite, i.e., selecting relevant training data. We present an algorithm for picking relevant pseudo-labels for improving a cascade speech-to-text translation system. We also analyzed the effectiveness of Influence Functions, a training data attribution technique, for filtering training data responsible for badly translated instances.

Materials in this chapter have been drawn from this two publications: ICASSP 2021 (Lam et al., 2021b) and WMT 2022 (Lam et al., 2022a).

5.1 Introduction and Overview

In the last chapter, we present a novel data augmentation method to increase the size of the effective training data for relieving the problem of data scarcity. Typically, the performance of DNN is positively correlated to its amount of training data, but not every training data have positive impact. In this chapter, we switch our focus from quantity to quality; we aim at picking training instances relevant to particular model performance and prediction. Since this selection algorithms focus on the performance of specific instances rather than generic quality of the training corpus, we call them *instance-specific data selection*. We present the concept in two scenarios: 1) selection of pseudo-labels in a cascade speech-to-text translation system and 2) back-tracking of model's prediction back to its training data via Influence Functions.

Similar to augmentation, data selection is another commonly used approach in improving NeurS2S model. In training data selection, it helps to discard noisy parallel data (Khadivi and Ney, 2005; Taghipour et al., 2010; Denkowski et al., 2012), to retain

domain relevant training data (Moore and Lewis, 2010; Axelrod et al., 2011; Banerjee et al., 2011), to select augmented training data or to back-track model's prediction back to its training data. In test data selection, e.g., quality estimation (Specia et al., 2009; Kreutzer et al., 2015; Ueffing et al., 2018), it helps to ensure the quality of the model output before being shown to the users. In this chapter, we focus on training data selection.

Training data selection algorithms can be further classified into off-the-fly or on-the-fly, depending on if the algorithm is applied before or during training. Off-the-fly selection algorithms, such as Moore and Lewis (2010) and Junczys-Dowmunt (2018), are applied before training, making the training process simpler. In case of machine translation, it usually includes a pipeline of rule-based and neural-based selection algorithms. In opposite, on-the-fly algorithm is adaptive to the state of the underlying model so that it better use the training data at the cost of a possibly more complicated training process, e.g., curriculum learning (Wang et al., 2020d; Dou et al., 2020; Lichtarge et al., 2020). Typically, both off-the-fly and on-the-fly selection algorithms are applied together. In our work of pseudo-label selection (Section 5.2), the selection occurs during training and is adaptive to the state of the underlying model. It is thus *on-the-fly*. The use of Influence Functions on neural machine translation 5.3, in opposite, presents an example of off-the-fly data selection.

5.2 Cyclic Feedback for Cascade Speech-to-Text Translation

Direct end-to-end ASTT models (Weiss et al., 2017) have been shown to overcome the error propagation issues of traditional cascades of ASR and MT *if enough in-domain parallel data of source audio and text translation are available* (Sperber et al., 2019b).

Considerable research effort has thus been invested in improving the data efficiency of direct ASTT, either by better exploitation of out-of-domain speech and translation resources in sophisticated information passing in multi-task approaches (Weiss et al., 2017; Sperber et al., 2019b; Bérard et al., 2018; Anastasopoulos and Chiang, 2018), or by synthesizing parallel data by back-translation approaches (Jia et al., 2019; Pino et al., 2019). However, as recently shown by Sperber and Paulik (2020b), problems like domain mismatch and error propagation might be re-introduced by exploiting out-of-domain data and by information-passing in end-to-end ASTT. On the other hand, cascaded models seem still to benefit more from out-of-domain data to directly

improve their MT and ASR components, than end-to-end systems can exploit such data by multi-task learning (Sperber et al., 2019b; Bérard et al., 2018; Pino et al., 2019). One question in the ongoing competition between end-to-end and cascaded models is which paradigm is preferable in low-resource scenarios where only a few thousand parallel data of recorded speech and text translation, but no further in-domain data to train MT and ASR separately, are available. Such a scenario is real for endangered languages (Duong et al., 2016), and it corresponds to the status quo in speech translation where copious amounts of training data are mostly available only in form of out-of-domain MT or ASR data. In this paper, we focus on low-resource direct speech translation. We confirm common knowledge that end-to-end systems can better exploit in-domain direct speech translation data, while cascades outperform end-to-end systems *if enough out-of-domain MT and ASR data* are available.

The main contribution of our paper is a novel adaptation cycle that allows ASR-MT cascades to also exploit direct speech translation data (that are useless for traditional cascades), and to eventually improve over end-to-end ASTT models by a wide margin. The crucial ingredients of our cyclic feedback approach are firstly to train the MT system on k-best ASR outputs. This will teach the MT system to translate imperfect ASR outputs into correct foreign sentences. Furthermore, the evaluation performance of the produced MT output is used as signal to improve the ASR system by selecting and weighting transcriptions leading to top-scoring translations as targets in self-training. This learning cycle will tune the ASR system towards producing transcriptions that perform well as translation inputs, thus improving the whole pipeline, without explicit parameter sharing or back-propagation. Our experiments on German-to-English speech translation on audio books (Beilharz et al., 2020) and diverse domain (Wang et al., 2020a) corpora show that 3.8 or 5.1 BLEU points can be gained over end-to-end systems on the respective datasets.

5.2.1 Related Work

The problem of closing the domain gap between ASR output and text input to MT and has been addressed already in the framework of Statistical Machine Translation (SMT), by training SMT systems on automatically transcribed speech (Peitz et al., 2012), or by augmenting SMT translation models with simulated acoustic confusions (Tsvetkov et al., 2014). In the area of NeurS2S learning, similar approaches have been applied to ASR error correction, either directly by monolingual sequence-to-sequence transformation (Mani et al., 2020), or by adapting the framework of generative

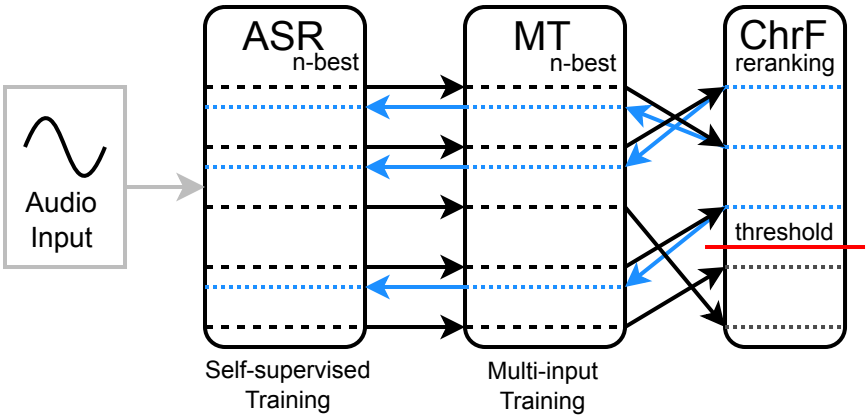

Figure 5.1: Cyclic feedback in ASR-MT cascade.

adversarial networks to provide a language-model critic to improve ASR (Liu et al., 2019a). Our work extends these ideas by using the performance improvement of downstream MT as learning signal in self-training of ASR.

The idea of tuning ASR parameters for optimal downstream translation performance is even older, and has been applied successfully via minimum error rate training (Och, 2003) of models that integrate ASR and MT features (Zhang et al., 2004; He et al., 2011). In recent years, deep reinforcement learning has been the machine learning approach of choice in order to tune a NeurS2S model to task-specific evaluation metrics (Keneshloo et al., 2020). The crucial difference to our approach is that we reward the ASR system by an evaluation score that is grounded in a downstream application, and not by an evaluation metric that is task-specific to ASR. Furthermore, we do not rely on the machinery of reinforcement learning, but we follow a much simpler and more efficient training scheme where the downstream reward signal is used to select and weight ASR outputs for self-training.

Recently, out-of-domain pre-training has been combined with in-domain triplets for end-to-end fine-tuning (Liu et al., 2019c). We use only speech-translation pairs to fine-tune our cascade. Speech-translation pairs for fine-tuning have also shown to be effective in a meta-learning scenario (Indurthi et al., 2020). Their method was applied to larger datasets of 229k-275k pairs while our method works for very small datasets of 6.7k and 59k pairs.

5.2.2 Cyclic Feedback

Our cyclic feedback idea is based on self-training with a twist. The algorithm has two parts, where in one part the MT system is tuned to produce better translations

for potentially noisy ASR outputs, and in the other part the ASR system is guided by the MT-output to generate transcriptions that led to higher scoring translations. Figure 5.1 shows a cascaded model for a single German (*de*) audio input. The model first produces k-best ASR transcriptions, which are fed as multiple inputs into an MT system. The translations are reranked according to their ChrF-score, and a list of indices of translations exceeding a ChrF threshold (indicated by red dashed line) is kept. These indices are used as learning signal in a feedback cycle to select data to improve the ASR and MT components (indicated by blue dotted arrow). The MT system takes the selected data as input to learn how to translate imperfect ASR outputs into foreign reference sentences. The ASR system is trained in a self-training fashion using the selected data as reference transcriptions.

In our concrete implementation, at the beginning of the MT-adaptation loop, the ASR-system generates k-best ($k = 8$) transcriptions via beam search. These outputs are then passed to the MT-system for translation. Based on the ChrF-score threshold of 0.4, we create a fine-tuning dataset by selecting corresponding ASR transcriptions as inputs for multi-input training. We additionally apply a weighting scheme that gives more weight to transcriptions that lead to better translations. The MT-system is then fine-tuned on this dataset via cross-entropy loss. This process is repeated until no further improvement on the dev set is obtained. At the beginning of the ASR-adaptation loop, we again generate k-best ASR transcriptions and their translations (in the first loop, we can re-use the already generated transcriptions and translations from the last loop of MT-tuning). The ChrF-score of corresponding MT-outputs against the reference translations is used to indicate a dataset of audio data and the generated ASR transcriptions. A similar weighting scheme as for MT-adaptation is also applied, but as the ASR system is more sensible to errors, we increased the ChrF threshold to 0.6, i.e., an *adaptive* selection scheme. The ASR-model is then fine-tuned with the generated data. This process can be repeated until the ASR training converges on the dev set. Finally, we start another cycle and enter the MT-adaptation loop until we observe no further improvement.

5.2.3 Experimental Setup

Datasets The first ASTT dataset that we used for fine-tuning on a new domain was LibriVoxDeEn (Beilharz et al., 2020). This dataset consists of 86 classical German audio books with 547h of speech data. Of these 86 books, 19 books were published with German to English text alignments and can thus be used in ASTT training.

Dataset	Purpose	No. Sentences			Avg. Words
		Train	Dev	Test	per Sentence
Spoken Wikipedia	ASR PT	160k	4k	4k	11.02
WMT14	MT PT	2.1M	23k	22k	23.99
LibriVoxDeEn	AST FT	6.7k	355	1.1k	14.78
CoVoST De-En	AST FT	59.0k	15.4k	145.5k	8.01

Table 5.1: Statistics of datasets used for pre-training (PT) and fine-tuning (FT) in our experiments.

As we need very clean data in our experiments, we applied strong filtering on the LibriVoxEnDe data and selected only four out of the 19 audio books for fine tuning. For each book, we analyzed 30 randomly sampled parallel sentences and annotated the pairs as "perfectly aligned", "partly aligned", or "not aligned". We then selected the books which had perfectly aligned sentences in 80% or more of the cases. These are *The Picture of Dorian Gray* by Oscar Wilde, *Casanovas Heimfahrt* by Arthur Schnitzler, *Die Verwandlung* by Franz Kafka, and *Undine* by Friedrich de la Motte Fouqué. These audio books are from the early 20th century except for the last one.

The other dataset we used was the German-English portion of the CoVoST dataset (Wang et al., 2020a). This dataset is built upon Common Voice (Ardila et al., 2020). For each transcription-translation pair, there are multiple recordings for each transcription spoken by different speakers. Throughout our experiments on CoVoST we used the original data splits so our results are comparable to the baselines reported by Wang et al. (2020a). For efficiency, we reduced the dev set size to 15,351 examples by downsampling by 1/5.

For pre-training of ASR and MT components in an additional experiment, we employed large scale datasets from both areas. For ASR pre-training, we used the German Spoken Wikipedia (2.0) (Baumann et al., 2019) corpus, a dataset containing more than 250h of aligned German sentences. For MT pre-training, we concatenated data from the official WMT14 English-German parallel data, namely Europarl v9, News Commentary v14, and IWSLT 2014. Table 5.1 summarizes the statistics of each dataset and its splits.[1]

Model: Direct End-to-End Speech-to-Text We used a variant of a joint CTC-attention end-to-end framework (Liu et al., 2019a) which its encoder contains a VGG network and 3 blocks of BiLSTM layers. The VGG network reduces the temporal resolutions by a factor of 4. In each block, there is a BiLSTM layer with 256 hidden

[1]Our LibriVoxDeEn data split: www.cl.uni-heidelberg.de/librivoxdeen/

Target domain	Fine-tuning data ASR	MT	End-to-end ASTT	Untuned cascade	Cascade	w/o ASR-tuning
LibriVoxDeEn	6.7k	6.7k	9.1	8.1	12.9	11.0
CoVoST De-En	59.0k	59.0k	7.8	11.3	12.9	12.4

Table 5.2: Best results for end-to-end and cascade approaches.

units per direction followed by a feed forward neural network with ELU activation (Clevert et al., 2016). There is one LSTM layer of size 256 in the decoder, and it is connected to the encoder via location-based attention (Chorowski et al., 2015).

We applied Locked Dropout (Merity et al., 2018) to each BiLSTM block with a value of 0.2 and embedding dropout of 0.1 (Gal and Ghahramani, 2016). We used 40-dimensional log Mel filter bank features with z-score normalization as input to the encoder. Data instances containing audio longer than 2,000 frames were excluded.

For the speech recognition system, the German textual data was lowercased, all punctuation removed, and numbers were normalized to their spoken form using pre-processing tools from the `marytts`[2] toolkit.

Model: Machine Translation In order to better compare to the direct end-to-end ASTT system and to share pre-trained components, we used an LSTM-based architecture instead of a more sophisticated model such as the Transformer architecture. There are 3 BiLSTM layers and one single LSTM layer for the encoder and the decoder respectively; we set their per-direction dimension to be 256. Similar to the VGG encoder, we applied Locked Dropout with a value of 0.1 to the BiLSTM encoder. Dropout of 0.1 was applied to the target embedding. In addition, we shared parameters of the embeddings and the output layer. Since both cascaded system and end-to-end ASTT system share the same data, we used a universal vocabulary of 10,000 subword units created with SentencePiece for all tasks.

Training of End-to-End Speech-to-Text Translation In our low-resource scenario we assumed that there are no audio-transcription-translation triplets in the target domain. This makes multi-task learning (Bérard et al., 2018) on in-domain data impossible. We instead used a transfer-learning based method called the adapter (Bahar et al., 2019), which connects the pre-trained ASR encoder with the pre-trained MT decoder in a separate learned layer.

[2]https://github.com/marytts/marytts/

Experiment name	Pre-training data ASR	MT	End-to-end ASTT	Untuned cascade	+cyclic feedback
librivox-100-100	159.5k	2.1M	9.1	8.1	12.9
librivox-100-10	159.5k	213.4k	8.8	6.2	9.4
librivox-100-2.5	159.5k	53.3k	8.8	3.9	5.4
librivox-25-100	39.9k	2.1M	8.8	6.2	9.8
librivox-25-10	39.9k	213.4k	8.9	5.2	7.5
librivox-25-2.5	39.9k	53.3k	8.1	3.2	5.4
librivox-10-100	15.9k	2.1M	8.4	4.9	6.8
librivox-10-10	15.9k	213.4k	8.3	4.1	5.4
librivox-10-2.5	15.9k	53.3k	7.1	2.6	4.6

Table 5.3: Results on four audio books from LibriVoxDeEn under different data sizes for pre-training.

5.2.4 Results and Analysis

Baselines and Best Results Table 5.2 gives an overview of our results compared to baselines. Wang et al. (2020a) reported 7.6 BLEU points for the end-to-end model on CovoST German-English data. Our end-to-end model performs at 7.8 BLEU, despite differences in experimental conditions such as character-level vs. sub-word units, different pre-training data, and our smaller decoder.

The results for our out-of-domain pre-trained cascade is at 11.3 BLEU for CoVoST, and improved to 12.9 BLEU by cyclic feedback.

To date no external baselines are available for a comparison on LibriVoxDeEn. We show improvements of 3.8 BLEU over our own end-to-end system, and of 4.8 BLEU over the untuned cascade.

LibriVoxDeEn Table 5.3 shows the performance of end-to-end ASTT and cascaded system fine-tuned on LibriVoxDeEn under different sizes of pre-training data. The untuned cascaded system is 1 BLEU point behind the fine-tuned end-to-end ASTT system if both the ASR and MT components have access to all available data.

The end-to-end system is much less sensitive to the amounts of pre-training data: while the untuned cascade quickly drops in performance if both ASR and MT data are reduced, the end-to-end system suffers a reduction of at most 2 BLEU points.

As soon as we add *cyclic feedback*, the results of the cascaded system improve significantly and we see gains of up to 4.8 and 3.8 BLEU points over the untuned cascaded and the end-to-end ASTT system, respectively. The cyclic feedback approach is always better than the end-to-end system if the amount of pre-training data exceeds

Experiment name	Pre-training data ASR	MT	End-to-end ASTT	Untuned cascade	+cyclic feedback
covost-100-100	159.5k	2.1M	7.8	11.3	12.9
covost-100-10	159.5k	213.4k	7.5	9.4	10.6
covost-100-2.5	159.5k	53.3k	7.0	6.7	7.6
covost-25-100	39.9k	2.1M	7.0	7.8	9.1
covost-25-10	39.9k	213.4k	6.8	6.6	7.6
covost-25-2.5	39.9k	53.3k	6.5	4.9	5.7
covost-10-100	15.9k	2.1M	6.5	5.2	6.1
covost-10-10	15.9k	213.4k	6.3	4.5	5.2
covost-10-2.5	15.9k	53.3k	5.9	3.5	4.2

Table 5.4: Results on the German-English part of the CoVoST dataset under different data sizes for pre-training.

100k for both ASR and MT.

CoVoST De-En Table 5.4 shows that the untuned cascaded system matches or surpasses the end-to-end ASTT system if both the MT components have access to all available data. When using all pre-training data, the untuned cascaded system lies 3.5 BLEU points above the end-to-end system. We observe the following trend when the pre-training data is reduced: the end-to-end system has the largest drop when ASR-data is reduced, while the cascaded system is more sensitive to the quality of the MT-system, which is where the feedback signal for fine-tuning comes from. With *cyclic feedback*, we again observe considerable improvements of up to 1.6 BLEU points for fine-tuning, but all in all lower than in the previous experiment.

We assume that the simple textual structure of the CoVoST dataset is one main reason for the good performance of the untuned systems, and that the low number of unique examples in CoVoST's multi-speaker data provides too little variation for larger gains by cyclic feedback.

Ablation Study We also evaluated the contribution of ASR tuning based on feedback from the MT model to the cyclic feedback method. The results are listed in the last column of Table 5.2 titled "without ASR-tuning", where we applied fine-tuning only on the MT-part until no further improvement was observed. The resulting system get considerable improvements over the corresponding untuned cascaded but is of 1.9 and 0.5 BLEU points worse than its cyclic feedback counterpart on LibriVoxDeEn and Covost De-En, respectively. In our side experiments, we also see improvements between 0.5 and 0.8 BLEU points in other cases with reduced data. This underlines

#		**Translation**
1	Untuned	i wanted to say that i did not talk about cloabel.
	Tuning	i wanted to say i don't talk to you..
	Final	i wish i had not spoke of sibyl vane.
	Reference	*i wish now i had not told you about sibyl vane.*
2	Untuned	i'm going to stay in the domain. he sadly said.
	Tuning	i shall remain in the real dorian, he said.
	Final	i shall stay with the real dorian, he said.
	Reference	*i shall stay with the real dorian, he said, sadly.*
3	Untuned	he had the life of his life, his life and his own resilience news of life.
	Tuning	he had life in his life, his life, and his own vicious news of life.
	Final	he life was determined to him, life and his own countless curiosity about life.
	Reference	*yes, life had decided that for him life, and his own infinite curiosity about life.*
4	Untuned	it is not only up to the bed in the bed-growth.
	Tuning	i don't end up in the bed's misused, said gregor.
	Final	it is only not to restraint in the bed, said gregor.
	Reference	*but i must not stay in bed uselessly, said gregor to himself.*

Table 5.5: Four examples taken from our experiments on LibriVoxDeEn that illustrate the different steps in fine-tuning.

the contribution of ASR tuning and shows the effectiveness of combining ASR and MT tuning in a cyclic feedback manner.

Examples Table 5.5 lists four examples from the LibriVoxDeEn experiments to illustrate the process of fine-tuning via cyclic feedback. The *untuned* translation comes from the untuned system and often contains misspelled words such as "cloabel" in Example 1 or "bed-growth" in Example 4. The *tuning* translation is generated by the system after few steps of fine-tuning and already shows significant changes, e.g. "domain" is correctly transcribed as "dorian" in Example 2. In the same example we see that the almost correct sentence "he sadly said" is transformed to the subclause "he said", dropping the correct "sadly" adverb. The *final* translation is the translation we receive from the final fine-tuned system. In Example 1, the nonsense word "cloabel" is correctly transformed to "sibil vane". Example 2 gives an almost perfect translation after the system is fine-tuned. The final examples in Examples 3 and 4 show typical examples in the test set, where the final translation is significantly better than the untuned version to the extend that the translation becomes understandable, but it is still far from perfect when compared to the *reference*. Together with the modest BLEU scores we achieve overall, this again underlines the difficulty of the speech translation task on different domains.

5.2.5 Summary

We presented a novel domain adaptation technique for fine-tuning of cascaded ASTT models without the use of transcriptions. The key idea is to exploit translation quality as a signal to guide the ASR system to generate transcriptions that lead to better translations, and at the same time make the MT model more robust against transcription errors. In two low-resource domain adaptation scenarios on largely different domains we observe considerable gains over a comparable end-to-end system and the untuned out-of-domain cascaded system.

The proposed cyclic feedback can also be viewed as an augmentation method since the ASR is trained via self-training and the MT is trained via back-translation. Its success, however, relies on the adaptive selection from the n-best pseudo-labels. Such selection enhances the alignment of ASR and MT toward the same goal, resulting in better speech-translation performance. Additionally, the selection of the pseudo-labels is guided by the performance on the input training instance; this makes cyclic feedback instance-specific.

5.3 Influence Functions for Neural Machine Translation

In this section, we discuss the second scenario of back-tracking model's prediction back to its training data in NMT. This is especially important for commercial machine translation systems where customer feedback can be an important signal for improvement.

NMT is the de facto standard for recent high-quality MT systems, but it requires abundant amount of bi-text for supervised training. One common approach to increase the amount of bi-text is via data augmentation, such as pseudo-labelling (Sennrich et al., 2016a; Edunov et al., 2018b; He et al., 2020). Another approach is the use of web-crawled data (Bañón et al., 2020) but since crawled data is known to be notoriously noisy (Khayrallah and Koehn, 2018; Caswell et al., 2020), a plethora of data filtering techniques (Junczys-Dowmunt, 2018; Wang et al., 2018a; Ramírez-Sánchez et al., 2020, *inter alia*) have been proposed for retaining a cleaner portion of the bi-text for training.

While standard data filtering techniques aim to improve the quality of the overall training data, *instance-specific data filtering* aims quality improvement towards specific instances via removal of the related (erroneous) training data. Manual filtering in commercial MT system, for an example, involves human annotators to identify translation errors on sentences reported by customer and to design filtering scheme,

such as regular expressions, to search related training examples for removal from the training set.

In this work, we attempt to apply a more automatable technique called influence functions (IF) which is shown to be effective on image classification (Koh and Liang, 2017), and certain NLP tasks such as sentiment analysis, entailment and toxic speech detection (Han and Tsvetkov, 2020; Guo et al., 2021). Given a probing example, influence functions (IF) search for the influential training examples by measuring the similarity of the probing example with a set of training examples in gradient space. Schioppa et al. (2022) use a low-rank approximation of the Hessian to speed up the computation of IF and apply the idea of self-influence to NMT. However, self-influence measures if a training instance is an outlier rather than its similarity with another instance. Akyürek et al. (2022) question the back-tracing ability of IF on the fact-tracing task. They compare IF with heuristics used in Information Retrieval and attribute the worse performance of IF to a problem called *saturation*. Compared to fact-tracing, the target sides of machine translation can be more diverse which complicates the application of IF.

We apply an effective type of IF called *TracIn* (Pruthi et al., 2020) to NMT for instance-specific data filtering and analyze its behaviour by constructing synthetic training examples containing simulated translation errors. In particular, we find that

- the gradient similarity, also called the influence[3], is highly sensitive to the network component.
- vanilla IF may not be sufficient to achieve good retrieval performance. We proposed two contrastive methods to further improve the performance.
- training examples consisting of copied source sentences have similar gradients even when they are lexically different. This indicates that the use of influence functions can go beyond what can be achieved with regular expressions.
- an effective automation of the instance-specific data filtering remains challenging.

To the best of our knowledge, we are the first to investigate applying IF for instance-specific data filtering to NMT.

[3]In this work, we use gradient similarity or influence interchangeably to denote the result of IF. Be aware that TracIn is also one type of IF.

5.3.1 Influence Functions

IF is a technique from robust statistics (Hampel, 1974; Cook and Weisberg, 1982, *inter alia*). It aims to trace a model's predictions back to the most responsible training examples without repeated re-training of the model, aka Leave-One-Out. Koh and Liang (2017) extend this idea from robust statistics to DNN that requires only the gradient of the loss functions L and Hessian-vector products so that the influence $\mathcal{I}(z, z')$ of two examples z and z' is approximated as

$$\mathcal{I}(z, z') \approx \nabla_\theta L(z')^T H_{\hat{\theta}}^{-1} \nabla_\theta L(z) \tag{5.1}$$

where $\hat{\theta}$ is the model parameters at optimum and $H_{\hat{\theta}} = \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta^2 L(\theta)$ is the Hessian of the model parameters at $\hat{\theta}$. Given n number of training instances and p number of model parameters, the inverse of Hessian has a complexity of $\mathcal{O}(np^2 + p^3)$ which is expensive to compute for DNN. There are several proposed methods to speed up the computation of IF, e.g., by computing on a training subset selected by KNN-search (Guo et al., 2021), by approximating the Hessian with LISSA (Agarwal et al., 2017), by computing on a subset of model parameters (Koh and Liang, 2017), or by replacing the Hessian with some other procedures (Pruthi et al., 2020). In this work, we focus on TracIn which is shown to be better than some other variations (Han and Tsvetkov, 2020; Schioppa et al., 2022) in terms of retrieval performance.

TracIn, denoted by $\mathcal{I}_{\text{TracIn}}(z, z')$, replaces the computationally costly Hessian matrix with an identity matrix. The remained gradient dot product, or called the gradient similarity, is instead computed over C number of checkpoints, followed by averaging:

$$\mathcal{I}_{\text{TracIn}}(z, z') = \frac{1}{C} \sum_{i=1}^{C} \nabla_\theta L(z')^T \nabla_\theta L(z) \tag{5.2}$$

In NMT, given the same source sentence, the magnitude of the gradient in general is positively correlated to the length of the target sentence. In order to reduce the effect of the target length, we normalize equation (5.2) by the product of $\|\nabla_\theta L(z')\|$ and $\|\nabla_\theta L(z)\|$, or equivalently, we compute the cosine similarity of $\nabla_\theta L(z')$ and $\nabla_\theta L(z)$.

Given a probing instance z' and its probing gradient $\nabla_\theta L(z')$, instances in the training set that yield a positive value of $\mathcal{I}_{\text{TracIn}}(z, z')$ are called the positively influential training instances (+IFTrain) whereas those that yield a negative value of $\mathcal{I}_{\text{TracIn}}(z, z')$ are called the negatively influential training instances (-IFTrain). Taking a gradient step on +IFTrain reduces the loss on the probing example while taking a gradient step on -IFTrain increases it. IF can be used for data filtering by removing the +IFTrain

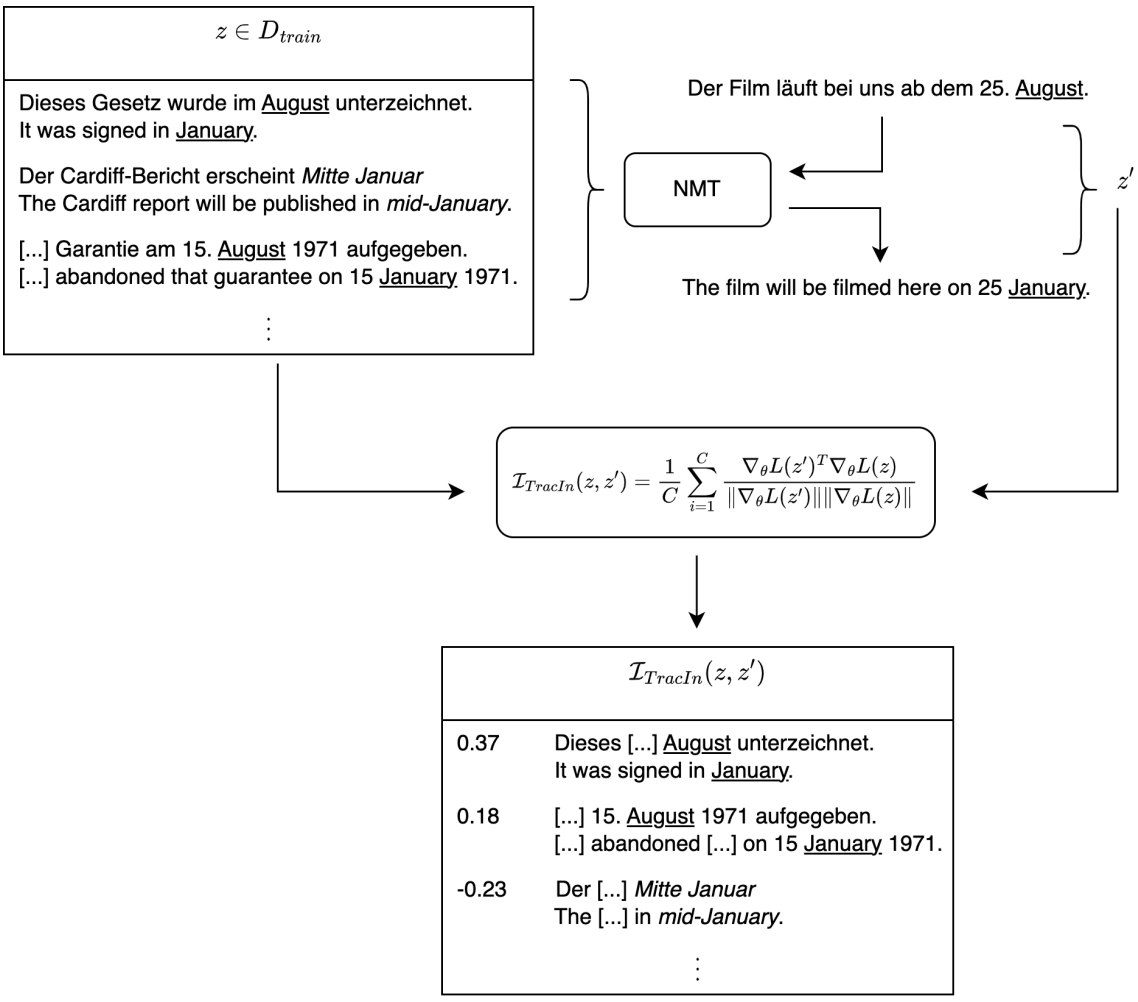

Figure 5.2: Diagram showing the workflow of using Influence Functions for data selection in NMT.

examples of low quality probing samples since their gradients have similar direction. Conversely, if the probing sample is of high quality, removing -IFTrain examples from the training data would be expected to increase translation quality w.r.t. the probing sample. Figure 5.2 illustrates the process of using Influence Functions for instance-specific data filtering in NMT.

	Samples	Shared parameters		Non-shared parameters			
		∇_{Full}	∇_{Emb}	∇_{srcEmb}	∇_{trgEmb}	∇_{output}	∇_{concat}
Probing	Noch kommt Volkswagen glimpflich durch. Volkswagen gets off lightly.	1	1	1	1	1	1
1	Das £ 1,35 Mrd. teure Projekt soll bis Mai 2017 fertiggestellt werden Volkswagen gets off lightly.	0.153	0.240	0.006	0.287	0.437	0.339
2	Alle in Frage kommenden Produkte wurden aus dem Verkauf gezogen. Volkswagen gets off lightly.	0.238	0.320	0.013	0.230	0.401	0.319
3	Noch kommt Volkswagen glimpflich durch. In 2008, most malware programmes were still focused on sending out adverts.	-0.021	-0.030	-0.149	-0.022	-0.017	-0.040
4	Noch kommt Volkswagen glimpflich durch. We've made a complete turnaround.	-0.007	-0.016	-0.120	-0.003	0.011	-0.013
5	Noch kommt Volkswagen glimpflich durch. Volkswagen gets off lightly!	0.950	0.894	0.973	0.927	0.843	0.873
6	Noch kommt Volkswagen glimpflich durch! Volkswagen gets off lightly.	0.899	0.912	0.873	0.915	0.940	0.927

Table 5.6: Example showing the changes of influence by network components. Segments that are marked in red are perturbed from the probing example. ∇_X indicates the network components used in computing the influence, ∇_{concat} indicates the concatenation of ∇_{srcEmb}, ∇_{trgEmb} and ∇_{output}.

5.3.2 Experimental Setup

Model configuration and training We used Transformer BASE configuration as described in Vaswani et al. (2017) with default setting and implementation in FAIRSEQ. We used a SENTENCEPIECE model to create subword units of size 32k. Unless otherwise specified, we pre-trained our NMT on Europarl-v7 data and News Commentary-v12 data in German-English direction from WMT17 for 100 epochs, about 112K updates, using Adam optimizerion training of 16-bit[4]. The effective mini-batch size was 4096 x 16 tokens and it took a p3.16xlarge[5] machine on AWS 6 hours for training. We evaluated the MT model on the newstest2017 test set with a checkpoint averaged over the 10-best checkpoints, measured by the validation loss on the newstest2014-2016 dev set. On the test set, our NMT model with non-shared parameters with the two word embeddings and the output layer scores 29.99 BLEU whereas the one with shared parameters scores 29.78 BLEU. We used beam search with beam size of 5 in decoding.

[4]We used 32-bit precision to compute the gradient similarity once the training is done.
[5]See https://aws.amazon.com/ec2/instance-types/ for details.

TracIn We selected 5 checkpoints, i.e., at epoch 5, 8, 15, 30 and 100 for computing TracIn[6]. The checkpoints were selected based on their relatively large changes in the validation loss, i.e., usually in the earlier phrase of training; the last checkpoint was included to cover information at the end of the training. We computed the per-sample gradient with a batch size of 1 parallelized over multiple processes with several g4dn.2x[5] machines on AWS.

5.3.3 Results and Analysis

This section describes our findings on the properties of applying IF on NMT for instance-specific data filtering.

R1: Sensitivity of gradient similarity to the network components

In previous works, the influence, or called the gradient similarity, is usually computed with respect to a small part of the network parameters, especially the last or the last few layers (Han and Tsvetkov (2020);Barshan et al. (2020); *inter alia*). In NMT, we find that the resulting influence is highly sensitive to the network components used in computing the gradients (or gradient component). For illustration, we constructed a set of perturbed instances, computed its influence by different gradient components and observed their changes. The perturbed instances were not included during the NMT training. This independence between the NMT and the perturbed instances provides a simpler setting for checking how gradient components and the perturbed examples affect the influence.

Table 5.6 shows the gradient similarities of a probing example from newstest2017 with six artificially created instances. We use two NMT models, 1) trained with shared parameters between the two word embeddings and the output layer and 2) trained without parameter sharing, to compute the similarities.

We notice that gradient similarity for the model with shared parameters is more strongly influenced by lexical matches on the target side, as shown by the larger magnitude of influence values for probing examples 1 and 2 with random source sides compared to probing examples 3 and 4 with random target sides. For non-shared parameters, we observe that the gradient w.r.t. the output layer (∇_{output}) has stronger response (0.437 and 0.401) to the probing instances with random source side whereas

[6]It is tempting to just use the deployed checkpoint to compute the influence. As shown by Liang et al. 2017, however, the Hessian term in equation (5.1) captures more accurately the effect of model training than the dot product of the optimal checkpoint. In TracIn, the Hessian is approximated by the average over a set of checkpoints, and we follow their guidelines for checkpoints selection.

the gradient w.r.t. source embedding (∇_{srcEmb}) has stronger response (-0.149 and -0.120) to the instances with random target sides. On the same probing example, we repeat this random sampling of source and target sentences by using the other 3003 instances in the newstest2017 set. We find that the mean magnitude of ∇_{srcEmb} is 0.04 for random target whereas it is 0.004 for random source. In the case of ∇_{output}, the mean magnitude for random target is 0.021 whereas it is 0.428 for random source. This indicates that ∇_{output} has a tendency of scoring sentence pairs higher when their target side overlaps with the target side of the probing instance and is less influenced by source-side overlap. This may be suboptimal for retrieving problematic training examples that are relevant to a given probing instance.

When using a gradient vector ∇_{concat} which is the concatenation of ∇_{srcEmb}, ∇_{trgEmb} and ∇_{output}, its similarity is dominated by ∇_{output} rather than equally shared between the three given that they have the same number of parameters. This may explain why, in the case of shared parameters, instances with random source side have higher similarities than those with random target side.

Instance 5 and 6 are minor edits of the probing instance with changes to punctuation. For instance 5, it is not easy to interpret the results for the model with shared parameters. However, in the non-shared parameter setting, we observe a higher similarity for ∇_{srcEmb} than for ∇_{trgEmb} and ∇_{output}. This is more interpretable because the punctuation change is on the target side. For instance 6, the punctuation change is on the source side and we see a higher TracIn value for ∇_{output} than for ∇_{srcEmb} and ∇_{trgEmb}. As before, the value of ∇_{concat} is more similar to the value of ∇_{output}. Further examples can be found in Table 5.7.

These qualitative results show that the choice of network component is crucial in computing the gradient similarity. As shown in the next experiment, this affects the retrieval of training examples.

R2: Contrastive signal is crucial for better retrieval performance

In this section, we try to illustrate how different gradient components affect the retrieval of the noisy instances with TracIn. We add control to the retrieval outcome by adding synthetic noisy training instances to the training data. In addition, we show that vanilla IF may not be sufficient to achieve good performance because the gradients are aggregated over all tokens in the target sentence. We thus propose two contrastive methods to sharpen the gradient signal.

	Samples	∇_{Full}	∇_{Emb}	∇_{srcEmb}	∇_{trgEmb}	∇_{output}	∇_{concat}
Probing	Selbst die britische Queen hat ihn schon geadelt. Even the British Queen has bestowed an honour upon him.	1	1	1	1	1	1
1	Nur fehlten die Beweise. Even the British Queen has bestowed an honour upon him.	0.358	0.284	0.024	0.225	0.401	0.319
2	Biologen haben in Hannover untersucht, welchen Effekt das Rufen von Katzenbabys auf erwachsene Tiere hat. Even the British Queen has bestowed an honour upon him.	0.275	0.168	0.004	0.219	0.280	0.200
3	Selbst die britische Queen hat ihn schon geadelt. The German branch of the Gülen movement also fears that many Turks will flee abroad.	-0.035	-0.038	-0.125	0.025	-0.043	-0.036
4	Selbst die britische Queen hat ihn schon geadelt. Demonstrators demanding political change in Ethiopia have been met with violent resistance by the government.	-0.039	-0.013	-0.141	0.039	0.001	-0.003
5	Selbst die britische Queen hat ihn schon geadelt. Even the British Queen has bestowed an honour upon him!	0.962	0.924	0.992	0.981	0.905	0.924
6	Selbst die britische Queen hat ihn schon geadelt! Even the British Queen has bestowed an honour upon him.	0.908	0.899	0.912	0.949	0.935	0.935

Table 5.7: Another example showing the changes of gradient similarity by selected network components. Segments that are marked in red are perturbed from the probing example. The notation ∇_X indicates the network components used in computing the gradient similarity. ∇_{srcEmb} has a mean magnitude of 0.051 and 0.007 on random target and random source respectively whereas ∇_{output} has respectively a mean magnitude of 0.0145 and 0.350. This shows that ∇_{output} has a tendency of scoring sentence-pairs containing random source higher.

Synthetic noisy examples We used the error template $X \rightarrow Y$ which stands for *X is translated to Y* to construct synthetic noise examples for the training set. We created four simple error patterns: 1) *August* $\rightarrow$ *January*, 2) *Deutschland* $\rightarrow$ *Italy*, 3) *Oktober* $\rightarrow$ *December* and 4) *Türkei* $\rightarrow$ *New Zealand*.

In the training set, we replaced the translation of the sentences containing the source pattern by the erroneous translation with a probability of 60% so that the total number of training data is unchanged. We selected these error patterns because translation errors of months and country names can easily result from noisy training examples and are therefore suitable to simulate real customer issues. In addition, there are related source sentences in the test set, i.e., newstest2017, which can be used as probing examples. In order to speed up the computation of IF, we extracted a subset of training data containing the original pattern, the perturbed pattern and some

Error pattern	Number of instances		
	train	synthetic noisy	probing
August → *January*	8,017	925	9
Deutschland → *Italy*	15,360	4,891	30
Oktober → *December*	11,927	2,422	8
Türkei → *New Zealand*	14,963	7,417	22

Table 5.8: Number of instances per error pattern

randomly sampled training sentences. For example, in the error pattern *Oktober* → *December*, the training subset contains sentences with *Oktober*, *Dezember*, *October* and *December* on either the source or target side together with some randomly sampled sentences. Table 5.8 gives the exact number of instances for each case. We followed the same training procedure as section 3 to pre-train a NMT model on the training corpus perturbed by the synthetic noises.

Contrastive-IF The gradient of a source-target pair in NMT involves complicated mapping between the source tokens and the target tokens. That is, the gradient vector does not just contain the information of the error pattern but also other context. In order to isolate the gradient of the error pattern from the aggregated signal, we propose two methods: 1) gradient masking and 2) gradient difference. Both methods leverage a cleaner translation either in the form of a gold-reference translation or a corrected hypothesis, i.e. the hypothesis with the error pattern corrected. We refer to them as *Contrastive Influence Functions* (Contrastive-IF).

The idea of gradient masking (*Mask*) is to apply a 0/1 token-level mask to the loss function so as to remove the contribution of irrelevant tokens from the gradient computation. We assign the mask based on which tokens differ between hypothesis and reference. If the 0-mask is applied everywhere except for the location of the error according to a corrected translation, we refer to it as *MaskExact*. Table 5.9 illustrate the differences with error pattern "August → January".

We can use the difference between two hypotheses in a continuous fashion by simply subtracting their gradients. Specifically, we compute the difference of the gradient of a sentence A and the gradient of a sentence B as the probing gradient: $GD(A, B) = \nabla(A) - \nabla(B)$. In this work, we use the hypothesis as A and a cleaner translation as B (either the reference or the corrected hypothesis) so that positively influential training instances w.r.t. to $GD(A, B)$ are the synthetic noisy training instances.

Reference:	The	film	is	released	in	German	on	25	August	.
Hypothesis:	The	film	will	be	filmed	here	on	25	January	.
Mask:	0	0	1	1	1	1	0	0	1	0
MaskExact:	0	0	0	0	0	0	0	0	1	0
Corrected Hypothesis:	The	film	will	be	filmed	here	on	25	August	.
MaskExact:	0	0	0	0	0	0	0	0	1	0

Table 5.9: An illustration of gradient masking

Results Table 5.10 shows the retrieval performance of vanilla IF, gradient masking and gradient difference where the gradient is computed w.r.t. to either the source embedding, output layer or the full model. We evaluate the performance with precision over the top-X% influential training instances, i.e. the number of synthetic training instances successfully retrieved given top-X% of the influential training samples. We combine results of the four error patterns by (macro) averaging their precision.

The first three rows show results for vanilla IF (TracIn) when either the hypothesis, the reference or a corrected hypothesis is used for probing the training data. Using ∇_{srcEmb} or ∇_{output} obtain substantially higher precision for each variant than using ∇_{Full}, i.e., the gradient w.r.t. the entire model, which demonstrates the importance of the choice of gradient component(s) in vanilla-IF for retrieval performance. Using the corrected hypotheses to retrieve negatively-influential examples yields the best precision for both top-1% and top-10% of retrieved training examples.

We qualitatively examine the influential instances retrieved. By using the source-hypothesis pair as the probing instance, we find that instances retrieved via ∇_{output} have less similarity on the source side. In the first probing example, *Januar* $\rightarrow$ *January* occurs more frequently in the ranking than *August* $\rightarrow$*January*. In the second example, *Italien* $\rightarrow$ *Italy* appears as the third influential training instance when using ∇_{output} whereas all top-3 influential instances obtained by ∇_{srcEmb} contain the desired error pattern of *Deutschland* $\rightarrow$ *Italy*, see Table 5.11.

We find that both gradient masking, $\nabla(\text{HYP}_{\text{Mask}})$, and gradient difference, $\nabla(\text{HYP}) - \nabla(\text{REF})$, perform better than the vanilla IF given the same gradient component. $\nabla(\text{HYP}_{\text{Mask}})$ always outperforms the comparable vanilla IF variants $\nabla(\text{HYP})$ and $\nabla(\text{REF})$. If we can identify the exact location of the error pattern, with the probing gradient $\nabla(\text{HYP}_{\text{MaskExact}})$ or $\nabla(\text{CorrHYP}_{\text{MaskExact}})$, the precision can be further boosted and this is consistent for gradients ∇_{srcEmb}, ∇_{output} and ∇_{Full}. While the gradient difference variants do not always outperform the comparable masking variants for all ∇_X, $\nabla(\text{HYP}) - \nabla(\text{CorrHYP})$ yields the overall best result using ∇_{srcEmb}.

An interesting finding is the improvement brought by the corrected hypothesis (CorrHYP). Applying vanilla-IF on it already achieves a precision of 0.930 under ∇_{srcEmb} considering the top-1% influential instances. By applying *MaskExact* or gradient difference on it, we achieve very high precisions of 0.989 and 1.0 under ∇_{srcEmb} considering the top-1% influential training instances. One notable gain brought by the proposed approaches is that for ∇_{Full}, the precision increases from 0.531 to around 0.987 for the $\nabla(\text{HYP}) - \nabla(\text{CorrHYP})$ variant, bringing it on-par to the performance of ∇_{output}. We include results for additional gradient components in Table 5.12.

We also conducted a side experiment with a NMT model with shared parameters between the embeddings and the output layer. Similar to the case of a NMT model with non-shared parameters, gradient difference improves over the vanilla-IF when averaging precisions over all error patterns as shown in Table 5.13.

To summarize, both our contrastive-IF variants improve retrieval performance regardless of the network component used in computing gradients and whether the NMT model has shared parameters.

∇(Probing)	+/-	Precision		
		∇_{srcEmb}	∇_{output}	∇_{Full}
∇(HYP)	+	0.846	0.720	0.503
∇(REF)	-	0.876	0.794	0.481
∇(CorrHYP)	-	0.930	0.905	0.531
$\nabla(\text{HYP}_{\text{Mask}})$	+	0.893	0.840	0.654
$\nabla(\text{HYP}_{\text{MaskExact}})$	+	0.957	0.910	0.862
$\nabla(\text{CorrHYP}_{\text{MaskExact}})$	-	0.989	**0.992**	0.924
∇(HYP) - ∇(REF)	+	0.930	0.856	0.584
∇(HYP) - ∇(CorrHYP)	+	**1.000**	0.971	**0.987**

(a) Retrieval performance for top-1% influential training examples

∇(Probing)	+/-	Precision		
		∇_{srcEmb}	∇_{output}	∇_{Full}
∇(HYP)	+	0.765	0.644	0.442
∇(REF)	-	0.799	0.693	0.437
∇(CorrHYP)	-	0.844	0.781	0.455
$\nabla(\text{HYP}_{\text{Mask}})$	+	0.848	0.829	0.567
$\nabla(\text{HYP}_{\text{MaskExact}})$	+	0.936	0.904	0.825
$\nabla(\text{CorrHYP}_{\text{MaskExact}})$	-	0.962	**0.958**	0.875
∇(HYP) - ∇(REF)	+	0.855	0.764	0.515
∇(HYP) - ∇(CorrHYP)	+	**0.986**	0.935	**0.931**

(b) Retrieval performance for top-10% influential training examples

Table 5.10: Retrieval performance measured in (macro) averaged precision over all error patterns. $\nabla(Probing)$ refers to the gradient with input 'source-Probing'. HYP, REF and CorrHYP stands for hypothesis, reference and corrected hypothesis respectively. "+" ("-") indicates that positively (negatively) influential training instances were retrieved. ∇_X indicates network components used in computing the gradient. We mark the best result per column in bold.

probing	1	Der Film läuft bei uns ab dem 25. August. The film will be filmed here on 25 January.
∇_{srcEmb}	1	Die Vereinbarung läuft am 31. Januar ab. This agreement formally expires on 31 January.
	2	Dieses Gesetz wurde im August unterzeichnet. It was signed in January.
	3	Die Vereinigten Staaten haben diese Garantie am 15. August 1971 aufgegeben. The United States abandoned that guarantee on 15 January 1971.
∇_{output}	1	Der Cardiff-Bericht erscheint Mitte Januar. The Cardiff report will be published in mid-January.
	2	Eine zweite Tagung ist für Januar 2004 vorgesehen. A second meeting will be held in January 2004.
	3	Ich hoffe, dass die Dynamik beibehalten und das Siebte Rahmenprogramm am 1. Januar 2007 auf den Weg gebracht wird. I hope that the momentum will be maintained and the Seventh Framework Programme will be launched on 1 January 2007.
probing	2	Auch in Deutschland finde eine "Hexenjagd" gegen Erdogan-Kritiker statt. A 'witch hunt' against Erdogan critics is also taking place in Italy.
∇_{srcEmb}	1	Deutschland ist dagegen. Italy is opposed to this.
	2	Dies wäre ein besseres Wirtschaftsmodell für Deutschland. This would be a better economic model for Italy.
	3	Deutschland und China können mehr tun als andere. Italy and China can do more than others.
∇_{output}	1	Eine weitere Lehre für Sarkozy aus Deutschland ist, dass ein aufgeklärter korporatistischer Staat unterstützender politischer Führung ebenso bedarf wie entgegenkommender Gewerkschaften. A further lesson for Sarkozy from Italy is that an enlightened corporate state needs supportive political leadership as well as accommodating trade unions.
	2	Insgesamt wurden fast 2 300 Tonnen möglicherweise kontaminiertes Futtermittelfett an 25 Futtermittelhersteller in Deutschland geliefert. A total of almost 2 300 tonnes of potentially contaminated feed fat was delivered to 25 feed manufacturers in Italy.
	3	Leider Gottes ist der Titel der heutigen Debatte Italien. Alas, the title of today's debate is Italy.

Table 5.11: Two probing examples with source-hypothesis as input and their top-3 positively influential training instances. ∇_{output} has a tendency to assign higher scores to sentence-pairs which target side has overlapped tokens but ignoring the similarity of the source side. For example, the pattern "*Januar -> January*" occurs more frequently in the ranking than "*August -> January*" in probing 1.

∇(Probing)	+/-	Precision ∇_{srcEmb}	$\nabla_{encoder}$	∇_{trgEmb}	∇_{output}	∇_{concat}	∇_{Full}
∇(HYP)	+	0.846	0.485	0.334	0.720	0.722	0.503
∇(REF)	-	0.876	0.432	0.303	0.794	0.805	0.481
∇(CorrHYP)	-	0.930	0.494	0.324	0.905	0.919	0.531
$\nabla(\text{HYP}_{\text{Mask}})$	+	0.893	0.581	0.347	0.840	0.844	0.654
$\nabla(\text{HYP}_{\text{MaskExact}})$	+	0.957	0.862	**0.474**	0.910	0.916	0.862
$\nabla(\text{CorrHYP}_{\text{MaskExact}})$	-	0.989	0.903	0.467	**0.992**	**0.994**	0.924
∇(HYP) - ∇(REF)	+	0.930	0.523	0.321	0.856	0.855	0.584
∇(HYP) - ∇(CorrHYP)	+	**1.000**	**0.985**	0.458	0.971	0.980	**0.987**

(a) Retrieval performance for top-1% influential training examples

∇(Probing)	+/-	Precision ∇_{srcEmb}	$\nabla_{encoder}$	∇_{trgEmb}	∇_{output}	∇_{concat}	∇_{Full}
∇(HYP)	+	0.765	0.399	0.301	0.644	0.646	0.442
∇(REF)	-	0.799	0.382	0.297	0.693	0.700	0.437
∇(CorrHYP)	-	0.844	0.402	0.299	0.781	0.789	0.455
$\nabla(\text{HYP}_{\text{Mask}})$	+	0.848	0.478	0.311	0.829	0.831	0.567
$\nabla(\text{HYP}_{\text{MaskExact}})$	+	0.936	0.794	**0.380**	0.904	0.908	0.825
$\nabla(\text{CorrHYP}_{\text{MaskExact}})$	-	0.962	0.821	0.372	**0.958**	**0.960**	0.875
∇(HYP) - ∇(REF)	+	0.855	0.442	0.307	0.764	0.765	0.515
∇(HYP) - ∇(CorrHYP)	+	**0.986**	**0.884**	0.371	0.935	0.939	**0.931**

(b) Retrieval performance for top-10% influential training examples

Table 5.12: Retrieval performance measured in (macro) averaged precision over all error patterns (extended version of Table 5.10). $\nabla(Probing)$ refers to the gradient with input 'source-Probing'. HYP, REF and CorrHYP stands for hypothesis, reference and corrected hypothesis respectively. "+" ("-") indicates that positively (negatively) influential training instances were retrieved. ∇_X indicates network components used in computing the gradient, ∇_{concat} indicates concatenation of ∇_{srcEmb}, ∇_{trgEmb} and ∇_{output}. We mark the best result per column in bold.

∇(Probing)	top-X% influential training samples	+/-	Precision ∇_{Emb}	Precision ∇_{Full}
∇(HYP)	1%	+	0.660	0.502
	10%		0.596	0.444
∇(CorrHYP)	1%	-	0.877	0.541
	10%		0.746	0.463
∇(HYP) - ∇(CorrHYP)	1%	+	0.891	0.691
	10%		0.808	0.607

Table 5.13: Retrieval performance measured in averaged precision across all error patterns for an NMT model with *shared* parameters between the word embeddings and the output layer.

∇(Probing)	+/-	Precision		
		∇_{srcEmb}	$\nabla_{encoder}$	∇_{Full}
∇(HYP)	+	0.930	0.972	0.994
∇(REF)	-	0.525	0.452	0.548
∇(HYP) - ∇(REF)	+	0.708	0.712	0.949

(a) Retrieval performance for top-10% influential training examples

∇(Probing)	+/-	Precision		
		∇_{srcEmb}	$\nabla_{encoder}$	∇_{Full}
∇(HYP)	+	0.888	0.932	0.986
∇(REF)	-	0.508	0.449	0.504
∇(HYP) - ∇(REF)	+	0.670	0.647	0.895

(b) Retrieval performance for top-20% influential training examples

Table 5.14: Retrieval performance measured in averaged precision over the probing instances, on copied training instances. $\nabla(Probing)$ refers to the gradient with input 'source-Probing'. HYP, REF stands for hypothesis, reference. "+" ("-") indicates that positively (negatively) influential training instances were retrieved. ∇_X indicates the network components used in computing the gradient.

R3: Copied source sentences have similar gradient signature

Our initial motivation for applying influence functions to NMT was to arrive at a more automatable way of retrieving relevant training examples for reported translation problems. We were also hoping to generalize over what can be achieved by applying manually composed regular expressions which are limited to detecting lexical overlap. In this section, we focus on the latter and investigate whether Influence Functions can retrieve training examples that cause an undesired copy behaviour in the decoder.

Experimental settings On top of the Europarl-v7 and News Commentary-v12 data, we append a set of 176,004 copied source sentences provided by Khayrallah and Koehn (2018) to the training set. Following the training recipe in section 3, our NMT with non-shared parameters has a degradation of translation quality from 29.99 BLEU to 17.64 BLEU on the newstest2017 data, showing the detrimental effect of the untranslated target sides.

We selected 40 probing instances from the newstest2017 data where their translation by the above NMT model is a copy of the source sentence. We again reduced the computation time by running TracIn over a training subset which contains the newly added noisy data, i.e., 176,004 instances and a set of randomly sampled training instances. This created a training subset of 476,004 instances.

probing	1	Golfer Langer erhält die Sportpyramide Golfer Langer erhält die Sportpyramide
∇_{srcEmb}	1	Binnenmarktanzeiger Binnenmarktanzeiger
	2	Vollständige Liste der ausgewählten Aussteller: Vollständige Liste der ausgewählten Aussteller:
	3	Dimiter TZANTCHEV Ständiger Vertreter Dimiter TZANTCHEV Ständiger Vertreter
∇_{Full}	1	Erstellung einzelstaatlicher Aktionspläne für die Verhütung von Verletzungen durch die Mitgliedstaaten. Erstellung einzelstaatlicher Aktionspläne für die Verhütung von Verletzungen durch die Mitgliedstaaten.
	2	Für weitere Informationen wenden Sie sich bitte an die Dienststelle Außenbeziehungen Europäischer Rechnungshof Für weitere Informationen wenden Sie sich bitte an die Dienststelle Außenbeziehungen Europäischer Rechnungshof
	3	Dimiter TZANTCHEV Ständiger Vertreter Dimiter TZANTCHEV Ständiger Vertreter
probing	2	Die demokratische Bewerberin kündigt gar die größte Investition in neue Arbeitsplätze seit dem Zweiten Weltkrieg an. Die demokratische Bewerberin kündigt gar die größte Investition in neue Arbeitsplätze seit dem Zweiten Weltkrieg an.
∇_{srcEmb}	1	Die Krise hat die großen Unterschiede innerhalb der EU deutlich gemacht. Die Krise hat die großen Unterschiede innerhalb der EU deutlich gemacht.
	2	Die Regierungskonferenz ist nur eine Versammlung aller Regierungen. Die Regierungskonferenz ist nur eine Versammlung aller Regierungen.
	3	Die Entschließung wird uns dabei helfen, auf einer soliden Grundlage in die nächste Phase der Entwicklung einer Meeresstrategie einzutreten. Die Entschließung wird uns dabei helfen, auf einer soliden Grundlage in die nächste Phase der Entwicklung einer Meeresstrategie einzutreten.
∇_{Full}	1	Die Partei für Freiheit möchte dafür sorgen, dass die niederländische Öffentlichkeit nicht länger als Geldautomat Europas behandelt wird. Die Partei für Freiheit möchte dafür sorgen, dass die niederländische Öffentlichkeit nicht länger als Geldautomat Europas behandelt wird.
	2	Die russische Regierung hat geschätzt, dass ein Drittel aller Wasserleitungen dringend ersetzt werden muss. Die russische Regierung hat geschätzt, dass ein Drittel aller Wasserleitungen dringend ersetzt werden muss.
	3	Die internationale Gemeinschaft erkannte ihn einstimmig an. Die internationale Gemeinschaft erkannte ihn einstimmig an.

Table 5.15: Two probing examples with copied training instances as input and their top-3 positively influential training instances. Both ∇_{srcEmb} and ∇_{Full} can retrieve copied instances in the training subset given a probing instance of copied source sentence which is lexically different.

Results Table 5.14 shows the retrieval performance on copied source sentences in the training subset with probing gradients of ∇(HYP), ∇(REF) and ∇(HYP) - ∇(REF) computed over source embedding (∇_{srcEmb}), the encoder ($\nabla_{encoder}$), or the entire model (∇_{Full}). We skip the masking strategy in this case since it would mask all target tokens, resulting in a loss of 0. Different from our results so far, the vanilla IF using only the hypothesis preforms better than using the reference for retrieval and better than the gradient difference variant for all network components. For example, when considering only the top-10% influential training instances, the precision is 0.930 for ∇(HYP) with ∇_{srcEmb} and only 0.525 for ∇(REF). This may indicate that instances of copied source sentence have similar gradient signature despite their lexical difference (see Table 5.15 for some examples) and that the reference translation is less useful in this setting because it cannot provide a specific contrastive signal.

A surprising finding in this setting is that using gradients computed over the entire network is better than the source embedding or the entire encoder. This is in contrast

Error pattern	∇(HYP) - ∇(CorrHYP)		∇(HYP)	
	∇_{srcEmb}	∇_{Full}	∇_{srcEmb}	∇_{Full}
August → *January*	0.399 ± 0.104	0.199 ± 0.041	0.059 ± 0.023	0.119 ± 0.042
Oktober → *December*	0.524 ± 0.192	0.397 ± 0.123	0.056 ± 0.028	0.143 ± 0.043
Deutschland → *Italy*	0.576 ± 0.126	0.428 ± 0.047	0.097 ± 0.061	0.135 ± 0.046
Türkei → *New Zealand*	0.527 ± 0.100	0.540 ± 0.118	0.080 ± 0.044	0.165 ± 0.051

Table 5.16: Statistics showing the mean and standard deviation of the largest influence per configuration. The large standard deviation of the maximum influence value for probing examples of the same error pattern shows the difficulty of defining a comparable filtering threshold across probing instances.

to the previous findings in the synthetic training instances. This possibly indicates that the copy mechanism is spread over the entire model or parts beyond the source embedding or the encoder.

R4: An effective IF-based instance-specific data filtering is hard to automate

Many data filtering algorithms require a threshold to decide which instances are to be filtered. This threshold can be a model score in an offline filtering algorithm (Junczys-Dowmunt, 2018) or a dynamic formula that is changed according to the learning state of the model (Wang et al., 2018a). In both cases, a desirable threshold should be effective as measured in the downstream model performance and be easily computed and generalized to other situations. In the case of IF-based instance-specific data filtering, we observe two properties in the ranking of the influence which makes the automation of the data filtering algorithm challenging.

1: The range of influence varies across probing examples Although the influence is bounded between $[-1, 1]$ because of the cosine similarity, the maximum magnitude of the influence for each probing example can still be very different. Table 5.16 shows the mean and standard deviation of the maximum influence value of positively influential training instances computed over probing examples of the same configuration. Firstly, the mean value is quite diverse across different gradient components, and across different probing gradients of the same error pattern. For example, the mean value of the error pattern *August* → *January* computed with ∇_{srcEmb} is 0.399 or 0.059 depending on which probing gradient is used. Secondly, the standard deviation within each configuration is relatively large when compared to the corresponding mean value. For example, it is about 26%, 36%, 22% and 19% in the case of ∇_{srcEmb} using gradient difference as the probing gradient. This large standard

Error pattern	∇(HYP) - ∇(CorrHYP)		∇(HYP)	
	∇_{srcEmb}	∇_{Full}	∇_{srcEmb}	∇_{Full}
August → January	1.44 ± 0.50	3.33 ± 1.76	1.78 ± 1.55	1.44 ± 0.69
Oktober → December	2.25 ± 0.43	2.00 ± 0.00	2.88 ± 1.76	2.00 ± 1.58
Deutschland → Italy	1.00 ± 0.00	1.77 ± 0.62	1.67 ± 1.22	2.70 ± 2.62
Turkei → New Zealand	3.05 ± 1.46	1.32 ± 1.26	2.27 ± 2.09	2.32 ± 1.66

Table 5.17: Mean and standard deviation of the number of influential training instances to be removed per configuration, using the largest consecutive difference found in the ranking as clustering criterion.

deviation indicates the difficulty of setting an effective threshold for filtering even for probing examples with the same type of error pattern.

2: The influence value drops abruptly at the top-of the ranking Apart from a fixed threshold across different probing example, we also examine the possibility of automatically setting a threshold for each probing example.

We first examined a simple clustering strategy by searching for the position where the consecutive difference is the largest in the ranking of influence. Table 5.17 shows the result of the mean and standard deviation of the number of most influential training instances to be removed per configuration. By considering only the largest consecutive difference, less than 5 training instances would be removed which is far less than the number of synthetic training instances.

We examined further by investigating the shape of the influence of the positively influential training instances in the ranking. Figure 5.3 shows the influences, computed via TracIn, of the top-500 positively influential training instances per error pattern. For each error pattern, we randomly select a probing example to examine its influence under different gradient conditions. In all these cases, the influence drops sharply in the first few instances, especially in the case of vanilla IF, denoted by "GradHYP" in the figures. After the sharp drop, the influence becomes quite steady for the remaining instances. This steady behaviour holds even for instances of much lower rank, see Figure 5.4. The "elbow" occurs before the first 50 influential training instances, which includes only a tiny portion of the synthetic noisy training instances.

How about Top-K filtering? In previous work, the authors have used either Top-K or Top-X% as the filtering threshold which is not realistic for NMT where 1) there can be billions of training instances, and 2) the error types are more diverse than the prediction of wrong classes. In spite of the good retrieval performance demonstrated in the previous section, our results here show that an effective automation of the

IF-based instance-specific data filtering for NMT remains a challenge.

5.3.4 Summary

We have analyzed the use of Influence Functions for NMT as instance-specific data filtering. By constructing synthetic instances, we find that 1) the gradient similarity is very sensitive to the selected network components, 2) vanilla Influence Functions are not sufficient for good retrieval performance, 3) our proposed contrastive-IF can boost the retrieval performance regardless of the gradient component or parameter sharing, 4) finding an effective automation of IF for instance-specific data filtering is difficult. This is because the proper choice of gradient component with respect to the type of error in the probing example is crucial for the effectiveness of Influence Functions. Despite the reported effectiveness for certain classification tasks in previous literature, our results show that applying IF to NMT poses some practical difficulties that we have not yet been able to solve.

5.3.5 Limitations

In this work, we provide an analysis of using Influence Functions for Neural Machine Translation as instance-specific data filtering for the purpose of cost saving and finding a more generally applicable solution. Despite the reported success of some previous works in NLP/Vision-related classification tasks, we face several challenges in applying Influence Functions to NMT. We are aware of the following limitations to our analysis:

- Our analysis focuses on TracIn rather than other influence functions because TracIn is reported to be very effective.
- Our analysis is based on a fixed set of checkpoints, following the practice of previous works. The selection and the number of checkpoints used in TracIn are computationally costly hyper-parameters.
- Our analysis focuses on major network components such as embeddings, encoder and the output layer, excluding other possible combinations.
- The scale of our experiments is limited, e.g., only the De-En language direction with 3M training instances and the synthetic examples are relatively simple. However, given such simple setting, we can already see the challenges of applying IF on NMT as instance-specific data filtering or as an attribution/interpretable method.

- The proposed contrastive IF requires a corrected translation, e.g., reference translation.

We hope that our analysis can inspire further evaluation and modification of the technique.

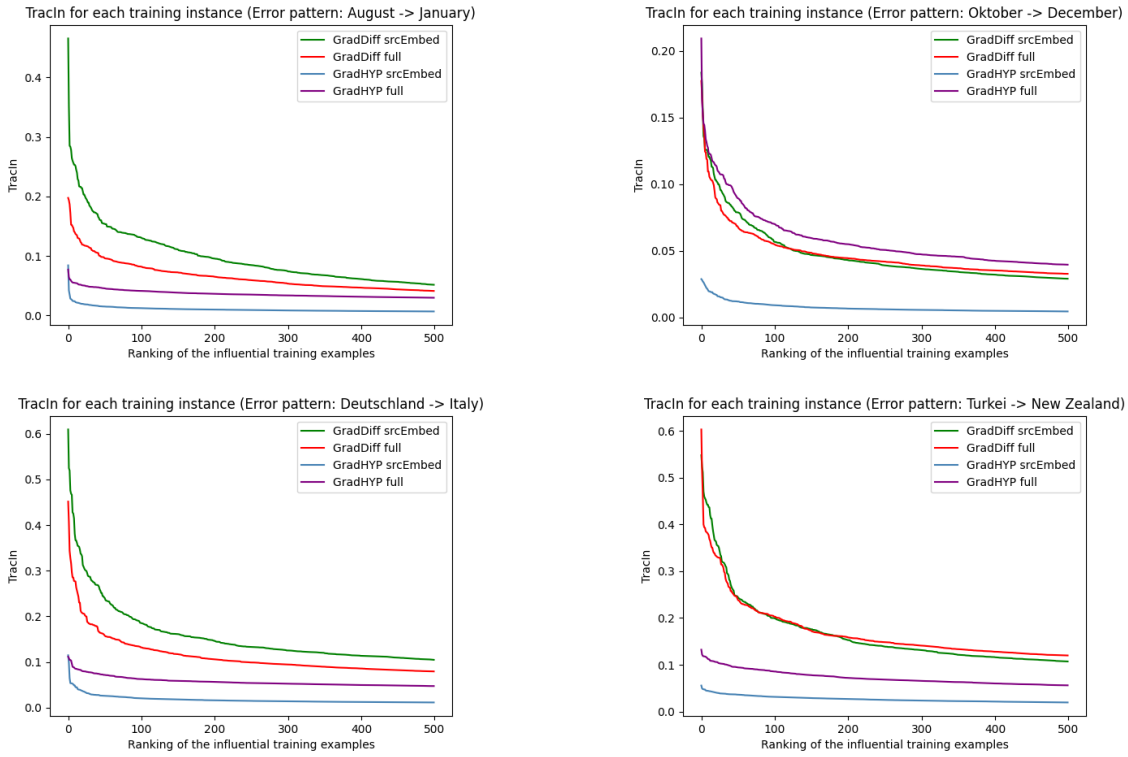

Figure 5.3: TracIn of the top-500 positively influential training examples. In each subfigure, we randomly select a probing example from each error pattern to compute its influence using gradient difference w.r.t. 1) source embedding (GradDiff srcEmbed) & 2) entire model (GradDiff full) and using vanilla-IF with source-hypothesis as input w.r.t. 1) source embedding (GradHYP srcEmbed) & 2) entire model (GradHYP full).

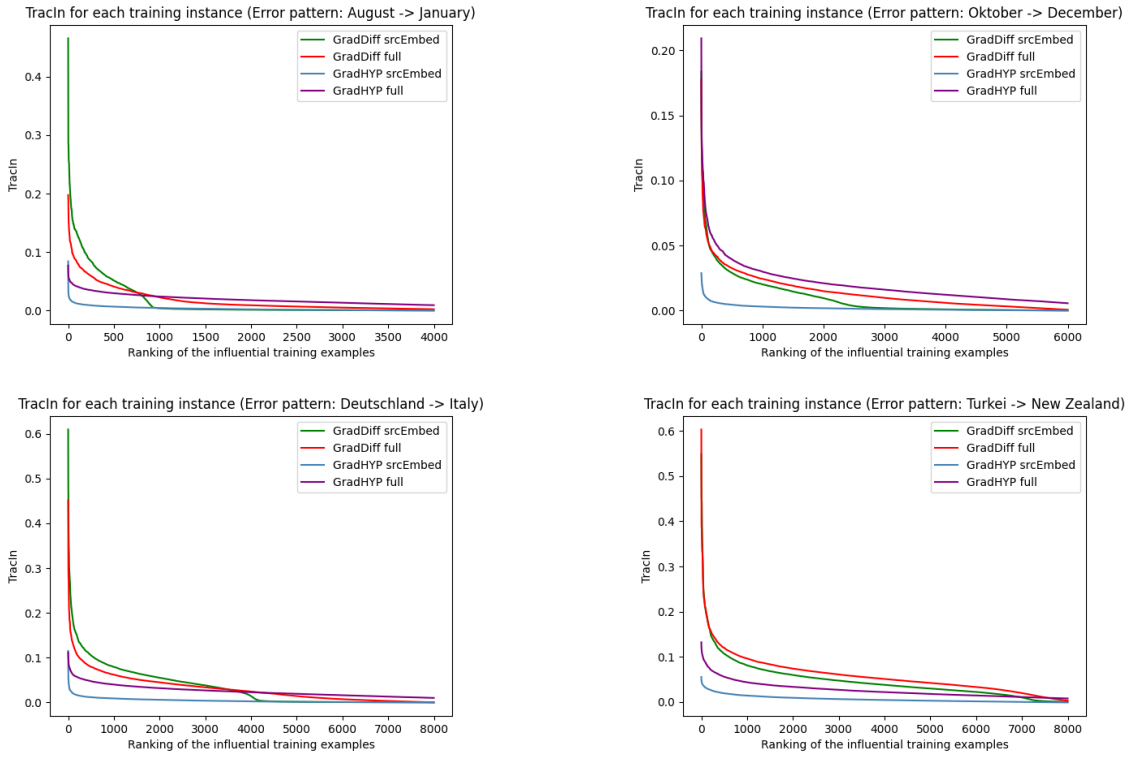

Figure 5.4: TracIn of the top-50% positively influential training examples. In each subfigure, we randomly select a probing example from each error pattern to compute its influence using gradient difference w.r.t. 1) source embedding (GradDiff srcEmbed), and 2) entire model (GradDiff full) as well as using vanilla-IF with source-hypothesis as input w.r.t. 1) source embedding (GradHYP srcEmbed), and 2) entire model (GradHYP full).

5.4 Chapter Summary

In this chapter, we present instance-specific data selection. We illustrate this idea via selection of pseudo-labels for training cascade speech translation and via Influence Functions for interpreting relevant training examples on a model's prediction. In the scenario of cyclic feedback, it is *instance-specific* because we select pairs of pseudo-labels which maximizes the translation performance of a given speech-translation training instance. In domain adaptation experiments on CoVoST and LibriVoxDeEn, we show that cyclic feedback on a cascade ASTT improves over the baselines in spite of the shortage of gold-reference transcriptions. In the second scenario, we call the use of Influence Functions for back-tracking model's prediction as *instance-specific*. It is because the training data selection is with respect to the probing instance instead of the general training data quality. This, in principle, helps to spot or interpret the "nuanced error", such as the mis-translation of country names or dates as reported by the customers. However, by analyzing Influence Functions on synthetically created examples, we find several properties which make its use in practise challenging. In particular, it is computationally costly to obtain the per-instance information.

Future works may to investigate better approximations to lower the computational complexities of this algorithms.

Chapter 6

Interactive Data Correction

In the previous two chapters, we have presented algorithms for adjusting the amount of effective training data from a purely algorithmic manner. In this chapter, we focus on data creation by means of interactive correction, i.e., human-in-the-loop, and present interactive learning protocols for neural machine translation. We examine feedback modes, simulated by gold-reference translations, in form of scoring under different granularity and of lexical constraint. We show that simple feedback on uncertain locations lead to a competitive model to the one trained on gold-reference translations.

Materials in this chapter have been drawn from two publications: EAMT 2018 (Lam et al., 2018) and MT Summit 2019 (Lam et al., 2019).

6.1 Introduction and Overview

Both within-corpus data augmentation and instance-specific data selection are purely algorithmic approaches for relieving the data scarcity problem. An obvious advantage of purely algorithmic approaches are their automation, i.e., without the need of human annotators. However, as discussed in instance-specific data selection, some nuanced but critical translation errors, such as the mis-translation of country names and dates, cannot be easily detected by algorithms alone.

In commercial MT system, the cost of expert annotation can be reduced via post-editing that shifts the task of translation from scratch to modification of the machine translated outputs. In spite of this, annotators may have to re-translate everything from scratch because of the domain gap between the pre-trained translation model and the new data. In addition, annotators need to correct and re-translate similar mistakes multiple times because of the lack of model adaptation or model personalization. This thus calls the need of learning algorithms which can efficiently

incorporate human feedback.

Interactive-predictive machine translation[1] (IPMT) is such a paradigm that allows interaction between the human annotator and the MT system. IPMT goes back to early approaches for IBM-type (Foster et al., 1997, 2002) and phrase-based machine translation (Barrachina et al., 2008; Green et al., 2014). Knowles and Koehn (2016) and Wuebker et al. (2016) presented neural interactive translation prediction — a translation scenario where translators interact with an NMT system by accepting or correcting subsequent target tokens suggested by the NMT system in an auto-complete style. However, in their work the system parameters are not updated based on the prefix. This idea is implemented in Turchi et al. (2017), Michel and Neubig (2018), Wuebker et al. (2018), Karimova et al. (2018), or Peris et al. (2017). Notably, these approaches use *complete post-edited sentences* to update their system.

Another closely related approach is interactive pre-post-editing (Marie and Max, 2015; Domingo et al., 2017). The core idea is to ask the translator to *mark good segments and use these for a more informed re-decoding.*

IPMT can also be combined with active learning (AL) to further reduce annotation effort by focusing on informative training instances. González-Rubio et al. (2011; 2012) applied AL for interactive machine translation, where a user interactively finishes translations of a statistical MT system. Their AL component decides which sentences to sample for translation and receive supervision for, and the MT system is updated on-line (Ortiz-Martínez et al., 2010).

In this chapter, we present our interactive learning protocols for NMT which aims to further reduce human effort in full post-editing. A key component of our protocol is to reduce annotation effort by using scoring rather than correction or a combination of correction and scoring as the feedback mode. Our protocols also used AL to decide which prefixes, rather than the entire sentence, to receive feedback for based on the entropy of the policy distribution. Such sub-sequence feedback helps to correct error occurring at the early parts of the sentence while minimizing the number of feedback requested. Furthermore, the protocol applies online update in sub-sequence level, resulting in faster and more accurate model adaptation or personalization.

In section 6.2, we present Bandit Interactive-Predictive NMT (BIP-NMT) which replaces post-editing entirely by scoring on sub-sequences. In section 6.3, we present another protocol which also integrates the substitution feedback.

[1]or called interactive translation prediction, interactive machine translation, text prediction or target-text mediated interactive translation prediction (Knowles et al., 2019)

6.2 Bandit Interactive-Predictive Neural Machine Translation

Our first attempt for human effort reduction is to use rating, or called scoring, in place of post-editing in an interactive-predictive manner. The scoring feedback combined with gradient descent in training neural network can be connected to policy gradient in reinforcement learning (RL). Such training helps to reinforce/penalize a targeted set of actions. Kreutzer et al. (2018a) presented an approach were ratings from human users on full translations are used successfully for NMT domain adaptation. Simulations of NMT systems interacting with human feedback have been presented firstly by Kreutzer et al. (2017), Nguyen et al. (2017), or Bahdanau et al. (2017), who apply different policy gradient algorithms, REINFORCE (Williams, 1992) or advantage-actor-critic methods (Mnih et al., 2016), respectively. However, their scoring and model update are executed only after the generation of the full sentence, resulting in noisy feedback, especially on long output sentences. Furthermore, their systems are not interactive-predictive, ignoring the benefit of faster adaptation.

In this paper, we use advantage-actor-critic update strategies for simulated bandit feedback on the sub-sentence level in an interactive-predictive setting. Because of the integration of bandit learning and interactive-predictive NMT, the proposed protocol is called *Bandit Interactive-Predictive Neural Machine Translation* (BIP-NMT). More specifically, BIP-NMT aims

1. to obtain a MT system which requires less human effort than post-editing.
2. to obtain user-adapted translations before post-editing.
3. to obtain a MT system with faster adaptability.

Our approach is based on combining a set of existing learning methods, e.g., RL and interactive-predictive learning, together with a set of developed tools, e.g. prefix buffer and entropy baseline. In domain adaptation using simulated human feedback, BIP-NMT shows better results, measured in both chrF and BLEU, than a sentence-level feedback system based on advantage actor-critic algorithm. In the followings, we present the essential components of BIP-NMT that help to reach the stated objectives.

6.2.1 Reduction of human effort via RL and AL

Replacement of post-editing by a score Instead of correction on the suggested hypotheses, BIP-NMT follows the idea from RL which only requires user ratings. This

score, usually normalised to a range of [0, 1], reflects the judgement of users on the quality of translations. It becomes the weight, or called advantage after subtraction of a baseline, for each token generated. Policy gradient based algorithm then updates the system according to this scored trajectory, see equation (2.21).

Assistance from human when the system is uncertain In principle, we can increase the resolution of feedback from sentence level to word level, that is, an individual score for each sampled token. In the case of interactive-predictive learning, we can even request feedback immediately after generating a single token. In both cases, however, the human effort is still not significantly reduced because of the large number of feedback per sentence required.

In order to optimize the number of feedback, BIP-NMT leverages AL to explore and request feedback when it is uncertain about the quality of its partial translations, or called prefix. Once an input is given, our BIP-NMT keeps exploring its action space by generating prefixes of increasing length using multinomial sampling until <eos> is captured. For each n-word prefix generated, BIP-NMT computes its average entropy per word $\bar{H}$ and record its cumulative average γ over previous prefixes.

$$\bar{H}(\hat{\mathbf{y}}_{\mathbf{1:n}}) = \frac{1}{n}\sum_{t=1}^{n}\left[-\sum_{v\in\mathcal{V}} p_\theta(v|s_t)\log p_\theta(v|s_t)\right], \tag{6.1}$$

where $\hat{\mathbf{y}}_{\mathbf{1:n}} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n\}$ is a sequence of n predicted tokens, $\mathcal{V}$ is the output vocabulary, and $p_\theta(v|s_t)$ is the probability of predicting a word in $\mathcal{V}$ at state s_t of the autoregressive decoder. The uncertainty of a partial translation is then quantified by a comparison of $\bar{H}$ and γ.

$$\bar{H}(\hat{\mathbf{y}}_{\mathbf{1:t}}) - \gamma_{t-1} \geq \epsilon \times \gamma_{t-1}, \tag{6.2}$$

where ϵ is a hyper-parameter, tuned on the validation set, which controls the triggering of feedback request. The higher its value, the higher is system's tolerance to its uncertainty about the current partial translations; this also implies a smaller number of requests per sentence on average. If equation (6.2) is 0 or positive, the prefix is presented to the user for feedback.

Multinomial sampling serves two purposes here. On one hand, it is used for exploration of the action space. On the other hand, it represents re-decoding of the partial translations in case of a bad feedback. If the feedback is positive, BIP-NMT stores the prefix in a Prefix Buffer Ξ.

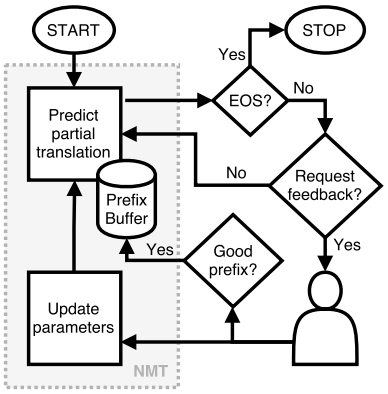

Figure 6.1: Human-system interaction in BIP-NMT.

6.2.2 Partial feedback and prefix buffer

The human-system interaction in BIP-NMT is strengthened by partial feedback & prefix buffer mechanism. Partial feedback refers to feedback on the partial translations. A prefix buffer Ξ is used for storing partial translations that is rated *good* by the user. The good prefix stored in Ξ is used for generation of its suffix via forced decoding. This property allows BIP-NMT to deliver translations that suits users needs. Feedback of the good prefix is also saved so that, in principle, only feedback on the suffix is sufficient for the model update. In our current design, the capacity of Ξ is one prefix for each input, and the stored prefix will be replaced by another good prefix found in latter stage of the interaction. However, Ξ can be extended to store multiple prefixes, especially in case of long input sentence.

6.2.3 Online updates offer faster adaptability

When BIP-NMT receives feedback, it updates itself immediately. This makes BIP-NMT improving itself through each interaction, i.e., partial translations, with the user, and offers faster adaptability than both off-line system and sentence-level based online system.

6.2.4 Algorithm

Figure 6.1 visualizes the interaction of the NMT system with a human for requesting and processing feedback for a single translation: Feedback is requested when the model is uncertain. It is then directly used for a model update and, in case it was good, for filling the prefix buffer, before the model moves to generating the next (longer) partial

Algorithm 3: Bandit Interactive-Predictive Neural Machine Translation

1: **Input:** θ_0, ϕ_0, α_A, α_C, ϵ
2: **Output:** Estimates θ^*, ϕ^*
3: $k \leftarrow 1$
4: **for** i $\leftarrow 1, \ldots$ N **do**
5: Receive $\mathbf{x}_i$, Initialize $\gamma_0 \leftarrow 0$, $\Xi \leftarrow \emptyset$
6: **for** t $\leftarrow 1 \ldots T_{\max}$ **do**
7: Sample $\hat{y}_t \sim p_{\theta_{k-1}}(\cdot|\mathbf{x}_i, \hat{\mathbf{y}}_{<t}, \Xi)$
8: Compute $\bar{H}(\hat{\mathbf{y}}_{\mathbf{1:t}})$
9: **if** $\bar{H}(\hat{\mathbf{y}}_{\mathbf{1:t}}) - \gamma_{t-1} \geq \epsilon \cdot \gamma_{t-1}$ or <eos> in $\hat{\mathbf{y}}_{\mathbf{1:t}}$ **then**
10: Receive feedback R($\hat{\mathbf{y}}_{\mathbf{1:t}}$)
11: **if** R($\hat{\mathbf{y}}_{\mathbf{1:t}}$) $\geq \mu$ **then**
12: $\Xi \leftarrow \hat{\mathbf{y}}_{\mathbf{1:t}}$
13: **end if**
14: Update $\theta_k \leftarrow \theta_{k-1} - \alpha_A \nabla L_{\theta_{k-1}}(\hat{\mathbf{y}}_{\mathbf{1:t}})$
15: Update $\phi_k \leftarrow \phi_{k-1} - \alpha_C \nabla L_{\phi_{k-1}}(\hat{\mathbf{y}}_{\mathbf{1:t}})$
16: $k \leftarrow k + 1$
17: **end if**
18: Update $\gamma_t \leftarrow \gamma_{t-1} + \frac{1}{t}\left(\bar{H}(\hat{\mathbf{y}}_{\mathbf{1:t}}) - \gamma_{t-1}\right)$
19: **break** if <eos> in $\hat{\mathbf{y}}_{\mathbf{1:t}}$
20: **end for**
21: **end for**

translation.

Algorithm 3 presents pseudo-code of BIP-NMT. The algorithm receives an input source sequence $\mathbf{x_i}$ (line 5), and incrementally predicts a sequence of output target tokens up to length $T_{\max}$ (line 6). At each step t, we generate a partial translation $\hat{\mathbf{y}}_{\mathbf{1:t}}$ by sampling a new target token $\hat{y}_t$ from the policy distribution $p_\theta(\cdot|\mathbf{x_i}, \mathbf{y}_{<t}, \Xi)$ that implements an auto-regressive encoder-decoder with an additional prefix buffer Ξ for forced decoding (line 7). User feedback is requested only when the average entropy $\bar{H}(\hat{\mathbf{y}}_{\mathbf{1:t}})$ of the policy is larger than or equal to a running average by a factor of ϵ (line 9). If the reward R($\hat{\mathbf{y}}_{\mathbf{1:t}}$) is larger than or equal to a threshold μ, the prefix is stored in a buffer for forced decoding (lines 11-12). Next, updates of the parameters of the policy (line 14), critic (line 15), and average entropy (line 18) are performed. Actor and critic each use a separate learning rate schedule (α_A and α_C).

Dataset	EP (v.5)	$\bar{n}$	NC (WMT07)	$\bar{n}$
Training (filt.)	1,346,679	23.5	9,216	21.9
Validation	2,000	29.4	1,064	24.1
Test	-	-	2,007	24.8

Table 6.1: Number of parallel sentences and average number of words per sentence in target language (en), denoted by $\bar{n}$, for training (filtered to a maximum length of 50), validation and test sets for French-to-English translation for Europarl (EP) and News Commentary (NC) domains.

6.2.5 Experimental Setup

In this chapter, we analyse and compare the performance of BIP-NMT with other NMT systems. BIP-NMT is designed to involve human translators in the interaction loop to produce better translations in an online manner. In this thesis, instead of hiring human translators, we simulate the human feedback by using the true references, which are not presented to the translation system(s).

Simulations In order to simulate the true online setting, we limited the batch size and number of training epoch in bandit training to be 1. This is based on the assumption that a human translator input sentences, in source language, one by one to the system, i.e., batch size of one. Moreover, each input sentence is seldom translated multiple times in a short time framework, i.e., a single epoch. Character-F score with character-n-grams of length 6 and $\beta = 2$, the importance factor over precision, was a substitute of human feedback in the simulation.

Data and Preprocessing We considered data from two domains with translation from French to English. Europarl (EP), $\sim$ 1.3M sentences, was used in pre-training the actor under standard supervised learning. In bandit training, we used a subset of News Commentary (NC) data, $\sim$ 10K. The domain shift and the small data set were to simulate a realistic scenario of costly human feedback and diverse sources of data. We also removed sentences having more than 50 tokens and applied Moses tools for tokenization and cleaning. The vocabulary of each language is the 50K most frequent words extracted from the two training sets. Table 6.1 gives a summary of the data sets.

Configuration of NMT Systems In our experiments, we would compare three models: 1) Out-of-domain, 2) NED-A2C (Nguyen et al., 2017) and 3) BIP-NMT. The

ϵ	**chrF (std)**	**BLEU (std)**	**Avg # Requests**	**Δ chrF**	**Δ BLEU**	**Δ Avg # Requests**
0	61.86 (0.06)	25.54 (0.17)	15.91 (0.01)	0	0	0
0.25	62.15 (0.17)	25.84 (0.13)	11.06 (0.07)	+0.29	+0.3	-5
0.5	61.95 (0.05)	25.46 (0.09)	7.26 (0.03)	+0.09	-0.08	-9
0.75	62.15 (0.04)	25.07 (0.12)	4.94 (0.02)	+0.29	-0.47	-11

Table 6.2: Impact of entropy margin ϵ on average sentence-level chrF score, corpus BLEU and average number of feedback requests per sentence on the NC validation set. The feedback quality threshold μ is set to 0.8 for all models.

first one represents a model without online update, which is in contrast to NED-A2C and BIP-NMT. NED-A2C represents algorithms trained with sentence level bandit feedback. We used uni-directional, single-layer LSTM with global attention mechanism for all models; the size of word embedding and LSTM hidden cells were set to 500. The exception was on the output layer that had size of the vocabulary for the actor (classification) or size of 1 for critic (regression). We used Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

Training In supervised training, we applied mini-batch learning, i.e., a batch size of 64, with Adam's $\alpha = 10^{-3}$. A decay factor of 0.5 is applied to α, starting from the fifth pass, when perplexity on the validation set increases. During bandit training, a constant value of $\alpha = 10^{-5}$ was applied to both the actor and critic. In both training cases, we clipped the Euclidean norm of gradients to a value of 5.

Model Selection We used the Out-of-domain model as the baseline that was chosen based on the highest corpus BLEU score on the validation set; it was also the warm-start for the subsequent models using bandit/RL related training. In BIP-NMT, several values of ϵ were tested, and each setting was run three times under three different random seeds. We reported the mean value (over the three runs) of average sentence chrF, corpus BLEU and average number of requests per sentence. The judgement parameter, μ, was set to 0.8 in our simulations. Table 6.2 summarizes BIP-NMT's validation performance with four different values of ϵ and a constant value of $\mu = 0.8$. There is a clear inversely proportional relationship between ϵ and average number of requests. In general, the more the number of requests the higher is the value of corpus BLEU except in the case of ϵ equals 0. A possible explanation is that the system suffers from over-fitting.

Evaluation Average sentence-level chrF score and corpus BLEU are two major evaluation metrics for our systems. In BIP-NMT, we consider an extra metric: the average number of requests per sentence because of the scarcity of human feedback. In order to encourage exploration and, at the same time, represent system's response to external feedback, we used multinomial sampling during training but greedy decoding in model validation and testing.

6.2.6 Results and Analysis

Table 6.3 and figure 6.2 present model performance on test data. BIP-NMT achieves more than 2 points in both chrF and BLEU than the Out-of-domain model which has 61.30 points in chrF and BLEU of 24.77 points. An improvement of 1 point in chrF and 0.55 point in corpus BLEU are observed in NED-A2C showing the usefulness of partial feedback with active learning.

Example Protocols Table 6.4 presents user-interaction protocols for three examples encountered during training of BIP-NMT with $\epsilon = 0.75, \mu = 0.8$. Words that are underlined are good prefixes, i.e, a score $\geq \mu$. They are stored in Ξ and re-used for generation of its suffixes. Words highlighted by red have negative advantage scores which indicate that they should be discouraged, i.e., lower probability of being sampled.

In the first example, the model made frequent feedback requests (in 8 of 17 decoding steps) and filled the prefix buffer due to the high quality of the samples. Maybe, one improvement is to avoid feedback request in consecutive iterations. In the second example, only the first two tokens were put in Ξ since the feedback varied quite a bit for subsequent partial translations. Another possible improvement is to forbid the system to sample the same token in the same position once it is deemed to be bad, e.g., the token 'we' in the third position. Note how the token-based critic encouraged a few phrases of the translations, but discouraged others. The final example shows a translation where the model was very certain and hence requested feedback only after the first and last token (minimum number of feedback requests). The critic correctly identified problematic parts of the translations regarding the choice of prepositions.

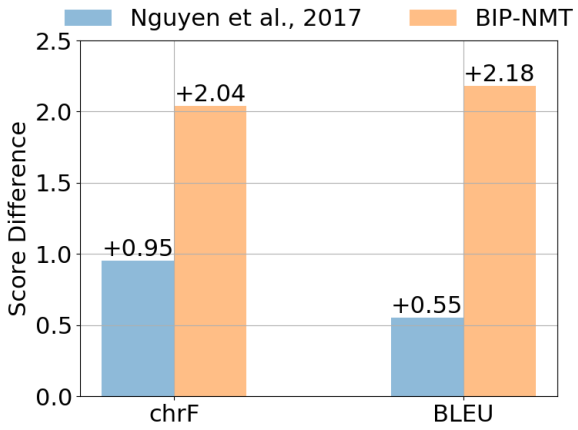

Figure 6.2: Performance of NED-A2C (Nguyen et al., 2017) and BIP-NMT over out-of-domain NMT on the NC test set of 2000 sentences.

System	**chrF (std)**	**BLEU (std)**	**Δ chrF**	**Δ BLEU**
Out-of-domain NMT	61.30	24.77	0	0
Nguyen et al.	62.25 (0.08)	25.32 (0.02)	+0.95	+0.55
BIP-NMT ($\epsilon = 0.75$, $\mu = 0.8$)	63.34 (0.12)	26.95 (0.12)	+2.04	+2.18

Table 6.3: Evaluation of pre-trained out-of-domain baseline model, actor-critic learning on one epoch of sentence-level in-domain bandit feedback (Nguyen et al., 2017) and BIP-NMT with settings $\epsilon = 0.75$, $\mu = 0.8$ trained on one epoch of sub-sentence level in-domain bandit feedback. Results are given on the NC test set according to average sentence-level chrF and corpus-level BLEU. Result differences between all pairs of systems are statistically significant according to `multeval` (Clark et al., 2011).

6.2.7 Summary

In this section, we presented an interactive-learning protocol for NMT for reducing human effort and improving adaptation. Our protocol, called BIP-NMT, uses scoring on subsequence-level, active learning via entropy measure and a prefix-buffer for enhancement. In simulation, BIP-NMT outperforms a NMT model fine-tuned on sentence-level feedback in domain adaptation settings, showing the potential advantage of incremental partial-feedback and active learning. One limitation is the use of prefix buffer, imposing a hard prefix to the latter suffix. Another limitation is the restriction of substitution or lightweight post-editing feedback. In the next section, we present another protocol to address this.

SRC depuis 2003 , la chine est devenue le plus important partenaire commercial du mexique après les etats-unis . < /s>
REF since 2003 , china has become mexico 's most important trading partner after the united states . < /s>

Partial sampled translation	**Feedback**
since	1
since 2003 , china has	1
since 2003 , china has become	1
since 2003 , china has become mexico	1
since 2003 , china has become mexico 's	1
since 2003 , china has become mexico 's most	1
since 2003 , china has become mexico 's most important	1
since 2003 , china has become mexico 's most important trading partner after the us . < /s>	0.8823

SRC la réponse que nous , en tant qu' individus , acceptons est que nous sommes libres parce que nous nous gouvernons nous-mêmes en commun plutôt que d' être dirigés par une organisation qui n' a nul besoin de tenir compte de notre existence . < /s>
REF the answer that we as individuals accept is that we are free because we rule ourselves in common , rather than being ruled by some agency that need not take account of us . < /s>

Partial sampled translation	**Feedback**
the	1
the answer	1
the answer we	0.6964
the answer we ,	0.6246
the answer we as individuals allow to 14 are	0.6008
the answer we , as individuals , go down to speak 8 , are being free because we govern ourselves , rather from being based together	0.5155
the answer we , as people , accepts is that we principle are free because we govern ourselves , rather than being led by a organisation which has absolutely no need to take our standards . < /s>	0.5722

SRC lors d' un rallye "journée jérusalem" tenu à l' université de téhéran en décembre 2001 , il a prononcé l' une des menaces les plus sinistres du régime . < /s>
REF at a jerusalem day rally at tehran university in december 2001 , he uttered one of the regime 's most sinister threats . < /s>

Partial sampled translation	**Feedback**
in	0
in a round of jerusalem called a academic university in teheran in december 2001 , he declared one in the most recent hostility to the regime . < /s>	0.5903

Table 6.4: Interaction protocol for three translations. These translations were sampled from the model when the algorithm decided to request human feedback (lines 9-10 in Algorithm 3). Tokens that get an overall negative reward (in combination with the critic), are marked in red, the remaining tokens receive a positive reward. When a prefix is good (i.e. $\geq \mu$, here $\mu = 0.8$), it is stored in the buffer and used for forced decoding for later samples (underlined).

6.3 Interactive-predictive Neural Machine Translation through Reinforcement and Imitation

In this section, we discuss our modification of BIP-NMT which was designed to reduce human effort in post-editing. BIP-NMT replaces the annotation process by scoring on (incremental) sub-sequences. Once a prefix of good translation quality is identified, it would be stored in a prefix-buffer as constrained prefix for generating latter suffixes in a forced-decoding manner. An advantage of such incremental feedback is that errors at the beginning of the translation can be earlier penalized. However, such entire constrained prefix may contain irrelevant tokens, limiting the translation flexibility of latter sub-sequences. In addition, multi-nominal sampling does not guarantee the generation of good prefixes in a realistic number of trials. To make the search of good (sub-)sequences more efficient, we introduce substitution feedback, i.e., correction, together with constrained beam-search for guidance.

Our goal here is to combine both feedback modes — corrections and scoring — by treating them as expert demonstrations and reward values in an interactive protocol that combines imitation learning (IL) (Ross et al., 2011) and reinforcement learning (RL) (Sutton and Barto, 2018), respectively, using only limited human edits. Our protocol allows natural instructions: 1) "keep", 2) "delete", and 3) "substitute" as provision of feedback. Both "keep" and "delete" actions are converted to numeric scores for policy gradient update, similar to BIP-NMT. On top of that, the "delete" action acts as a lexical constraint to prevent the model from generating it again at the same location. The "substitute" feedback instructs the model where to generate the desired tokens. A further difference of our framework to BIP-NMT is our use of token-level entropy and relative change of entropy, rather than average sentence-level entropy, to reduce the amount of feedback requests. The token-level entropy assists provision of feedback by exposing its most uncertain tokens whereas the relative change helps to capture abrupt change in the model's confidence upon generating the current token. Lastly, we replace the prefix-buffer in BIP-NMT by constrained beam-search so that the model can adjust the previously generated tokens according to the new constraints.

Our domain adaptation experiments show that weak feedback in form of keep/delete rewards on translation outputs yields consistent improvements of between 2.6 and 3.46 BLEU points over the pre-trained baseline. On one language pair (Fr-En), it even matches the improvements gained by forcing word substitutions from reference translations into the re-decoded output. Furthermore, both feedback scenarios considerably reduce human effort.

6.3.1 Interactive-Predictive Learning from Rewards and Demonstrations for NMT

As shown in Cheng et al. (2018), IL and RL can be viewed as a single algorithm that only differs in the choice of the oracle, based on objective functions that are defined as the expected value function with respect to the current model's policy π_n in case of RL, and as the expected value function with respect to an expert policy π^* in case of IL. Applied to NMT, both IL and RL are based on a Markov Decision Process where a deterministic sequence of states consisting of the source input and the history of the model's predictions (possibly incorporating expert's demonstrations) serves as conditioning context to predict the respective word (Bahdanau et al., 2017).

We instantiate rewards and demonstrations to the feedback types in interactive-predictive translation as follows: In the first case, uncertain words predicted by the system receive a positive or negative reward based on "keep" or "delete" feedback respectively. In the second case, uncertain words can additionally be corrected based on an expert policy in the form of "substitute" feedback associated with a positive reward. This feedback is integrated in context of the model's own predictions by adding rules to constrained beam search decoding (Hokamp and Liu, 2017; Post and Vilar, 2018).[2]

Learning Objective We formalize the objective of interactive-predictive NMT as maximizing the value function V of a parametrized policy π_θ, i.e., we seek to maximize the expected (future) reward obtainable from interactions of the NMT system with a human translator who, by editing translations, implicitly assigns rewards $r(\hat{\mathbf{y}}_i)$ to system predictions $\hat{\mathbf{y}}_i$ given source sentences $\mathbf{x}_i$:

$$\max_\theta V_{\pi_\theta}(s_0) = \max_\theta \mathbb{E}_{\hat{\mathbf{y}}\sim\pi_\theta(\cdot|\mathbf{x}_i)}[r(\hat{\mathbf{y}})] \tag{6.3}$$

where s_0 is the initial state of decoding, i.e., given the source sequence $\mathbf{x}_i$ and the <bos> token in the decoder. Following the policy gradient theorem (Sutton et al., 2000), and especially actor-critic for sequence prediction by Bahdanau et al. (2017), its derivative is

[2]We observe that the distinction between weak feedback and expert feedback is difficult to make in the "keep" feedback case: on the one hand, this type of feedback refers to an action generated by the system, and on the other hand, it can be seen as a form of expert demonstration. From this perspective, our first system is closer to RL while our second system is closer to IL. For brevity, we will refer to our models as "RL model" and "IL model", respectively.

$$\begin{aligned}\nabla_\theta V_{\pi_\theta}(s_0) &= \mathop{\mathbb{E}}_{\hat{\mathbf{y}}\sim\pi_\theta(\cdot|\mathbf{x}_i)}\left[\sum_{t=1}^{T}\sum_{y\in\mathcal{V}}\nabla_\theta\pi_\theta(y|s_{t-1})Q(y,s_{t-1})\right] \\ &= \mathop{\mathbb{E}}_{\hat{\mathbf{y}}\sim\pi_\theta(\cdot|\mathbf{x}_i)}\left[\sum_{t=1}^{T}\sum_{y\in\mathcal{V}}\pi_\theta(y|s_{t-1})\nabla_\theta\log\pi_\theta(y|s_{t-1})Q(y,s_{t-1})\right] \qquad (6.4)\end{aligned}$$

where $s_{t-1} = (\mathbf{x}_i, \hat{\mathbf{y}}_{<\mathbf{t}})$ is the state, $\mathcal{V}$ is a set of target-word vocabularies, $Q(y, s_{t-1})$ is the action value function on an action y given the state s_{t-1}. In our application, we ask for feedback on a single trajectory at each round of interactions. This naturally represents the 1-sample estimate method used in the REINFORCE algorithm so that both the expectation over all possible target sequences $\hat{\mathbf{y}}$ and also the inner sum over all the target tokens per timestep t are reduced. Furthermore, we approximate $Q()$ directly by our designed reward (see below) for computation efficiency and for our designed reward system that operates mainly on specific tokens only.

Depending on the type of feedback, the instantaneous reward $r(\hat{y}_t)$ for a system output $\hat{y}_t$ (a word for an example) is set to the following values[3]:

$$r(\hat{y}_t) = \begin{cases} 0.5 & \text{if SUBSTITUTE/KEEP,} \\ -0.1 & \text{if DELETE.} \end{cases} \qquad (6.5)$$

In addition, we found that flooring rewards for tokens that do not receive explicit feedback to a small number[4] stabilizes the training and improves performance on the dev set. Combining the simplifications on equation (6.4) and the designed reward scheme in (6.5), our final objective is

$$\nabla_\theta V_{\pi_\theta}(s_0) = \sum_{t=1}^{T} r(\hat{y}_t)\nabla_\theta\log\pi_\theta(\hat{y}_t|\mathbf{x}, \hat{\mathbf{y}}_{<\mathbf{t}}) \qquad (6.6)$$

The equation is essentially the same as equation (2.23) when the mini-batch size is one. However, a notable difference is that *the above equation is on a (human) corrected trajectory, which can be different from the one generated by π_θ, because of the substitution feedback.*

Interactive-predictive workflow Figure 6.3 gives a graphical illustration of the proposed workflow. In contrast to existing approaches where full sentences are

[3]The values are tuned on the validation set.

[4]We apply Gaussian noise with mean 0.1 and standard deviation of 0.05.

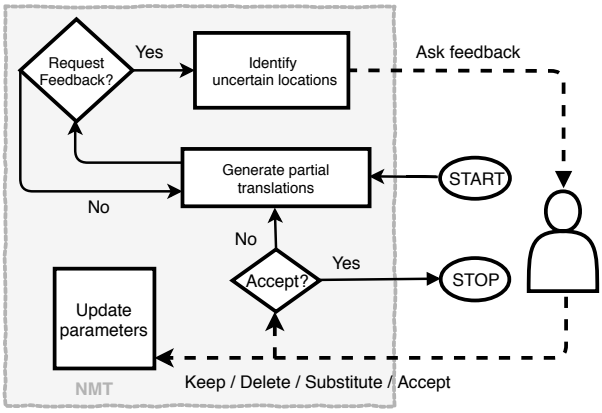

Figure 6.3: A graphical illustration of the interactive-predictive workflow of our system. Dotted arrows indicate interactions between human and system; solid arrows indicate procedures within the system

Algorithm 4: Interactive-predictive workflow for a single sentence using constrained beam search. *Input:* model parameters θ, source sentence $\mathbf{x}_i$, beam size k, learning rate α. *Output:* updated θ^*.

$t_{prefix} \leftarrow 1$, $n \leftarrow 1$;
$\theta_0 \leftarrow \theta$, $\xi \leftarrow \emptyset$;
SET-NMT-SOURCE($\mathbf{x}_i$);
repeat
  $\hat{\mathbf{y}}_{1:t} \leftarrow$ BEAM-SEARCH(k, t_{prefix}, $T_{\max}$, ξ);
  for $i \leftarrow 1$ **to** t **do**
    if UNCERTAIN-LOCATION($\hat{\mathbf{y}}_{1:t}, i$) **then** Collect feedback rules ξ_i;
  Get rewards for $\xi_i \in \{keep, delete, substitute\}$ according to Eq. 6.5 ;
  $\theta_n \leftarrow \theta_{n-1} + \alpha \nabla_\theta V$ (Eq. 6.6);
  $t_{prefix} \leftarrow |\hat{\mathbf{y}}_{1:t}|$, $n \leftarrow n + 1$;
until $\hat{\mathbf{y}}_{1:t}$ *accepted*;

corrected in each round, our system stops decoding when the generated segment meets several (un)certainty criteria. Our system then identifies uncertain words within the generated segment and asks the user to edit these words. The idea is to direct the user to possible translation errors in the segment, and to collect feedback on these highly informative locations, effectively implementing an AL strategy. The collected feedback is used twice: firstly, it is used to perform an online update of the system's parameters, and secondly, it is integrated as rules into constrained beam search. The full translation is reached after several interactive rounds when the translator finally accepts the translation.

Algorithm 5: Constrained beam search for uncertain partial translation.
Input: beam size k, prefix length p, maximum length N, feedback rules ξ.
Output: partial translation.

```
1  function BEAM-SEARCH(k, p, N, ξ)
2      beam ← DECODER-INIT(k);
3      for t ← 1 to N do
4          scores ← DECODER-STEP(beam);
5          beam ← KBEST(scores, k, ξ);
6          if LENGTH(beam[0]) > p and IS-UNCERTAIN(beam[0]) then break ;
7      return beam[0];
8  function KBEST(scores, k, ξ)
9      scores_c ← APPLY-CONSTRAINTS(scores, ξ);
10     beam ← ARGMAX_k(scores_c);
11     return beam;
```

Algorithms Algorithm 4 describes the implementation of our interactive-predictive workflow. In the first round, the system starts with initial model parameters θ_0, and an empty set of feedback rules ξ, and calls BEAM-SEARCH (line 6) to first generate an unconstrained partial translation of length t by evaluating the uncertainty criteria in function IS-UNCERTAIN. The algorithm then evaluates each token within the partial translation and asks for user feedback if the token is considered uncertain w.r.t. the function UNCERTAIN-LOCATION (lines 6-7).

Feedback is captured in form of rules that correspond to edits on specific locations, e.g., KEEP token at position i, DELETE token at position i, or SUBSTITUTE token at position i with another token. After collecting the rewards for feedback rules ξ_i according to Equation 6.5 (line 8), the model parameters are updated by taking a gradient step as defined in Equation 2.23 (line 9).

The updated system then proceeds to the next round by calling BEAM-SEARCH again, this time with a set of feedback rules ξ to generate a constrained partial translation exceeding the previous length t_{prefix}. The uncertainty criterion of tokens is evaluated again and the user is asked for feedback on these tokens, extending the set of feedback rules ξ, which are used to update the system parameters and generate the next partial translation until the user is satisfied with the translation.

Measuring uncertainty We define a measure of uncertainty based on the entropy at a time step t given a set of actions $\mathcal{V}$ (i.e., the target vocabulary) where

$$H_t = -\sum_{y \in \mathcal{V}} \pi_\theta(y|\mathbf{x}, \hat{\mathbf{y}}_{<\mathbf{t}}) \log \pi_\theta(y|\mathbf{x}, \hat{\mathbf{y}}_{<\mathbf{t}}).$$

The idea is that learning from edits on high entropy time steps is more helpful than learning from edits on low entropy time steps because updating parameters based on uncertain regions better stabilizes the model over time. Furthermore, entropy is computationally simple and far less expensive than external reward estimators such as a quality estimation system, a critic, or a discriminator.

A single token at time step t is considered uncertain if the entropy exceeds a defined threshold ϵ, i.e., $H_t > \epsilon$. We use this criterion to identify informative locations of a partial translation on which the user is asked for feedback.

In case of partial translations, a sequence of length t is considered uncertain if the token at time t is uncertain as defined above, and there is an abrupt change in entropy at t, formally $\frac{H_t - H_{t-1}}{H_{t-1}} > \delta$. Both criteria are applied to determine the length of a partial translation shown to the user.

Constrained beam search A central component is a modified beam search algorithm that takes positional constraints into account (Algorithm 5). The user constraints force the system to generate alternative translations and can thus be interpreted as an exploration strategy.

An efficient alternative exploration strategy is multinomial sampling. In our interactive-predictive scenario, however, it is crucial that translations on locations without explicit user feedback are preserved, and this cannot be modeled easily with multinomial sampling. Beam search on the other hand ensures stable translations due to its deterministic nature, and the idea of constrained beam search provides the tools to improve the translation interactively. As a side effect, higher quality translations can be obtained by increasing the beam size at the cost of computational power.

After initializing k beams, the algorithms generates a partial translation by calling DECODER-STEP (line 4) to retrieve the next token and score all hypotheses. The constraints (provided in the form of feedback rules) are applied in the function KBEST (line 5) by filtering out all hypotheses that do not satisfy the constraints before the ARGMAX_k operation selects the k highest scoring remaining hypotheses. The single best partial translation is shown to the user only if two conditions are met: (1) the length exceeds the length of the previous partial translation, and (2) the current partial translation is considered an uncertain sequence (line 6). In case one condition is not met, the system iteratively extends the partial translation up to a maximum hypothesis length.

	Data	Training	train / dev / test	∅ en-length
fr-en	EP	pre-training	1.3M / 2k / –	25.5
	NC	interactive	18.4k / 3k / 5k	22.8
de-en	EP	pre-training	1.7M / 2.7k / –	24.0
	NC	interactive	18.9k / 1k / 2k	22.6

Table 6.5: Data used in pre- and interactive training for French-English (fr-en) and German-English (de-en).

6.3.2 Experimental Setup

To demonstrate the effectiveness of our reinforcement and imitation strategies, we simulated the interactive-predictive workflow described above in a domain adaptation setup. A human translator was simulated by comparing partial translations with corresponding gold translation to extend the set of feedback rules in every round. In the RL setting, the simulated human translator provided only weak feedback (KEEP and DELETE edits) on tokens generated by the system, while in the IL setting the simulated translator additionally injected expert feedback (SUBSTITUTE edit) by demonstrating how the system should act at a specific time step.

In our simulation experiments, feedback was collected on the uncertain tokens of the partial translation. An exact match between the uncertain token and the reference generated a KEEP edit, while differing tokens generated either a DELETE or SUBSTITUTE edit depending on the type of system. Tokens exceeding the sentence length of the reference received a DELETE feedback. We refer to the first system as KEEP+DELETE, and the second system as +SUBSTITUTE. While the system parameters were updated online after every such simulated interaction, system evaluation was done by a standard offline translation of an unseen test set.

Dataset For pre-training, we used the Europarl (EP) corpus version 5 for the French-English system, and version 7 for German-English. For interactive training, we used the News Commentary (NC) 2006 corpus. Both corpora are publicly available on the WMT13's homepage.[5] All experiments were conducted on two language pairs, i.e., German-English (de-en) and French-English (fr-en). Data sets were tokenized and lowercased using MOSES preprocessing scripts (Koehn et al., 2007). We applied compound splitting on the German source sentences using CDEC's tool (Dyer et al., 2010). Our data sets for interactive training differ from the original News Commentary

[5] `https://www.statmt.org/wmt13/`

Pair	System	ChrF (σ)	ΔChrF	BLEU (σ)	ΔBLEU	∅ rounds	∅ keep+delete / subst.
fr-en	Pre-trained	61.08	–	24.70	–	–	–
	Full Post Edits	61.96 (0.15)	+0.88	29.10 (0.09)	+4.40	–	–
	KEEP+DELETE	**62.72** (0.11)	+1.64	28.16 (0.14)	+3.46	3.2	13.7 / –
	+SUBSTITUTE	62.24 (0.08)	+1.16	**28.52** (0.10)	+3.82	3.3	1.8 / 5.6
de-en	Pre-trained	59.34	–	22.66	–	–	–
	Full Post Edits	60.24 (0.25)	+0.9	27.40 (0.22)	+4.74	–	–
	KEEP+DELETE	59.57 (0.19)	+0.23	25.28 (0.09)	+2.62	3.3	13.1 / –
	+SUBSTITUTE	**60.73** (0.14)	+1.39	**26.91** (0.1)	+4.25	3.3	1.8 / 5.9

Table 6.6: Character-F (ChrF), and BLEU test results on the French-English (fr-en) and German-English (de-en) translation tasks. Highest scores on RL and IL systems are printed in bold. The Δ columns indicate the score differences to the pre-trained baseline system. All scores are averaged over three runs with standard deviation σ in parentheses.

data splits as follows: (1) we sampled a subset of the original training set to reduce the number of parallel sentences to 18,432 for French-English and 18,927 for German-English, and (2) we increased both validation and test set for French-English to 3,001 and 5,014 parallel sentences by moving data from the original training set excluding sentences that were sampled for training. Note that a training set size of less than 19,000 parallel sentences is very small even in a domain adaptation setup. Table 6.5 summarizes the statistics of our datasets.

Model Architecture We used a single uni-directional LSTM layer with global attention mechanism between encoder and decoder. We set the dimensionality of the LSTM hidden states and the word embeddings to 500 and built the vocabulary using the most frequent 50,000 words in each language.

We used Adam optimizer in all training scenarios. In supervised training, we used a mini-batch size of 64 and an initial learning rate of 0.001. Starting from the 5^{th} epoch, the rate was reduced by half in each epoch if the validation perplexity increases. In interactive training, we trained for a single epoch and applied a constant learning rate of 10^{-5} with a mini-batch size of 1.

In all experiments we set entropy parameters to $\epsilon = 1$, $\delta = 0.5$, and used a beam size of 5 during training. For testing, we applied greedy decoding.

6.3.3 Results and Analysis

On both language pairs, the optimal pre-trained NMT models were obtained in the 6^{th} training epoch, forming the out-of-domain baseline. We also compared our RL/IL strategies with full post-edits simulated by supervised training on the in-domain News

Commentary data, forming an in-domain upper bound. We repeated each experiment three times and report mean and standard deviation for both Character-F[6] (ChrF) and corpus BLEU.

In the French-English experiments, both our imitation and reinforcement strategies show improvements of more than 3 points in BLEU and 1 point in ChrF over the out-of-domain baseline. Both strategies achieve lower BLEU score than training on full post-edits, in particular, 0.94 points lower in the KEEP+DELETE setting, and 0.58 points lower in +SUBSTITUTE setting. However, both strategies achieve higher ChrF scores, i.e., 0.76 points for KEEP+DELETE and 0.28 points for +SUBSTITUTE. See upper half of Table 6.6 for a summary.

In the German-English experiments, there is a bigger performance gap between the KEEP+DELETE and the full post-edits system, concretely, 0.67 points in ChrF score and 2.12 points in BLEU lower than full post-edits. However, the improvement over the pre-trained model amounts to 2.62 BLEU points and 0.23 points in ChrF score. Our +SUBSTITUTE system is comparable in performance to the full post-edits system, yielding a result that is 0.49 lower in BLEU but 0.49 points higher in ChrF. See lower half of Table 6.6 for the summary.

We also report average of feedback rounds and rules per sentence in Table 6.6. We optimized the maximum number of allowed feedback rules per round on the dev set and use 9 (fr-en) and 7 (de-en) for the KEEP+DELETE and 3 for the +SUBSTITUTE systems. Even for the simpler model based on only weak feedback, the number of user clicks is between 13.7 and 13.1, which is well below the average target sentence length of 22.8 and 22.6. By allowing expert SUBSTITUTE feedback that actively generates better tokens in the next round, the number of rules is reduced to 7.4 and 7.7. Our experiments indicate that focusing on uncertain locations can reduce human translation effort substantially.

Effect of online learning We also examine the effect of online learning on average cumulative entropy of the model's policy distribution over time. Figure 6.4 visualizes the change of entropy during interactive training. At the beginning, the system is in regions of high entropy but quickly learns from human edits and the curves become smooth and monotonic. After this initial phase, the overall better performing French-English task shows consistently lower entropy than the German-English task, indicating a connection between model's entropy and translation quality. However, the comparison between the KEEP+DELETE and the better performing +SUBSTITUTE

[6]Using parameters $ngram = 6$ and $\beta = 2$.

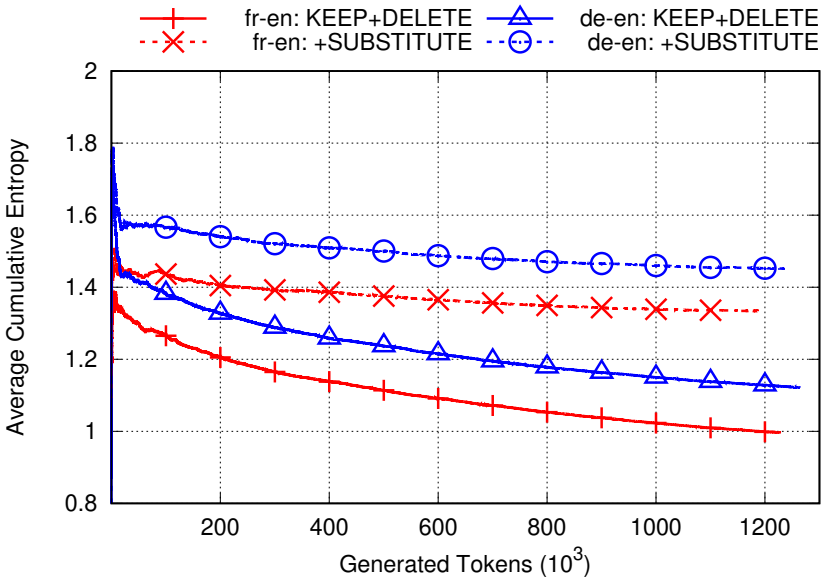

Figure 6.4: Average cumulative entropy of the model's policy distribution over time during simulated interactive learning. Plots are shown for the French-English (fr-en) and the German-English (de-en) task, and for the KEEP+DELETE and the +SUBSTITUTE system, respectively.

systems shows the opposite trend and requires a different explanation. We conjecture that the +SUBSTITUTE system's expert demonstrations at uncertain locations help the system to find better translations, but such demonstrations also move the system to higher entropy regions, effectively implementing a useful exploration strategy. In contrast to this, the KEEP+DELETE system always stays in more certain regions by selecting another high probability token if the original token receives a DELETE feedback by the user.

Effect of beam size The observations on model's entropy over time in the previous paragraph and the implementation details described in Section 11 show that our constrained beam search implements exploration in a user-controlled manner. We conjecture that beam size also influences the exploration and should have a different effect on different feedback strategies. We thus conduct additional experiments using beam sizes of 2, 5, 10 and 20 on all language pairs and the two systems. The results are summarized in Figure 6.5. In both KEEP+DELETE and +SUBSTITUTE systems, a beam size of 2 is sufficient to achieve substantial gains over the baselines in both language pairs. In case of the KEEP+DELETE system, increasing beam sizes only marginally influence the translation performance.

In case of the +SUBSTITUTE system, there are considerable gains of almost 1 BLEU

point and 1 Character-F point when increasing the beam size from 2 to 5. Here, the larger beam size enables the system to connect the expert demonstrations with better prefixes which helps the system to explore higher scoring trajectories. Increasing the beam size to 10 or 20 further improves performance but the gains are small.

Decoding Speed The total runtime of each of our simulated interactive experiments is roughly 6 hours when simulated on a Nvidia P40, while training of the KEEP+DELETE system is slightly slower than of the +SUBSTITUTE system due to the higher number of feedback rules. Looking at the sentence level this means the total decoding time of our system for all partial translations of a single sentence is $6 \times 1\text{h}/(18,432 \times 3.3) = 0.361\text{s}$ for the French-English task, and even less for the German-English task. This estimate does not account for the time our system conducts validation tests or constructs simulated feedback, thus the actual averaged processing time is lower. Knowles and Koehn (2016) argue that beam search is usually too slow to be used for training in interactive live systems, however, recent hardware developments together with our strategy of partial decoding makes constrained beam search applicable even in training. As a side effect, corrections on early time steps reduce the problem of error propagation and thus improve both usability of the system and satisfaction of the translator.

Leveraging BPE or character-level NMT Our current implementation of interactive-predictive NMT uses a word-based translation approach and presents word units to users for feedback. An adaptation of our algorithm to sub-word or character level NMT is possible and requires to redistribute the reward associated to the word level to sub-word units or characters, and to maintain their location information in the constrained beam search. We leave this extension to future work.

Examples Table 6.7 illustrates the translation workflow of our interactive-predictive protocol by listing four examples: the upper half shows example translations of the two systems for the German-English task, the lower half shows two examples of the systems for the French-English task.

The first example is taken from the KEEP+DELETE system, where our simulated user provides only KEEP and DELETE feedback on suggested locations. In interactive round 1 on the German-English task, the system stops after generating the uncertain partial translation “the core” and asks the user for feedback specifically on the term “core”. The simulated user returns a DELETE feedback and the system is able to

generate the more appropriate translation "heart of the problem" in round 2. In round 3, however, a weakness of the simulated feedback becomes apparent: our user gives a negative DELETE feedback on the token "amount" because the token differs from the given reference word "quantity", even though it is an appropriate translation for the German word "Menge" in this context. The system then generates "volume" in round 4 and "supply" in the final round 5, although both translations are worse than the initially proposed translation "amount". One explanation for this behavior is the way online updates are applied to the NMT system: while the constrained beam search implements feedback rules on token level, the online updates of the NMT system take place on the word embedding level. An update based on negative feedback actually forces the NMT system to avoid semantically similar words. In the above example, the negative feedback for "amount" downgrades the optimal translation "quantity" because of the semantic similarity of both words, and instead upgrades the more diverse translations "volume" and "supply". In our example, this strategy has an immediate negative impact on translation quality, but it also illustrates the positive exploration effect which is helpful in the long run.

The second example is taken from the +SUBSTITUTE system, where the simulated user additionally provides "substitute" feedback. In interactive round 1, the system generates the uncertain partial translation "the south koreans are" and identifies "the" and "are" as uncertain tokens. The user suggests to change "the" to "as", and "are" to "south" by providing SUBSTITUTE feedback. Again, a limitation of our simulation becomes apparent: our simulated substitutions are based on reference translations, but a real translator would not change the given partial translation to "as south korean south". Still, based on the two feedback rules and the online update, the NMT system is able to follow a better trajectory in round 2. We observe that SUBSTITUTE feedback is a very strong signal that supports the system to quickly get close to the translation our simulated user has in mind (which is the reference in our simulation).

The French-English task examples illustrate a noteworthy property of our algorithm: In round 3 of the KEEP+DELETE system, the simulated user provides DELETE feedback on the tokens "to hate their" only because they occur at different positions compared to the reference. However, the system is able to recover and re-generate the tokens at the correct position in round 5. A similar behavior can be observed for the +SUBSTITUTE system in round 3, where the phrase "bring about macro-economic" is first substituted and then generated again in the final round 4.

6.3.4 Summary

In this work, we propose an integration of interactive-predictive neural machine translation with imitation learning and reinforcement learning. The goal of such integration is to bring model learning and effort-reduced human feedback for faster model adaptation/personalization purpose. Our results indicate that online learning from (simulated) human edits on uncertain locations of the partial translations can train a competitive model to the one using supervised learning on in-domain data but with substantially less human effort.

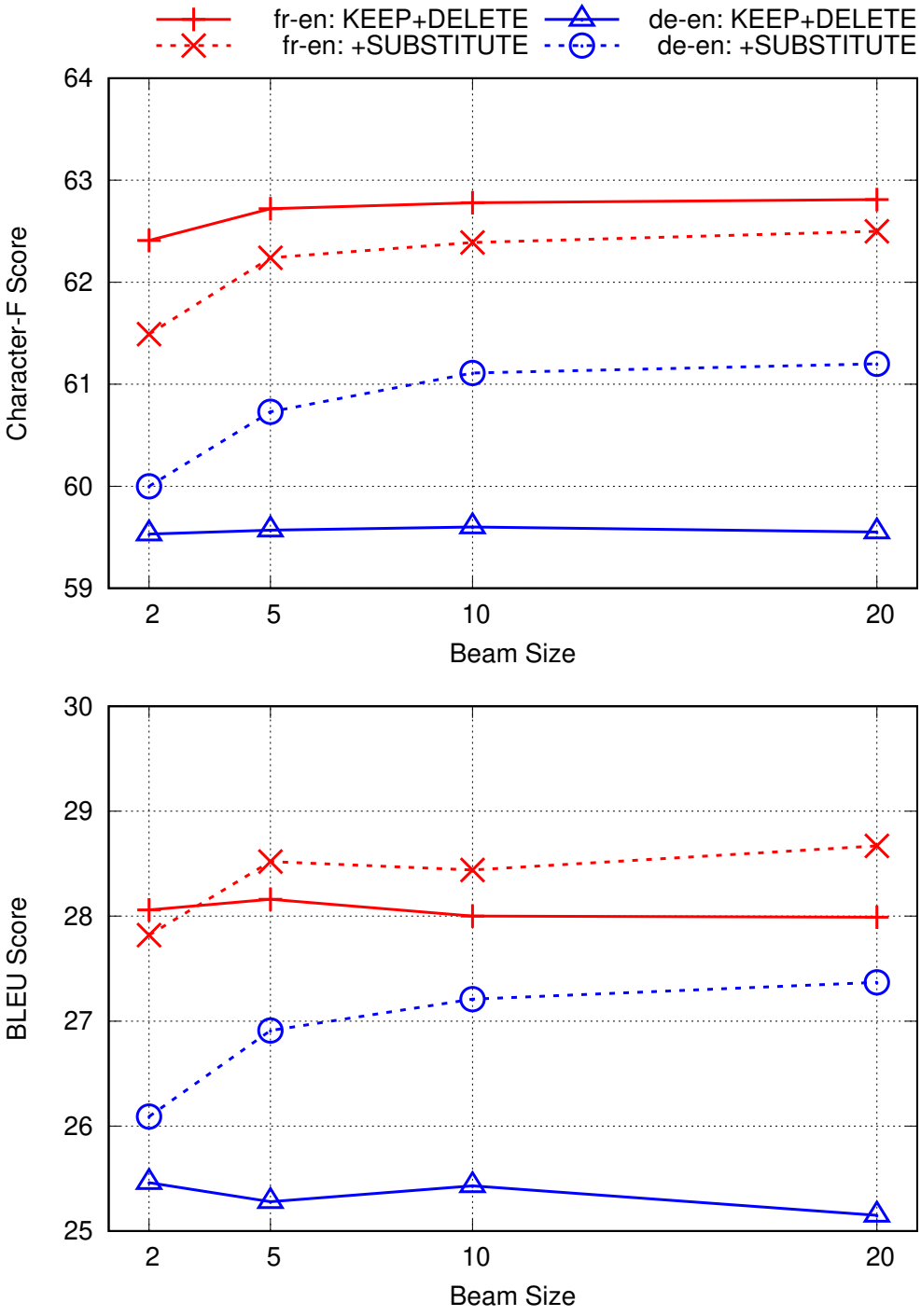

Figure 6.5: The two figures show the effect of different beam sizes on Character-F score (top) and BLEU score (bottom). We conduct experiments on French-English (fr-en) and German-English (de-en) and both systems (KEEP+DELETE and +SUBSTITUTE). All scores are averaged over two runs.

German-English	**Source**	der kern des problems ist nicht die gesamt_menge des öls , sondern seine lage .
	Reference	the heart of the problem is not the overall quantity of oil , but its location .
	Round	**Partial translation → FEEDBACK**
	1	the core$_{2}$ → DELETE(2)
	2	the *heart*$_{2}$ of the problem is not the total$_{9}$ → KEEP(2), → DELETE(9)
	3	the *heart* of the problem is not the *overall* amount$_{10}$ of oil , but its$_{15}$ → DELETE(10), → KEEP(15)
	4	the *heart* of the problem is not the *overall* volume$_{10}$ of oil , but *its* situation$_{16}$. → DELETE(10, 16)
	5	the *heart* of the problem is not the *overall supply* of oil , but *its position* . → accepted.
German-English	**Source**	die süd_koreaner ihrerseits verlassen sich darauf , dass china mit der nuklearen krise in nord_korea fertig wird .
	Reference	as for the south koreans , they are counting on china to deal with the north korean nuclear crisis .
	Round	**Partial translation → FEEDBACK**
	1	the$_{1}$ south koreans are$_{4}$ → SUBSTITUTE(1:as, 4:south)
	2	*as* for the$_{3}$ *south* koreans , china$_{7}$ → KEEP(3:the), → SUBSTITUTE(7:they)
	3	*as* for *the south* koreans , *they* are relying$_{9}$ on china to be$_{13}$ → SUBSTITUTE(9:counting, 13:deal)
	4	*as* for *the south* koreans , *they* are *counting* on china to *deal* with the nuclear crisis in north korea . → accepted.
French-English	**Source**	il est dur d' aimer ou de respecter un peuple et de haïr son état .
	Reference	it is hard to love or respect a people and hate their state .
	Round	**Partial translation → FEEDBACK**
	1	it is hard to love$_{5}$ → KEEP(5)
	2	it is hard to *love* or to$_{7}$ → DELETE(7)
	3	it is hard to *love* or *comply*$_{7}$ with a people and to$_{12}$ hate$_{13}$ their$_{14}$ → DELETE(7, 12, 13, 14)
	4	it is hard to *love* or *respect*$_{7}$ a$_{8}$ people and hatred$_{11}$.$_{12}$ → KEEP(7, 8),→ DELETE(11, 12).
	5	it is hard to *love* or *respect a* people and *to hate their state* . → accepted.
French-English	**Source**	un gouvernement qui n' est pas en mesure d' équilibrer ses propres finances ne peut pas apporter une stabilité macroéconomique .
	Reference	a government that cannot balance its own finances cannot be relied on to provide macroeconomic stability .
	Round	**Partial translation → FEEDBACK**
	1	a government that is$_{4}$ → SUBSTITUTE(4:cannot)
	2	a government that *cannot* balance its own$_{7}$ → KEEP(7)
	3	a government that *cannot* balance its *own* finances cannot bring$_{10}$ about$_{11}$ macro-economic$_{12}$ stability . → SUBSTITUTE(10:be,11:relied,12:on)
	4	a government that *cannot* balance its *own* finances cannot *be relied on* to bring about macro-economic stability . → accepted.

Table 6.7: Interaction protocol illustrating translation progress of the two learning systems on the German English task (upper half) and French-English (lower half). For each language pair, the first example illustrates interactions with the KEEP+DELETE system, while the second example shows interactions with the +SUBSTITUTE system. In each round, the user is asked for feedback on uncertain locations of the current partial translation. Tokens printed in blue with their position in subscript indicate uncertain locations. At the end of each round, the system is updated given the user's feedback (KEEP, DELETE, SUBSTITUTE). In the next round, it generates a constrained (partial) translation with respect to this feedback. Tokens generated based on feedback rules are printed in *italics*.

6.4 Chapter Summary

In this chapter, we discuss interactive data correction as a mean for data enhancement. We present BIP-NMT which replaces post-editing by scoring partial translation during auto-regressive decoding. In our experiments, BIP-NMT performs better than a model fine-tuned on sentence-level feedback. In the second protocol, we remove the prefix-buffer which limits the model's flexibility in generating the suffix. In addition, we integrate substitution feedback to guide better translation in a limited feedback setting. We show that such substitution feedback together with simple "keep" and "delete" feedback can train a model with better chrF score and less (simulated) human effort than a system trained on gold-reference translations. In both protocols, we use active learning with entropy based uncertainty measure to reduce the number of feedback requested.

In spite of the shown effectiveness in the experiments, only simulated human feedback is used. As indicated by (Kreutzer et al., 2018b), real human feedback is noisy and not reliable; thus, the claim of human effort reduction remains questionable and requires further investigation in future work.

Another interesting line of research would be the evaluation on improvement that cannot be detected by commonly used evaluation metrics. For an example, Wang and Sennrich (2020) found that Minimum Risk Training, a technique similar to REINFORCE, can reduce hallucinations.

Chapter 7

Conclusion

In this thesis, we present data enhancement techniques as a solution to the data scarcity problem in neural sequence-to-sequence learning. We categorize our proposed techniques into three research directions: 1) increasing the size of effective training data, 2) selecting instance relevant training data, and 3) incorporating simple but effective human feedback. We develop corresponding algorithms and evaluate their effectiveness on major tasks in natural language processing and speech processing such as speech recognition and speech/text-to-text translation.

To increase the size of effective training data, we propose within-corpus data augmentation that works in a 2-step procedure: 1) segmentation of the parallel corpus and 2) recombination of the segments. Unlike augmentation methods such as pseudo-labeling and noise injection, within-corpus data augmentation remains simple and effective through possessing three properties: 1) *on-the-fly*, 2) *memory-efficient* and 3) *source-target alignment*. Following these properties, we have developed ADA, STR and concatenation-based methods for speech-to-text applications. On widely used datasets, our augmentation techniques achieve significant improvement over their baselines.

The next direction is instance-specific data selection. It differs from generic data filtering, which focuses on overall training data quality, by selecting relevant training instances in relation to specified data instances. This enhances model performance on the specified instances and potentially reduce nuanced errors, such as mis-translation of country names and dates. We demonstrate its usefulness through building algorithms for two scenarios: 1) selection of pseudo-labels in a cascade speech-to-text translation system and 2) back-tracking model's prediction back to its training data in neural machine translation. While being effective, we identify the costly instance-specific computation as the major bottleneck.

The last direction combines minimal human feedback and model learning in an

interactive manner to fix problems that require human intervention. Our proposed interactive learning protocols take simple human instructions guided by active learning and perform online learning in sub-sequence level, resulting in better sample efficiency and faster model adaptation/personalization. In experiments of using simulated human feedback, our algorithms show competitive performance with substantial effort reduction to their baselines, e.g., a model trained with gold-reference translations.

For future work, we would like to extend our works to other sequence-to-sequence tasks such as speech synthesis and image translation. Last but not the least, we would like to make our algorithms more sample-efficient and interpretable.

Bibliography

Naman Agarwal, Brian Bullins, and Elad Hazan. 2017. Second-order stochastic optimization for machine learning in linear time. *The Journal of Machine Learning Research*, 18(1):4148–4187.

Ekin Akyürek, Tolga Bolukbasi, Frederick Liu, Binbin Xiong, Ian Tenney, Jacob Andreas, and Kelvin Guu. 2022. Tracing knowledge in language models back to the training data. In *arXiv preprint arXiv: 2205.11482*.

Antonios Anastasopoulos and David Chiang. 2018. Tied multitask learning for neural speech translation. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL:HLT)*, pages 82–91, New Orleans, LA, USA. ACL.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. 2020. Common voice: A massively-multilingual speech corpus. In *Language Resources and Evaluation Conference (LREC)*, pages 4218–4222, Marseille, France. ELRA.

Mikel Artetxe and Holger Schwenk. 2019. Margin-based parallel corpus mining with multilingual sentence embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3197–3203, Florence, Italy. Association for Computational Linguistics.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 355–362. ACL.

Parnia Bahar, Tobias Bieschke, and Hermann Ney. 2019. A comparative study on end-to-end speech to text translation. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 792–799, Sentosa, Singapore. IEEE.

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 5th 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 *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Pratyush Banerjee, Sudip Kumar Naskar, Johann Roturier, Andy Way, and Josef van Genabith. 2011. Domain adaptation in statistical machine translation of user-forum data using component level mixture modelling. In *Proceedings of Machine Translation Summit XIII: Papers, MTSummit 2011, Xiamen, China, September 19-23, 2011*.

Marta Bañón, Pinzhen Chen, Barry Haddow, Kenneth Heafield, Hieu Hoang, Miquel Esplà-Gomis, Mikel L Forcada, Amir Kamran, Faheem Kirefu, Philipp Koehn, et al. 2020. Paracrawl: Web-scale acquisition of parallel corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4555–4567.

Sergio Barrachina, Oliver Bender, Francisco Casacuberta, Jorge Civera, Elsa Cubel, Shahram Khadivi, Antonio Lagarda, Hermann Ney, Jesús Tomás, Enrique Vidal, and Juan-Miguel Vilar. 2008. Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1):3–28.

Sergio Barrachina, Oliver Bender, Francisco Casacuberta, Jorge Civera, Elsa Cubel, Shahram Khadivi, Antonio Lagarda, Hermann Ney, Jesús Tomás, Enrique Vidal, et al. 2009. Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1):3–28.

Elnaz Barshan, Marc-Etienne Brunet, and Gintare Karolina Dziugaite. 2020. Relatif: Identifying explanatory training samples via relative influence. In *International Conference on Artificial Intelligence and Statistics*, pages 1899–1909. PMLR.

Andrew G Barto, Richard S Sutton, and Charles W Anderson. 1983. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846.

Timo Baumann, Arne Köhn, and Felix Hennig. 2019. The spoken wikipedia corpus collection: Harvesting, alignment and an application to hyperlistening. *Lang. Resour. Evaluation*, 53(2):303–329.

Daniel Beck, Trevor Cohn, and Gholamreza Haffari. 2019. Neural speech translation using lattice transformations and graph networks. In *Proceedings of the Thirteenth*

Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13), pages 26–31, Hong Kong. Association for Computational Linguistics.

Benjamin Beilharz, Xin Sun, Sariya Karimova, and Stefan Riezler. 2020. LibriVoxDeEn: A corpus for german-to-english speech translation and german speech recognition. In *Language Resources and Evaluation Conference (LREC)*, pages 3590–3594, Marseille, France. ELRA.

RICHARD Bellman. 1957. Dynamic programming, princeton univ. *Press Princeton, New Jersey*.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2000. A neural probabilistic language model. *Advances in neural information processing systems*, 13.

Alexandre Bérard, Laurent Besacier, Ali Can Kocabiyikoglu, and Olivier Pietquin. 2018. End-to-end automatic speech translation of audiobooks. In *Proc. ICASSP*, pages 6224–6228, Calgary, AB, Canada. IEEE.

N. Bertoldi and M. Federico. 2005. A new decoder for spoken language translation based on confusion networks. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU 2005)*, pages 86–91. IEEE.

Maximilian Bisani and Hermann Ney. 2004. Bootstrap estimates for confidence intervals in ASR performance evaluation. In *ICASSP*, pages 409–412. IEEE.

John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2004. Confidence estimation for machine translation. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 315–321, Geneva, Switzerland. COLING.

Leo Breiman and Nong Shang. 1996. Born again trees. *University of California, Berkeley, Berkeley, CA, Technical Report*, 1(2):4.

Rich Caruana. 1998. *Multitask learning*. Springer.

Isaac Caswell, Theresa Breiner, Daan van Esch, and Ankur Bapna. 2020. Language ID in the wild: Unexpected challenges on the path to a thousand-language web text corpus. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 6588–6608. International Committee on Computational Linguistics.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. 2016a. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4960–4964. IEEE.

William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals. 2016b. Listen, attend

and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, China, March 20-25, 2016*, pages 4960–4964. IEEE.

Yang Chen, Weiran Wang, and Chao Wang. 2020. Semi-supervised ASR by end-to-end self-training. In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pages 2787–2791. ISCA.

Zhehuai Chen, Andrew Rosenberg, Yu Zhang, Heiga Zen, Mohammadreza Ghodsi, Yinghui Huang, Jesse Emond, Gary Wang, Bhuvana Ramabhadran, and Pedro J. Moreno. 2021. Semi-supervision in ASR: sequential mixmatch and factorized tts-based augmentation. In *INTERSPEECH*, pages 736–740. ISCA.

Ching-An Cheng, Xinyan Yan, Nolan Wagener, and Byron Boots. 2018. Fast policy learning through imitation and reinforcement. In *Uncertainty in Artificial Intelligence (UAI)*, Monterey, CA, USA.

Chung-Cheng Chiu, Tara N. Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J. Weiss, Kanishka Rao, Ekaterina Gonina, Navdeep Jaitly, Bo Li, Jan Chorowski, and Michiel Bacchiani. 2018. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*, pages 4774–4778. IEEE.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the properties of neural machine translation: Encoder-decoder approaches. In *Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014*, pages 103–111. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. 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 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL.

Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. In *Proc. NIPS*, pages 577–585, Montreal, QC, Canada.

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*, pages 176–181, Portland, Oregon, USA. Association for Computational Linguistics.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2016. Fast and accurate deep network learning by exponential linear units (elus). In *Proc. ICLR*, San Juan, Puerto Rico.

David Cohn, Les Atlas, and Richard Ladner. 1994. Improving generalization with active learning. *Machine learning*, 15:201–221.

R Dennis Cook and Sanford Weisberg. 1982. *Residuals and influence in regression*. New York: Chapman and Hall.

Ekin D. Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. 2019. Autoaugment: Learning augmentation strategies from data. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 113–123. Computer Vision Foundation / IEEE.

Xiaodong Cui, Vaibhava Goel, and Brian Kingsbury. 2015. Data augmentation for deep neural network acoustic modeling. *IEEE ACM Trans. Audio Speech Lang. Process.*, 23(9):1469–1477.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. 2017. Language modeling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 933–941. PMLR.

Michael J. Denkowski, Greg Hanneman, and Alon Lavie. 2012. The cmu-avenue french-english translation system. In *Proceedings of the Seventh Workshop on Statistical Machine Translation, WMT@NAACL-HLT 2012, June 7-8, 2012, Montréal, Canada*, pages 261–266. The Association for Computer Linguistics.

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, Volume 1*

(Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Miguel Domingo, Álvaro Peris, and Francisco Casacuberta. 2017. Segment-based interactive-predictive machine translation. *Machine Translation*, 31(4):163–185.

Zi-Yi Dou, Antonios Anastasopoulos, and Graham Neubig. 2020. Dynamic data selection and weighting for iterative back-translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5894–5904. Association for Computational Linguistics.

Long Duong, Antonios Anastasopoulos, David Chiang, Steven Bird, and Trevor Cohn. 2016. An attentional model for speech translation without transcription. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL:HLT)*, pages 949–959, San Diego, CA, USA. ACL.

Chris Dyer, Jonathan Weese, Hendra Setiawan, Adam Lopez, Ferhan Ture, Vladimir Eidelman, Juri Ganitkevitch, Phil Blunsom, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the ACL 2010 System Demonstrations (ACL Demo)*, Uppsala, Sweden.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018a. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018b. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 489–500. Association for Computational Linguistics.

Michael Evans and Timothy Swartz. 2000. *Approximating integrals via Monte Carlo and deterministic methods*, volume 20. OUP Oxford.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017a. Data augmentation for low-resource neural machine translation. In *ACL*, volume 2, pages 567–573, Vancouver, Canada. Association for Computational Linguistics.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017b. Data augmentation for low-resource neural machine translation. In *ACL*, pages 567–573. ACL.

Ronald A. Fisher. 1935. *The Design of Experiments*. Hafner, New York, NY, USA.

George Fishman. 2013. *Monte Carlo: concepts, algorithms, and applications*. Springer Science & Business Media.

George Foster, Pierre Isabelle, and Pierre Plamondon. 1997. Target-text mediated interactive machine translation. *Machine Translation*, 12(1):175–194.

George Foster, Philippe Langlais, and Guy Lapalme. 2002. User-friendly text prediction for translators. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Philadelphia, PA.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Proc. NIPS*, pages 1019–1027, Barcelona, Spain.

Simona Gandrabur and George Foster. 2003. Confidence estimation for translation prediction. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 95–102.

Rashmi Gangadharaiah, Ralf D. Brown, and Jaime Carbonell. 2009. Active learning in example-based machine translation. In *Proceedings of the 17th Nordic Conference of Computational Linguistics (NODALIDA 2009)*, pages 227–230, Odense, Denmark. Northern European Association for Language Technology (NEALT).

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 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252. PMLR.

L. Gillick and Stephen J. Cox. 1989. Some statistical issues in the comparison of speech recognition algorithms. In *ICASSP*, pages 532–535. IEEE.

Jesús González-Rubio and Francisco Casacuberta. 2014. Cost-sensitive active learning for computer-assisted translation. *Pattern Recognition Letters*, 37:124–134.

Jesús González-Rubio, Daniel Ortiz-Martínez, and Francisco Casacuberta. 2011. An active learning scenario for interactive machine translation. In *Proceedings of the 13th International Conference on Multimodal Interfaces (ICMI)*, Barcelona, Spain.

Jesús González-Rubio, Daniel Ortiz-Martínez, and Francisco Casacuberta. 2012. Active learning for interactive machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Avignon, France.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Machine Learning, Proceedings of the Twenty-Third*

International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006, volume 148 of *ACM International Conference Proceeding Series*, pages 369–376. ACM.

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 of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented transformer for speech recognition. In *INTERSPEECH*, pages 5036–5040. ISCA.

Han Guo, Nazneen Rajani, Peter Hase, Mohit Bansal, and Caiming Xiong. 2021. Fastif: Scalable influence functions for efficient model interpretation and debugging. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 10333–10350. Association for Computational Linguistics.

Gholamreza Haffari, Maxim Roy, and Anoop Sarkar. 2009. Active learning for statistical phrase-based machine translation. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, May 31 - June 5, 2009, Boulder, Colorado, USA*, pages 415–423. The Association for Computational Linguistics.

Frank R Hampel. 1974. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393.

Xiaochuang Han and Yulia Tsvetkov. 2020. Fortifying toxic speech detectors against veiled toxicity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7732–7739. Association for Computational Linguistics.

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, et al. 2018. Achieving human parity on automatic chinese to english news translation. *arXiv preprint arXiv:1803.05567*.

Tomoki Hayashi, Shinji Watanabe, Yu Zhang, Tomoki Toda, Takaaki Hori, Ramón Fer-

nandez Astudillo, and Kazuya Takeda. 2018. Back-translation-style data augmentation for end-to-end ASR. In *2018 IEEE Spoken Language Technology Workshop, SLT 2018, Athens, Greece, December 18-21, 2018*, pages 426–433. IEEE.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2020. Revisiting self-training for neural sequence generation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Xiaodong He, Li Deng, and Alex Acero. 2011. Why word error rate is not a good metric for speech recognizer training for the speech translation task? In *Proc. ICASSP*, pages 5632–5635, Prague, Czech Republic. IEEE.

Wassily Hoeffding. 1952. The large-sample power of tests based on permutations of observations. *Annals of Mathematical Statistics*, 23:169–192.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *ACL*, Vancouver, Canada.

Takaaki Hori, Ramón Fernandez Astudillo, Tomoki Hayashi, Yu Zhang, Shinji Watanabe, and Jonathan Le Roux. 2019. Cycle-consistency training for end-to-end speech recognition. In *ICASSP*, pages 6271–6275. IEEE.

Junjie Hu and Graham Neubig. 2021. Phrase-level active learning for neural machine translation. In *Proceedings of the Sixth Conference on Machine Translation*, pages 1087–1099, Online. Association for Computational Linguistics.

Ting-Yao Hu, Ashish Shrivastava, Jen-Hao Rick Chang, Hema Koppula, Stefan Braun, Kyuyeon Hwang, Ozlem Kalinli, and Oncel Tuzel. 2021. Sapaugment: Learning A sample adaptive policy for data augmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pages 4040–4044. IEEE.

H. J. Scudder III. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Trans. Inf. Theory*, 11(3):363–371.

Hirofumi Inaguma, Tatsuya Kawahara, and Shinji Watanabe. 2021. Source and target bidirectional knowledge distillation for end-to-end speech translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 1872–1881. Association for Computational Linguistics.

Sathish Reddy Indurthi, Houjeung Han, Nikhil Kumar Lakumarapu, Beomseok Lee, Insoo Chung, Sangha Kim, and Chanwoo Kim. 2020. End-end speech-to-text

translation with modality agnostic meta-learning. In *Proc. ICASSP*, pages 7904–7908, Online. IEEE.

Javier Iranzo-Sánchez, Joan Albert Silvestre-Cerdà, Javier Jorge, Nahuel Roselló, Adrià Giménez, Albert Sanchis, Jorge Civera, and Alfons Juan. 2020. Europarl-st: A multilingual corpus for speech translation of parliamentary debates. In *Proceedings of ICASSP 2020*, pages 8229–8233, Barcelona, Spain. IEEE.

Navdeep Jaitly and Geoffrey E. Hinton. 2013. Vocal tract length perturbation (vtlp) improves speech recognition. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Ye Jia, Melvin Johnson, Wolfgang Macherey, Ron J. Weiss, Yuan Cao, Chung-Cheng Chiu, Naveen Ari, Stella Laurenzo, and Yonghui Wu. 2019. Leveraging weakly supervised data to improve end-to-end speech-to-text translation. In *Proceedings of ICASSP 2019*, Brighton, UK. IEEE.

Ye Jia, Michelle Tadmor Ramanovich, Tal Remez, and Roi Pomerantz. 2022. Translatotron 2: High-quality direct speech-to-speech translation with voice preservation. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 10120–10134. PMLR.

Ajay J. Joshi, Fatih Porikli, and Nikolaos Papanikolopoulos. 2009. Multi-class active learning for image classification. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 2372–2379. IEEE Computer Society.

Marcin Junczys-Dowmunt. 2018. Dual conditional cross-entropy filtering of noisy parallel corpora. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pages 888–895. Association for Computational Linguistics.

Jacob Kahn, Ann Lee, and Awni Hannun. 2020. Self-training for end-to-end speech recognition. In *ICASSP*, pages 7084–7088. IEEE.

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.

Shigeki Karita, Xiaofei Wang, Shinji Watanabe, Takenori Yoshimura, Wangyou Zhang, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang,

Masao Someki, Nelson Enrique Yalta Soplin, and Ryuichi Yamamoto. 2019. A comparative study on transformer vs RNN in speech applications. In *ASRU*, pages 449–456. IEEE.

Yaser Keneshloo, Tian Shi, Naren Ramakrishnan, and Chandan K. Reddy. 2020. Deep reinforcement learning for sequence-to-sequence models. *IEEE Trans. Neural Networks Learn. Syst.*, 31(7):2469–2489.

Shahram Khadivi and Hermann Ney. 2005. Automatic filtering of bilingual corpora for statistical machine translation. In *International Conference on Application of Natural Language to Information Systems*, pages 263–274. Springer.

Huda Khayrallah and Philipp Koehn. 2018. On the impact of various types of noise on neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation, NMT@ACL 2018, Melbourne, Australia, July 20, 2018*, pages 74–83. Association for Computational Linguistics.

Suyoun Kim, Duc Le, Weiyi Zheng, Tarun Singh, Abhinav Arora, Xiaoyu Zhai, Christian Fuegen, Ozlem Kalinli, and Michael L. Seltzer. 2022. Evaluating user perception of speech recognition system quality with semantic distance metric. In *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, pages 3978–3982. ISCA.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1317–1327. The Association for Computational Linguistics.

Hajime Kimura, Shigenobu Kobayashi, et al. 1998. An analysis of actor/critic algorithms using eligibility traces: Reinforcement learning with imperfect value function. In *ICML*, volume 98.

Hajime Kimura, Kazuteru Miyazaki, and Shigenobu Kobayashi. 1997. Reinforcement learning in pomdps with function approximation. In *ICML*, volume 97, pages 152–160.

Hajime Kimura, Masayuki Yamamura, and Shigenobu Kobayashi. 1995. Reinforcement learning by stochastic hill climbing on discounted reward. In *Machine Learning Proceedings 1995*, pages 295–303. Elsevier.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR 2015*, San Diego, CA, USA.

Rebecca Knowles and Philipp Koehn. 2016. Neural interactive translation prediction.

In *North American component of the International Association for Machine Translation (AMTA)*, Austin, TX, USA.

Rebecca Knowles et al. 2019. *Interactive and Adaptive Neural Machine Translation*. Ph.D. thesis, Johns Hopkins University.

Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. 2015. Audio augmentation for speech recognition. In *INTERSPEECH*, pages 3586–3589. ISCA.

Tom Kocmi, Christian Federmann, Roman Grundkiewicz, Marcin Junczys-Dowmunt, Hitokazu Matsushita, and Arul Menezes. 2021. To ship or not to ship: An extensive evaluation of automatic metrics for machine translation. In *Proceedings of the Sixth Conference on Machine Translation, WMT@EMNLP 2021, Online Event, November 10-11, 2021*, pages 478–494. Association for Computational Linguistics.

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 Companion Volume Proceedings of the Demo and Poster Sessions (ACL Demo)*, Prague, Czech Republic.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR.

Vijay Konda and John Tsitsiklis. 1999. Actor-critic algorithms. *Advances in neural information processing systems*, 12.

Vijay R Konda and John N Tsitsiklis. 2003. Onactor-critic algorithms. *SIAM journal on Control and Optimization*, 42(4):1143–1166.

Wouter M Kouw and Marco Loog. 2018. An introduction to domain adaptation and transfer learning. *arXiv preprint arXiv:1812.11806*.

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 2020, Lisboa, Portugal, November 3-5, 2020*, pages 135–144. European Association for Machine Translation.

Julia Kreutzer, Shigehiko Schamoni, and Stefan Riezler. 2015. Quality estimation from scratch (QUETCH): deep learning for word-level translation quality estimation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation,*

WMT@EMNLP 2015, 17-18 September 2015, Lisbon, Portugal, pages 316–322. The Association for Computer Linguistics.

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. 2018a. 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)*.

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 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1777–1788. Association for Computational Linguistics.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of EMNLP: System Demonstrations*, pages 66–71. Association for Computational Linguistics.

Tsz Kin Lam, Eva Hasler, and Felix Hieber. 2022a. Analyzing the use of influence functions for instance-specific data filtering in neural machine translation. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 295–309, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

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, Mayumi Ohta, Shigehiko Schamoni, and Stefan Riezler. 2021a. On-the-fly aligned data augmentation for sequence-to-sequence asr. In *Proceedings of INTERSPEECH 2021*, Brno, Czech Republic. ISCA.

Tsz Kin Lam, Shigehiko Schamoni, and Stefan Riezler. 2019. Interactive-predictive neural machine translation through reinforcement and imitation. In *Proceedings*

of Machine Translation Summit XVII: Research Track, pages 96–106, Dublin, Ireland. European Association for Machine Translation.

Tsz Kin Lam, Shigehiko Schamoni, and Stefan Riezler. 2021b. Cascaded models with cyclic feedback for direct speech translation. In *Proceedings of ICASSP 2021*, pages 7508–7512, Toronto, ON, Canada. IEEE.

Tsz Kin Lam, Shigehiko Schamoni, and Stefan Riezler. 2022b. Sample, translate, recombine: Leveraging audio alignments for data augmentation in end-to-end speech translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 245–254, Dublin, Ireland. Association for Computational Linguistics.

Tsz Kin Lam, Shigehiko Schamoni, and Stefan Riezler. 2023. Make more of your data: Minimal effort data augmentation for automatic speech recognition and translation. In *2023 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2023, Rhodes Island, Greece, June 4-10, 2023, To appear*.

Aleksandr Laptev, Roman Korostik, Aleksey Svischev, Andrei Andrusenko, Ivan Medennikov, and Sergey Rybin. 2020. You do not need more data: Improving end-to-end speech recognition by text-to-speech data augmentation. In *CISP-BMEI*, pages 439–444. IEEE.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324.

Ann Lee, Hongyu Gong, Paul-Ambroise Duquenne, Holger Schwenk, Peng-Jen Chen, Changhan Wang, Sravya Popuri, Yossi Adi, Juan Pino, Jiatao Gu, and Wei-Ning Hsu. 2022. Textless speech-to-speech translation on real data. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 860–872, Seattle, United States. Association for Computational Linguistics.

Jared Lichtarge, Chris Alberti, and Shankar Kumar. 2020. Data weighted training strategies for grammatical error correction. *Trans. Assoc. Comput. Linguistics*, 8:634–646.

Alexander Liu, Hung-yi Lee, and Lin-shan Lee. 2019a. Adversarial training of end-to-end speech recognition using a criticizing language model. In *Proc. ICASSP*, pages 6979–6983, Brighton, UK. IEEE.

Xuebo Liu, Longyue Wang, Derek F. Wong, Liang Ding, Lidia S. Chao, Shuming Shi, and Zhaopeng Tu. 2021. On the complementarity between pre-training and back-translation for neural machine translation. In *Findings of the Association*

for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021, pages 2900–2907. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Yuchen Liu, Hao Xiong, Jiajun Zhang, Zhongjun He, Hua Wu, Haifeng Wang, and Chengqing Zong. 2019c. End-to-end speech translation with knowledge distillation. In *Proc. INTERSPEECH*, pages 1128–1132, Graz, Austria. ISCA.

Yiping Lu, Zhuohan Li, Di He, Zhiqing Sun, Bin Dong, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Understanding and improving transformer from a multi-particle dynamic system point of view. *CoRR*, abs/1906.02762.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421. The Association for Computational Linguistics.

Christoph Lüscher, Eugen Beck, Kazuki Irie, Markus Kitza, Wilfried Michel, Albert Zeyer, Ralf Schlüter, and Hermann Ney. 2019. RWTH ASR systems for librispeech: Hybrid vs attention. In *INTERSPEECH*, pages 231–235. ISCA.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Anirudh Mani, Shruti Palaskar, Nimshi Venkat Meripo, Sandeep Konam, and Florian Metze. 2020. ASR error correction and domain adaptation using machine translation. In *Proc. ICASSP*, pages 6344–6348, Barcelona, Spain. IEEE.

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.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. 2017. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Proceedings of INTERSPEECH 2017*, volume 2017, pages 498–502, Stockholm, Sweden. ISCA.

Arya D. McCarthy, Liezl Puzon, and Juan Miguel Pino. 2020. Skinaugment: Auto-

encoding speaker conversions for automatic speech translation. In *Proceedings of ICASSP 2020*, pages 7924–7928, Barcelona, Spain. IEEE.

Linghui Meng, Jin Xu, Xu Tan, Jindong Wang, Tao Qin, and Bo Xu. 2021. Mixspeech: Data augmentation for low-resource automatic speech recognition. In *ICASSP*, pages 7008–7012. IEEE.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018. Regularizing and optimizing LSTM language models. In *Proc. ICLR*, Vancouver, BC, Canada.

Paul Michel and Graham Neubig. 2018. Extreme adaptation for personalized neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, Melbourne, Australia.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Akiva Miura, Graham Neubig, Michael Paul, and Satoshi Nakamura. 2016. Selecting syntactic, non-redundant segments in active learning for machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 20–29, San Diego, California. Association for Computational Linguistics.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, New York, NY.

Robert C. Moore and William D. Lewis. 2010. Intelligent selection of language model training data. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden, Short Papers*, pages 220–224. The Association for Computer Linguistics.

Nathan Ng, Kyunghyun Cho, and Marzyeh Ghassemi. 2020. SSMBA: Self-supervised manifold based data augmentation for improving out-of-domain robustness. In *EMNLP*, pages 1268–1283. ACL.

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.

Thai-Son Nguyen, Sebastian Stüker, Jan Niehues, and Alex Waibel. 2020. Improving

sequence-to-sequence speech recognition training with on-the-fly data augmentation. In *ICASSP*, pages 7689–7693. IEEE.

Toan Q. Nguyen, Kenton Murray, and David Chiang. 2021. Data augmentation by concatenation for low-resource translation: A mystery and a solution. In *Proceedings of IWSLT*, pages 287–293. Association for Computational Linguistics.

Toan Q. Nguyen and Julian Salazar. 2019. Transformers without tears: Improving the normalization of self-attention. In *Proceedings of the 16th International Conference on Spoken Language Translation*, Hong Kong. Association for Computational Linguistics.

Eric W. Noreen. 1989. *Computer Intensive Methods for Testing Hypotheses. An Introduction*. Wiley, New York.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 160–167, Sapporo, Japan. ACL.

Daniel Ortiz-Martínez, Ismael García-Varea, and Francisco Casacuberta. 2010. Online learning for interactive statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, Los Angeles, CA.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT: Demonstrations*, pages 48–53. Association for Computational Linguistics.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An ASR corpus based on public domain audio books. In *ICASSP*, pages 5206–5210. IEEE.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL 2002*, pages 311–318, Philadelphia, PA, USA. Association for Computational Linguistics.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin Dogus Cubuk, and Quoc V. Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. In *Proceedings of INTERSPEECH 2019*, pages 2613–2617, Graz, Austria. ISCA.

Daniel S. Park, Yu Zhang, Ye Jia, Wei Han, Chung-Cheng Chiu, Bo Li, Yonghui Wu, and Quoc V. Le. 2020. Improved noisy student training for automatic speech recognition. In *INTERSPEECH*, pages 2817–2821. ISCA.

Stephan Peitz, Simon Wiesler, Markus Nussbaum-Thom, and Hermann Ney. 2012. Spoken language translation using automatically transcribed text in training. In *International Workshop on Spoken Language Translation (IWSLT)*, pages 276–283, Hong Kong, China. ISCA.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Álvaro Peris, Luis Cebrián, and Francisco Casacuberta. 2017. Online learning for neural machine translation post-editing. *CoRR*, abs/1706.03196.

Juan Miguel Pino, Liezl Puzon, Jiatao Gu, Xutai Ma, Arya D. McCarthy, and Deepak Gopinath. 2019. Harnessing indirect training data for end-to-end automatic speech translation: Tricks of the trade. In *Proceedings of IWSLT 2019*, Hong Kong, China.

Maja Popović. 2015. chrF: character n-gram f-score for automatic MT evaluation. In *Proceedings of WMT*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Maja Popović. 2016. chrF deconstructed: beta parameters and n-gram weights. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 499–504, Berlin, Germany. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Matt Post, Gaurav Kumar, Adam Lopez, Damianos Karakos, Chris Callison-Burch, and Sanjeev Khudanpur. 2013. Improved speech-to-text translation with the fisher and callhome spanish-english speech translation corpus. In *Proceedings of the 10th International Workshop on Spoken Language Translation: Papers*.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. 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.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33:19920–19930.

Christopher Quirk. 2004. Training a sentence-level machine translation confidence measure. In *LREC*.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. 2018. Searching for activation functions. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net.

Gema Ramírez-Sánchez, Jaume Zaragoza-Bernabeu, Marta Bañón, and Sergio Ortiz-Rojas. 2020. Bifixer and bicleaner: two open-source tools to clean your parallel data. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 291–298.

Yi Ren, Jinglin Liu, Xu Tan, Zhou Zhao, Sheng Zhao, and Tie-Yan Liu. 2020. A study of non-autoregressive model for sequence generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 149–159, Online. Association for Computational Linguistics.

Stefan Riezler and John T. Maxwell. 2005. On some pitfalls in automatic evaluation and significance testing for MT. In *ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 57–64. ACL.

Herbert E. Robbins. 1951. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407.

Andrew Rosenberg, Yu Zhang, Bhuvana Ramabhadran, Ye Jia, Pedro J. Moreno, Yonghui Wu, and Zelin Wu. 2019. Speech recognition with augmented synthesized speech. In *IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2019, Singapore, December 14-18, 2019*, pages 996–1002. IEEE.

Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics (AISTATS)*, Fort Lauderdale, FL, USA.

Nick Rossenbach, Albert Zeyer, Ralf Schlüter, and Hermann Ney. 2020. Generating synthetic audio data for attention-based speech recognition systems. In *ICASSP*, pages 7069–7073. IEEE.

Nicholas Ruiz, Qin Gao, Will Lewis, and Marcello Federico. 2015. Adapting machine translation models toward misrecognized speech with text-to-speech pronunciation

rules and acoustic confusability. In *Proceedings of INTERSPEECH 2015*, pages 2247–2251, Dresden, Germany. ISCA.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

Elizabeth Salesky, Matthias Sperber, and Alan W Black. 2019. Exploring phoneme-level speech representations for end-to-end speech translation. In *ACL*, pages 1835–1841. ACL.

Víctor M. Sánchez-Cartagena, Miquel Esplà-Gomis, Juan Antonio Pérez-Ortiz, and Felipe Sánchez-Martínez. 2021. Rethinking data augmentation for low-resource neural machine translation: A multi-task learning approach. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8502–8516, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. 2022. Scaling up influence functions. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 8179–8186. AAAI Press.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia. In *Proceedings of EACL 2021: Main Volume*, pages 1351–1361, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Improving neural machine translation models with monolingual data. In *ACL*, pages 86–96. ACL.

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.

Patrice Y. Simard, David Steinkraus, and John C. Platt. 2003. Best practices for

convolutional neural networks applied to visual document analysis. In *ICDAR*, pages 958–962. IEEE.

Lucia Specia, Marco Turchi, Nicola Cancedda, Nello Cristianini, and Marc Dymetman. 2009. Estimating the sentence-level quality of machine translation systems. In *Proceedings of the 13th Annual conference of the European Association for Machine Translation, EAMT 2009, Barcelona, Spain, Map 14-15, 2009*. European Association for Machine Translation.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2019a. Attention-passing models for robust and data-efficient end-to-end speech translation. *Transactions of the Association for Computational Linguistics*, 7:313–325.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alexander H. Waibel. 2019b. Attention-passing models for robust and data-efficient end-to-end speech translation. *Transactions of the Association for Computational Linguistics (TACL)*, 7:313–325.

Matthias Sperber and Matthias Paulik. 2020a. Speech translation and the end-to-end promise: Taking stock of where we are. In *Proceedings of ACL 2020*, pages 7409–7421, Online. Association for Computational Linguistics.

Matthias Sperber and Matthias Paulik. 2020b. Speech translation and the end-to-end promise: Taking stock of where we are. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 7409–7421, Online. ACL.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning. An Introduction*, second edition. The MIT Press.

Richard S. Sutton, David A. McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pages 1057–1063. The MIT Press.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processings Systems (NIPS)*, Denver, CO, USA.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Kaveh Taghipour, Nasim Afhami, Shahram Khadivi, and Saeed Shiry. 2010. A discriminative approach to filter out noisy sentence pairs from bilingual corpora. In *2010 5th International Symposium on Telecommunications*, pages 537–541. IEEE.

Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. 2017. Listening while speaking: Speech chain by deep learning. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2017, Okinawa, Japan, December 16-20, 2017*, pages 301–308. IEEE.

Shubham Toshniwal, Anjuli Kannan, Chung-Cheng Chiu, Yonghui Wu, Tara N. Sainath, and Karen Livescu. 2018. A comparison of techniques for language model integration in encoder-decoder speech recognition. In *2018 IEEE spoken language technology workshop (SLT)*, pages 369–375. IEEE.

Yulia Tsvetkov, Florian Metze, and Chris Dyer. 2014. Augmenting translation models with simulated acoustic confusions for improved spoken language translation. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 616–625, Gothenburg, Sweden. ACL.

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 (PBML)*, 108(1):233–244.

Nicola Ueffing, José G. C. de Souza, and Gregor Leusch. 2018. Quality estimation for automatically generated titles of ecommerce browse pages. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 3 (Industry Papers)*, pages 52–59. Association for Computational Linguistics.

Nicola Ueffing, Klaus Macherey, and Hermann Ney. 2003. Confidence measures for statistical machine translation. In *Proceedings of Machine Translation Summit IX: Papers*.

Nicola Ueffing and Hermann Ney. 2007. Word-level confidence estimation for machine translation. *Comput. Linguistics*, 33(1):9–40.

Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. In *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, 13-15 September 2016*, page 125. ISCA.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In

Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 5998–6008.

Changhan Wang, Juan Miguel Pino, Anne Wu, and Jiatao Gu. 2020a. Covost: A diverse multilingual speech-to-text translation corpus. In *Proceedings of LREC*, pages 4197–4203. European Language Resources Association.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020b. Fairseq S2T: Fast speech-to-text modeling with fairseq. In *AACL: System Demonstrations*, pages 33–39. ACL.

Changhan Wang, Anne Wu, Jiatao Gu, and Juan Pino. 2021a. CoVoST 2 and Massively Multilingual Speech Translation. In *Proceedings of INTERSPEECH 2021*, pages 2247–2251, Brno, Czech Republic.

Changhan Wang, Anne Wu, Juan Pino, Alexei Baevski, Michael Auli, and Alexis Conneau. 2021b. Large-scale self- and semi-supervised learning for speech translation. In *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pages 2242–2246. ISCA.

Chaojun Wang and Rico Sennrich. 2020. On exposure bias, hallucination and domain shift in neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3544–3552, Online. Association for Computational Linguistics.

Gary Wang, Andrew Rosenberg, Zhehuai Chen, Yu Zhang, Bhuvana Ramabhadran, Yonghui Wu, and Pedro J. Moreno. 2020c. Improving speech recognition using consistent predictions on synthesized speech. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 7029–7033. IEEE.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. 2019. Learning deep transformer models for machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1810–1822, Florence, Italy. Association for Computational Linguistics.

Wei Wang, Ye Tian, Jiquan Ngiam, Yinfei Yang, Isaac Caswell, and Zarana Parekh. 2020d. Learning a multi-domain curriculum for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7711–7723. Association for Computational Linguistics.

Wei Wang, Taro Watanabe, Macduff Hughes, Tetsuji Nakagawa, and Ciprian Chelba. 2018a. Denoising neural machine translation training with trusted data and online data selection. In *Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pages 133–143. Association for Computational Linguistics.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018b. SwitchOut: an efficient data augmentation algorithm for neural machine translation. In *EMNLP*, pages 856–861. ACL.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. Espnet: End-to-end speech processing toolkit. In *INTERSPEECH*, pages 2207–2211. ISCA.

Shinji Watanabe, Takaaki Hori, Suyoun Kim, John R Hershey, and Tomoki Hayashi. 2017. Hybrid ctc/attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 11(8):1240–1253.

Ron J. Weiss, Jon Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. 2017. Sequence-to-sequence models can directly translate foreign speech. In *Proc. INTERSPEECH*, pages 2625–2629, Stockholm, Sweden. ISCA.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256.

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. *CoRR*, abs/1609.08144.

Joern Wuebker, Spence Green, John DeNero, Sasa Hasan, and Minh-Thang Luong. 2016. Models and inference for prefix-constrained machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, Berlin, Germany.

Joern Wuebker, Patrick Simianer, and John DeNero. 2018. Compact personalized models for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Brussels, Belgium.

Qiantong Xu, Alexei Baevski, Tatiana Likhomanenko, Paden Tomasello, Alexis Con-

neau, Ronan Collobert, Gabriel Synnaeve, and Michael Auli. 2021a. Self-training and pre-training are complementary for speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pages 3030–3034. IEEE.

Qiantong Xu, Tatiana Likhomanenko, Jacob Kahn, Awni Y. Hannun, Gabriel Synnaeve, and Ronan Collobert. 2020. Iterative pseudo-labeling for speech recognition. In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pages 1006–1010. ISCA.

Weijia Xu, Shuming Ma, Dongdong Zhang, and Marine Carpuat. 2021b. How does distilled data complexity impact the quality and confidence of non-autoregressive machine translation? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4392–4400, Online. Association for Computational Linguistics.

Lingxuan Ye, Gaofeng Cheng, Runyan Yang, Zehui Yang, Sanli Tian, Pengyuan Zhang, and Yonghong Yan. 2022. Improving recognition of out-of-vocabulary words in E2E code-switching ASR by fusing speech generation methods. In *INTERSPEECH*, pages 3163–3167. ISCA.

Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV 2019*, pages 6022–6031, Seoul, Korea (South). IEEE.

Xiangkai Zeng, Sarthak Garg, Rajen Chatterjee, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Empirical evaluation of active learning techniques for neural MT. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 84–93, Hong Kong, China. Association for Computational Linguistics.

Pei Zhang, Xueying Xu, and Deyi Xiong. 2018. Active learning for neural machine translation. In *2018 International Conference on Asian Language Processing, IALP 2018, Bandung, Indonesia, November 15-17, 2018*, pages 153–158. IEEE.

Ruiqiang Zhang, Genichiro Kikui, Hirofumi Yamamoto, Frank Soong, Taro Watnabe, and Wai Kit Lo. 2004. A unified approach in speech-to-speech translation: Integrating features of speech recognition and machine translation. In *International Conference on Computational Linguistics (COLING)*, Geneva, Switzerland. ICCL.

Yu Zhang, James Qin, Daniel S. Park, Wei Han, Chung-Cheng Chiu, Ruoming Pang,

Quoc V. Le, and Yonghui Wu. 2020. Pushing the limits of semi-supervised learning for automatic speech recognition. *CoRR*, abs/2010.10504.