Strategies for Effective Utilization of Training Data for Machine Translation

Praveen Dakwale

548661-L-bw-Dakwale
Processed on: 18-9-2020

Strategies for Effective Utilization of Training Data for Machine Translation

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. K.I.J. Maex
ten overstaan van een door het College voor Promoties ingestelde
commissie, in het openbaar te verdedigen in
de Agnietenkapel
op dinsdag 27 oktober 2020, te 12:00 uur

door

Praveen Dakwale

geboren te Ujjain

Promotiecommissie

Promotor:		
	dr. C. Monz	Universiteit van Amsterdam
Co-promotor:		
	prof. dr. M. de Rijke	Universiteit van Amsterdam
Overige leden:		
	dr. A. Bisazza	Rijksuniversiteit Groningen
	dr. M. Marx	Universiteit van Amsterdam
	prof. dr. K. Sima'an	Universiteit van Amsterdam
	prof. dr. C. G. M. Snoek	Universiteit van Amsterdam
	prof. dr. A. Way	Dublin City University, Ireland

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

The research was supported by Netherlands Organization for Scientific Research (NWO) under project numbers 639.022.213 and 612.001.218.

Cover by Shailendra Pathak
Printed by Printer Name, City

ISBN: 978-94-6421-048-4

Acknowledgements

Yes. This is it. After years of hopes and doubts about this moment, it has finally arrived. And at this moment, I owe gratitude to all those whose intellectual and emotional support has helped me to complete this dream.

Thanks Christof, for your trust in me. Besides guiding how to do quality research, you have also taught me to become slightly more independent and pragmatic. And no amount of gratitude would be enough for your patience during the proofreading of my papers and this thesis. I would like to thanks Maarten for encouragement and guidance on research and the valuable feedback on the thesis. Also, thanks for creating a great group of people at ILPS.

Being part of the MT group has been one of the best things about these years. Hamid and Marzieh, my paranymphs, have been true companions in my journey over these years. Your academic and emotional support has always been a source of motivation for me. My eternal gratitude for Arianna for her academic advice and mental encouragement. And yeah, our lunch discussions about cuisines, languages, and numerous other subjects have been the main highlight on many days. Thanks Ke, for inspiring and directing me towards the world of neural network and for developing *Tardis*. Thanks to Marlies and Katya for enlightening me with numerous research ideas, discussions, and motivations. And last but not least, Ivan: discussions and arguments with you have always provided me new perspectives on life.

I will always be grateful for being in the company of great minds at ILPS. I have enjoyed the numerous paper readings, academic discussions, presentations, coffee breaks, and beer meet-ups, which have enhanced my academic as well as social skills. I would like to thanks all current and former members of the ILPS with whom I got a chance to learn and grow: Evangelos, Maarten, Rolf, Ridho, Mostafa, Hosein, Amir, Ilya, Isaac, Bob, Nikos, Tom, Xinyi, Alexey, Christophe, Harrie, Chang, Chuan, Evgeny, Manos, Richard, Zhaochun, Ziming, Yifan, Fei, Anne, and Shangsong. A special thanks to Petra for relieving us from all the hassles of official stuff and organizing great lunches and ILPS-outings. Also, thanks to colleagues and friends from other departments at UvA: Mihir and Yash, for entertaining discussions on machine learning and Indian politics. During my Ph.D., I have also spent time in two internships: at Microsoft and Nuance communications, which have enhanced my skills in NLP and machine learning. Thanks to all colleagues and managers at these two companies.

Besides the colleagues at UvA, during the Ph.D., I have sustained myself on the advice and encouragement of my friends and colleagues from thousands of miles away. No amount of words would be enough to thank my friend Riyaz Bhat whose constant guidance and moral support have kept my spirit high. Thanks to Himanshu and Aditya for their academic and research advice and to Shailendra and Ravish for their emotional support. A special note of thanks to Shailendra for designing the cover page of this thesis.

At this point, I would like to thanks one of my best friends in the Netherlands, without whose help, I would have never survived the harsh weather and loneliness of the Netherlands. Fransisco Ramos, your selfless support, not only helped me to overcome the struggles of moving to a new country but also inspired me to improve my personality.

I would also like to thanks my ex-colleagues and managers at Wolters Kluwer and Elsevier, who provided me with the required support and motivation during the writing of this thesis. Thanks to Tjerk, Bill, Harry, Murli, Juan, Aliaksandr, Mohit, Umesh, Dicle, and Camillo. I would also like to thank my former advisor at IIIT-H, Prof. Dipti M Sharma, for igniting the love for linguistics in me.

I am heartily grateful to the members of the review committee for investing their time in reading the thesis and providing their valuable feedback.

Finally, no achievement in my life would have been possible without the love and blessings of my family. Their belief in me and unconditional support has been the most important driving force of my life.

Puurnnam-Adah Puurnnam-Idam Puurnnaat-Puurnnam-Udacyate
Puurnnasya Puurnnam-Aadaaya Puurnnam-Eva-Avashissyate

This is complete, that is complete. From the complete, originates the completeness.
When the completeness is subtracted from the complete, whatever remains is just the completeness.

—Isavasya Upanishad

Contents

1 Introduction

Machine translation (MT) is a sub-field of natural language processing that aims to develop technologies for automatically translating text from a source language to a target language. Early methods developed for machine translation were rule-based approaches, often designing explicit linguistic rules involving morphological, syntactic, and semantic structures to model the correspondences between source and target languages (Arnold et al., 1994; Hutchins, 1986; Trujillo, 1999; Forcada et al., 2011). These rules are usually designed by human experts. However, in the last 25 years, the bulk of research in MT has been focused on developing data-driven MT systems (Koehn, 2010; Brown et al., 1988, 1993). In data-driven MT, the idea is to learn a translation model from a given sample of source-target sentence pairs, which is known as a parallel corpus or bitext (Brown et al., 1988). Data-driven approaches enable the translation system to learn models for any language pair without explicit knowledge about the grammar and syntax of the source and target languages (Koehn, 2010). These trained models represent the patterns of translations observed in semantically equivalent sentence pairs in the training bitext. For example, these patterns could represent how source words are mapped to the target words and how the target words are ordered with respect to source words (Marcu and Wong, 2002).

Research in data-driven MT deals with several problems. The main issues among them are efficient and robust training of models (DeNero et al., 2006; DeNero and Klein, 2008; Wuebker et al., 2010), inferencing (Wang and Waibel, 1997; Germann et al., 2001; Aziz et al., 2014), selection of data (Eetemadi et al., 2015; Lü et al., 2007; van der Wees et al., 2017), domain adaptation (Cuong and Sima'an, 2017), and evaluation (White and O'Connell, 1993; Dorr et al., 2011). However, the most crucial bottleneck for training data-driven MT is the availability of substantial amounts of training data with reasonably high-quality (Koehn and Knowles, 2017). As a result, the translation performance of data-driven MT systems is good for language pairs and domains for which a substantial amount of training data is available. However, for other low-resource scenarios such as language pairs or domains with a small amount of training data, the translation performance is of poor quality (Irvine and Callison-Burch, 2013; Turchi et al., 2008).

Although the availability of high-quality annotated data is certainly important for training high-performing MT systems, an important related aspect of this problem is also the ineffectiveness of the standard training strategies in learning robust models by leveraging all available training data compiled from various sources (Lü et al., 2007).

Most of these training algorithms treat the entire training corpus as uniform and attempt to learn models in a single pass (Koehn, 2010, 2020). This results in problems such as overfitting (Wuebker et al., 2010; Koehn, 2020; Zoph et al., 2016), imbalance of domains or genres in the training data (Koehn and Schroeder, 2007; Chu and Wang, 2018) and inclusion of incorrect features from samples of relatively low-quality (Khayrallah and Koehn, 2018). These problems arise due to variations in data quality with respect to the intended translation task. For example, forcing the learning algorithm to rely only on the given samples in the training data is based on the assumption that all samples in the training data are noise-free and belong to the intended domain. However, the reality in industrial settings can be completely different, where the training data is often compiled from multiple sources and tends to be noisy (Axelrod et al., 2011). In such scenarios, a common solution is to improve the quality of the training data with respect to the intended task or domain through techniques such as data selection (Rauf and Schwenk, 2011). However, data selection techniques require robust oracle systems in order to decide the quality of training data sample (van der Wees et al., 2017). Moreover, these techniques only select a small subset as the relevant training data. Therefore, instead of relying only on improving the quality and quantity of training data, another more beneficial solution could be to explore algorithms and training strategies that can potentially refine the model quality to minimize the effect of training data variations and to investigate regularization techniques that can provide some kind of stabilization during training.

An alternative approach to single-pass learning is to consider the segments of data separately depending on domains and data quality and incrementally improve the quality and robustness of the model by addition or refining of the knowledge in each incremental step (Lü et al., 2007; Gao et al., 2011). Based on this idea, this thesis aims to propose solutions to diverse problems such as the quality of translation models in phrase-based SMT (PBSMT), degradation of performance during domain adaptation for neural MT (NMT), and efficient exploitation of low-quality training data for neural MT. While addressing these problems, the common theme in this thesis is to address the following question: "To what extent can transfer of knowledge in incremental steps help to improve the performance of machine translation models?"

The problems addressed in this thesis and the solution proposed for these problems draw inspiration from two active research concepts in other machine learning-based tasks: transfer learning and incremental learning. In transfer learning, the idea is to use models learned for one specific task towards solving another different but related task (Pan and Yang, 2010). In scenarios where sufficient data for a target domain or task is not available, transfer learning provides transfer of common features from a source domain to the target domain. Similarly, incremental training is a dynamic technique in machine learning that can be used when training data becomes available gradually (Geng and Smith-Miles, 2009). The aim is to adapt the model to new data without forgetting the existing knowledge. Although the approaches we propose in this thesis are closely related to transfer learning and incremental learning, we diverge from these concepts by aiming to explore to what extent the knowledge gathered by the models in initial training steps can be refined and used for solving varied problems such as improvement of the quality of the models, efficient domain adaptation and efficient utilization of noisy training data.

For a long time, research in data-driven machine translation has been dominated by the phrase-based MT paradigm (Koehn, 2010). However, during the last few years, neural machine translation (NMT) has achieved comparable or even better performance than phrase-based MT for the majority of MT tasks (Koehn, 2020; Toral and Sánchez-Cartagena, 2017). These two paradigms are significantly different from each other. Training phrase-based MT involves learning probabilistic phrase translation rules from sentence pairs in the training data along with additional features like re-ordering (Bisazza and Federico, 2016) and n-gram language models (Koehn, 2010). The weights for each individual model are then optimized on a development set, and finally, test sentences are decoded using beam search over a hypothesis space (Neubig and Watanabe, 2016). On the other hand, neural machine translation is an end-to-end neural network and training involves learning of parameters of the neural network by minimizing a loss function such as perplexity, given the source and target sentence pairs. As this paradigm shift from phrase-based MT to neural MT happened during the course of this thesis, the problems addressed in this thesis belong to both paradigms. Therefore, the first problem that we address in this thesis falls under phrase-based MT, and the rest of the problems addressed in this thesis fall under the neural MT paradigm.

1.1 Research outline and questions

In this thesis, we address four diverse problems in MT. The solutions that we propose to these problems revolve around the idea of transferring information or knowledge from initial models to guide the training of efficient resultant models. These research problems are discussed as follows:

RQ1: *How does heuristic training of translation models affect the performance of phrase-based MT and to what extent can alternative training strategies based on re-estimation of models improve phrase-based MT performance?*

The first problem we address is about the limitations of standard training of translation models for phrase-based MT, which results in the extraction of unintuitive features that affect the performance and compactness of the trained models (DeNero and Klein, 2008; Wuebker et al., 2010). We focus on improving the quality and compactness through the re-estimation of the initial models by observing the features it would prefer to achieve the best possible translations. We show that such a re-estimation allows the training process to incorporate knowledge from different phrase-based MT models such as re-ordering models and language models for the selection of high-quality phrase segmentations. We divide this research question into three sub-questions and answer those:

RQ1.1 *What are the limitations of heuristic training algorithms for phrase-based MT, and how is the translation performance of the models affected by them?*

In Chapter 3, we address RQ1.1 by first discussing the limitations of the heuristic training strategy in phrase-based MT, along with examples. This standard training strategy is based on the extraction of phrase translation pairs that are *'consistent'* with the alignments of words between the sentence

pairs in the training bitext. The translation probabilities of the phrase pairs are calculated based only on their occurrence frequency in the training data (Och and Ney, 2000). We show that the initial models trained with standard heuristic training algorithm result in the extraction of unintuitive and noisy phrase segmentations, which affect the translation performance of the SMT system.

RQ1.2 *Can the reliability and performance of translation models be improved by re-estimating the phrase translation table through oracle-BLEU decoding of the training data?*

Some remedies, such as forced decoding, have been proposed in the literature to overcome the limitations of the heuristic-based approach (Wuebker et al., 2010). However, these previously proposed solutions have their own limitations. For example, in the case of forced decoding, the unreachability of exact reference for noisy sentence pairs leads to incorrect and unreasonable distributions of translation probabilities. While addressing RQ1.2, we propose a more robust training strategy based on oracle-BLEU re-estimation of the translation models. We show that instead of re-estimation through forced decoding, oracle-BLEU re-estimation results in improved robustness and compactness of models.

RQ1.3 *To what extent can oracle-BLEU re-estimation of re-ordering models and language models provide additional improvements in translation performance?*

Another limitation of forced decoding based training is that it does not allow for the re-estimation of re-ordering and language models due to the constraint of generating the exact reference. On the other hand, oracle-BLEU translation based training allows for re-estimation of re-ordering and language models. To answer RQ1.3, we experiment with oracle-BLEU re-estimation of re-ordering models and language models and evaluate whether their re-estimation provides any additional improvements in translation quality.

The idea of our proposed method can be generalized as a re-estimation strategy to improve the quality of the translation model by learning from the model's own predictions over the training data. This method re-distributes the probabilities of the phrase segmentations according to their likelihood of being selected by the decoding algorithm to reach the best possible translation.

As already stated in the introduction section, recently, the focus of the research community has rapidly shifted from phrase-based MT to NMT. Unlike phrase-based MT, NMT is a single end-to-end neural network with a large number of parameters (Bahdanau et al., 2015; Luong et al., 2015). NMT models are known to be data-hungry and prone to overfitting if trained on training data of small amounts or low-quality (Koehn and Knowles, 2017). Usually, there is a scarcity of data for a specific domain. Research has shown that models trained on general domains do not perform well on test sentences from a specific domain (van der Wees et al., 2017). Our second research question in this thesis, addresses domain adaptation for NMT. A straightforward way to

adapt a general domain model to a specific domain is *'fine-tuning,'* where the idea is to continue training a general model on the training data for a specific domain (Freitag and Al-Onaizan, 2016). However, *'fine-tuning'* leads to a degradation of performance on the source or general domain. In real scenarios, one would prefer a model that is equally good on different domains. To address this problem, we devise a technique to transfer knowledge from a source domain to a target domain in a way that leads to reasonable performance across multiple domains. We frame the second research question as follows:

RQ2: *How can we apply fine-tuning to adapt NMT models to new domains while retaining the performance on the source (original) domain?*

We answer RQ2 by dividing it into three sub-questions and answering those:

RQ2.1 *How do NMT models trained on a general domain perform on data from a specific domain?*

In Chapter 4, we discuss the problem of low performance of the NMT model when used to translate sentences from unseen domains. While addressing RQ2.1, we demonstrate the effect of domain variation between training and test data for neural machine translation. Our experiments show that the performance of a model trained on general domain data is significantly lower when tested on multiple target domains.

RQ2.2 *How is the performance on the original source domain affected by fine-tuning on target domains?*

Fine-tuning is an easy way to adapt the NMT model to new domains. We evaluate the performance of fine-tuning based domain adaptation on the target domain and also evaluate whether the performance of the adapted model on the source domain is retained. We demonstrate by answering RQ2.2 that the fine-tuning strategy results in a degradation of model performance on the source or original domain.

RQ2.3 *Can we apply knowledge distillation as a remedy to minimize the degrading effect of fine-tuning?*

As a remedy to the degradation problem, we propose the transfer of knowledge through a distillation procedure in order to minimize the degradation on the source domain while improving the performance on the target or new domain. In answer to RQ2.3, we evaluate the performance of domain adaptation through knowledge distillation on the source and target domains and show that this strategy helps to adapt to new domains with improved performance, while at the same time, the model retains its performance on the source domains.

Although domain adaptation is a well-known and thoroughly researched problem in NMT, a related problem that has gained less attention from the research community is that of the quality of the training data. Irrespective of the intended domain, low-quality of the training data is known to severely affect the performance of neural network-based

tasks (Khayrallah and Koehn, 2018; Koehn et al., 2018). Therefore, as a third research problem, we address the issue of noise or low-quality of the training data for NMT. Neural network-based tasks require a large amount of training data in order to improve performance. Some parts of such large training corpora can be noisy. To reduce the effect of noise in the training data, the standard practice is to select only relevant or high-quality data from the noisy corpus. We propose a knowledge transfer strategy to improve NMT performance that aims at leveraging all available noisy data without any filtering. We frame the third research question as follows:

RQ3: *How do noisy training corpora affect the performance of NMT and how can we leverage them to improve NMT performance?*

To answer RQ3, we divide it into three sub-research questions that we answer individually:

RQ3.1 *What is the effect of comparable or noisy training data on the performance of a recurrent NMT system?*

Here, we address the effect of low-quality training data on NMT performance. Many language pairs do not have high-quality annotated data for training neural MT systems. However, many of these language pairs only use slightly low-quality parallel data known as comparable data that is built by crawling the web and aligning source and target sentences. However, before using this noisy data, one should investigate what kind of effect this noisy data could have on the performance of trained models. We show that, at least for recurrent architecture based NMT, low-quality data such as comparable data can have negative effects on the performance of trained NMT models.

RQ3.2 *To what extent can we leverage noisy comparable data to improve NMT performance through the transfer of knowledge from a high-quality model?*

An obvious solution to mitigate the negative effect of noisy data is data selection or filtering, where the idea is to use only relevant samples from the noisy data. However, these filtering or selection techniques are based on heuristic measures that decide the level of noise based on various factors. Here, we raise the following question: Is it possible to use comparable data without any heuristic-based filtering? We propose that the concept of knowledge distillation can be used as a strategy to minimize the effect of noisy data and show that such a knowledge transfer can help to leverage the low-quality data with minimal degrading effect. Knowledge distillation (Hinton et al., 2014) is a framework for training compressed "student" networks by using supervision from a large teacher network.

RQ3.3 *What is the relative variation in performance of recurrent vs. non-recurrent NMT models when noisy data is added to the training pool?*

Although in RQ3.1, we show that noisy data has a negative effect on the performance of the recurrent neural network-based NMT models, we also aim to investigate its effect on the more recently proposed non-recurrent architectures for NMT. Our intuition suggests that non-recurrent architectures

achieve relative robustness against training noise. In RQ3.3, we also explore the relative difference with respect to the effect of noisy data on recurrent and non-recurrent architectures. We show that non-recurrent NMT architectures such as transformers are relatively robust against noise in the training data as compared to recurrent architectures.

In RQ1, RQ2, and RQ3, we aim to explore the idea of improving the reliability of MT by transfer of knowledge across models, domains, or data sources. However, another interesting way of combining the capabilities of multiple models is the combination of different NMT architectures. Therefore, with the fourth research question, we address the problem of the limited capability of recurrent neural network-based NMT to accurately compose the source sequence representation using an attention mechanism. The attention mechanism was proposed for NMT to enable the network to focus on the relevant words or fragments of the source sentence corresponding to each target position (Bahdanau et al., 2015; Luong et al., 2015). However, even with the attention mechanism, the source representation fed to the decoder at each time step is a compact summary of the previous history of the source sequence. As a result, the encoder memory is shared across multiple words and is prone to have a bias towards the recent past (Miculicich Werlen et al., 2018; Cheng et al., 2016). We propose that combining recurrent and convolutional networks on the encoder side can allow the network to create a representation of the source tokens with both global and local features. We frame the fourth research question as follows:

RQ4: *What are the limitations of using only recurrent layers towards effective modeling of source sentences in NMT and can NMT performance be improved by the addition of convolutional layers?*

We divide RQ4 in two sub-questions and answer those:

RQ4.1 *Can a combination of recurrent and convolutional layers for the encoder provide the model with improved effective guidance to focus on relevant information in the source sequence?*

Here, we discuss the limitations of the NMT architecture, where both encoder and decoder are solely based on a recurrent neural network. We propose to add multiple convolutional layers on top of the recurrent layers of the encoder and demonstrate that it enables the encoder to capture relevant features of the source sequence leading to improved neural MT performance. Our experiments demonstrate that a combined recurrent and convolutional encoder model shows significantly better performance than the vanilla recurrent encoder model.

RQ4.2 *Are the improvements observed with additional convolutional layers due to properties of convolutions or merely due to the increased number of parameters?*

Even if the proposed combination demonstrates an improvement over a vanilla recurrent encoder model, it raises the question of whether the observed improvements are a result of the increased number of NMT parameters or

are a result of increased capability of the model to capture complex features due to the addition of convolutional layers. For this purpose, we evaluate the proposed model against another fully recurrent model with an increased number of layers as compared to the baseline. Our experiments suggest that the addition of convolutional layers is indeed more beneficial than simple additions of recurrent layers.

1.2 Main contributions

Here, we summarize the main contributions of the thesis. We categorize the contributions in algorithmic and empirical contributions:

1.2.1 Algorithmic contributions

1. We demonstrate that for phrase-based MT, phrase translation models trained with a heuristic extraction method are unreliable and fail to incorporate the features from other MT models such as re-ordering and language models. We propose an improved re-estimation strategy based on oracle-BLEU translations of the training data that also allows for the re-estimation of re-ordering models and language models. [Chapter 3]

2. We address the requirement of training a neural MT model with consistent performance across domains and demonstrate that domain adaptation methods such as fine-tuning suffer from degradation of performance on the original domain. We propose knowledge distillation as a remedy to minimize this degradation while adapting to new domains. [Chapter 4]

3. We address the issue of negative effects of noisy training data on NMT and demonstrate that NMT performance can be severely affected by the presence of noise in the training data. We propose knowledge distillation in incremental steps as a training strategy to leverage noisy comparable data while minimizing its negative effects. [Chapter 5]

4. We propose a combination of recurrent and convolutional neural networks for neural machine translation in order to provide effective guidance to the network to focus on the relevant parts of the source sentence. [Chapter 6]

1.2.2 Empirical contributions

1. We compare the translation performance of the proposed oracle-BLEU re-estimation to the standard heuristic training paradigm as well as to the previously proposed forced decoding based training method and show that oracle-BLEU re-estimation provides improved translation performance as compared to the other two baselines. [Chapter 3]

2. We demonstrate that oracle-BLEU re-estimation also yields higher compression rates compared to forced decoding as it focuses on the selection of only those

segmentations that are suitable to generate the best possible translation of the source sentences in training data. [Chapter 3]

3. We demonstrate that oracle-BLEU re-estimation of re-ordering models provides additional gains in translation performance as compared to the re-estimation of phrase translation models only. [Chapter 3]

4. For NMT, we compare the improvements achieved by vanilla fine-tuning for domain adaptation in two different domains to an approach using knowledge distillation. We show that while vanilla fine-tuning suffers from catastrophic degradation on source domains, adaptation by knowledge distillation not only sustains the improvements on the target domains but also retains good performance on the source domain. [Chapter 4]

5. We evaluate the effect of directly using noisy data for training NMT models in addition to the high-quality data along with some well-known filtering techniques. Further, we demonstrate that additional improvements can be achieved through knowledge distillation without the requirement of filtering techniques. [Chapter 5]

6. We compare the change in translation performance of recurrent vs. non-recurrent NMT architectures when trained on low-quality noisy data and demonstrate that non-recurrent NMT models are less susceptible to training noise and can learn efficiently from noisy data without suffering a large drop in translation quality. [Chapter 5]

1.3 Thesis overview

1. Chapter 2 – **Background: MT paradigms, Related concepts, Evaluations:** This chapter provides the required background about MT systems. It discusses two main MT paradigms: Statistical Machine Translation and Neural Machine Translation. After describing the basic concepts of the two paradigms along with their similarities and differences, the rest of the chapter provides details on the concepts related to this thesis, such as standard training strategies for both paradigms, baseline models, and evaluation metrics.

2. Chapter 3 – **Model Re-estimation for Statistical Machine Translation:** This is a research chapter where we propose oracle-BLEU model re-estimation as a strategy to overcome limitations of the standard training for phrase-based SMT.

3. Chapter 4 – **Efficient Domain Adaptation for Neural Machine Translation:** This is a research chapter where we propose knowledge distillation for efficient domain adaptation of neural MT models.

4. Chapter 5 – **Neural Machine Translation with Noisy Data:** This is a research chapter in which we explore the effect of noisy training data on training neural machine translation systems as well as a novel technique to effectively leverage

noisy data with minimal degrading effect. We also compare the negative effects of noisy data on recurrent vs. non-recurrent neural machine translation architectures.

5. Chapter 6 – **Convolution over Recurrent Encoder for Neural Machine Translation:** This is a research chapter where we propose modifications to the standard neural MT architecture by combining recurrent and convolutional networks in order to provide effective guidance to the network to focus on the relevant parts of the source sentence.

6. Chapter 7 – **Conclusions:** In this chapter, we summarize the thesis and revisit all research questions. Finally, we provide an outlook for possible extensions and future work corresponding to each of the research questions.

1.4 Origins

The chapters in this thesis are based on the previously published papers as described below:

1. **Chapter 3** is based on Dakwale and Monz (2016). *Improving Statistical Machine Translation Performance by Oracle-Bleu Model Re-estimation.* In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 38–44, 2016. Monz proposed the model and concepts. Dakwale carried out the experiment and analysis. Dakwale did most of the writing.

2. **Chapter 4** is based on Dakwale and Monz (2017a). *Fine-tuning for Neural Machine Translation with Limited Degradation across In-and Out-of-domain Data.* In Proceedings of the 16th Machine Translation Summit, 2017, pages 156–169. The idea was proposed by Dakwale. Experiments and analysis were carried out by Dakwale. Dakwale did most of the writing.

3. **Chapter 5** is based on Dakwale and Monz (2019). *Improving Neural Machine Translation Performance using Noisy Data through Distillation.* In Proceedings of the 17th Machine Translation Summit 2019. The idea was proposed by Dakwale. Experiments and analysis were carried out by Dakwale. Dakwale did most of the writing.

4. **Chapter 6** is based on Dakwale and Monz (2017b). *Convolutional over Recurrent Encoder for Neural Machine Translation.* The Prague Bulletin of Mathematical Linguistics, vol. 108, pages 37-48, 2017. The idea was proposed by Dakwale. Experiments and analysis were carried out by Dakwale. Dakwale did most of the writing.

2
Background

In this chapter, we provide the necessary background for this thesis. We mainly discuss two machine translation paradigms: Statistical machine translation (SMT) and Neural machine translation (NMT). Besides this, we also provide a brief introduction to the evaluation metrics, MT systems, and tools used in this thesis.

2.1 Statistical machine translation foundations

The idea of statistical machine translation is to learn translation models from a given sample of source-target sentence pairs known as the parallel training corpus or bitext. Statistical machine translation consists of models and architectures that can learn translation rules and features from a training dataset without any explicit knowledge about the linguistic features of the language pairs.

Training standard statistical machine translation starts with an assumption that a large sentence-aligned bitext is available. A *bitext* is a corpus in which sentences that are identical in meaning in source and target languages are paired together. The performance of phrase-based MT systems is highly dependent on the quantity and quality of the parallel corpora on which it is trained. These parallel corpora are obtained either by human annotation or by automatic alignment techniques such as lexical matching approaches (Tiedemann, 2011). Public and private institutions invest in the creation of annotated parallel corpora for the languages and domains of intended applications. However, the availability of high-quality data sets is limited to relatively few language pairs and usually come from parliamentary proceedings such as Europarl (Koehn, 2005) and the Linguistic Data Consortium.[1]

The earliest models introduced for SMT are word based models attributed to Brown et al. (1993) who formalized the translation task in terms of a noisy channel model (Shannon, 1948). Formally, given a sentence **f** in a source language, the corresponding translation **e** in a target language can be generated as:

$$\mathbf{e}^* = \underset{\mathbf{e}}{\operatorname{argmax}}\, p(\mathbf{e}|\mathbf{f}) = \underset{\mathbf{e}}{\operatorname{argmax}}\, \frac{p(\mathbf{f}|\mathbf{e})p(\mathbf{e})}{p(\mathbf{f})} = \underset{\mathbf{e}}{\operatorname{argmax}}\, p(\mathbf{f}|\mathbf{e})p(\mathbf{e}) \tag{2.1}$$

[1] https://www.ldc.upenn.edu/

The above equation can be interpreted as the task of finding the most probable target language translation for a given source language sentence **f**. Since the probability of the source sentence remains constant for all possible translations, $p(\mathbf{f})$ can be ignored. Therefore, the probability of translation can be defined in terms of two models: the translation model $p(\mathbf{f}|\mathbf{e})$, which assigns a higher probability to a hypothesis translation that is similar in meaning to the given source language sentence, and the language model $p(\mathbf{e})$ that assigns a higher probability to more fluent or grammatical sentences in the target language. Thus, a more probable translation for a given source language sentence is one that is similar in meaning to the given source sentence and at the same time is more appropriate in the target language in terms of fluency and grammaticality.

The translation model $p(\mathbf{f}|\mathbf{e})$ can be trained by counting how many times a source language sentence **f** appears in the training corpora corresponding to a target sentence **e**. However, given the limited quantity of training corpora, it is difficult to estimate translation probabilities of sentence pairs directly. Therefore, the first instantiations of the noisy channel model are known as word-based models or *IBM models* (Brown et al., 1993), which were proposed to estimate the sentence translation probabilities using correspondences between the source and target words in a sentence pair. The first IBM model, known as IBM model-1, introduced the concept of word alignments. Alignment refers to the set of links between source and target words in a sentence pair. Given a sentence pair $(\mathbf{f}, \mathbf{e})$ where $\mathbf{f} = (f_1, f_2, \ldots, f_n)$ and $\mathbf{e} = (e_1, e_2, \ldots, e_m)$ and an alignment function $(a : j \rightarrow i)$, the translation probability is defined as:

$$p(\mathbf{e}, a|\mathbf{f}) = \frac{\epsilon}{(n+1)^m} \prod_{j=1}^{m} t(e_j|f_{a(j)}). \tag{2.2}$$

Here, $t(e_j|f_{a(j)})$ are the translation probabilities of the generated target words conditioned on the source word and the alignment link. The term $\frac{\epsilon}{(n+1)^m}$ represents the normalization over sentence lengths. The main modeling problem is, therefore, the estimation of word translation probabilities $t(e_j|f_{a(j)})$. The assumption here is that we know the alignment links between the source and target words of the sentence pairs in the training corpus. However, in the parallel corpus, we only have the sentence alignments but not the word alignment links. Therefore, to estimate the alignments and their probabilities, Brown et al. (1993) proposed to use the Expectation-Maximization (EM) algorithm (Dempster et al., 1977). The idea of using the EM algorithm for word alignment training is to infer the most plausible alignment links between words of a sentence pair. Starting by assigning uniform probabilities to all possible alignments, the EM algorithm iteratively learns which source words frequently co-occur with which other target words across the training corpus. Details of the implementation of the EM algorithm for word alignments can be found in Brown et al. (1993).

IBM model-1 only considers word alignment for the computation of translation probabilities, but it ignores word order differences. As a refinement, IBM model-2 (Brown et al., 1993) uses an additional parameter to represent the position of a word in the target string. IBM model-3 includes a fertility parameter to capture one-to-many mappings between words in a sentence pair. IBM model-4 further refines the relative positioning (or distortion) by using word classes. A discussion about the details of each of these models can be found in (Koehn, 2010).

2.2 Phrase based machine translation

As discussed in Section 2.1, to model the frequent one-to-many mappings between the words in language pairs, word-based models apply complex concepts such as fertility, insertion, and deletions of the words. However, these concepts increase the complexity of the model. Moreover, it is a common observation that the semantics of words is highly dependent on the context in which they occur (Firth, 1935). To a large extent, translation between language pairs is a sentence-level task where the semantics of each word is dependent on other words in a sentence. Therefore, instead of learning the direct translation between individual words, modeling groups of words as basic translation units could be more beneficial. This idea was implemented as a solution, known as *phrase-based machine translation* by Koehn et al. (2003). Phrase-based MT considers phrases as the basic translation units and models the translation probability between a given sentence pair as the product of all the phrase pairs into which the sentence pair is segmented. In this model, phrases do not correspond to the linguistic definition of syntactic phrases; instead, they are contiguous sequences of words with a fixed maximum length. Further, to capture different aspects of translation, phrase-based MT computes sentence translation probabilities by combining scores from several other models using a log-linear combination as follows:

$$\mathbf{e}^* = \underset{\mathbf{e}}{\operatorname{argmax}} \left[\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f}) \right]. \tag{2.3}$$

In Equation 2.3, h_m represents the log-scores from various feature functions, most important of which are the phrase translation probabilities and the language model scores. λ_m are the optimized model weights for each feature. One of the most important variations of these models is the reordering or distortion model that addresses word-order variations between language pairs. In the following subsections, we will describe the important models used in this thesis.

2.2.1 Phrase translation model

The standard practice of training a phrase translation model is to obtain alignments between sentence pairs (using the IBM models discussed in Section 2.1) followed by the extraction of phrases pairs consistent with these word alignments. Finally, the probabilities of these phrases are estimated by simply counting and normalizing their occurrences. Below, we explain the two steps in detail.

Word alignments: For training statistical machine translation systems (including word-based MT and phrase-based MT), the first step is to obtain word alignments between the tokens of sentence pairs in a given parallel corpus. Note that given a sentence pair, there are a large number of possible alignments. This is due to the inherent translation ambiguities of the words and differences in the word orders of the source and target languages. The idea of word alignment training is to infer the most plausible alignment links between words of a sentence pair by using IBM models (Brown et al., 1993), and the parameters of the models are estimated using EM algorithm

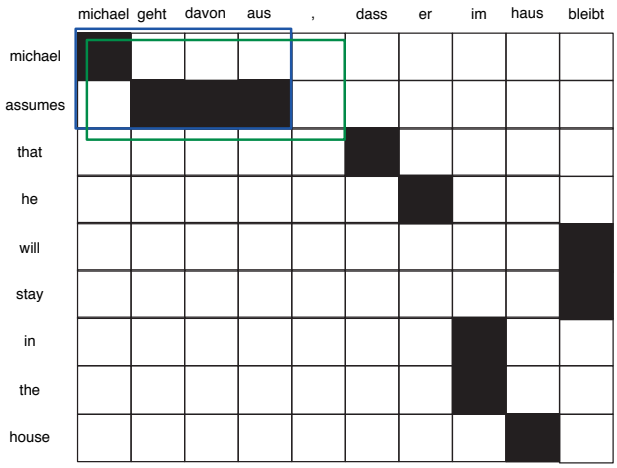


Figure 2.1: Phrase extraction from an English-German sentence alignment. This example is taken from (Koehn, 2010). The English sentence is represented on the vertical axis and the German sentence on the horizontal axis. Dark squares represent the presence of an alignment link between words. Colored boundaries represent the borders of consistent phrase segmentation. For simplicity of visualization, only two phrases are shown in this figure.

(Dempster et al., 1977). The quality of trained word alignment is highly dependent on the quality and quantity of the training data.

Note that alignments obtained for the source-to-target direction will be different from those of the target-to-source direction. Therefore, it is common practice to train the alignments in both directions and merge the two to obtain symmetric many-to-many alignments.

Phrase segmentation: The end goal of training a phrase translation model is to obtain source-target phrase pairs, which are most likely translations of each other. Given a word-aligned sentence pair, the simplest approach is to extract all those phrase translation pairs which have intersecting alignments within themselves. In other words, in an extracted phrase pair, any word within the source phrase should only be aligned to one of the words in the target phrase and vice versa. However, since there are many unaligned words in a sentence pair, this strategy will be able to extract only very precise phrase pairs, and a lot of the phrase pairs will be missed. Therefore, Koehn et al. (2003) proposed a strategy known as *grow-diag-final* heuristics, which allows unaligned words on either side to be included in the phrase pairs if they are neighboring the word boundaries of alignment intersections. Formally, the notion of consistency with this heuristic is defined in (Koehn et al., 2003) is as follows:

Phrase pair $(\bar{e}, \bar{f})$ is consistent with alignment **A**, if all words $f_1, f_2, \ldots, f_n$ in $\bar{f}$ that have alignment points in **A** have their alignment links with words $e_1, \ldots, e_m$ in $\bar{e}$

English phrases	**German phrases**
michael	michael
michael assumes	michael geht davon aus
michael assumes	michael geht davon aus ,
michael assumes that	michael geht davon aus , dass
michael assumes that he	michael geht davon aus , dass er
michael assumes that he will stay in the house	michael geht davon aus , dass er im haus bleibt
assumes	geht davon aus
assumes	geht davon aus ,
assumes that	geht davon aus , dass
assumes that he	geht davon aus , dass er
assumes that he will stay in the house	geht davon aus , dass er im haus bleibt
that	dass
that	, dass
that he	dass er
that he	, dass er
that he will stay in the house	dass er im haus bleibt
that he will stay in the house	, dass er im haus bleibt
he	er
he will stay in the house	er im haus bleibt
will stay	bleibt
will stay in the house	im haus bleibt
in the	im
in the house	im haus
house	haus

Table 2.1: List of phrase pairs extracted from English-German alignments shown in Figure 2.1.

and vice versa:

$$\begin{aligned}(\bar{e}, \bar{f}) \text{ consistent with } A \iff & \forall e_i \in \bar{e} : (e_i, f_i) \in A \rightarrow f_i \in \bar{f} \\ & \text{AND} \quad \forall f_i \in \bar{f} : (e_i, f_i) \in A \rightarrow e_i \in \bar{e} \\ & \text{AND} \quad \exists e_i \in \bar{e}, f_j \in \bar{f} : (e_i, f_i) \in A.\end{aligned} \tag{2.4}$$

In other words, according to Equation 2.4, a phrase pair extracted from a sentence pair is consistent if it has:

- a consecutive sequence of words within the source and target sentences;
- words which do not have any alignment links outside the phrase pair; and
- at least one alignment link between any source and target word.

Figure 2.1 shows an example of a word-aligned source-target sentence pair and Table 2.1 provides a list of all phrase pairs that can be extracted from this sentence pair based on the above heuristics. Note that these heuristics allow for a large number of phrase pairs, many of which are overlapping and some of them are very unlikely to be used when

decoding a test sentence. In Chapter 3, we will propose a technique to improve the reliability of the translation models by extracting more reliable phrase segmentations.

After extracting the consistent phrase pairs from all sentences in the training bitext, a phrase translation table is built that represents the translation probabilities for each phrase pair. These translation probabilities are calculated as the relative frequency of the source phrase given a target phrase. Formally, given a phrase pair $(\bar{e}, \bar{f})$, the phrase translation probability is defined as:

$$\phi(\bar{f}|\bar{e}) = \frac{\text{count}(\bar{e}, \bar{f})}{\sum_{\bar{f}_i} \text{count}(\bar{e}, \bar{f}_i)}, \tag{2.5}$$

where $\text{count}(\bar{e}, \bar{f})$ is the joint count of observing phrase pair $(\bar{e}, \bar{f})$ in the training data and the denominator is the total count of observing phrase $\bar{e}$ in the bitext. Inverse phrase translation probabilities $\phi(\bar{e}|\bar{f})$ are also computed in a similar manner and added to the phrase translation table.

For phrase pairs that are rarely observed in the training data, it is not sufficient to rely only on the phrase translation probabilities. To obtain a better estimate of probabilities for such phrase pairs, one solution is to use *"lexical weightings"* which are a product of the word translation probabilities of the aligned target words within a phrase pair (Koehn et al., 2003). Formally, the lexical weighting for a phrase pair $(\bar{e}, \bar{f})$ with alignment function a is defined as:

$$\text{lex}(\bar{e}|\bar{f}, a) = \prod_{i=1}^{|\bar{e}|} \frac{1}{|\{j\}|(i,j) \in a|} \sum_{(i,j)\in a} w(e_i|f_j). \tag{2.6}$$

The lexical weighting of a phrase is calculated as the product of all individual lexical weightings of each target word. If a target word is unaligned, its lexical weighting probability is considered as 1. If a target word is aligned to multiple source words, its lexical weighting probability is calculated as the sum of the translation probabilities over all its alignment links. Similarly, inverse lexical weightings $lex(\bar{e}|\bar{f}, a)$ are calculated by considering word translation probabilities for all words in a source sentence and added to the phrase translation table.

Note that theoretically, there is no limit on the length of extracted phrase pairs according to the heuristics defined in Equation 2.4. This can result in full sentence pairs being extracted as phrase pairs that can reduce the generalization capability of the model while increasing its size. In order to avoid this, standard phrase-based systems place a constraint on the maximum length of phrases on both the source and target side. Moreover, it is also customary in phrase translation tables to include *phrase-penalty* and *word-penalty* features in order to favor shorter phrases. These features are pre-defined by the user and fire whenever a phrase is applied during decoding.

Finally, while building models for our experiments, for each phrase pair, we also store the alignment links between the words of source and target phrases. Although it is not used as a training feature, it is useful in our experiment in Chapter 3 to deduce full alignment links between the source sentence and generated translation hypotheses.

2.2.2 N-gram language model

As discussed in Section 2.1, in MT, the language model is used to ensure the fluency of the translation output in the target language. In Equation 2.1, $p(e)$ represents the language model. A language model (LM) assigns probabilities to a sequence of words. Traditionally, language models used in phrase-based MT are n-gram language models.

Using the probabilities of individual words, we can estimate the probability of an entire sentence by multiplying them using the chain rule. Given a sentence $S = (w_1, w_2, \ldots w_t)$, using an LM of order n, the probability of the sentence can be estimated as follows:

$$p(s) = \prod_{i=1}^{t+1} p(w_i | w_{i-n+1}, \ldots, w_{i-1}) \tag{2.7}$$

While computing sentence probability, n-grams also include tokens representing start and end of the sentence (<s> and </s>). In the above equation, the additional term at position $t + 1$ is the end of sentence token (</s>). Note that the probability of each word in the sentence is estimated conditioned only on $n - 1$ previous words (where n is the order of the LM). Ideally, one would like to estimate this probability conditioned on the entire history observed in the sentence. However, to keep the size of the LM small, the order n is usually fixed to a given value.

There is an important problem with the above approach for estimating sentence probabilities. Although language models are trained by collecting n-gram counts over very large monolingual corpora, there is always a possibility of encountering an n-gram sequence in the candidate translations that has never been observed in the training corpus. Based on pure relative frequency, the probability of such an n-gram will be assigned zero. Based on Equation 2.7, this will result in the likelihood of the entire sentence to be zero as well. This problem can be solved by two well-known techniques, namely "Smoothing" and "Back-off." Smoothing techniques assign a small probability to the unseen n-grams and re-distribute the probabilities of the observed n-grams. The most well-known smoothing techniques are Kneser-Ney smoothing (Kneser and Ney, 1995), Witten-Bell smoothing (Witten and Bell, 1991), and Good-Turing smoothing (Good, 1953). We refer the reader to (Chen and Goodman, 1996) for a detailed discussion of the different smoothing techniques. Similarly, the probability of an unseen higher-order n-gram can be estimated by backing off to lower order n-grams that are possibly seen in the training data (Katz, 1987).

2.2.3 Reordering models

Languages differ in the order of different syntactic and semantic elements when used in a sentence. This implies that the order of the positioning of constituents like *Subject*, *Object* and *Verb* maybe different in source and target languages. This creates the problem of reordering in statistical machine translation. In terms of phrase-based MT, reordering decisions define the order in which the phrases in the target side should be generated in order to produce a sentence translation that follows the correct word order of the target language (Koehn, 2010). For this purpose, various approaches have been proposed in the SMT literature. Here we describe the most popular ones, which are also used in this thesis.

Ideally, MT systems must explore all possible reordering permutations between words of a given sentence pair because the word orders between many language pairs differ substantially. However, exploring all reordering possibilities is computationally too expensive. Therefore standard phrase-based MT systems only focus on exploring local reorderings. For this purpose, these systems impose a *distortion limit*, which is the maximum permissible distance between two consecutive phrase applications. Although this reduces the computational cost of the system, it limits the possibility of capturing long-distance reorderings. Further, Koehn et al. (2003) introduced the concept of *distortion cost* as a feature. This is modeled as a function of the distance between consecutive phrase applications.

Lexicalized reordering models (LRM): Distortion limit and *distortion cost* are just linear constraints to control the complexity of the model. However, an effective reordering model should model what kind of reordering decisions will lead to a more likely and fluent translation. Lexicalized reordering models (Tillman, 2004; Axelrod et al., 2005) define the probabilities of three different orientations between the current phrase pair and the next phrase to be selected. Orientation defines the relative order of the application of the source side of a phrase pair at adjacent points of time. Given a phrase pair $(\bar{e}, \bar{f})$ and a given orientation, the reordering probability is formally defined by Koehn et al. (2003) as:

$$p(\text{orientation}|\bar{e}, \bar{f}) = \frac{\text{count}(\text{orientation}, \bar{e}, \bar{f})}{\sum_o count(o, \bar{e}, \bar{f})} \quad \text{where} \quad o \in [\text{monotone}, \text{swap}, \text{discontinuous}]. \tag{2.8}$$

The model defines three types of orientation: *Monotone, Swap* and *Discontinuous*. *Monotone* orientation implies that the source side of the phrase applied at time $t + 1$ immediately follows the source side of the phrase pair applied at time t on the right hand side. Similarly, *Swap* orientation implies that the source side of the phrase pair applied at time $t + 1$ is adjacent to the phrase pair applied at time t on the left side. If the two source phrases are not adjacent to each other, then the orientation is simply considered as *Discontinuous*. For all phrase pairs learnt in the translation model, each of these orientations is estimated separately from the training bitext. Note that this model does not distinguish between different positions in a *Discontinuous* orientation, i.e., all positions that are not adjacent either on the left or right side of the current phrase pair have equal reordering probabilities.

Hierarchical reordering models (HRM): Although lexicalized models can efficiently handle swaps between adjacent phrases, they are not efficient to model the long-distance reorderings. Galley and Manning (2008) proposed an easy way to capture the long-distance dependencies without depending on complex parsing algorithms. They instead proposed to modify the LRM by including hierarchical phrases while estimating the probabilities of the orientations. For example, by treating two adjacent target phrases as one single phrase, the orientation of the subsequent phrase would be *Swap*, which otherwise would have been *Discontinuous* by considering them separately. Galley and Manning (2008) showed that modeling hierarchical phrases can significantly improve the translation performance for language pairs such as Chinese-English, which have substantially different word orders.

2.2.4 Bilingual language model (BiLM)

A Bilingual Language model (Niehues et al., 2011) is an n-gram language model that has elements from both the source and target language. The advantage of BiLM's for phrase-based MT is that while n-gram language models can only predict the probability of the next word given the previous context in the same language, BiLM's are capable of using context information from both sides to score the translation hypothesis. Niehues et al. (2011) have shown that BiLM's significantly increased phrase-based MT performance. The simplest approach to model a BiLM is to compute n-grams for bilingual tokens instead of tokens from one language. These bilingual tokens can be learned by relating tokens in the source and target language through trained word alignments. Therefore, in simple terms, a bilingual token is a target word along with all the source words it is aligned to. If, however, a source word is aligned to multiple target words, multiple bilingual tokens are created corresponding to each of the alignments. Unaligned words on the target side are just represented as monolingual tokens, while unaligned words on the source side are ignored.

2.2.5 Tuning

As defined in Equation 2.3, individual specialized models are trained separately, and the scores from each of these models are combined in a log-linear fashion to estimate the translation probability of a candidate translation. One important aspect of this training process is the estimation of the values for the weight parameters λ_i. These parameters define the importance of each of the models or features in deciding the probability of a candidate translation. They are optimized by generating the translations for a development set and updating the values of λ's with respect to quality metrics such as BLEU (described in Section 2.5.1). In standard optimization algorithms, λ's are initialized to a random or uniform value, and the development set is decoded (process described in Section 2.2.6) generating the k best scoring translation hypotheses. Given the target references in the development set, the λ's are then iteratively updated so that the updated values lead the decoder to generate a more likely translation in terms of BLEU. Various optimization algorithms have been proposed for machine translation tuning. The most popular of these are MERT (Och, 2003), MIRA (Watanabe et al., 2007) and PRO (Hopkins and May, 2011).

It is important to note that in phrase-based MT, individual models are independently trained. They are combined by weighting them with respect to their importance determined by tuning. Therefore, the training process has no knowledge of the overall likelihood of the candidate translation when calculated with the combined scores in Equation 2.3. As a result, the effect of the model becomes highly dependent on the correct optimization of the values of model weights λ's.

2.2.6 Decoding

Decoding is the process of finding the highest-ranking translation from all conceivable translations of a given test sentence. With reference to Equation 2.1, decoding implies the application of the *argmax* function. In simple terms, given a test sentence, the

decoding process involves segmenting the sentence into all possible phrase segmentations, looking up the corresponding phrase translations from the translation model, and arranging them in the correct order. However, each sentence can be segmented in multiple ways. Moreover, for each possible phrase segmentation, there are multiple possible translations in the phrase table, and there are many possible reorderings. All these possible permutations result in a vast search space of possible translations, and searching for the best possible translation in this search space is computationally too expensive. In fact, it has been shown that a complete search through the entire space is NP-complete (Knight, 1999). Therefore, phrase-based MT decoding (Koehn, 2010) applies various intelligent heuristics to keep the search space tractable. The most important heuristics is the use of a beam-search strategy where partial hypotheses are stored in stacks, which are organized by the number of translated source words.

Given a source sentence, decoding starts with an empty hypothesis. Hypotheses are expanded by adding phrase translations for different phrase segmentations of the source sentence. These partial hypotheses are stored in multiple stacks, which are organized according to the number of source words that have been covered by that partial hypothesis (Wang and Waibel, 1997). Whenever a partial hypothesis is expanded, it is moved to another stack based on the number of source words covered. However, given a stack with a large number of partial hypotheses, it is critical to decide which hypotheses should be expanded further. Attempting to expand all hypotheses could be computationally expensive or even infeasible. On the other hand, one would like not to drop a partial hypothesis that could lead to a highly likely translation. To overcome this tradeoff, decoding applies pruning strategies to remove low scoring hypotheses from each stack. The most common way is *threshold pruning*, where the idea is to discard those hypotheses which fall below a pre-defined beam-width α. In other words, if the score of a hypothesis is worse than α times that of best scoring hypothesis in a stack, it is discarded. The highest-scoring hypothesis in the final stack (which represents all source words being translated) is generated as the final output translation. Note that due to heuristics-based pruning, beam-search is prone to search error and hence does not guarantee an optimal solution.

Each partial hypothesis is represented by a data structure with the following information: (i) Coverage vector: This simply represents all source words that have been translated so far. This is important to ensure that all words of the source sentence have been translated just once. (ii) Cost (negative log probability) of the partial hypothesis: This is the main parameter that is used to rank the partial hypothesis and to make pruning decisions. (iii) Future cost: It may not always be beneficial to rank or prune the hypothesis solely on the current cost (or probability of the translation). For example, it is possible that a partial hypothesis may have low-probability (or high cost) for the translation generated so far due to the use of difficult phrases or reorderings. However, generating the translation of the remaining part of the source sentence may lead to a low cost or more optimal final translation. Therefore, we also consider *future-cost* while applying threshold pruning. Again, calculating future-cost is computationally intractable; therefore, the decoding algorithm applies intelligent heuristics through dynamic programming to determine the approximate value of the future cost of translating the remainder of the source sentence.

Another heuristics that the beam-search algorithm applies in order to reduce the

number of partial hypotheses is the idea of *hypothesis-recombination*. If two partial hypotheses have identical translations but different translation probabilities, the lower scoring hypothesis can be discarded as it will never lead to best scoring full translations. Further details about details of the decoding algorithm are provided by Koehn (2010).

2.3 Neural machine translation

As discussed in Section 2.2, the phrase-based MT paradigm is based on training individual specialized models such as phrase translation models, reordering models and language models, which are combined in a log-linear fashion to compute the likelihood of a candidate translation. In this way, phrase-based MT systems are still inherently rule-based systems where the rules and their probabilities are learned directly from the training data. There are various limitations of this paradigm. First, the training of individual models increases the complexity of the system. In this paradigm, all features are learned in isolation, and during training, there is no feedback to learn which model features could lead to a more likely translation. Further, to limit the size and complexity of models, a phrase-based paradigm puts constraints on model parameters such as phrase length and distortion limit, which restricts the capacity of the models to learn important linguistic phenomena such as "long-distance reordering" resulting in low fluency of the generated translations.

To overcome the limitations of phrase-based MT, a fully neural network-based model was first proposed by Sutskever et al. (2014). The main idea was to develop an end-to-end model that will avoid the requirement of training explicit models for translation, reordering, alignments, etc. It has been shown that recurrent neural networks (RNNs) have the capacity to model word sequences of variable length and hence have the capacity to easily capture long-distance dependencies (Mikolov et al., 2010). A recurrent network generates a hidden state representation for a given input vector based on the previous history observed in the sequence. The hidden state representation at each timestep is influenced by the hidden states in the previous steps. This property makes them suitable to capture long-distance dependencies and, therefore, to model sequential objects such as sentences.

Mikolov et al. (2010) proposed an RNN language model that predicts the probability of the next word given a sequence of words in a sentence. For this purpose, the words in a sentence are represented as high-dimensional dense vectors known as *word embeddings*, which are then sequentially fed to multiple non-linear recurrent layers stacked on each other. The hidden state representation of the last word is then projected to an output layer, which has a size equal to that of a pre-defined vocabulary. This output layer generates scores corresponding to each word in the vocabulary. These scores can be converted to probabilities using a *softmax* function. The RNN can then be trained by maximizing the log-likelihood of the output distribution with respect to the given target word and updating the parameters of the model using optimization algorithms such as *stochastic gradient descent*.

One limitation of an RNN is the problem of vanishing gradients. For very long sequences, the gradients for the parameters of an RNN can become very small, which would result in practically no updates of the parameters. As a solution, Hochreiter

and Schmidhuber (1997) proposed *Long-short-term-memory* (LSTM) networks, which consist of various gates to control the information flow between the nodes of the network. This property allows LSTM networks to model relatively long sequences by avoiding the vanishing gradient problem.

In Section 2.3.1 we explain the fundamentals of the NMT paradigms, followed by training and decoding procedures in Section 2.3.2 and Section 2.3.3, respectively. Finally, in Section 2.3.4 and Section 2.3.5 we discuss important modifications and enhancements to NMT that are relevant to this thesis.

2.3.1 Recurrent neural machine translation

The earliest and simplest instantiation of the recurrent NMT model is the sequence-to-sequence or encoder-decoder model of Sutskever et al. (2014), where a multi-layer recurrent network known as encoder converts an input sentence **x** = $[x_1, x_2, \ldots, x_n]$ into a sequence of hidden states $[h_1, h_2, \ldots, h_n]$.

$$h_i = f_{enc}(x_i, h_{i-1}). \tag{2.9}$$

Here, f_{enc} is a recurrent unit. Another multi-layer recurrent network known as decoder predicts a target sequence $y = [y_1, y_2, \ldots, y_m]$. Each word in the sequence is predicted based on the last target word y_{j-1}, the current hidden state of the decoder s_j and the hidden state representation of the final state of the encoder h_n:

$$p(y_j|y_1, \ldots, y_{j-1}, \mathbf{x}) = \text{softmax}(W_s\ \tilde{s}_j), \tag{2.10}$$

where

$$s_j = f_{dec}(s_{j-1}, y_{j-1}, h_n). \tag{2.11}$$

As stated in (Bahdanau et al., 2015), a limitation of the approach of Sutskever et al. (2014) is that it attempts to compress all necessary information of the source side into one single fixed-length vector. This may be problematic when dealing with long sentences where the final representation will tend to lose the information from the start of the sentence. Therefore, Bahdanau et al. (2015) introduced an "attention mechanism" in which each time the decoder generates a target word, it uses a different representation of the source sentence which is known as the *context vector*. This context vector is computed as the weighted sum of the hidden state representations corresponding to each individual word in the source sequence. These weights are calculated based on the relevance of each source position with respect to the target position being generated. This enables the neural model to focus on the local information in the source sequence instead of squeezing all information into one single compact vector. Formally, the context vector c_j with respect to each target position is calculated as follows:

$$c_j = \sum_{i=1}^{n} \alpha_{ji} h_i, \tag{2.12}$$

where α_{ji} are attention weights corresponding to each encoder hidden state output h_i:

$$\alpha_{ji} = \frac{\exp(a(s_{j-1}, h_i))}{\sum_{k=1}^{n} \exp(a(s_{j-1}, h_k))}, \tag{2.13}$$

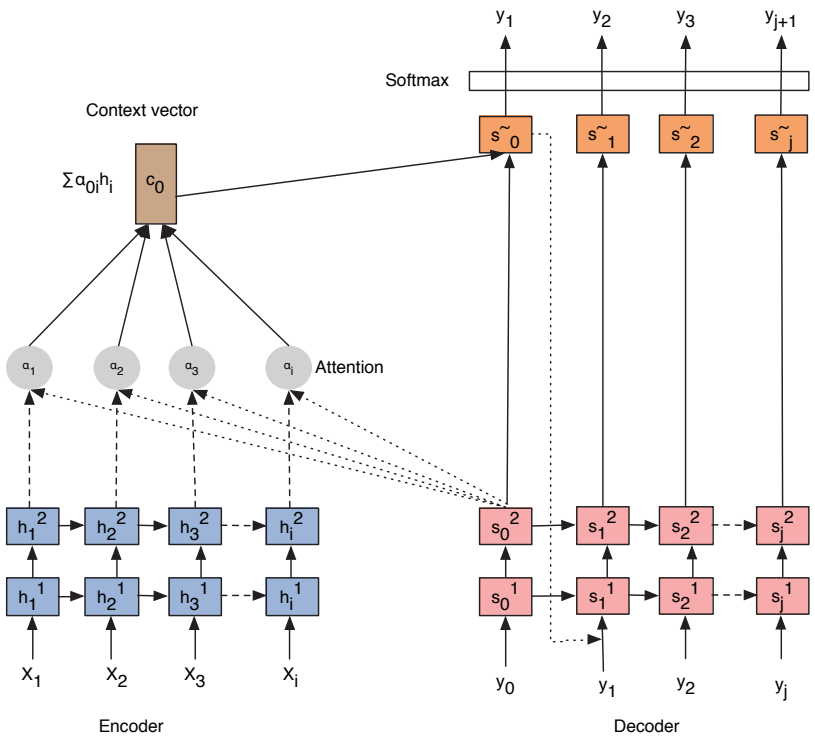


Figure 2.2: Recurrent NMT encoder-decoder framework of Bahdanau et al. (2015). For simplicity of visualisation, attention mechanism is shown only for a single decoder hidden state representation s_0^2.

where s_j is the decoder hidden state generated by RNN units similar to the encoder:

$$s_j = f_{dec}(s_{j-1}, y_{j-1}, c_j). \tag{2.14}$$

Given the target hidden state and the context vector, a simple concatenation combines the information from both vectors into an attentional hidden state $\tilde{s}_j$:

$$\tilde{s}_j = \tanh(W_c[c_j; s_j]). \tag{2.15}$$

This attentional vector $\tilde{s}_j$ is then projected to the output vocabulary size using a linear transformation and then passed through a softmax layer to produce the output probability of each word in the target vocabulary:

$$p(y_j|y_1, \ldots, y_{j-1}, \mathbf{x}) = \text{softmax}(W_s\ \tilde{s}_j). \tag{2.16}$$

Figure 2.2 shows the basic architecture of recurrent NMT model with attention as proposed in (Bahdanau et al., 2015).

2.3.2 Training

Training an NMT system implies searching for the optimal values for the parameters (weights) of the neural network. The performance and the quality of the NMT model largely depend on searching for the most optimal values of the network parameters. Note that in phrase-based MT, training an MT system implies first learning the individual specialized models followed by optimization of the model weight. On the other hand, for NMT, training implies end-to-end optimization of all parameters over the training bitext. The standard way of training any neural network is the "back-propagation algorithm." For NMT, at each training step, output probabilities $p(y_j)$ corresponding to each target position j are generated by the softmax layer by running a forward pass through the network. Given the correct word from the training data, the end-to-end network is trained by maximizing the log-likelihood over the training data. In (Koehn, 2020), the log-likelihood loss for each timestep is defined as:

$$\mathrm{L}_{\mathrm{NLL}}(\theta) = -\sum_{k=1}^{|V|} y_k \cdot \log\Big(p(y_k|x;\theta)\Big), \tag{2.17}$$

where $p(y_k)$ is the output probability distribution generated by the network and y_k are the binary labels corresponding to each word in the target vocabulary V. Note that for only one value of k, value of y_k will be 1. This will be the index corresponding to the true class label, i.e., the word at position j in the reference target sentence. Therefore, this comes down to the probability p_k being given to the correct word k. The total loss for the complete target sequence can then be calculated by summing losses over each individual target position:

$$\mathrm{L}_{\mathrm{NLL}}(\theta) = -\sum_{j=1}^{n}\sum_{k=1}^{|V|}(y_{jk}) \cdot \log\Big(p(y_{jk}|x;\theta)\Big). \tag{2.18}$$

The training results in learning of optimal values of model parameters θ.

The loss calculated above is then used to calculate gradients corresponding to each layer of the network through back-propagation, and weights of the network are updated correspondingly. Note that the above optimization calculates the loss corresponding to each target position or word. However, the intention of an NMT model is to generate a target sequence. Therefore a more principled approach would be either to optimize a loss corresponding to the error in the generated sequence or to optimize quality metrics such as BLEU (Ranzato et al., 2016). However, due to the simplicity of the approach, log-likelihood based optimization is still the most popular training approach for NMT. Moreover, well-known optimization algorithms such as *Stochastic gradient descent* or *Adam* (Kingma and Ba, 2015) can be used for training the model.

2.3.3 Decoding

Similar to phrase-based MT, the NMT model of Bahdanau et al. (2015) also uses beam-search to decode an unseen test sentence. However, unlike phrase-based MT, NMT does not stack the hypotheses by the number of translated words but by the output length.

The aim of decoding is to search a hypothesis with the highest log probability score, where the score for a candidate translation sequence is defined as follows:

$$\begin{aligned} score(y_1, y_2, \ldots, y_t) &= \log(p(y_1, y_2 \ldots y_t|x)) \\ &= \sum_{i=1}^{t} \log(p(y_j|y_1, \ldots, y_{j-1}, \mathbf{x})). \end{aligned} \tag{2.19}$$

The ideal situation would be to do an exhaustive search over all possible output sequences. However, an exhaustive search would have a complexity of $O(V^T)$ (where V is the target vocabulary size, and T is the output sequence length), which is very expensive. Therefore an efficient approach is beam-search. Given a beam size b, the decoder assumes a dummy (start-of-the-sentence) token as the input and generates b highest scoring words for the first output position. For the next position, the network is instantiated b times, corresponding to each of the words generated in the previous step as input along with the respective context vectors. Thus at each position, hypotheses are expanded, and decoding continues until the network generates an EOS (end-of-sentence) token for each possible hypothesis or a maximum number of timesteps has been reached. This is unlike SMT, where the search continues until all words in the source sentence have been covered. The maximum scoring hypothesis is produced by the system as the translation of the given input sentence. Note that beam-search is an approximate search algorithm and does not guarantee an optimal solution. Further, beam-search tends to generate short translations. However, this can be avoided by normalizing the log probability of the sentence by the length of the input source sentence.

The recurrent network-based encoder-decoder model, along with an attention mechanism, has shown comparable or even better performance than phrase-based MT (Luong et al., 2015). However, there are several limitations to this model. Various enhancements and modifications have been proposed to overcome these limitations. The most important of these modifications are proposals for non-recurrent NMT architectures.

One major limitation of the recurrent architecture is the sequential processing of the input sentence by the recurrent network. Due to sequential processing, each hidden state of the encoder network depends on the previous hidden state. Therefore to process each position, the network has to wait until inputs from all the previous positions have been processed. This restricts parallel processing of the input and hence slows down the whole training process. Moreover, although the attention mechanism enables the decoder to model dependencies between input and output sequences by allowing access to full input sequence, it still restricts the capability of the model to handle long-distance dependencies among the input or output tokens themselves. This is due to the fact that the encoder does not have access to the full source sequence at any specific position. To resolve this, Vaswani et al. (2017) proposed a fully-attentional model, commonly known as the Transformer model. We briefly describe the Transformer model architecture in Section 2.3.4. Similarly, convolutional neural networks also allow parallel processing of input. Gehring et al. (2017b) proposed a convolutional sequence-to-sequence model for NMT. We discuss the model of Gehring et al. (2017b) in Section 2.3.5.

2.3.4 Transformer NMT model

In order to avoid recurrent modeling and allow the encoder and decoder access to the full input sequence at each position, Vaswani et al. (2017) proposed the fully-attentional neural MT model, commonly known as the *Transformer model*. Vaswani et al. (2017) used the idea of "self-attention." In a self-attention mechanism, the representation corresponding to each word is calculated as a weighted sum of every word in the input sequence. In this way, at each position, the encoder has access to all other words in the input sequence, which enables it to capture the relevant information from different parts of the sequence.

Along with self-attention, Vaswani et al. (2017) also proposed the concept of "Multi-head attention." In this mechanism, an individual head is calculated as a different linear transformation of the input representation. To compute each head, the individual vectors representing the inputs are mapped into lower dimensions using weight matrices, and the output of each transformation is used to compute the attention head. These heads are then concatenated and transformed using another output weight matrix. The use of these multiple heads enables different parts of the input to interact with other parts.

However, with no recurrence, the model is devoid of any sense of the order of the input sequence. To incorporate information about the order of the sequence of words, Vaswani et al. (2017) use "positional embeddings" which are calculated as sinusoidal functions of the relative positions of the words. These positional embeddings are combined with the word embeddings to compute the initial representation corresponding to each word. Other important features of the Transformer model are the "residual connections"(He et al., 2016) and layer-normalization. The decoder has an architecture similar to the encoder except that in the decoder, all the values to the right of the current position in the dot-product attention are masked in order to prevent access to information from the future target words. We refer the reader to (Vaswani et al., 2017) for a complete description of the Transformer model. Vaswani et al. (2017) shows that the fully-attentional model performs comparably to the recurrent model while reducing the computation time.

2.3.5 Convolutional sequence-to-sequence model

The convolutional sequence-to-sequence model is another approach to NMT and was proposed by Gehring et al. (2017b) to avoid sequential processing and dependence on a recurrent network. Since the convolutional network does not depend on the processing of previous timesteps, it allows for parallelization over every element in the sequence. Though convolutions create representations for a fixed length of context, Gehring et al. (2017b) proposed that the effective context size can be made larger by stacking multiple layers of convolutions and therefore enable capturing long-distance dependencies. In the first layer, word embeddings for each input token are fed to the convolutional functions which capture local context with fixed kernel size. Then, higher layers allow for interactions between distant layers. This model also uses convolutions for both encoders and decoders. Similar to the Transformer model, this model also uses positional embedding to capture the order in the input sequence. Moreover, this model uses gated linear units (GLUs) (Dauphin et al., 2017) as non-linearities. These

GLUs allow the model to control the flow of information from only those inputs in the current context that are relevant. An important aspect of this model is the use of multi-step attention. This implies that at each decoder layer, a context vector is calculated for each position using a dot-product similarity with all encoder outputs. This is similar to the Transformer model but different from the standard recurrent model of Bahdanau et al. (2015), which computes attention only for the final layer. We refer the reader to (Gehring et al., 2017b) for a complete description of the convolutional sequence-to-sequence model.

2.4 Knowledge distillation

An interesting concept in deep learning is the idea of "knowledge distillation" proposed by Hinton et al. (2014). The concept of knowledge distillation is used in two chapters in this thesis (Chapter 4 and 5). Therefore in this section, we briefly describe the main idea behind knowledge distillation.

Knowledge distillation is a framework for training compressed "student" networks by using supervision from a large teacher network. As discussed in (Hinton et al., 2014), large, cumbersome neural network models with millions of parameters are computationally too expensive to be deployed in environments with a large number of users. Therefore, such environments require smaller models due to limited computational resources. However, smaller models with reduced dimension sizes usually fail to perform comparably to large models. Knowledge distillation provides a straightforward approach to train smaller networks with performance comparable to large networks.

Assuming we have a teacher network with large dimension sizes trained on a large amount of data, a smaller student network with much smaller dimension sizes can be trained to perform comparable or even better than the teacher network by learning to mimic the output distributions of the teacher network on the same data. This is done by minimizing the cross-entropy or KL-divergence loss between the two distributions. Formally, if we have a teacher network trained on a given training data with a learned distribution $q(y|x;\theta_T)$, the student network (model parameters represented by θ) can be trained on the same data by minimizing the following loss:

$$\mathrm{L_{KD}}(\theta,\theta_{\mathrm{T}}) = -\sum_{k=1}^{|V|} \mathrm{KL}\Big(q(y|x;\theta_T)\; p(y|x;\theta)\Big), \tag{2.20}$$

where θ_T is the parameter distribution of the teacher network and KL represents the KL-divergence loss function. Commonly, this loss is interpolated with the log-likelihood loss, which is calculated with regard to the target labels in the training data:

$$\mathrm{L}(\theta,\theta_{\mathrm{T}}) = (1-\lambda)\mathrm{L_{NLL}}(\theta) + \lambda \mathrm{L_{KD}}(\theta,\theta_{\mathrm{T}}). \tag{2.21}$$

In Equation 2.21, the first term L_{NLL} represents the loss of the student network with respect to the given target (similar to the teacher network), while the second term $L_{KD}(\theta,\theta_T)$ represents the difference or divergence of the output distribution of the student network with respect to that of the teacher network. The second term forces the student network to mimic the output of the teacher network. Hinton et al. (2014)

discussed that the generalization capability of the teacher model could be transferred to the student network using these output probability distributions, which they termed "soft-targets." These soft-targets result in less variance in the gradients as compared to the hard targets, therefore, enabling faster convergence even when the size of the training data is relatively small. An important aspect of the knowledge distillation framework is the increased temperature of the softmax layer. The higher temperature allows the student network to encode the similarities among the output classes. Hinton et al. (2014) suggest generating a smoother distribution by increasing the temperature of the softmax of both teacher and student networks.

2.5 Experimental setup

2.5.1 Evaluation metrics

Various evaluation matrics have been proposed in the literature for automatic as well as human evaluation of the translation quality of MT output. These include *BLEU* (Papineni et al., 2002), *METEOR* (Banerjee and Lavie, 2005) and *TER* (Snover et al., 2006). However, BLEU remains the most popular metric for automatic MT evaluation due to its simplicity and independence from any hyper-parameter. Throughout this thesis, we only use BLEU for the evaluation in all our experiments. We briefly explain BLEU as follows.

BLEU stands for *Bilingual Understudy Evaluation*. Essentially, it is based on the calculation of n-gram precision. It counts how many n-grams in the translation output of a given test set match with the n-grams in one or more reference translations usually provided by human annotators. Precision is calculated by normalizing the number of matching n-grams by the total number of n-grams in the reference set. This precision is calculated for each value of n and finally multiplied. However, using only n-gram precision will reward short sentences that have a perfect match in the reference translations. To avoid this, the n-gram precision is multiplied by a 'brevity-penalty,' which penalizes shorter translations. The brevity-penalty is calculated as the sum of the ratios of output length to that of the reference lengths of all candidate sentences. Formally, BLEU is defined as:

$$BLEU_n = BP \times \prod_{i=1}^{n} precision_i, \tag{2.22}$$

$$BP = \sum_{j=1}^{c} \min\Big(1, \frac{l(ouput_j)}{l(reference_j)}\Big). \tag{2.23}$$

Note that standard BLEU is defined only for a given test set and not for individual sentences. In Chapter 3, we define some modifications to the standard BLEU metric, where we require the calculation of approximate BLEU for each individual sentence in a generated output set. These modifications will be described in the appropriate section in Chapter 3. Other than this, all results are reported using the standard definition of BLEU over the entire test set. In all our experiments, we use case-insensitive BLEU up to and including n-grams of length 4.

2.5.2 Phrase based SMT system

For all experiments related to phrase-based MT conducted in this thesis, we use a phrase-based MT system designed in Perl, which is very similar in its architecture and implementation to the Moses SMT toolkit (Klein et al., 2017). However, this SMT system requires some other tools for training individual models, which we describe below. Although the specific settings and parameter values for different models will be described in the respective research chapters, here we describe some of the standard parameters that are used identically across all experiments.

- Word alignments for the training bitexts are obtained using GIZA++ (Och and Ney, 2003) in both directions.
- N-gram language models are trained using the SRILM toolkit (Stolcke et al., 2011). Unless otherwise specified, all n-gram LMs are of order 5 with linear interpolation and Kneser-Ney smoothing.
- Optimization of weights or tuning is performed through an algorithm called pairwise ranking optimization (PRO) (Hopkins and May, 2011)
- Unless otherwise specified, the maximum phrase length in the phrase translation models is 7, and the distortion limit in the reordering models is fixed as 5.

2.5.3 Neural machine translation systems

In this thesis, we use two different NMT systems. In Chapter 4 and Chapter 6, we use *Tardis,*[2] which is based on the system described in (Luong et al., 2015) and developed in the *Torch* deep learning framework. In Chapter 5, we use the Open-NMT-py toolkit (Klein et al., 2017), which is an open-source NMT system developed in Python. The specifics of the used models and hyper-parameters will be described in the respective chapters.

[2]https://github.com/ketranm/tardis

3

Model Re-estimation for Statistical Machine Translation

3.1 Introduction

In this chapter, we focus on the problem of improving the reliability and quality of translation models trained from a word-aligned parallel corpus for phrase-based MT. This problem is based on the first research-question RQ1 stated in Chapter 1:

RQ1: *How does heuristic training of translation models affect the performance of phrase-based MT and to what extent can alternative training strategies based on re-estimation of models improve phrase-based MT performance?*

As explained in Section 2.2, standard phrase-based SMT systems translate source sentences by segmenting them into phrases and translating each phrase separately. These systems are usually composed of at least three models: (a) a phrase translation model consisting of bilingual phrase pairs extracted from a parallel corpus and their corresponding translation probabilities, (b) a re-ordering model to score movements between the phrase applications, and (c) a language model used to score the fluency of the candidate translation in the target language. To determine the overall likelihood of a candidate translation, probabilities from these models are combined in a log-linear fashion:

$$\mathbf{e}^* = \underset{\mathbf{e}}{\operatorname{argmax}} \left[\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f}) \right]. \tag{3.1}$$

In Equation 3.1, h_m represents the log-scores from various feature functions, most important of which are the phrase translation probabilities and the language model scores. The λ_m factors are the optimized model weights for each feature.

As described in Section 2.2, the phrase pairs in the translation model are trained using a heuristic extraction method from a word-aligned bilingual training data (Och and Ney, 2000). This heuristic method extracts phrase pairs based on alignment consistency, i.e., all possible phrase pairs are extracted in which the word alignments are within the boundaries of the extracted pairs on both the source and the target side, and at least

one source and target word within a phrase are aligned with each other. The word-level alignments are statistically learned from the bitext using the EM algorithm. Formally, the consistent phrase segmentations are defined in (Koehn, 2010) as follows:

Phrase pair $(\bar{e}, \bar{f})$ is consistent with alignment **A**, if all words $f_1, f_2, \ldots f_n$ in $\bar{f}$ that have alignment points in **A** have these with words $e_1, \ldots e_{ms}$ in $\bar{e}$ and vice versa:

$$\begin{aligned}(\bar{e}, \bar{f}) \text{ consistent with } A \iff & \forall e_i \in \bar{e} : (e_i, f_i) \in A \rightarrow f_i \in \bar{f} \\ & \text{AND} \quad \forall f_i \in \bar{f} : (e_i, f_i) \in A \rightarrow e_i \in \bar{e} \\ & \text{AND} \quad \exists e_i \in \bar{e}, f_j \in \bar{f} : (e_i, f_i) \in A.\end{aligned} \tag{3.2}$$

After extracting the consistent phrase pairs from all sentences in the training bitext, a phrase translation table is built that represents the translation probabilities for each phrase pair. The probabilities of the translation model are then calculated for each phrase translation as the relative frequencies of the extracted phrases:

$$\phi(\bar{f}|\bar{e}) = \frac{\text{count}(\bar{e}, \bar{f})}{\sum_{\bar{f}_i} \text{count}(\bar{e}, \bar{f}_i)} \tag{3.3}$$

A notable limitation of heuristic extraction based training has been discussed in the SMT literature: The translation model probabilities calculated using the heuristic method can be unintuitive and non-reliable (Marcu and Wong, 2002; Foster et al., 2006). Heuristic method simply extracts all phrase translation pairs that satisfy the criteria described in Equation 3.2. This includes overlapping phrases. The heuristic extraction is based on simple counting and does not consider whether the extracted phrases correspond to a highly probable or unlikely alignment. Moreover, the extracted phrase tables reflect only the distribution over the phrase pairs observed in the training data.

On the other hand, during translation (decoding), hypotheses are generated by considering each relevant phrase translation in the phrase table, which are then combined with the probability score from language models, re-ordering models, and other possible features and the highest-scoring hypothesis is selected as the output translation. Therefore, phrase table training and decoding are not necessarily consistent with each other. While training only considers the distribution of phrase pairs in the training data, during decoding, the hypotheses are ranked on a weighted combination of translation probabilities, language model scores, and re-ordering probabilities. Due to this combination, a phrase translation extracted with high translation probability using the heuristic method may not get selected in the generation of a highly ranked translation of the source sentence. This implies that a considerable probability mass is wasted on phrase translation pairs that are less likely to be used during decoding.

To understand the problem described above, consider the example shown in Figure 3.1. Since the heuristic method allows the extraction of phrases which include unaligned words, this results in 3 possible translations of source phrase يين (*yyn*). The proba-

bilities are calculated based on their occurrence frequency in the bitext, as shown in Table 3.1 (which is a small excerpt of the complete phrase-table extracted from an Arabic→English bitext). In order to analyze which of the possible translations of the

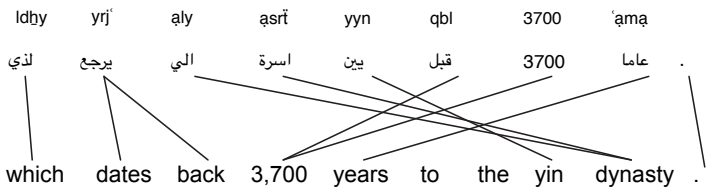


Figure 3.1: Word aligned Arabic-English sentence pair example.

Table 3.1: Example phrase pairs extracted from sentence alignment in Figure 3.1.

source	target	Forward probability
يين (yyn)	the yin	0.0946896
يين (yyn)	to the yin	0.00160451
يين (yyn)	**yin**	**0.476923**

given phrase are actually used during translation, we use the decoder optimized using this phrase table to translate the source side of the complete training bitext. We observe that within the 100 best scoring hypotheses, only the last phrase translation is used. The first two translations are never used. This implies that during training, considerable probability mass is 'wasted' on the first two phrase pairs. One simple solution, which is widely used to reduce the phrase-table size, is "phrase-table pruning" where the idea is to simply discard the phrase translations below a threshold probability score. However, this kind of pruning is again unreliable because these low-scoring phrase pairs could appear in a highly probable translation when model scores from the language model and re-ordering models are combined.

This implies that the choice of phrase segmentation used to achieve the highest scoring translations is also affected by the scores of other models. In order to train a compact phrase translation model with a reliable probability distribution, these scores should be taken into account. This brings us to our first sub-research-question.

RQ1.1 *What are the limitations of heuristic training algorithms for phrase-based MT, and how is the translation performance of the models affected by them?*

It can be concluded from the discussion above that the translation models trained with heuristic training are unreliable because of following two reasons:

- A large number of overlapping and unusable phrase segmentations increase the size of the translation models and hence can be inefficient due to memory and time complexities.

- The count-based calculation of phrase translation scores is based only on their occurrence in the training data and does not have an intuition about which phrase segmentations are more likely to lead to better MT output.

In the next section, we discuss strategies proposed in the literature that aim to overcome the limitations of the heuristic training strategy.

3.2 Related work

Several methods have been proposed to learn a more reliable phrase translation model. One of the most intuitive solutions is to use the word alignment probabilities to score the likelihood of the phrase segmentations. DeNero et al. (2006) proposed direct training of phrase segmentation using the EM-algorithm, where the likelihood of the phrase pairs is calculated using word alignment probabilities. However, they found that this kind of highly constrained training leads to overly deterministic probability estimates, ultimately leading to overfitting and that it performs worse than heuristic training. In other words, constraining the translation model to only have one translation of each source phrase in a given sentence and no overlaps results in only a few translation options for each phrase segmentation, leading to poor generalization and overfitting.

As a more reliable solution, Wuebker et al. (2010) proposed "forced-decoding" with a leave-one-out procedure, which attempts to learn more probable phrase alignments and segmentations by forcing a decoder to produce the reference translations for the source sentences in the training bitext and then re-estimating phrase translation probabilities based on these alignments. Their idea is to first train all models using heuristic training and then use these models to translate the source sentences in the training data. Finally, the phrase-table is re-estimated based only on those segmentations, which lead to a hypothesis exactly matching the target sentence. In forced-decoding, given a sentence pair (f_1^J, e_1^I), the best phrase segmentations and alignments are searched that will cover both sentences. The best segmentation is defined in (Wuebker et al., 2010) as follows:

$$\hat{s_1^K} = \operatorname{argmax} \left\{ \sum_{m=1}^{M} \lambda_m h_m(e_1^I, s_1^K, f_1^J) \right\}. \tag{3.4}$$

The above equation implies that out of all possible phrase segmentations of the source and target sentences, the best segmentation of the sentence pairs is the one that will maximize the probability of translating the source sentence into the given target sentence using the weighted sum of all model features used to score the sentence likelihood. The phrase table is re-estimated by counting all the best phrase segmentations defined above for all sentence pairs in the training bitext.

The reason behind the success of forced-decoding is the fact that it results in alignments and segmentations that are considered more probable by the decoder based on the scores from all the translation models (phrase translation model, language model, and re-ordering model) and features and thus results in a more reliable translation probability distribution. Thus, forced-decoding is a re-estimation technique that refines the phrase table by re-distributing the probability mass among those phrase segmentations that lead to translations of the source sentences equivalent to the given target sentences with high

probability. As discussed in (Wuebker et al., 2010), using forced-decoding an improvement of up to 1.4 BLEU points can be achieved over a baseline for German→English. An additional advantage of forced-decoding is that the re-estimated phrase table size is significantly reduced compared to the original phrase table.

However, there are some limitations to forced-decoding based estimation. First, it is too conservative in selecting phrase segmentations by restricting the decoder to only produce the given reference translation. In a related line of research, Liang et al. (2006) used forced-decoding for an end-to-end discriminative MT model and called this restrictive selection "bold updating" where only the "Viterbi approximation" leading to the target translation is considered. They discover that bold updating is not always the best strategy, especially in the case of noisy alignments. For example, in case of an incorrect alignment for a noisy sentence pair, the initial phrase segmentations leading to the exact target reference will have a very low probability, and these segmentations are already unreliable due to the noisy alignments. Reinforcing such segmentations during re-estimation may be undesirable to improve the reliability and compactness of translation models.

As an alternative to "bold updating", Liang et al. (2006) proposed "max-BLEU updating," where instead of updating the model weights towards the given exact target, the updates should be made with respect to the hypothesis with the highest score. With "max-BLEU updating," although the best hypothesis is far from exact reference, its correspondence to the correct segmentation would be more reasonable. However, as pointed out in (Chiang et al., 2008), even updating towards max-BLEU may be undesirable since the max-BLEU translation may contain some peculiar rules due to noise in the training data. As a result, Liang et al. (2006) proposed a third strategy called "local updating," where the idea is to first generate an n-best list based on model scores and select the highest-BLEU translation from this list. However, the limitation of this approach is that the max-BLEU translation search is limited to the space of the n-best hypotheses based on model probabilities. Therefore, Chiang et al. (2008) proposed oracle-BLEU updating, where instead of searching for max-BLEU translations from the n-best translations, the idea is to select the translation that maximizes the optimal combination of BLEU and model score. Note that neither Liang et al. (2006) nor Chiang et al. (2008) have used the max-BLEU or oracle-BLEU for the training of generative phrase translation models. Liang et al. (2006) compared the update strategies for an end-to-end discriminative model, whereas Chiang et al. (2008) proposed oracle-BLEU update for the optimization of model weights, λ's.

As a second limitation, forced-decoding requires searching the exact hypotheses within an exponential space, which will require relaxation of most of the decoding constraints of phrase-based SMT such as distortion limit, beamwidth, and stack size resulting in slow training time.

A third limitation is that in forced decoding phrase segmentations are re-estimated only from hypotheses that match the given target reference; therefore, the word order and hence the re-ordering of phrases remain the same as the initial model. As a result re-estimation of lexicalized re-ordering models is not possible with forced decoding.

Motivated by the success of forced-decoding and observing its limitations, we propose another solution, which is to re-estimate the translation models by aligning the source sentence with the oracle-BLEU hypothesis similar to the one proposed in (Chiang

et al., 2008). We rank the possible hypotheses on a score, which is the combination of model score and BLEU score. We propose that the segmentations observed in the hypothesis with the highest BLEU score, with respect to the target reference, provide a more realistic estimate of the translation models from the decoding perspective than the estimate obtained with "bold updating" or "local updating." In this way, we consider a middle ground between the approaches of Wuebker et al. (2010) and Liang et al. (2006). Thus, our aim is to re-estimate the phrase segmentations in such a way that unreliable phrase translations are not reinforced in the case of noisy data, however, at the same time, the extracted segmentation and their probabilities are not completely different from those extracted by heuristic extraction.

To understand this, consider Figure 3.2, which shows word alignments for a noisy example from the training data for Arabic→English. The connecting lines between the source (Arabic) and target (English) words or tokens show the set of alignments for this sentence pair obtained using EM-algorithm. However, note that this is a partially noisy sentence pair, i.e., the given target is not a complete translation of the given source sentence and has some fragments missing when compared to a translation provided by a human annotator. The additional fragments missing in the aligned target are marked in red in the human translation. Because of the noisy fragments, many words in the source sentence remain unaligned. When phrase pairs are extracted from this noisy sentence alignment, the presence of these unaligned words leads to the extraction of overlapping unreliable phrase segmentations.

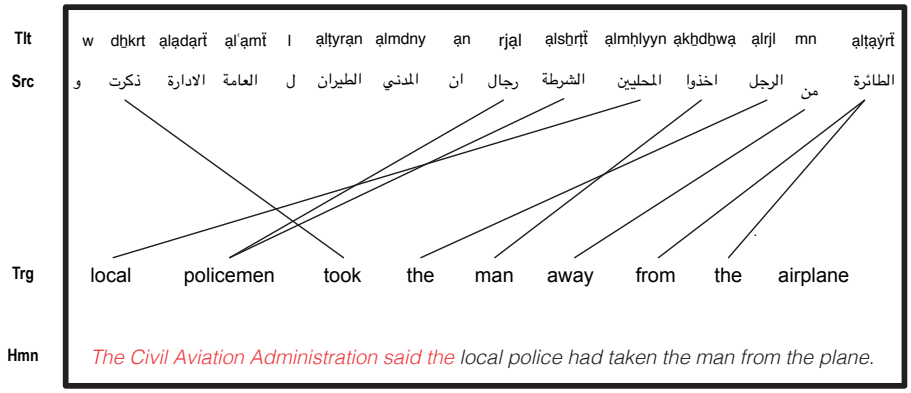


Figure 3.2: Word alignments; **Tlt**= Transliteration of source sentence, **Src**=Source (Arabic) sentence, **Trg**=Aligned target sentence in bitext, **Hmn**=Human translation of source sentence; Words in red have no correspondence on source side.

In forced-decoding, the idea is to re-estimate only those phrase segmentations which lead to the highest ranking hypothesis translation equivalent to the given target. As shown in Figure 3.3, only three phrase pairs will be re-estimated from the given sentence pair and the rest of the phrase pairs which were extracted in heuristic training will be discarded. However, since in this case, the original reference itself is noisy, the decoder selects unreliable phrases translation such as $\left[(w\ dẖkrt\ ạlạdạrĭ\ ạl\acute{ạ}mĭ) \leftrightarrow \text{the} \right]$ and $\left[(l\ ạltyrạn\ ạlmdny\ ạn\ rjạl\ ạlsẖ) \leftrightarrow \text{local policemen took} \right]$. In this way, forced-decoding

can lead to reinforcement of the probability of such noisy segmentations.

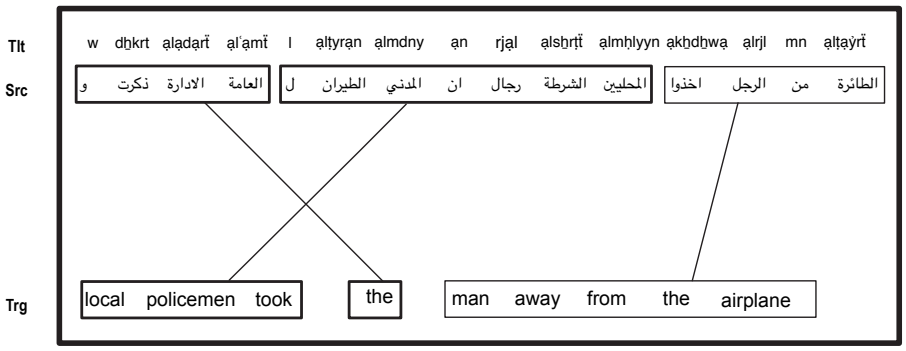


Figure 3.3: Phrase segmentation from forced-decoding.

With the re-estimation strategy based on the model score, as shown in Figure 3.4, the best hypothesis is closer to human translation. However, due to fewer unaligned words, it is relatively farther than the given original reference translation. For example, for one of the source phrase, this approach selects a target phrase as [*'local police officer'*] while a correct target phrase with respect to the given reference sentence would be *"local policemen"*. On the other hand, as shown in Figure 3.5, the oracle-BLEU is

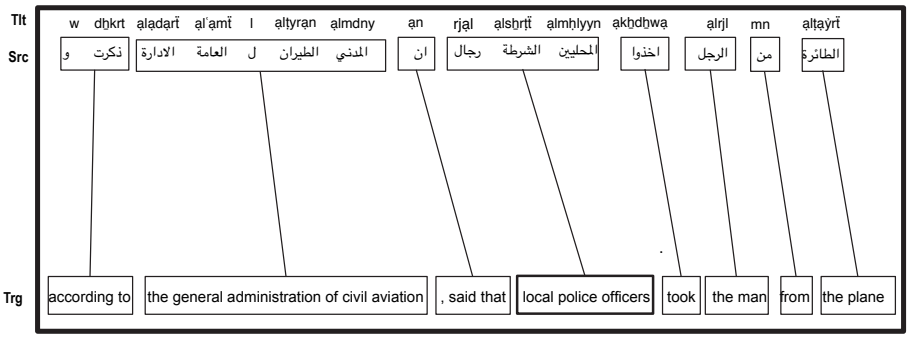


Figure 3.4: Model-best hypothesis segmentation.

midway between the original target reference and the best model hypothesis. Unlike forced-decoding, it does not select the noisy segmentations, and at the same time, aligned phrases are closer to the given reference. In oracle-BLEU re-estimation, the goal is not to search for the original target reference but to search for the best scoring hypotheses allowing for the pruning of the hypothesis space with general constraints of decoding. Therefore, unlike forced-decoding, relaxation of decoding constraints is not required, and hence the re-estimation can be done within the usual decoding time. Since the decoder is not restricted to generating the exact reference, the oracle-BLEU translation may have a different re-ordering from the reference. Thus, our approach also allows for the re-training of re-ordering models.

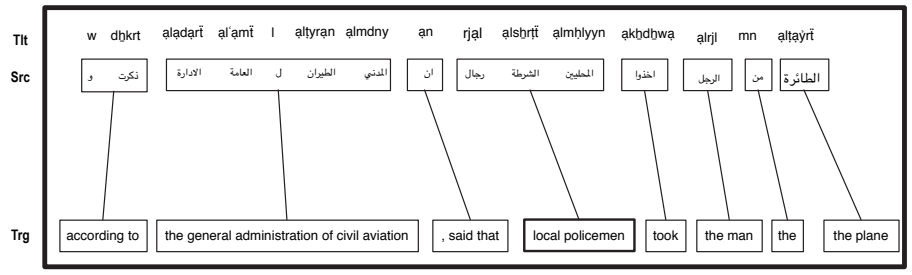


Figure 3.5: Oracle-BLEU-best hypothesis segmentation.

Oracle-BLEU translations have previously been used for weight optimization (tuning) based on n-best re-ranking (Srivastava et al., 2011), for comparing re-ordering constraints (Dreyer et al., 2007) and for the analysis of model errors (Wisniewski et al., 2010) in SMT. However, to the best of our knowledge, we present the first approach to re-estimate translation models from oracle-BLEU translations for SMT. Our approach for obtaining oracle-BLEU translations is based on (Chiang et al., 2008) who use an optimal-BLEU approximation for on-line large margin training of syntactic and structural features in a discriminative machine translation setting. Similar to Chiang et al. (2008), we use a weighted combination of model score and a pseudo-document approximation of the BLEU score to obtain oracle translations; see Section 3.3.1 for details.

3.3 Model re-estimation

The basic idea of our approach is to re-train or re-estimate the translation models from the phrase segmentations that are used to achieve n-best oracle-BLEU translations obtained by decoding with the initial models trained with heuristic extraction. Given a source and its reference translation, the oracle-BLEU translation is defined as the translation with the highest BLEU score with respect to the given reference sentence. It can be seen as a translation that is most similar to the reference translation, out of all the possible translations that an SMT system can produce for a given set of models and decoding constraints.

3.3.1 Optimal oracle-BLEU score

As discussed in the previous section, instead of absolute BLEU scores, we use an optimal-BLEU score that is a combination of model score and BLEU score for each hypothesis. The optimal-BLEU hypothesis is defined in Chiang et al. (2008) as follows:

$$\begin{aligned} e^* = & \operatorname*{argmax}_e (B(\mathbf{e}) - \mu(B(\mathbf{e}) - \mathbf{w} \cdot \mathbf{h}(\mathbf{e})), \\ & \operatorname*{argmax}_e \Big((1-\mu)(B(\mathbf{e})) - \mu(\mathbf{w} \cdot \mathbf{h}(\mathbf{e})) \Big), \end{aligned} \tag{3.5}$$

where B**(e)** is the BLEU score for each hypothesis **e**, **h(e)** are the model scores and **w** is the optimized weight for each model or feature, μ is a hyper-parameter in range $[0, 1]$ which controls the relative weight between BLEU score and model score (sentence translation probabilities). A value of $\mu = 0$ implies that the optimal hypothesis is the one with the highest BLEU score while $\mu = 1$ implies that the optimal hypothesis is the translation with the highest model score (sentence translation probability). We use a value of $\mu = 0.5$ which implies that the optimal hypothesis has BLEU scores almost as high as the max-BLEU translations, yet are not very far from the translations preferred by the model.

Ideally, one would like to calculate the optimal-BLEU score defined above for each hypothesis translation. However, there is a problem in calculating BLEU for individual sentences. As discussed in (Chiang et al., 2008), BLEU is not designed to be used for sentences in isolation. Instead, it is defined to be calculated over the entire test set or the entire test document. Chiang et al. (2008) and Watanabe et al. (2007) proposed to calculate BLEU for a sentence in the context of an exponentially-weighted moving average of previous translations. We follow the approach of (Chiang et al., 2008) for sentence-level BLEU computation which is briefly discussed as follows:

Given a source sentence **f**, and its reference translation **r**, for a hypothesis translation e^*, let $c(\mathbf{e}^*)$ be defined as the vector of the following counts:

- $|\mathbf{f}|$ (source length),
- $|\mathbf{r}|$ (effective reference length),
- $\mathrm{ngram}(\mathbf{e}^*)$ (the number of n-gram in $\mathbf{e}^*$),
- $\mathrm{match}(\mathbf{e}^*, \mathbf{r})$ (number of n-gram matches between $\mathbf{e}^*$ and **r**).

Then **O** is defined as "pseudo-document" and is calculated as an exponentially-weighted moving average of the vectors from previous sentences and the current sentence. This implies that for each training sentence if e^* is the hypothesis translation; **O** and input length $\mathbf{O}_f$ are updated as follows:

$$\begin{aligned} &\mathbf{O} \leftarrow 0.9(\mathbf{O} + c(\mathbf{e}^*)) \\ &\mathbf{O}_f \leftarrow 0.9(\mathbf{O}_f + |\mathbf{f}|) \\ &c(\mathbf{e}^*) = \textbf{[|r|, |f|, ngram(e), match(e*,r)]}. \end{aligned} \tag{3.6}$$

The above update equation simply means that while processing the sentence in a document, we maintain vectors to carry forward an approximation of the values (from preceding sentences) required for BLEU computation. Both the vectors O and O_f are updated and the final vector for the current sentence is computed as a fraction of the resultant values. However, as the document is processed, O will become larger and the effect of the current sentence on the BLEU score will be reduced. This is corrected by scaling the current BLEU score with current effective input length. With these values, the BLEU score for a sentence pair (f, r) and translation e^* is defined as:

$$B(\mathbf{e}^*; \mathbf{f}, \mathbf{r}) = (\mathbf{O}_f + |\mathbf{f}|) * BLEU\Big(\mathbf{O} + c(\mathbf{e}^*; \mathbf{f}, \mathbf{r})\Big). \tag{3.7}$$

3.3.2 Oracle-BLEU re-estimation

For obtaining the oracle-BLEU translations, we first train the translation models on the bitext using the standard pipeline of word alignment and heuristic extraction. For our experiments, along with the phrase translation model and language model, we also train a bilingual language model (BiLM) (Niehues et al., 2011), as well as lexicalized (Tillman, 2004) and hierarchical re-ordering models (Galley and Manning, 2008). The phrase table contains extracted phrase translations along with forward and backward phrase translation probabilities and lexical translation probabilities (Koehn et al., 2003). It also stores the word alignments for each translation pair. We use a bilingual language model (BiLM) specifically as an instance of a re-ordering model in order to determine the effect of re-estimation on re-ordering decisions from oracle-BLEU translations. We use the decoder trained on these models to translate the training bitext. Along with the 1-best translation (based on model probabilities), we also store search graphs or lattices generated during the translation process. The overall procedure can be summarized in the following steps:

Given bitext $\mathbf{D} = [(f_i, r_i, a_i)]$ where $i = 1, 2, \ldots, n$ and f_i, r_i and a_i are source sentence, target sentence and sets of word alignments between them respectively.

1. Train the initial translation model PT_0 and the Re-ordering model RM_0 using the heuristic alignment method for the given data and alignments (the subscript 0 stands for the initial models).

2. For each source sentence f_i:

   2.1. Using the models PT_0, RM_0 and language model LM_{mono}, decode source sentence f_i and generate a corresponding hyper-graph (lattice) H_i^{model} representing model scores.

   2.2. Convert the lattice H_i^{model} to H_i^{BLEU} by comparing the transitions against reference r_i.

   2.3. Extract n-best optimal-BLEU hypotheses $NB = [h_1^{OB}, h_2^{OB}, \ldots, h_n^{OB}]$ from lattice H_i^{BLEU}.

3. Extract phrase pairs used in NB and re-estimate the translation model $PT_{re-estimate}$.

Re-estimation of the translation models is straightforward by counting the joint and marginal counts for all phrase pairs in the n-best oracle-BLEU hypotheses and re-calculating the forward and backward probabilities. We retain the lexical weightings from the initial translation model.

3.3.3 Searching for oracle-BLEU

Finding the best scoring hypothesis based on translation probabilities (referred to as model scores hereafter) from the hypothesis space (known as lattice) is a relatively easy task. Traditional SMT systems implement this by using a beam search (Koehn, 2010) based on decoding as described in Section 2.1. A lattice is a hyper-graph where the nodes represent all partial hypotheses that were considered during decoding of the

source sentence. The edges represent the costs or scores of transitions between the partial hypotheses based on the model probabilities. In other words, each transition or edge in the lattice represents a translation corresponding to each phrase segment. The terminating node of the graph represents the complete hypothesis with all words/spans of the source sentence covered. The edge score is calculated based on the weighted combination of the model scores for each transition.

However, searching for the highest-scoring sentence within the full search space, specifically for sentence-level metrics such as BLEU, is non-trivial. This is due to the fact that computing the contribution of each edge towards the sentence-BLEU (which is based on the n-gram precision) requires taking into consideration all other edges on the same path. Various approaches have been proposed for efficiently searching the oracle-BLEU translation from the lattice (Dreyer et al., 2007), (Li and Khudanpur, 2009), (Sokolov et al., 2012). For our purpose, we use the dynamic programming-based algorithm of Dreyer et al. (2007) for which we first convert the translation lattice to an isomorphic oracle-BLEU lattice, which has the same set of nodes as the translation lattice. However, the edges in the oracle-BLEU lattice represent BLEU score differences corresponding to each transition. Finally, we extract n-best candidate translations from this graph according to the BLEU score, as defined in Equation 3.7.

We briefly explain the algorithm for the computation of the oracle-BLEU score of each partial hypothesis in a lattice. A detailed explanation is provided in (Dreyer et al., 2007). Consider the example shown in Table 3.2. Along with given source and

Source	لجنة النقل تحذر من سيطرة القطاع الخاص علي الموانئ
Target	transport committee warn of private sector control of ports
h_A	transport committee warn of control of the private sectors to ports
h_B	transport committee warn of the private sector control of ports

Table 3.2: Arabic-English sentence pair example.

target sentences, we show two partial hypothesis translations h_A and h_B of the given source sentence which are generated by the decoder. Note that usually there is a very large number of complete hypotheses in the decoding space. However, for the sake of brevity, here we show only two. The translation (model) lattice for this example is shown in Figure 3.6. Each edge in the model lattice represents the phrase translation corresponding to one partial expansion. Here the path corresponding to hypothesis h_A is the path with maximum model probability score [$path_A = 0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6$]. The path corresponding to hypothesis h_B is [$path_B = 0 \rightarrow 1 \rightarrow 2 \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow 6$]. Note that the actual scores are costs or negative log probabilities, however, here we use translation probabilities for the sake of illustration. The edges in Figure 3.6 are labelled with model scores.

To compute the optimal-BLEU score corresponding to each edge, we first update the n-gram matches corresponding to each edge. Here n-gram matches mean the number of n-grams matched with the reference sentence in the partial hypothesis generated up to the terminating node of the edge.

Figure 3.7 depicts the n-gram matching and update step. Consider node 1 for which

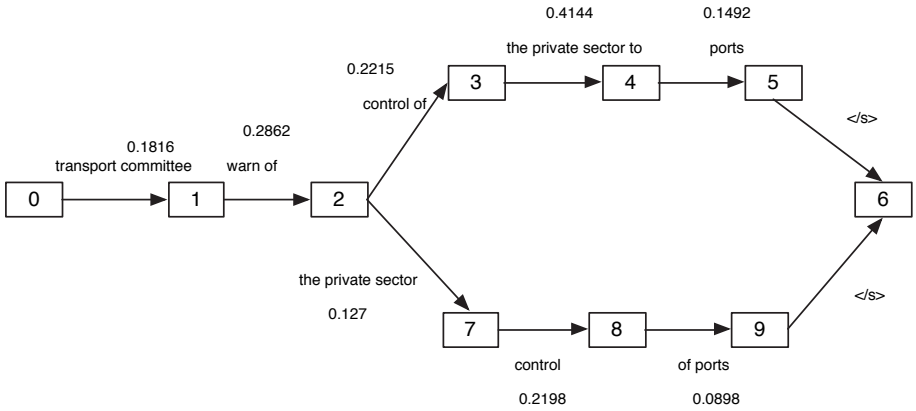


Figure 3.6: Model (translation probability scores) lattice for example in Table 3.2.

the partial hypothesis is [*'transport committee'*]. For this partial hypothesis, there are two unigram matches with the reference sentence, i.e., [*transport, committee*]. Similarly, for node 2, the partial hypothesis generated so far is [*'transport committee warn of'*] for which there are four unigram matches, three bigram matches, two trigram matches and one 4-gram match with the reference translation. These counts for each n-gram size match are represented at each edge. The total count of each n-gram size in the reference string is [9,8,7,6], i.e., nine unigrams, eight bi-grams, seven trigrams, and six 4-grams.

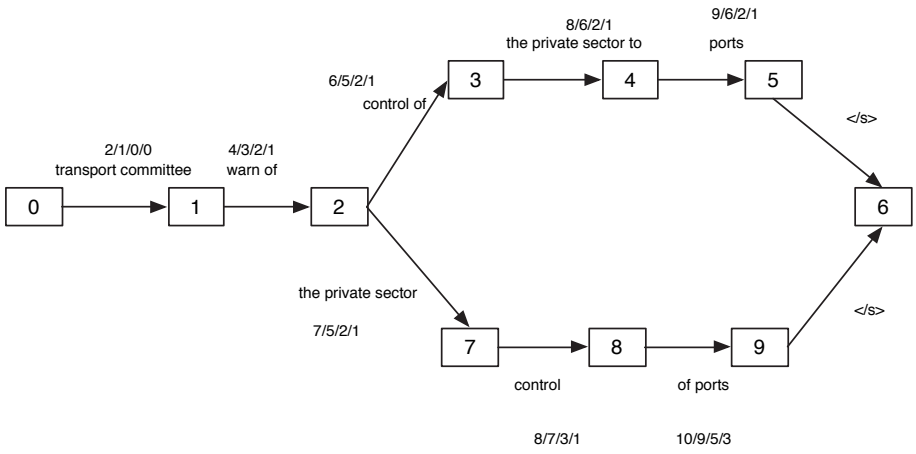


Figure 3.7: N-gram matching and updates for the example in Table 3.2.

In Figure 3.8, the value *'p'* labeled at each edge represents the corresponding n-gram precision calculated based on the n-gram counts of each edge (from Figure 3.7). Other than precision, calculation of BLEU requires the calculation of *'brevity-penalty.'* For sentence-level BLEU, the brevity penalty can be calculated as the ratio of the length of the full hypothesis to that of the reference length (based on the number of tokens). However, for partial hypotheses or subpaths, the total lengths of the resulting full hypotheses are not available. Hence, the exact value of the brevity-penalty for a partial hypothesis is not computable.

As a solution, to calculate the brevity-penalty for each subpath, we follow (Li and Khudanpur, 2009): the true reference length corresponding to an expansion is calculated as the product between the length of source sentence covered by that partial hypothesis and the ratio between the full source length and full reference length. Thus for the example in Table 3.2, the full source length to target ratio is 9/8. As shown in Figure 3.8, for the expansion $(1 \rightarrow 2)$, up to the node 2, the source length covered is 4; as a result, the approximated reference length for this partial expansion becomes $(9/8) * 4 = 4.5$, and the length of the hypothesis up to this node is 4. Hence the brevity-penalty $= (4/4.5) = 0.889$. The brevity-penalty for each expansion is represented as *'bp'* in Figure 3.8. With n-gram precision and brevity penalty, the partial BLEU *"pBLEU"* score for each edge is calculated. This is combined with the model score according to Equation 3.5 to calculate optimal-BLEU *"oBLEU"* as shown in Figure 3.8.

As shown in Figure 3.8, the final oracle-BLEU (oBLEU) for hypothesis h_B is 0.4534 while that for hypothesis h_A is 0.2871. Apparently, since the hypothesis h_B is more similar to the given reference as compared to h_A, it leads to a higher BLEU score for h_B.

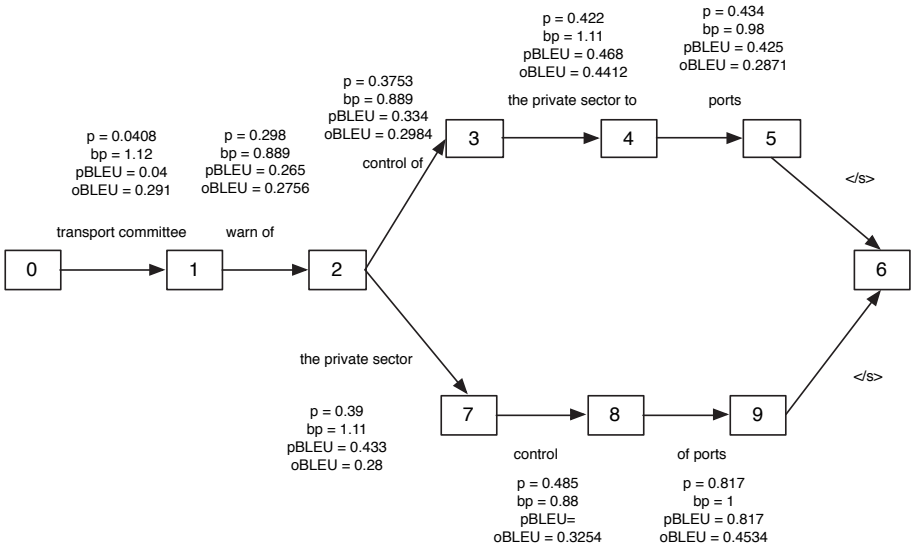


Figure 3.8: Oracle-BLEU lattice for the example in Table 3.2.

With this BLEU-lattice as input, we run Yen's algorithm (Yen, 1970) to search for n-best oracle-BLEU hypotheses.

3.3.4 Re-estimation of re-ordering model, language model, and BiLM

Word alignments for each phrase pair used in the n-best oracle-BLEU translation can be obtained from the initial phrase table. Using these alignments, we extract the sentence alignments between each source sentence and each of their n-best oracle-BLEU translations. With these word alignments, we re-train the re-ordering model and BiLM on these translations and alignments. To re-estimate the language model, we simply extract the n-best BLEU translations and build an n-gram language model from these translations, which is then interpolated with the baseline language model for evaluation.

3.3.5 Avoiding over-fitting

An important problem that arises during the re-estimation of the phrase tables is that of over-fitting. Many of the phrase translations that are only observed in a few sentences will have artificially high translation probabilities. In extreme cases, if a phrase translation is observed only in one sentence, such a phrase translation, known as a singleton, will have a probability of 1. While translating the sentences in the training bitext, it is highly likely that the highest-scoring hypothesis will have a phrase segmentation that is extracted from the same sentence due to these high scores. Re-estimating the translation models from such n-best translations could reinforce the probabilities of these phrase pairs in the resulting models leading to over-fitting.

For forced-decoding, Wuebker et al. (2010) address this problem by using a leave-one-out approach where they modify the phrase translation probabilities for each sentence pair by removing the counts of all phrases that were extracted from that particular sentence. By thus modifying the probabilities, they ensure that while translating a sentence, the model will not be biased to select a phrase pair that was extracted from the same sentence pair. However, in our oracle-BLEU re-estimation approach, there is no constraint to produce the exact translation. Instead, the optimal-BLEU translations may be very different from the references. Thus it is not strictly necessary to apply leave-one-out in our approach as a solution to over-fitting. Instead, we handle the problem by simply removing all the phrase pairs below a threshold count, which in our case is 2:

$$\phi_{init} = \phi_{baseline} - \phi_{C(e,f)<2}, \tag{3.8}$$

thus removing phrase pairs with high probability but low frequency.

3.4 Experimental set up

We conduct our experiments for an Arabic→English parallel corpus of approximately 1 million sentence pairs. First, we establish a baseline system by training models using heuristic extraction on this parallel corpus. We then compare this to a forced-decoding

implementation and to our approach, where we re-estimate the models by oracle-BLEU, all using the same bitext.

3.4.1 Baseline

The initial training corpus we use is a collection of parallel sentences taken from following data sources released by the Linguistic Data Consortium: LDC2006E25, LDC2004T18, several GALE corpora, LDC2004T17, LDC2005E46, LDC2007T08, and LDC2004E13. Table 3.3 provides statistics of the training data. Initial word-alignments

Table 3.3: Arabic-English parallel training data statistics.

#Sentences	Arabic Tokens	English Tokens
944,468	29,288,811	30,058,376

are produced with GIZA++ (Och and Ney, 2003). Along with the phrase translation models (including forward and backward probabilities and lexical weightings (Koehn et al., 2003)), we train two re-ordering models, namely lexicalized and hierarchical distortion models (8 features), and also a lexical and a part-of-speech BiLM.

The English 5-gram target language model is trained using SRILM (Stolcke et al., 2011) with Kneser-Ney smoothing trained on news data from various sources of nearly 1.6 Billion tokens.

We use an in-house implementation of a phrase-based SMT system similar to the Moses SMT toolkit (Koehn et al., 2007). A distortion limit of 5 and a maximum phrase length of 7 is used for the baseline system.

Model weights are optimized on NIST's MT04 data set using pairwise ranked optimization (Hopkins and May, 2011). For testing the performance of the re-estimated models, we tune (optimize model weights) different systems while replacing the baseline models with the corresponding re-estimated models incrementally. We also experiment with the interpolation of re-estimated models with the corresponding baseline model. For all the settings, weights are tuned on MT04. We evaluate results on 4 test sets MT05, MT06, MT08, and MT09.

Case-insensitive 4-gram BLEU (Papineni et al., 2002) is used as evaluation metric. Approximate randomization (Noreen, 1989; Riezler and Maxwell, 2005) is used to detect statistically significant differences.

3.4.2 Forced-decoding experiments

For forced-decoding, we use the existing implementation within the Moses SMT toolkit (Koehn et al., 2007). For this purpose, we use the baseline phrase translation model to decode its own training corpus, while constraining the output to exactly match the reference. Standard decoding constraints such as stack size limits, beamwidth, and distortion limit, which are necessary to maintain the efficiency of decoding, restrict the possibility of producing the reference translation exactly. In many cases, the exact reference is pruned out from the search space of the decoder. Therefore, to increase the chances of producing the exact reference requires relaxation of all constraints. This

also results in a decrease in decoding speed. Motivated by the discussion in (Foster and Kuhn, 2012), we relax the parameters as follows:

- Distortion Limit: ∞
- Stack size: 2000
- Beamwidth: 10e(-30)
- No threshold pruning of translation model

No re-ordering or language model is used for forced-decoding of the bitext. However, as observed in our experiments, even with such a high relaxation of constraints, only approximately 40% of the sentences can be decoded to match the exact reference which is much lower than the maximum success rate of 78% for Chinese-English and 81% for French-English as discussed in (Foster and Kuhn, 2012). This low success rate of forced-decoding for our data can be attributed to the presence of noisy sentence pairs taken from various sources. Therefore, additionally, we allow for partial covers as a feasible hypothesis for those sentence pairs for which exact reference can not be produced with the given decoding parameters. By partial covers, we mean the best-scoring hypotheses where a segment of the target sentence can be matched but not the full target sentence.

3.4.3 Oracle-BLEU re-estimation

As discussed in Section 3.3.2, we implement the re-estimation of the models by extracting phrase pairs from the n-best optimal BLEU translations of the source sentences in the training set, using target sentences as references. For this purpose, we first train an initial SMT system and use it to decode the bitext. This system is identical to the baseline system except for the removal of phrase pairs with low frequency from the baseline phrase table, as described in Equation 3.8. To obtain the n-best oracle-BLEU translations, we experiment with different values of n, where $n \in \{1, 10, 100\}$. From these oracle-BLEU translations and alignments, all phrases that were used in the derivation of these n-best sentences are extracted, and the models are re-estimated from these phrases only. Hierarchical and lexicalized re-ordering models, as well as the BiLMs, are re-trained using the source sentences, oracle-BLEU translations, and corresponding word alignments.

3.5 Results

In this section, we discuss the experimental evaluation of our proposed oracle-BLEU re-estimation approach for different model variants and settings. We focus on the effectiveness of oracle-BLEU re-estimation over heuristic model training and over forced-decoding.

3.5.1 Effect of n-best variation

As discussed in Section 3.2, we re-estimate the models from n-best oracle-BLEU translations with three different values of n (1, 10, 100). Models re-estimated for a value

of n were evaluated under three model settings: phrase table re-estimation, interpolation, and BiLM re-estimation. The best improvements over the baseline for oracle-BLEU re-estimation are obtained by using only the Viterbi translation (alignment), i.e., for $n = 1$ as shown in Table 3.4. This is in contrast with forced-decoding, as discussed in (Wuebker et al., 2010), where the best improvements are obtained for $n = 100$.

Table 3.4: Performance of the oracle-BLEU re-estimation with varying size n of n-best list for the NIST MT09 test set. Statistically significant improvements over baseline are marked ▲ at p < 0.01, and with △ at p < 0.05. ▼ represents a statistically significant drop at p < 0.01. Values in brackets show gain in BLEU scores over the baseline for each result. PT_{re}=Phrase table re-estimation, PT_{in}=phrase table interpolation and $BiLM_{re}$=BiLM re-estimation.

Baseline	50.1		
	n=1	n=10	n=100
PT_{re}	50.1(0.0)	50.1(0.0)	50.0(-0.1)
PT_{in}	$50.7^{\blacktriangle}(+0.6)$	$50.5^{\blacktriangle}(+0.4)$	50.0(-0.1)
$BiLM_{re}$ + PT_{in}	$50.9^{\blacktriangle}(+0.8)$	$50.5^{\blacktriangle}(+0.4)$	49.6(-0.5)

3.5.2 Comparison with forced-decoding

To compare the effectiveness of our approach against forced-decoding, we run both approaches on the same training data. Table 3.5 provides a comparison between BLEU improvements achieved by forced-decoding (100-best) and our oracle-BLEU re-estimation approach (1-best) over the baseline under two re-estimation settings. Here, we only list comparisons between settings that result in the best improvements over the baseline. For forced-decoding, this is the model re-estimated at $n = 100$, and for oracle-BLEU, this is the model re-estimated at $n = 1$.

First, we only re-estimate the phrase tables for both methods and evaluate SMT systems trained with these phrase tables. For both methods, the re-ordering model and language model are the same as the baseline. It can be observed from Table 3.5 that while for forced-decoding, the BLEU score for phrase table re-estimation drops substantially for all test sets (up to -1.4 for MT09), our oracle-BLEU phrase table re-estimation shows either slight improvements ($+0.2$) or negligible drops in BLEU compared to the baseline.

In the second setting, we linearly interpolate the re-estimated phrase table with the baseline phrase table (using uniform interpolation weights) for both approaches and evaluate the SMT systems trained with the interpolated phrase tables. Again, all other models remain the same as the baseline. In Table 3.5, for phrase-table interpolation, forced-decoding shows only a slight improvement for MT09 ($+0.3$), MT08 and MT06 and still suffers from a substantial drop for MT05 (-0.3). On the other hand, oracle-BLEU re-estimation shows consistent improvement for all test sets with a maximum gain of up to $+0.7$ for MT06.

Table 3.5: Comparison of BLEU scores for 4 test sets for *forced-decoding* and *oracle-BLEU re-estimation*. PT_{re}=Phrase table re-estimation, PT_{in}=phrase table interpolation, $BiLM_{re}$=BiLM re-estimation, FD=Forced-decoding and OB=oracle-BLEU.

	MT05		MT06		MT08		MT09	
Baseline	58.5		47.9		47.3		50.1	
	FD	OB	FD	OB	FD	OB	FD	OB
PT_{re}	$57.4^{\blacktriangledown}$ (-1.1)	$58.7^{\triangle}$ (+0.2)	46.3(-0.7)	$47.8^{\blacktriangledown}$ (-0.1)	$46.1^{\blacktriangledown}$ (-1.2)	$47.4^{\triangle}$ (+0.1)	$48.7^{\blacktriangledown}$ (-1.4)	50.1(0.0)
PT_{in}	$58.2^{\triangledown}$ (-0.3)	**58.8**$^{\blacktriangle}$ (+0.3)	48.0(+0.1)	**48.6**$^{\blacktriangle}$ (+0.7)	47.5(+0.2)	**47.7**$^{\blacktriangle}$ (+0.4)	$50.4^{\triangle}$ (+0.3)	**50.7**$^{\blacktriangle}$ (+0.6)

3.5.3 Comparison with forced decoding with leave-one-out

As already pointed out in Section 3.3.5, Wuebker et al. (2010) uses a leave-one-out strategy to mitigate over-fitting, which significantly improves the performance of forced-decoding. For additional analysis, we also compare oracle-BLEU re-estimation to forced-decoding with leave-one-out.

As can be observed from Table 3.6, even with leave-one-out, phrase table re-estimation through forced-decoding yields performance below the baseline (a drop of -0.8 for MT05, -0.3 for MT08 and -0.3 for MT09). In contrast, the phrase table trained through oracle-BLEU re-estimation still performs either similar to or slightly better (up to $+0.2$ for MT05) than the baseline, along with the benefit of a substantially reduced phrase table size. When interpolated with the baseline phrase table, both approaches show significant improvements over the baseline. This implies that forced-decoding in itself with or without leave-one-out may degrade performance as compared to heuristic extraction. As shown in the last row of Table 3.6, only when interpolated with the original phrase table does forced-decoding (with leave-one-out) outperform the baseline trained using heuristic extraction. On the other hand, oracle-BLEU re-estimation consistently performs better than forced-decoding (up to $+0.6$ for MT09) with leave-one-out when interpolated with the baseline phrase-table.

Table 3.6: Comparison of BLEU scores for Forced-decoding with leave one out and oracle-BLEU re-estimation. PT_{re}=Phrase table re-estimation, PT_{in}=phrase table interpolation, $BiLM_{re}$=BiLM re-estimation. FD_{LO}=Forced-decoding and OB=oracle-BLEU.

	MT05		MT08		MT09	
Baseline	58.5		47.3		50.1	
	FD_{LO}	OB	FD_{LO}	OB	FD_{LO}	OB
PT_{re}	$57.7^{\blacktriangledown}$ (-0.8)	$58.7^{\triangle}$ (+0.2)	$47.0^{\blacktriangledown}$ (-0.3)	$47.4^{\triangle}$ (+0.1)	$49.8^{\blacktriangledown}$ (-0.3)	50.1(0.0)
PT_{in}	$58.7^{\triangle}$ (+0.2)	**58.8**$^{\blacktriangle}$ (+0.3)	47.6(+0.3)	**47.7**$^{\blacktriangle}$ (+0.4)	$50.6^{\triangle}$ (+0.5)	**50.7**$^{\blacktriangle}$ (+0.6)

In summary, based on the results shown in Table 3.5 and Table 3.6, we can conclude that oracle-BLEU re-estimation performs better than standard heuristic training and also yields performance better than forced-decoding.

3.5.4 Re-estimation of re-ordering models

As discussed in Section 3.2, an important novelty of oracle-BLEU re-estimation is that it also allows re-training of other models alongside the phrase table. In Table 3.7, we provide the results for the re-estimation of re-ordering models (lexicalized and hierarchical distortion models) and the bilingual language models for test sets MT08 and MT09. Note that the forced-decoding procedure, by definition, only allows for re-training of phrase translation models. This is because their goal is to search for exact references in the hypothesis space. Therefore only the re-orderings observed in the target sentence would be permitted, and no new re-ordering model can be learned.

We observe that unlike phrase table re-estimation, re-estimated re-ordering models (DM_{re}) do not provide any significant improvements over the original models. Instead, a drop of up to -0.6 BLEU is observed. Interpolation (DM_{in}) of the re-estimated re-ordering models with the baseline models provides results equivalent to the baseline models. This implies that although oracle-BLEU training allows for re-estimation of re-ordering models, it does not provide any significant gain over the baseline. Interpolation of all the re-estimated models (phrase table, re-ordering models, and bilingual language models) shows an improvement of up to 0.8 BLEU over the baseline.

Finally, we evaluate the effectiveness of BiLM re-estimation in two additional settings, as shown in Table 3.7. The re-estimated BiLM on its own adds BLEU improvement of up to $+0.5$ (for MT09). The highest bleu improvement of $+0.8$ is achieved for a system trained on a re-estimated BiLM and all other models being interpolated with the baseline models, as shown in the last row.

Table 3.7: Oracle-BLEU re-estimation of re-ordering Models. DM_{re} = Re-estimated (Lexical Re-ordering Model + Hierarchical Re-ordering Model), DM_{in}=Interpolation of re-ordering models with baseline models. $BiLM_{re}$=Re-estimated Bilingual Language model. $BiLM_{in}$=Interpolation of BiLM.

	MT08	MT09
Baseline	47.3	50.1
FD	46.1$^{\blacktriangledown}$(-1.2)	48.7$^{\blacktriangledown}$(-1.4)
DM$_{re}$ + Baseline(PT, BiLM, LM)	46.9(-0.4)	49.5(-0.6)
DM$_{in}$ + Baseline(PT, BiLM, LM)	47.2(-0.1)	50.1(0.0)
BiLM$_{re}$ + Baseline(PT, DM, LM)	47.5(+0.2)	50.6(+0.5)
BiLM$_{in}$ + Baseline(PT, DM, LM)	47.4(+0.1)	50.4(+0.3)
All models interpolated	**47.7**$^{\blacktriangle}$(+0.4)	**50.9**$^{\blacktriangle}$(+0.8)

The results discussed in Table 3.7 demonstrate that although re-estimation of re-ordering models does not provide any significant gain over the baseline models, it yields performance similar to the baseline. Moreover, re-estimation of Bilingual language models (BiLM) which can also be considered an instance of the re-ordering model, provides additional improvements over the re-estimation of phrase translation models.

3.5.5 Model compression

As already discussed in Section 3.2, oracle-BLEU re-estimation results in the removal of unintuitive and overlapping phrases that may not be useful for decoding a test sentence. Oracle-BLEU training results in a re-estimated phrase table with a significantly reduced size as compared to the baseline. As shown in Table 3.8, the minimum phrase table size after re-estimation is only 3% of the baseline for 1-best oracle-BLEU. While for forced-decoding, the minimum phrase size obtained with 100 best translations is 8% of the baseline phrase table. Thus our approach provides a reduced phrase table as compared to forced-decoding. Since the re-ordering models define re-ordering probabilities for the same phrase translation as in the phrase table, the size of the re-ordering models is also reduced as a result of this phrase pruning. This reduction in size reduces memory requirements of the SMT system, thus making decoding feasible for devices with smaller memory.

Table 3.8: Phrase table size compared to Baseline for Oracle BLUE re-estimation for N=1,10,100 and Forced-decoding, OB=Oracle-BLEU, FD=Forced-decoding, FD_{LO}=Forced-decoding with leave-one-out.

	Phrase Table Size (% of baseline)
OB 100 best	5
OB 10 best	4
OB 1 best	**3**
FD 100 best	28
FD_{LO} 100 best	8

3.5.6 Phrase length comparison

Since forced-decoding requires generation of exact reference, decoding a noisy sample sentence will yield segmentations which have phrases with unaligned words. This results in an increased average phrase length (number of words or tokens in a phrase). Even with leave-one-out, inclusion of such phrases will negatively effect the generalization capability of the re-estimated model. On the other hand, we observed that the oracle-BLEU translations usually include smaller phrase segmentations which is a desirable property to achieve better generalization. Table 3.9 provides a comparison of source

Table 3.9: Phrase length comparison of translation models trained from Oracle-BLEU re-estimation and Forced-decoding.

	Source phrase length (avg)	Target phrase length (avg)
Oracle-BLEU 1-best	**3.2**	**3.5**
FD	4.7	5.0
FD Leave-one-out	3.8	4.2

and target phrase length sizes for oracle-BLEU re-estimated models and models re-estimated by forced-decoding. The average source phrase and target phrase sizes for 1-best oracle-BLEU re-estimated phrase table are 3.2 and 3.5 tokens, respectively. For forced-decoding with and without leave-on-out, it is 3.8 and 4.2 tokens, respectively. Therefore, it is evident that the oracle-BLEU translations prefer smaller phrases as compared to forced-decoding.

3.6 Conclusion

In this chapter, we have proposed a simple re-estimation strategy to improve the quality and robustness of translation models for phrase-based machine translation. We first discussed the limitations of heuristic training for phrase based MT as an answer to research question **RQ1.1**:

RQ1.1 *What are the limitations of heuristic training algorithms for phrase-based MT, and how is the translation performance of the models affected by them?*

We showed that extraction of phrases based on segmentations consistent with the word alignments leads to the extraction of overlapping noisy phrase translations, and the allocation of probability mass to these segmentations constrains the overall performance of the model. We also discussed forced-decoding, which is based on the re-estimation of phrase segmentations from hypotheses exactly matching the target reference. We discussed a limitation of forced-decoding, which leads to the re-estimation of incorrect segmentations in the case of noisy data. We also discussed the limitations of forced-decoding in terms of training complexity and its inability to reach the exact target in the case of noisy data.

As a remedy, we proposed oracle-BLEU re-estimation where phrase segmentations are selected from the best scoring hypotheses translations (in terms of BLEU) of the source sentences in the training bitext. To answer research question **RQ1.2**, we evaluated the performance of oracle-BLEU segmentations as compared to the heuristic training as well as forced-decoding based re-estimation:

RQ1.2 *Can the reliability and performance of translation models be improved by re-estimating the phrase translation table through oracle-BLEU decoding of the training data?*

Our experiments show that re-estimating translation models either by forced-decoding or from oracle-BLEU translations, yields translation performance comparable to the initial phrase translation table trained using the heuristic extraction method. At the same time, due to the removal of unintuitive overlapping phrases, the size of the phrase table is reduced significantly to just 3% of the original size. The most important aspect behind the success of re-estimated models is that the reliability or likelihood of a phrase segmentation is estimated not just on the occurrence frequency in the training data, but also on their ability to generate a high-quality translation, which includes the scores from re-ordering and language models. Further, our proposed re-estimation method performs better or comparable to the forced-decoding procedure. Forced-decoding insists on

considering only those segmentations, which lead to the target reference. This restriction of forced-decoding is based on the assumption that the training data has high translation equivalence and hence fails to reliably re-estimate from noisy sentence pairs. On the other hand, the proposed oracle-BLEU re-estimation considers phrase segmentation from hypotheses with a high optimal-BLEU score. Therefore, it encourages translations very similar to the given reference but is not restricted to generate the exact target reference. Hence, even in the case of noisy sentence pairs in the training data, it can lead to a more reliable translation instead of faulty segmentations extracted from the noisy training examples. In conclusion, our experiments show that the reliability of translation models can be improved by re-estimating phrase segmentations from translations of the source sentences in the training data that have a high score based on an optimal combination of model probabilities and BLEU score.

Finally, since the oracle-BLEU re-estimation method allows for re-estimation of re-ordering and language models, we proposed that it can yield additional improvements in the overall translation performance. To answer research question **RQ1.3**, we evaluated performance improvements through oracle-BLEU re-estimation of re-ordering and language models:

RQ1.3 *To what extent can oracle-BLEU re-estimation of re-ordering models and language models provide additional improvements in translation performance?*

While forced-decoding does not allow for the re-estimation of re-ordering models due to the restriction to generate the exact translation, the proposed oracle-BLEU re-estimation allows re-ordering of the best hypothesis to be different than the given reference and hence new re-ordering decisions can be observed and re-estimated. Re-estimation of re-ordering models, especially the Bilingual Language model (BiLm), provides additional improvements in translation performance as compared to only the re-estimation of phrase translation models. Therefore, our experiments indicate that along with the translation models, reinforcing those re-ordering decisions that lead to a high scoring translation of the source sentence is beneficial to improve the translation performance of the SMT system.

In this chapter, we addressed the question of the reliability of the training mechanism for phrase-based machine translation with respect to the efficient utilization of available training data. In the next chapter, we shift our discussion to a related problem: the effect of domain variation in the training data on machine translation.

4

Efficient Domain Adaptation for Neural Machine Translation

4.1 Introduction

In recent years, the focus of research in machine translation has rapidly shifted from statistical machine translation to neural machine translation (NMT). For many language pairs, NMT has shown better performance than phrase-based machine translation (Toral and Sánchez-Cartagena, 2017). There are multiple reasons behind the growing interest in NMT. First, the ability of recurrent neural networks to capture long-distance dependencies enables the model to generate translations with improved grammaticality without depending on explicit re-ordering models (Mikolov et al., 2010). Second, the end-to-end architecture of NMT provides an easy alternative to phrase-based MT, which requires training of different models in isolation and optimization of the weights for each model. However, neural methods, including NMT, are known to be data-hungry (Koehn and Knowles, 2017).

Standard approaches in NMT train on all available parallel data for a language pair without taking into consideration the domain of the intended translation task (Chu and Wang, 2018). This training data pool is compiled from parallel texts from various domains and genres. For example, for German→English translation, the training data provided for the news translation task at the annual conference on machine translation (WMT) (Bojar et al., 2019) is one of the most commonly used publicly available corpora for research in machine translation. The majority of sentence pairs in this corpus come either from multilingual news sources or parliamentary proceedings. However, in many industrial settings, the aim is to translate sentences from specific domains. For example, in the medical domain, the requirement is to translate medical journals, manuals, and prescriptions, etc. Undoubtedly, the terminology and the language style can differ substantially between the news domain and the medical domain (van der Wees et al., 2015; Haddow and Koehn, 2012; Cuong and Sima'an, 2017). A model trained on the data from the snews domain may not have learned the terminologies and nuances of specific domains such as medical journals. As a result, a model trained on data compiled from various sources may show substantially better performance on test data from a domain that is the most dominating in the training data pool. However, for other domains, such a general domain model will show poor performance (Koehn and

Knowles, 2017). Therefore, training a model for a specific task or domain requires a substantial amount of training data belonging to that specific domain. However, high-quality training data for every intended domain may not be available in large amounts. As a result, NMT systems trained for a domain with less available parallel data quickly overfit, resulting in poor performance (Zoph et al., 2016).

Research in domain adaptation addresses the problem of adopting a model trained on data from one specific domain to test instances from another domain. The majority of the strategies proposed for domain adaptation of phrase-based MT as well as NMT fall in one of two categories (van der Wees, 2017): (a) domain adaptation at the data level and (b) domain adaptation at the model level. Both strategies have shown significant improvements for domain adaptation for phrase-based machine translation. The details of these strategies are discussed in Section 4.2. With the increasing popularity of NMT, research in domain adaptation for NMT has also received significant attention. Since data-level adaptation methods mainly focus on the selection of training samples relevant to the intended domains, they do not require any modification either to the training strategy or model architecture, and hence they can be directly applied to NMT. However, due to the fundamental difference between the two MT paradigms, model-level adaptation for NMT would require rather different techniques than those used for phrase-based MT.

A fast and efficient method for domain adaptation for neural methods is *fine-tuning*, which has been applied to NMT (Freitag and Al-Onaizan, 2016; Chu et al., 2017). In *fine-tuning*, a neural network that is already trained on a large general domain corpus (also called out-of-domain data) is further trained on a small corpus available for the target domain (also called in-domain data). Fine-tuning yields significant improvements as compared to training on in-domain data only (Chu et al., 2017). The idea of *fine-tuning* is to first learn the low-level features or parameters common to both domains and then to transfer these parameters to the intended in-domain task. This allows the model to converge on the in-domain data in relatively few steps and hence reduces the amount of training data required for the in-domain task (Zoph et al., 2016). However, training the neural network involves optimizing the output distribution from the final layer of the network with respect to the given classes or target labels, which can be substantially different for different tasks. This results in significant drift in the parameter space corresponding to the events observed in the new data (Lee et al., 2017b; Kirkpatrick et al., 2017; Kemker et al., 2018). As a result of this transformation to the new parameter space, the performance of the resulting model decreases drastically for the test instances from the general (source) domain. This phenomenon is known as catastrophic forgetting in the machine learning literature (Li and Hoiem, 2018; Kemker et al., 2018). In real-world applications, such a degradation of translation performance on either of the tasks is undesirable because even after adaptation, one would like to use a single model for translating sentences from multiple domains. In this chapter, we address this problem of drastic degradation of the translation performance of an NMT model when fine-tuning it to a new (target) domain. This problem is stated in the second research question RQ2:

RQ2: *How can we apply fine-tuning to adapt NMT models to new domains while retaining the performance on the source (original) domain?*

In order to address the main research question, we first empirically demonstrate the effect of domain variation by evaluating the performance of model trained on general domain data over test sets from specific target domains. This is the main concern of the first sub-research question RQ2.1:

RQ2.1 *How do NMT models trained on a general domain perform on data from a specific domain?*

Next, we discuss *fine-tuning* as a straightforward method for domain adaptation. We evaluate the performance of *fine-tuning-based* domain adaptation on the source as well as target domains to determine its degrading effect on source domain test instances. This is the concern of our second sub-research question RQ2.2

RQ2.2 *How is the performance on the original source domain affected by fine-tuning on target domains?*

To address the problem of drastic degradation by fine-tuning, in this chapter, we propose two simple modifications to the fine-tuning approach. Both approaches are based on the knowledge-distillation framework of Hinton et al. (2014), where a smaller student network learns to mimic a large teacher network by minimizing the loss between the output distributions of the two networks. We are motivated by the idea that new tasks can be learned without degrading performance on old tasks, by preserving the parameters shared between the two tasks (Li and Hoiem, 2018). We conduct experiments for English→German translation for three different domains to answer the third sub-research question RQ2.3:

RQ2.3 *Can we apply knowledge distillation as a remedy to minimize the degrading effect of fine-tuning?*

Our first modification is a simple multi-objective learning approach which involves *fine-tuning* a general domain model on small in-domain data while at the same time minimizing the loss between the output distributions of the student network (the model learned by fine-tuning) and the baseline teacher network (a model trained on sizeable general domain data). In our second modification, we propose adding multiple output layers to the student network corresponding to the different tasks (domains) and learning task-specific parameters for both domains using only the in-domain data while simply fine-tuning the parameters shared between the two tasks. Our experiments demonstrate that both of the proposed approaches, namely multi-objective *fine-tuning* and multi-output *fine-tuning*, achieve translation performance comparable to vanilla fine-tuning on the target domain task and at the same time suffer little degradation on the original general domain task.

In Section 4.2, we discuss related work on domain adaptation for traditional statistical MT as well as recent approaches in neural MT. We briefly discuss the *fine-tuning* method for domain adaptation and its limitations in Section 4.3. We introduce our approaches in Section 4.4 and discuss experiments and results in Section 4.5 and 4.6, respectively.

4.2 Related work

4.2.1 Domain adaptation for phrase-based MT

As discussed in Section 4.1, the problem of domain adaptation has been explored extensively for phrase-based machine translation (Cuong and Sima'an, 2017; Etchegoyhen et al., 2018). For phrase-based MT, domain adaptation approaches fall mainly into two categories: (a) domain adaptation at the data level and (b) domain adaptation at the model level (van der Wees, 2017).

Domain adaptation at the data level is mainly based on the selection of data samples from the out-of-domain training data that are similar to the target domain. Computation of the similarity can be based on simple criteria such as cross-domain entropy (Moore and Lewis, 2010; Axelrod et al., 2011). Other approaches include instance weighting (Matsoukas et al., 2009; Foster et al., 2010) or feature weighting (Hoang and Sima'an, 2014). While the data selection approach aims at filtering and discarding sentences dissimilar to the intended domain, another useful approach is to obtain or generate additional training data for the specific domain. For example, Schwenk (2008) generates additional synthetic or pseudo-parallel data by translating a monolingual corpus by an SMT system trained on the available in-domain data. For phrase-based MT, these data selection approaches have yielded significant improvements in performance (Cuong and Sima'an, 2017; van der Wees, 2017). Recently these data selection methods have also been evaluated for NMT and have demonstrated results similar to phrase-based MT (Silva et al., 2018; van der Wees et al., 2017). Note that the data level adaptation approaches are independent of the training procedure and can be applied to both the paradigms without any change in the training procedure or model architecture.

Domain adaptation at the model level aims at changing the distribution of the model trained on the general domain so as to represent the distribution of the intended domain. These approaches involve more complex solutions such as model interpolation (Koehn and Schroeder, 2007; Foster and Kuhn, 2007), domain-specific classifiers (Banerjee et al., 2010) and fill-up (Bisazza et al., 2011). Since in phrase-based SMT, each model is trained in isolation, adapting the SMT system to a new domain requires adapting all involved models to the new domain. This is unlike standard NMT, where the model is trained in an end-to-end fashion. For phrase-based MT, model interpolation is the most effective (Sennrich, 2012). However, it is a complex approach because it requires training of multiple models, including phrase-tables, re-ordering, and language models corresponding to each of the domains and then learning the corresponding interpolation weights.

In the case of phrase-based MT, there has not been much research in addressing the problem of degradation in the source domain. To the best of our knowledge, Wang et al. (2012) is the only relevant work that has discussed this problem and has goals similar to that of our approach, i.e., to adapt an SMT system for the newer domain while at the same time preserving the performance on the original (source) domain. Their idea is to learn a different set of model weights corresponding to each intended domain and then classify the incoming sentence at test time and accordingly select the corresponding model weights to decode the test sentence. However, their results show that for phrase-based SMT, the degradation effect is not drastic. On the other hand, as

we will discuss in Section 4.6, for NMT, the drop in performance on the original domain after domain adaptation can be catastrophic.

4.2.2 Domain adaptation for NMT

The majority of recent research on domain adaptation in NMT has focused on fine-tuning a pre-trained model on the additional in-domain data. This is because the end-to-end architecture of NMT enables fast and efficient fine-tuning without the complexity of re-training individual components for each domain. The first attempt at applying fine-tuning to NMT was by Luong and Manning (2015). They adapted an NMT model trained on large general domain data by training it further on a small in-domain data set. Freitag and Al-Onaizan (2016) extended this fine-tuning approach by using an ensemble of the baseline general domain model and the fine-tuned model. They proposed that the ensemble provides better translations on the new domain and, at the same time, retains much of the performance on the general domain. Their experiments on TED-talks (Cettolo et al., 2012) as in-domain data demonstrated improvements on the in-domain test set. However, as we will discuss in Section 4.6, our experiments demonstrate that even with an ensemble, the performance of the fine-tuned model still degrades significantly on the general domain task, especially when adapted to a target domain such as medical documents for which vocabulary and style are substantially different from that of the source domain. On the other hand, our approach, which is based on "baseline supervision" during fine-tuning, not only retains general domain performance better than an ensemble for two different target domains but also provides a faster and more efficient decoding procedure by avoiding the requirement of having to compute two output distributions separately which is required in the ensemble approach.

Another approach for NMT domain adaptation is "mixed fine-tuning" by Chu et al. (2017). They propose to fine-tune the baseline model on a parallel corpus that is a mix of in-domain and general domain corpora instead of fine-tuning only on in-domain data. However, as they point out, mixed fine-tuning takes longer than vanilla fine-tuning, depending on the size of the "mixed" data. Also, it requires robust data selection heuristics to extract relevant sentences from the general domain. On the other hand, in our approach, we completely remove any dependence on the general domain data while fine-tuning. To generate additional data for specific domains, Sennrich et al. (2016a) apply back-translation of the target in-domain sentences, thus adding new data to the training corpus and then fine-tuning on the extended bitext. A similar approach is (Poncelas and Way, 2019), where the idea is to select only the most relevant sentence pairs from the back-translated corpus using data selection algorithms.

A related line of research to our approach is in computer vision where supervision from the baseline model based on the knowledge distillation framework has been extensively used in various domain adaptation paradigms such as "deep domain transfer" (Tzeng et al., 2015), "knowledge adaptation" (Ruder et al., 2017) and "learning without forgetting" (Li and Hoiem, 2018). The "learning without forgetting" approach of Li and Hoiem (2018) is very similar to our approach since they propose to reduce the general domain performance degradation by adding multiple nodes in the output layer of a convolutional neural network and fine-tune the parameters on the in-domain data using the supervision from the baseline model. However, we differ from their approach in that

we not only apply this method to neural machine translation, but we also experiment with training with multiple objectives as well as numerous output layers instead of simply adding new nodes corresponding to the new classes.

4.3 Background

4.3.1 Neural machine translation

In this chapter, we conduct domain-adaptation experiments with the recurrent NMT architecture discussed in Section 2.3. In order to explain the modifications proposed in this chapter, we briefly revisit the NMT architecture and training procedure. It is important to remember here that given a source sequence and previous history of the target sentence, the softmax layer of the end-to-end neural network generates a probability distribution for the next target position. This distribution represents the probabilities of each word in the target vocabulary for that target position. The log-likelihood of this distribution is calculated as defined in Equation 2.18:

$$\mathrm{L}_{\mathrm{NLL}}(\theta) = -\sum_{j=1}^{n}\sum_{k=1}^{|V|}(y_{jk}) \cdot \log\Big(p(y_{jk}|x;\theta)\Big), \tag{4.1}$$

In Equation 4.1, y_j corresponds to the output label generated by the network at each timestep, and k is the true class label, i.e., the reference target word at each timestep, which is selected from a fixed vocabulary V. With the outer summation, the total loss is computed as the sum over the entire target sequence. This loss is then back-propagated to each layer of the network. The corresponding parameters are then updated using the respective gradients. Training is performed iteratively over batches of training samples until model convergence is achieved.

4.3.2 Fine-tuning for domain adaptation

Fine-tuning (Hinton and Salakhutdinov, 2006; Yosinski et al., 2014) is a straightforward strategy to quickly adapt a model to a new task or domain. It belongs to a more general class of approaches known as *transfer learning* where the idea is to transfer the set of features or parameters learned for a specific domain (known as *out-of-domain-data*) to another target domain for which substantially large training data are not available (Pan and Yang, 2010). The motivation behind this strategy is that for two tasks with a similar set of output classes but different inputs, the low-level features have similar distributions while the higher-level features can be domain- or task-specific. Training on a small dataset does not provide sufficient supervision to learn optimal values for a large number of parameters. As an easy solution, using a network pre-trained for similar tasks results in better initialization of low-level parameters, and further training on the intended task by fine-tuning enables learning of task-specific features (Zhou et al., 2017; Li and Hoiem, 2018). Fine-tuning has been extensively used in many neural network applications such as image recognition, where it is common to use a pre-trained network trained on a large dataset sand fine-tune it to other domains for which sufficient training data is not available (Oquab et al., 2014; Radenović et al., 2019).

Although fine-tuning provides an easy way to quickly adapt to new tasks, an important shortcoming of this approach has been discussed in the deep learning literature. This problem is the catastrophic forgetting of the original domain (also known as *source domain*) (Li and Hoiem, 2018). Forgetting can manifest itself as degradation of the performance of a model on the source domain after fine-tuning on the target domain. Although this is not a problem when the only requirement for experimental settings is to adapt the model to the intended *in-domain* task, such a degradation is not desirable for an industrial setting where a model is expected to perform equally well for the *source* and *target* domain. The reason behind this catastrophic forgetting is discussed in (Chen and Liu, 2016) and (Goldberg and Hirst, 2017). When a pre-trained neural network is adapted to the new task, it tends to forget the representations learned for the original task. This is due to the fact that the weight updates for the new task superimpose or override the parameters or weights learned for the original task. This problem has been referred to as the *stability-plasticity dilemma* in neuroscience by (Abraham and Robins, 2005). They explain that a rigid model will not be able to learn from the new inputs for future tasks. This is also known as high stability. On the other hand, a model with high plasticity will easily update the weights for a new task but will forget the information for the original task. For NLP tasks, Goldberg and Hirst (2017) argue that when fine-tuned on data with a different distribution of words or tokens, the representations for the original words will change to the new parameter space.

4.4 Domain adaptation with baseline supervision

As described in Section 4.1, during fine-tuning, a model pre-trained on general domain data is further trained on the in-domain data, which largely shifts the parameter space of the model to the target domain resulting in performance degradation on the general domain test instances. We propose that fine-tuning on the target domain with an objective function similar to one defined in Equation 2.21 using the supervision from both the general domain and the target domain can avoid the performance degradation on the general domain while achieving performance comparable to vanilla fine-tuning on the target domain. We explore this idea by proposing two modifications to the fine-tuning approach.

4.4.1 Multi-objective fine-tuning (MCL)

In the first modification, we train the network on a joint objective function that includes the supervision provided by the hard target labels in the in-domain data as well as the output distribution of the general domain model on the in-domain data. We believe that with such a joint objective, the network can learn a parameter space common to both domains.

First, we pre-train the initial model, also known as 'teacher network,' on the general domain data. Then we fine-tune this baseline model on the in-domain data by minimizing the log-likelihood loss between the target references and the output distribution of the network. However, at the same time, for each sentence, we also minimize the KL-divergence (or cross-entropy) loss between the output distribution of teacher network

and the distribution produced by the student network on the in-domain training data, as shown in Figure 4.1.

Let the general domain data (on which the initial model is trained) be represented by x_{out} and the in-domain data be represented by x_{in}, then the final learning objective becomes:

$$\begin{aligned} &L(\theta, \theta_T) = \\ &(1-\lambda) \sum_{k=1}^{|V|} (y_j = k) \times \log\Big(p(y_j = k|x_{in};\theta)\Big) \\ &+ \lambda\Bigg(- \sum_{k=1}^{|V|} \mathrm{KL}\Big(q(y|x;\theta_T)\ p(y|x_{in};\theta)\Big)\Bigg). \end{aligned} \tag{4.2}$$

In Equation 4.2, KL is the KL-divergence function, q is the output distribution of 'teacher' model on the in-domain data and θ_T is the parameter space of the model learned over the general domain data. Note that both distributions here are obtained by increasing the temperature ϵ of the softmax as defined in Equation 4.3.

$$p_\epsilon(y|x;\theta) = \mathrm{softmax}\left(\frac{W_s\ \tilde{s}_t}{\epsilon}\right). \tag{4.3}$$

In the proposed approach, once the teacher model is trained, data from the source (general) domain is no more required while fine-tuning.

4.4.2 Multiple-output layer fine-tuning (MLL)

Although learning a joint objective as discussed in Section 4.4.1 can reduce the degradation on the general domain task to some extent, a much better solution to retain the performance on the old task is to preserve task-specific parameters corresponding to the old task and at the same time slightly transform parameters shared across the two tasks (Li and Hoiem, 2018). Therefore, as a second modification, we propose modifying the baseline model by adding parameters specific to the new task and learning the task-specific parameters with the respective learning objectives.

In this approach, we consider only the parameters of the final output-layer (W_s as defined in Equation 2.16) of the NMT network as task-specific, while all parameters corresponding to the encoder, decoder, attention mechanism, and the concatenation layer are considered to be shared. Let θ_s be the set of all the shared parameters and θ_o and θ_n be the task-specific output-layer parameters corresponding to the old (general domain) and new (in-domain) task, respectively.

Training the general domain baseline model results in initially learning the shared parameters θ_s and the output layer for the out-of-domain task θ_o. For training the in-domain student model, we first modify the network by adding another output layer to the standard NMT network, as shown in Figure 4.2. We first compute the output distribution of the general domain teacher model on the in-domain data. Then the shared parameters for the student network corresponding to the encoder-decoder and attention mechanism are initialized with θ_s, which are learned from the baseline model. Similarly, the parameters of the output layer corresponding to the old-domain task

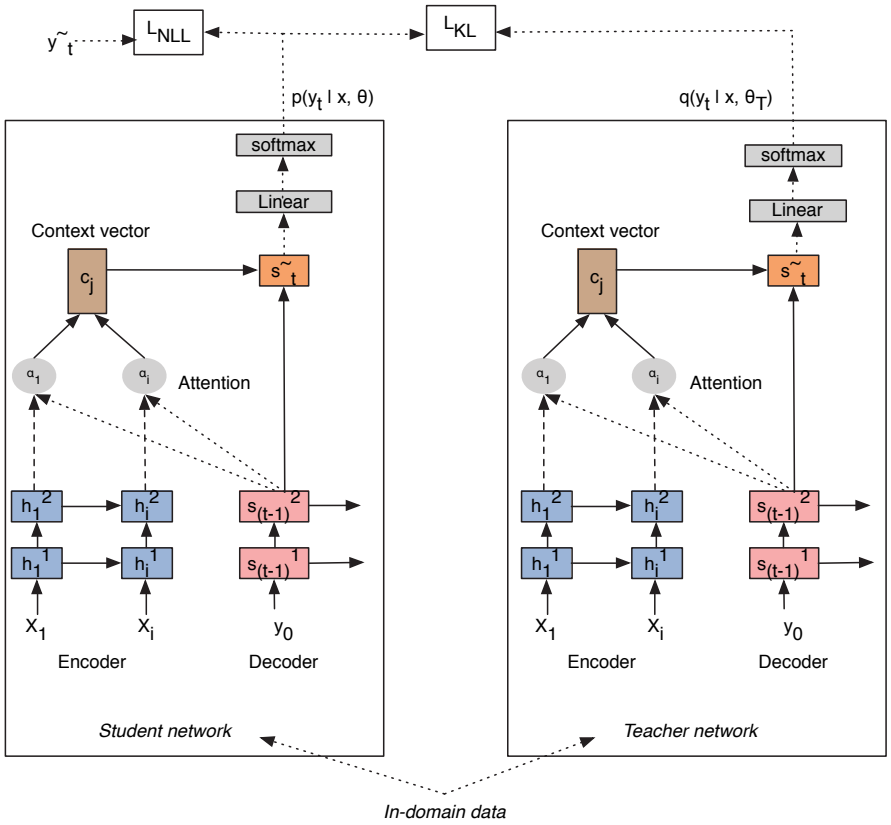


Figure 4.1: **Multi-objective fine-tuning**. Both teacher and student network have the same architecture. The student network is initialized with parameters trained for the teacher network. During fine-tuning the parameters of the teacher network are frozen.

are also initialized with parameters θ_o. Parameters specific to the in-domain task θ_n are initialized randomly. Then we first train the new parameters θ_n using the ground-truth with the standard log-likelihood objective as defined in Equation 4.1. This is a simple warm-up procedure that improves fine-tuning performance (Li and Hoiem, 2018). During this warm-up, we freeze θ_s and θ_n. Finally, all parameters are fine-tuned by minimizing the joint objective. Consider x_n, y_n to be a sentence pair from the in-domain data. Then, let y_o be the computed output of the teacher model for the new data, i.e.,

$$y_o = q(x_n, \theta_s, \theta_o). \tag{4.4}$$

For the student network, let y'_o and y'_n be the output distributions from the old and new output-layer, respectively corresponding to θ'_o and θ'_n:

$$y'_o = p(x_n, \theta'_s, \theta'_o) \tag{4.5}$$

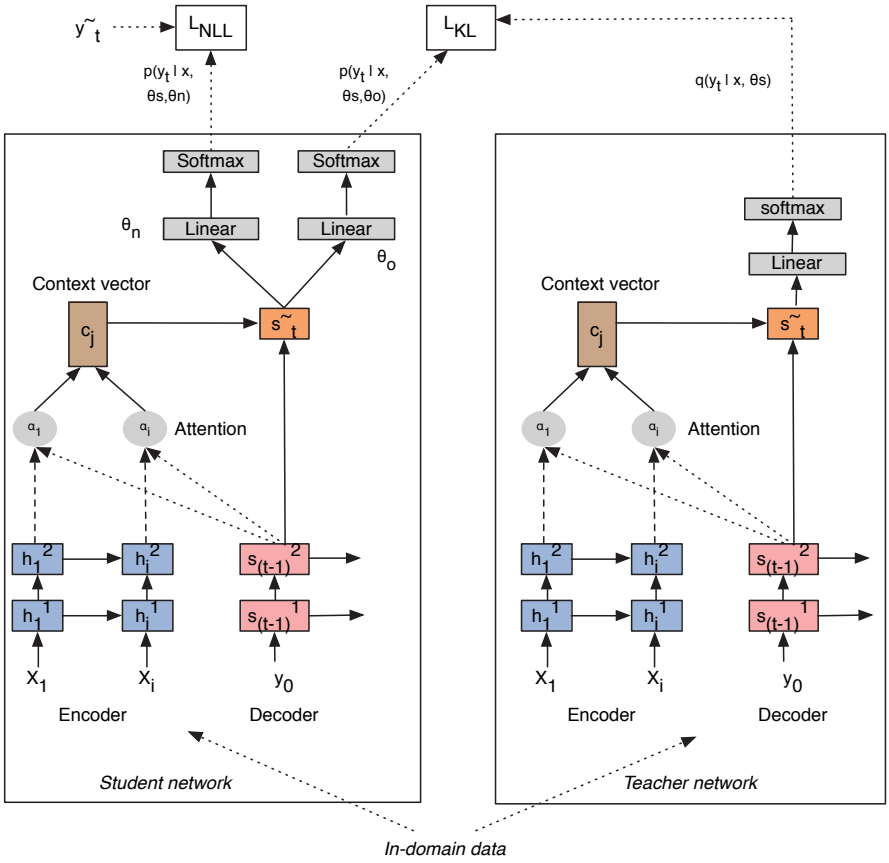


Figure 4.2: **Multi-output-layer learning**. The student network has two output layers. The additional layer is trained with respect to the distribution from the teacher network.

$$y'_n = p(x_n, \theta'_s, \theta'_n). \tag{4.6}$$

Then similar to Equation 4.2, we define two objectives: The first objective is the standard log-likelihood loss for the shared parameters θ_s and the parameters for new task θ_n with respect to the target references in the in-domain data

$$\mathrm{L_n} = -y_n \times \log(p(x_n|\theta'_s, \theta'_n)). \tag{4.7}$$

The second objective is the KL-divergence between the output distribution of the student network with respect to the output distribution of the teacher network on the same in-domain data:

$$\mathrm{L_o} = -\sum_{k=1}^{|V|} \mathrm{KL}\Big(y_o, p(y'_o|x_n; \theta'_s, \theta'_o)\Big). \tag{4.8}$$

The student network is finally trained on the joint objective defined as:

$$\mathrm{L}_{\mathrm{combined}} = (1-\lambda)\mathrm{L}_{\mathrm{n}} + \lambda \mathrm{L}_{\mathrm{o}}. \tag{4.9}$$

During decoding, the output layer corresponding to the domain label of the test sentences is used to compute the output distribution.

4.5 Experimental settings

4.5.1 NMT parameters

In all our experiments, we use an NMT system based on (Luong et al., 2015) implemented using the Torch7 deep learning framework. It is a two-layer unidirectional LSTM encoder-decoder with a global attention (dot product) mechanism. Both the encoder and decoder have input embeddings and hidden layers of size 1000. We limit the vocabulary sizes to 60k for the source and 40k for the target side. Parameters are optimized using stochastic gradient descent. We set the initial learning rate as 1 with a decay rate of 0.5 for each epoch after the 5th epoch. Model weights are initialized uniformly within $[-0.02, 0.02]$. A dropout value of 0.2 is applied to each layer. The mini-batch size for out-of-domain training is fixed to 64 sentences, while for each of the in-domain fine-tunings, we use a batch size of 32. We train for a maximum of 20 epochs and decode with a standard beam search with a beam size of 10. All models have been trained on NVIDIA Titan-X (Pascal) GPU devices.

For in-domain training, we use the same vocabulary extracted from the baseline general domain bitext. Note that our multiple-output layer approach also allows for the use of a different vocabulary (that can be extracted from the new data) for in-domain fine-tuning.

4.5.2 Data

We conducted experiments for English→German translation. For all settings we use the WMT-2015 (Bojar et al., 2015) training set as the general domain training data. This training set contains approximately 4.2 million parallel sentences from multiple sources including Europarl, news commentary and common crawl articles. We constrain the maximum sequence length to be 80 and remove duplicates. This results in a bitext of approximately 4 million sentence pairs, out of which we reserve 5000 sentence pairs for perplexity validation and use the rest for training the general domain baseline model. We use newstest15 (the official test set for WMT-2015) as a general domain test set.

For in-domain training, we consider three different domains namely the TED-talks bitext (Cettolo et al., 2012) (approximately 170k sentences, also known as the IWSLT-2013 corpus), the EMEA corpus (Tiedemann, 2009) (approximately 200k sentence) which is a parallel text of medical guidelines, and the information technology domain data provided from the IT-translation task for WMT-2016 (Bojar et al., 2016). From each of these three corpora, we reserve 2000 sentences as the validation set. For the TED talk domain, we use a combination of the official test sets for IWSLT 2011, 2012, and 2013 as an evaluation set. For the medical and IT-domain, we reserve a set of

Table 4.1: Training and evaluation sets for various domains. Sizes in number of parallel sentences.

Corpus	Train	Validation	Test
WMT	4M	2K	2.1K
Ted	170K	2K	4.7K
Medical	200K	2K	2K
IT	210K	5K	2K

2000 sentences each from the respective training corpus evaluation sets. Table 4.1 summarizes the statistics of training and test data sets for all domains. Results are reported in terms of case-insensitive BLEU-4 (Papineni et al., 2002). Approximate randomization (Noreen, 1989; Riezler and Maxwell, 2005) is used to detect statistically significant differences.

4.6 Results

We define multiple baselines to compare our proposed approaches. First, we obtain the BLEU scores for a model trained only on general domain data and measure its performance on the general domain (source domain) test set as well as on the in-domain test sets. Similarly, for baseline comparisons, we train models only on the small in-domain data for each of the target domains. We compare our approaches to vanilla fine-tuning and ensemble methods (Freitag and Al-Onaizan, 2016) in terms of BLEU improvements on the in-domain test sets as well as in terms of degradation on the general domain test-set. Finally, we also train baseline models on the combined general and in-domain bitexts and test them on both test sets. Note that this baseline acts as a ceiling for our approaches as this setting can only be used for training when data from the general, as well as the target domain, are available. Table 4.2 summarizes the notations used for different models and datasets used in this section.

Table 4.2: Notation for models and datasets.

Notation	Description
In-domain$_{only}$	Model trained only on in-domain data
General$_{only}$	Model trained only on General data
Combined	Model trained on combined in-domain and general domain data
FT	Vanilla fine-tuning
FTEns	Ensemble fine tuning
MCL	Multi-objective fine-tuning
MLL	Multi-output-layer fine-tuning

Since the main goal of our experiments is to adapt the model to the target domain, we first evaluate each model on the in-domain development set and finally evaluate the same

model on the evaluation set for both the source (in-domain) and general domain. Finally, we also aim to measure how the performance of the model varies during in-domain training. Therefore we report epoch-wise BLEU score for each adaptation technique on the evaluation set (test-set) corresponding to each domain.

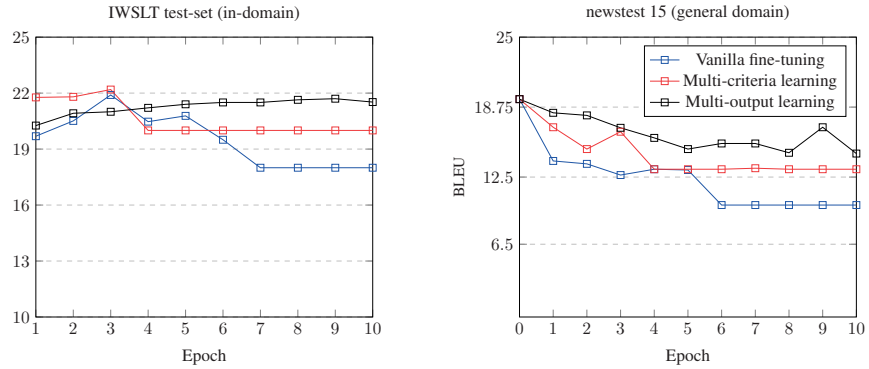
4.6.1 Domain adaptation for TED data

For domain adaptation for TED data, Table 4.3 summarizes the BLEU scores for different settings and also the highest and lowest BLEU scores for fine-tuning and the proposed approaches. The performance of the *General*$_{only}$ model (19.46) is higher than that of the *In-domain*$_{only}$ model (17.95) for the in-domain (*iwslt*) test set. Hence, we consider the BLEU scores of the *General*$_{only}$ model on the two test sets as our corresponding baseline scores. To compare the gradual degradation of vanilla fine-tuning with our proposed approaches, we report BLEU scores corresponding in the first epoch as well as highest BLEU score achieved for the in-domain test set. The last column shows the corresponding BLEU scores over the general domain test set.

Table 4.3: BLEU scores for different approaches for TED data domain adaptation. iwslt = IWSLT (2011+2012+2013). * represents the baseline setting for these experiments and $^{\blacktriangle}/_{\blacktriangledown}$ and $^{\triangle}/_{\triangledown}$ indicates a statistically significant gain/drop at $p < 0.01$ and $p < 0.05$ respectively over the baseline.

	Epoch	In-domain (TED)		General domain (WMT)
		development	iwslt(test)	newstest'15 (test)
Standard NMT baselines				
In-domain$_{only}$		15.51	17.95	
General$_{only}$		17.40	19.46*	18.54*
Combined		20.80	22.86$^{\blacktriangle}$ (+3.40)	17.57$^{\blacktriangledown}$ (-0.97)
Fine-tuning methods				
FT	1	17.60	19.69$^{\triangle}$(+0.23)	13.94$^{\blacktriangledown}$(-4.60)
	3	18.43	21.90$^{\blacktriangle}$(+2.44)	12.68$^{\blacktriangledown}$(-5.86)
FTEns		19.60	22.90$^{\blacktriangle}$(+3.44)	16.70$^{\blacktriangledown}$(-1.84)
Proposed approaches				
MCL	1	19.31	21.77$^{\blacktriangle}$(+2.21)	16.95$^{\blacktriangledown}$(-1.59)
	7	20.55	**22.19**$^{\blacktriangle}$(+2.73)	16.55$^{\blacktriangledown}$(-1.99)
MLL	1	17.29	20.26$^{\triangle}$(+0.80)	**18.24**$^{\triangledown}$(-0.30)
	9	18.40	21.70$^{\blacktriangle}$(+2.24)	16.90$^{\blacktriangledown}$(-1.64)

Vanilla fine-tuning improves over the baseline by up to $+2.4$ BLEU in the third epoch for the in-domain (iwslt) test set. However, it drops substantially in the first epoch by $+4.6$ BLEU on the general domain (newstest'15) test set and by $+5.8$ corresponding to the third epoch. On the other hand, for the in-domain test set, multi-objective learning (MCL) improves over the baseline by up to $+2.7$ BLEU (in the third epoch) which is slightly better than vanilla fine-tuning while at the same time, performance



(a) In-domain evaluation

(b) General domain evaluation

Figure 4.3: Epoch-wise BLEU (a) improvement over in-domain test set (TED) (b) degradation over general domain (WMT) test set.

degradation on the general domain test set is only -2.0 which is substantially less than for fine-tuning.

Similarly, for multi-output-layer learning (MLL), we observe improvements of up to $+2.2$ BLEU over the baseline on the in-domain test set, which is almost equal to fine-tuning and the drop in BLEU over the general domain test set for the 9th epoch is only -1.64.

The ensemble method (FTEns) achieves an improvement of $+1$ BLEU over simple fine-tuning for the in-domain test set (iwslt), which is higher than both proposed methods and also suffers similar degradation on the general domain (newstest'15) test set. This is due to the fact that although the TED talks are considered to be a different domain, the vocabulary and style of the TED data is very similar to that of the general domain data. Also, the model trained on the combined data performs comparable to the best setting, i.e., the ensemble method for both test sets.

Figure 4.3 shows the epoch-wise comparison of BLEU scores for source and target domain test sets for TED data domain adaptation. For the source domain, fine-tuning performance drops rapidly, starting from the first epoch. However, performance degrades relatively slower for our two proposed approaches on the general domain.

In conclusion, we observe that for the TED data, our approaches show comparable improvement to fine-tuning while suffering from less degradation over the source domain. However, the ensemble method of Freitag and Al-Onaizan (2016) also demonstrates similar results. Therefore, we repeat our experiments for domain adaptation for the EMEA corpus for which the vocabulary is substantially different from the source domain of news articles as compared to the TED data.

4.6.2 Domain adaptation for medical domain data

Table 4.4 summarizes our experiments for domain adaptation for the EMEA data. The first important observation is that the BLEU score for the *In-domain*$_{only}$ model on the

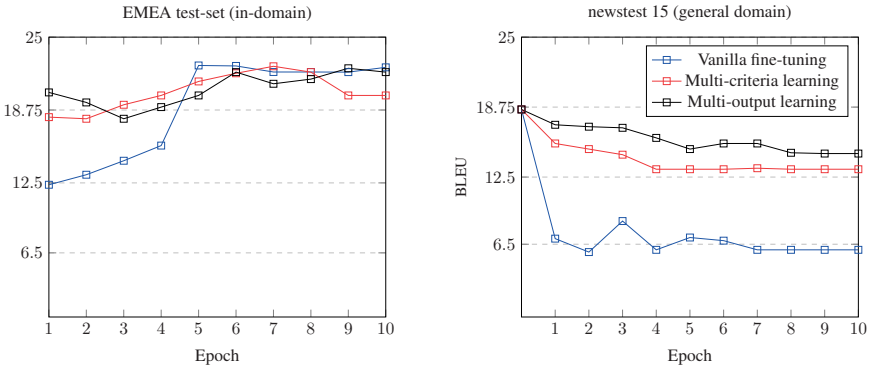
in-domain test set is 20.08 while for the *General*$_{only}$ model it is 13.54. That is, the performance of the *In-domain*$_{only}$ model is substantially higher than the *General*$_{only}$ model on the in-domain test set. This is in contrast to the experiments on TED data, where the *General*$_{only}$ model performed better than the *In-domain*$_{only}$ model. Hence, for this set of experiments, we consider the *In-domain*$_{only}$ model performance (20.08) as our baseline. Also, here the model trained on the combined data performs only slightly better than the *In-domain*$_{only}$ baseline for the EMEA test set while performing comparable to the *General*$_{only}$ model on the WMT test set.

Table 4.4: BLEU score for different approaches for EMEA data domain adaptation. * represents the baseline setting for these experiments. emea$_{test}$ = test set for medical domain, emea$_{dev}$ = development set for medical domain. * represents the baseline setting for these experiments and $^{\blacktriangle}/_{\blacktriangledown}$ and $^{\triangle}/_{\triangledown}$ indicates a statistically significant gain/drop at $p < 0.01$ and $p < 0.05$ respectively over the baseline.

	Epoch	In-domain (EMEA)		General domain (WMT)
		emea$_{dev}$	emea$_{test}$	newstest15 (test)
Standard NMT baselines				
In-domain$_{only}$		19.40	20.08 *	
General$_{only}$		13.13	13.54	18.54*
Combined		20.51	20.84$^{\triangle}$(+0.76)	17.60$^{\blacktriangledown}$(-0.94)
Fine-tuning methods				
FT	1	11.71	12.34	7.10$^{\blacktriangledown}$(-11.44)
	6	22.23	22.57$^{\blacktriangle}$(+2.49)	4.50$^{\blacktriangledown}$(-14.04)
FTEns		18.90	19.43$^{\blacktriangledown}$(-0.65)	13.63$^{\blacktriangledown}$(-4.91)
Proposed approaches				
MCL	1	17.75	18.14$^{\blacktriangledown}$(-1.94)	15.50$^{\blacktriangledown}$(-3.04)
	11	22.11	**22.50**$^{\blacktriangle}$(+2.42)	**13.29**$^{\blacktriangledown}$(-5.25)
MLL	1	20.20	20.60 (-0.02)	17.17$^{\blacktriangledown}$(-1.37)
	6	21.87	**22.33**$^{\blacktriangle}$(+2.25)	**14.67**$^{\blacktriangledown}$(-3.87)

Vanilla fine-tuning shows an improvement of up to $+2.5$ BLEU (in the 6th epoch) as compared to the *In-domain*$_{only}$ model on the in-domain (EMEA) test-set. However, for fine-tuning, the drop for the general domain test set is dramatic, i.e., -11.3 in the first epoch and -14 BLEU in the 6th epoch. Although the ensemble method suffers from less degradation (-4.9) for the general domain test set (newstest'15), it performs slightly lower (-0.6) than the baseline for the in-domain (EMEA) test as compared to vanilla fine-tuning. On the other hand, multi-objective learning (MCL) performs comparably to fine-tuning on the in-domain (EMEA) test set ($+2.4$ improvement over baseline) while the drop for the general domain test set (-5.25) is not as dramatic as for vanilla fine-tuning.

Similarly, for our multi-output-layer approach (MLL), while the improvement on the in-domain (EMEA) test-set is comparable to fine-tuning, it shows the smallest drop in performance for newstest'15. Figure 4.4 shows the comparison of BLEU scores for the in-domain (EMEA) test set and newstest'15. Similar to the TED data experimental



(a) In-domain evaluation

(b) General domain evaluation

Figure 4.4: Epoch-wise BLEU (a) improvement over in-domain test set (EMEA) (b) degradation over general domain test-set (WMT).

results, while fine-tuning performance drops rapidly on the general domain test set, it drops slower for our proposed methods,

In conclusion, for the medical domain, our proposed methods of multi-objective and multi-output-layer learning show improvements comparable to fine-tuning on the target domain with limited loss on the source (general) domain as compared to fine-tuning. On the other hand, the ensemble method fails to achieve any improvements for the target domain.

4.6.3 Domain adaptation for IT domain data

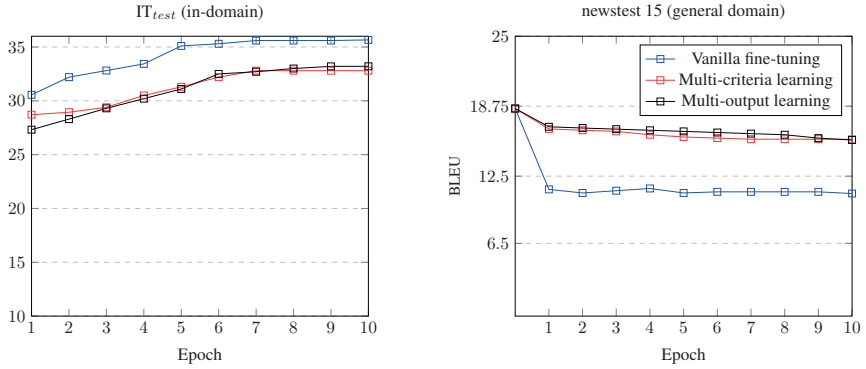
Table 4.5 summarizes the results of our experiments for domain adaptation for the IT domain data. Similar to the medical domain, for the IT-domain, the performance of a model trained only on in-domain data (*In-domain*$_{only}$) is higher ($+12.1$) than that of a model trained only on out-of-domain data (*General*$_{only}$) on the in-domain test set (IT$_{test}$). Therefore, for adaptation to the IT-domain, we consider the in-domain-only model *In-domain*$_{only}$ as our baseline. Surprisingly, a model trained on the combined data shows no improvement for test sets from either of the domains. Instead we observe a drop of -0.78 for the in-domain test set and a drop of -0.63 for the general domain test set as compared to the corresponding baselines.

Similar to the medical domain results, the best performance observed for the in-domain test set corresponds to vanilla fine-tuning, which shows an improvement of up to $+3.18$ BLEU (in the 10th epoch) over the in-domain test-set. However, again, the drop in performance over the general domain test set is very large, i.e., -7.22 in the first epoch and -7.59 in the 10th epoch.

Similar to the medical domain results, the ensemble method suffers less from degradation in performance for the general domain test-set (newstest'15), i.e., -3.78 BLEU. However, it performs slightly lower than the baseline for the in-domain (IT_{test}) test (-0.9) as compared to vanilla fine-tuning.

Table 4.5: BLEU score for different approaches for IT data domain adaptation. * represents the baseline setting for these experiments. IT_{dev} = development set for IT domain, IT_{test} = test set for IT domain. * represents the baseline setting for these experiments and $^{\blacktriangle}/^{\blacktriangledown}$ and $^{\triangle}/^{\triangledown}$ indicates a statistically significant gain/drop at $p < 0.01$ and $p < 0.05$ respectively over the baseline.

	Epoch	In-domain (IT)		General domain (WMT)
		IT_{dev}	IT_{test}	newstest'15 (test)
Standard NMT baselines				
In-domain$_{only}$		32.21	32.48*	
General$_{only}$		21.14	21.36	18.54*
Combined		31.50	31.70$^{\blacktriangledown}$(-0.78)	17.91$^{\blacktriangledown}$(-0.63)
Fine-tuning methods				
FT	1	30.21	30.56$^{\blacktriangledown}$(-1.92)	11.32$^{\blacktriangledown}$(-7.22)
	10	35.48	**35.66**$^{\blacktriangle}$(+3.18)	10.95$^{\blacktriangledown}$(-7.59)
FTEns		31.32	31.57$^{\blacktriangle}$(-0.91)	14.76 $^{\blacktriangle}$(-3.78)
Proposed approaches				
MCL	1	28.53	28.71$^{\blacktriangledown}$(-5.16)	16.72$^{\blacktriangledown}$(-1.82)
	10	32.73	**32.80**$^{\blacktriangle}$(+0.32)	**15.79**$^{\blacktriangledown}$(-2.75)
MLL	1	27.11	27.32$^{\blacktriangledown}$(-5.16)	16.91$^{\blacktriangledown}$(-1.63)
	8	32.90	33.21$^{\blacktriangle}$(+0.73)	15.73$^{\blacktriangledown}$(-2.81)



(a) General domain evaluation

(b) In-domain evaluation

Figure 4.5: Epoch-wise BLEU (a) improvement over in-domain test set (IT) (b) degradation over general domain test-set (WMT).

In contrast to the results for the medical domain, the performance of the proposed multi-objective learning (MCL) for the IT domain is not as high as fine-tuning on the in-domain test set (only $+0.32$ in the 10th epoch). However, it is still higher than the baseline, and the drop in general domain performance (-2.75) is much lower than for vanilla fine-tuning. Similarly, for a multi-output-layer approach (MLL), while

the improvement on the in-domain test-set is similar to multi-criteria-learning (MCL) ($+0.73$ BLEU), it shows the least drop in performance for the general domain test set (-2.81).

Figure 4.5 shows the comparison of BLEU scores over the IT domain test sets and the general domain test set for IT data domain adaptation. Similar to the TED and medical domain results, fine-tuning performance drops rapidly on the general domain test set, whereas multi-criteria learning shows smaller drops. For the IT domain, while the proposed methods perform lower than the fine-tuning method for the in-domain test set, they still show significant improvements over the baseline. Moreover, both methods result in a smaller drop in performance over the general domain test sets as compared to vanilla fine-tuning.

4.6.4 Decoding time comparison

Finally, we compare the average decoding time per sentence for ensemble decoding and multi-objective learning in Table 4.6 as calculated over the TED domain data. The decoding time for the ensemble method is twice that of fine-tuning because it requires computing two different models, while for multi-objective learning, it is the same as that of fine-tuning.

Table 4.6: Average decoding time (in milliseconds) on GPU devices per sentence for ensemble decoding and multi-objective learning. The average sentence length for the used test set is 20.9 tokens.

	Average time(ms)
Fine tuning	0.131
Ensemble decoding	0.277
Multi-objective learning	0.147

4.7 Conclusion

Although NMT has demonstrated better translation performance than statistical machine translation in general domain applications, the data-hungry nature of NMT models makes them prone to overfitting and can result in low performance for translation tasks focused on specific domains due to the scarcity of training data in these domains. Domain adaptation provides methods to utilize the general domain models for translating text in specific domains. For NMT and other neural network-based tasks, fine-tuning provides a straightforward method for fast domain adaptation. However, a particular shortcoming of fine-tuning is the large degradation of performance on general domain tasks. In this chapter, we addressed the question of how to retain the consistent performance of an NMT model across the source and target domains when adapting the general domain model to specific domains. We asked the general research question RQ2:

RQ2: *How can we apply fine-tuning to adapt NMT models to new domains while retaining the performance on the source (original) domain?*

To answer RQ2, we carried out three sets of experiments: First, we evaluated the performance of a model trained on the general domain (news data) on three different target domains: TED talks, medical domain, and IT-domain. As discussed in Section 4.6.2 and 4.6.3 for the medical domain and IT domain, the performance of the general domain model is significantly lower than that of the in-domain model when evaluated on the in-domain test set. Only for the TED-talks data, due to the substantially smaller size of the in-domain data, the performance of the general domain model is higher than the in-domain model.

Our first sub-research question in this chapter was:

RQ2.1 *How do NMT models trained on a general domain perform on data from a specific domain?*

To answer RQ2.1, we can conclude from the discussions in Section 4.6.2 and 4.6.3 that given a reasonable size of in-domain data, the performance of a model trained on general domain data will be substantially lower than training only on the in-domain data. Only in case of a very small amount of in-domain data, as observed for the TED-domain in Section 4.6.1, can a general domain model yield better performance than the in-domain model.

Domain adaptation aims at making the best use of general domain data for domain-specific tasks. As discussed in Section 4.3.2, although various data selection methods have been proposed for domain adaptation in SMT, due to the end-to-end neural network architecture of NMT, fine-tuning becomes an easy and straightforward solution. We can answer our second sub-research question about the evaluation of fine-tuning for NMT on the source (general) as well as target domains:

RQ2.2 *How is the performance on the original source domain affected by fine-tuning on target domains?*

The goal of addressing RQ2.2 is to evaluate whether fine-tuning provides any significant improvements over the in-domain baselines when adapting the models to multiple domains. At the same time, we also evaluate whether the model adapted through fine-tuning can retain its performance on the general domain task. It can be concluded from the results for all domains that fine-tuning indeed provides significant improvements as compared to the in-domain only or general domain only models, but the performance of the adapted model on the general domain tasks degrades drastically.

To address the problem of this drastic degradation, we proposed two modifications to the fine-tuning method for domain adaptation for NMT in order to retain the performance on the source domain. The third sub-research question addresses the evaluation of the proposed methods.

RQ2.3 *Can we apply knowledge distillation as a remedy to minimize the degrading effect of fine-tuning?*

From the results for all domains, we see that both proposed approaches achieve performance comparable to vanilla fine-tuning while retaining performance on the source (general) domain. Moreover, the decoding speeds of the proposed methods are the same as fine-tuning, while the ensemble method requires almost twice the decoding time of fine-tuning.

Similar to the negative effect of domain variation, the quality of the training data is another problem that negatively affects the performance of the NMT models. In the next chapter, we discuss the problem of utilizing noisy data for training NMT models and propose a solution also based on the idea of knowledge distillation.

5 Neural Machine Translation with Noisy Data

5.1 Introduction

In Chapter 4, we discussed the problem of domain adaptation for neural machine translation. We proposed a strategy to easily adapt NMT models to new domains without degradation on the source domain. A similar problem that has received limited attention is that of low-quality training data. In this chapter, our aim is to evaluate the effect of noisy training data on NMT performance and propose strategies to effectively utilize the noisy data for training NMT models. Therefore, our main research question in this chapter is:

RQ3: *How do noisy training corpora affect the performance of NMT and how can we leverage them to improve NMT performance?*

Recent research has shown that the negative effect of noise on neural machine translation performance as compared to phrase-based machine translation (Khayrallah and Koehn, 2018). Similar to phrase-based machine translation, most of the recent techniques proposed for de-noising training data for NMT are based on data filtering approaches that rank the samples in the training data in terms of their comparability or translation equivalence and then use only the highly ranked samples for training (Junczys-Dowmunt, 2018; Khayrallah et al., 2018; Rossenbach et al., 2018; Koehn et al., 2018). These ranking algorithms are based on combinations of heuristics-based filtering and data-driven classifiers (Junczys-Dowmunt, 2018; Barbu and Barbu Mititelu, 2018). The classifiers are usually trained on high-quality data that enable the machine learning algorithms to learn features from high-quality samples. Samples from noisy data that are ranked low by these algorithms are discarded, and only highly ranked samples are used for training. Although recent research has shown that data filtering techniques can improve the performance of NMT, we believe that these techniques yield inefficient utilization of low-quality data. This is due to the fact that in filtering techniques, samples that are identified as noisy by the data selection algorithm are discarded completely. However, as we will demonstrate in Section 5.3.1, a noisy sentence pair can still have some fragments that can provide useful contextual features for NMT models. Therefore,

it is important to investigate and experiment alternative techniques that could enable utilization of all samples in the available training data, even if they are of low quality. In this chapter, we propose one such technique based on knowledge-distillation and show that for recurrent NMT, the proposed technique allows for the utilization of low-quality training data without requiring any data filtering techniques.

For many language pairs, substantial amounts of high-quality parallel corpora are not available. For some of these language pairs, another useful resource known as comparable corpora can be obtained easily in substantially larger amounts. A comparable corpus is an aligned bitext created by crawling large monolingual data in the source and target languages from multilingual news portals such as Agence France-Presse (AFP), BBC news, and Euronews, etc. Source and target sentences in these monolingual corpora are then aligned by automatic document and sentence alignment techniques (Munteanu and Marcu, 2005). Such a bitext extracted from comparable data is usually not of the same quality as annotated parallel corpora. In recent research, Khayrallah and Koehn (2018) showed that building models from low-quality data could have a degrading effect on the performance of recurrent NMT models. Based on the observations reported by Khayrallah and Koehn (2018), we address the first sub-research question in this chapter:

RQ3.1 *What is the effect of comparable or noisy training data on the performance of a recurrent NMT system?*

Following Khayrallah and Koehn (2018), there is a growing interest in filtering and sampling techniques to extract high-quality sentence pairs from large, noisy parallel texts. Recently, the "Parallel corpus filtering" (Koehn et al., 2018) shared task was held at WMT-2018. This task aims at extracting high-quality sentence pairs from Paracrawl,[1] which is a large but noisy parallel corpus. Most of the participants (Junczys-Dowmunt, 2018; Barbu and Barbu Mititelu, 2018; Koehn et al., 2018) in this task use rule-based pre-filtering methods followed by a classifier-based scoring of sentence pairs. A subset sampled with a fixed number of target tokens is then used to train an NMT system in order to evaluate the relative quality of the filtered bitexts. Some of the submissions show good translation performance for the German→English translation task by training on the filtered bitext only (Koehn et al., 2018). In Section 5.2, we discuss in detail the relevant literature that aims to utilize low-quality training corpora for NMT.

Although results from Khayrallah and Koehn (2018) and the submissions at the "Parallel corpus filtering" task show that noisy training data indeed has a negative effect on NMT performance, there are some open questions in their conclusions. First, the majority of previous work has experimented with only one standard noisy dataset, which is the Paracrawl corpus for the German → English translation task. Second, the submissions at the "Parallel corpus filtering" task reported results for systems trained only on the samples filtered from the Paracrawl noisy corpus. Therefore, there is a requirement to further evaluate the effects of noisy data for other language pairs. Moreover, it is also important to evaluate the effect of noisy data when it is used in conjunction with clean, high-quality data. To answer RQ3.1, we experiment with

[1] https://paracrawl.eu/

noisy data for three language pairs and evaluate the effect of noisy data when used in conjunction with high-quality parallel data.

As already discussed, most previous work that aims to use noisy corpora for neural MT training select only high-quality subsamples from the noisy corpus and discard samples that are ranked lower. However, we argue that most of the noise in the training data is due to the low translation equivalence between aligned sentence pairs. Such noisy samples can still have some correct overlaps between the fragments of source and target sentences. Therefore, instead of discarding a sentence pair completely, using it with possible correction of noisy fragments can provide better gains in translation performance. In order to correct noisy fragments, we explore two approaches. The first one is to simply replace the source sentences in the noisy corpora with their back-translation equivalents, which are the translations of the corresponding target sentences generated through another pre-trained NMT model in reverse direction (Sennrich et al., 2016a). The second approach is knowledge distillation, where the correct targets for noisy fragments are replaced with a pseudo-label obtained through supervision from a strong teacher model. Both approaches aim to answer RQ3.2:

RQ3.2 *To what extent can we leverage noisy comparable data to improve NMT performance through the transfer of knowledge from a high-quality model?*

In our distillation strategy, we first train a teacher model on the clean parallel data, which then guides the training of a student model on the combination of clean and noisy data. Our experimental results demonstrate that for multiple language pairs, distillation helps to successfully utilize noisy comparable corpora without any performance degradation. Moreover, it also outperforms one of the best performing filtering techniques reported in (Koehn et al., 2018). In Section 5.3, we provide a brief discussion of the types of noise encountered in the comparable data and describe our strategy to use knowledge distillation for training with noisy data in Section 5.4. We discuss our experimental settings in Section 5.5 and results in Section 5.6.

The majority of related research on studying and handling noisy training data experimented only with recurrent architectures for NMT (Khayrallah and Koehn, 2018; Koehn et al., 2018). However, recently proposed non-recurrent architectures have not yet been studied in the context of noisy training data. Non-recurrent architectures such as the Transformer models (Vaswani et al., 2017) have achieved comparable or better performance as compared to recurrent NMT models. Moreover, they are more appealing due to their ability to process the input in parallel, resulting in faster computations. However, it is important to investigate whether these non-recurrent models also suffer from similar degradations when high-quality training bitexts are augmented with additional noisy data. We believe that due to the sequential processing of the input sequence in the training data, recurrent networks are more severely affected by noisy fragments as compared to non-recurrent architectures, where the input representations are more localized, and the overall representations of sequences are less severely affected by small noisy fragments. In Section 5.7, we discuss the differences between recurrent and non-recurrent NMT models and provide the basic intuition behind our argument. As an additional investigation, in Section 5.9, we empirically compare the effect of noisy data on the performance of recurrent vs. non-recurrent architectures for NMT. These experiments provide an answer to our third sub-research question:

RQ3.3 *What is the relative variation in performance of recurrent vs. non-recurrent NMT models when noisy data is added to the training pool?*

5.2 Related work

We divide the discussion of relevant literature into four categories: First, we discuss commonly used approaches in machine learning that are aimed at handling noisy training data. Second, we briefly discuss previous work that aims to utilize noisy or low-quality data for improving the performance of phrase-based MT. Next, we describe recently proposed approaches for the utilization of noisy data for training NMT. Finally, we discuss noise handling approaches used in other neural network-based tasks that can be directly applied to NMT.

In the machine learning literature, it is well-known that the performance of machine learning algorithms is highly susceptible to the quality of the training data (Nettleton et al., 2010). Although training noise can be categorized into various types, the most common noise type is caused by incorrect output labels. Label noise refers to sample instances where the given targets do not correctly represent or define the input samples (Frénay and Kabán, 2014). Frénay and Verleysen (2014) provided a comprehensive survey of the various techniques to handle label noise in the training data. The most common method for handling training noise is the filtering method, where the idea is to treat the noisy samples as outliers. These filtering methods detect noisy samples with different outlier detection algorithms (Liu and Tao, 2016) and finally exclude them from the training data. Some recent approaches suggest direct learning through noise-robust algorithms, which involves modifying the loss function (Beigman and Klebanov, 2009; Manwani and Sastry, 2012). These methods are successful in cases where label noise can be managed by avoiding overfitting on the noisy data. In situations where a small set of high-quality labeled data is available, semi-supervised learning is used. In these semi-supervised techniques, the unlabeled data can be annotated, or noisy data can be corrected by classifying the samples. These classifiers are trained on high-quality data. One variant of the semi-supervised method is the bootstrapping approach, where improved labels for noisy or unlabeled data can be obtained by predictions of another classifier (Reed et al., 2014). Another well-known approach is the transfer learning method, where the idea is to treat noisy data as samples from a different domain. This approach is quite successful in situations where the size of the noisy data is relatively large as compared to the size of the clean data. In such cases, using transfer learning, the model is first intialized by training on the noisy data and then fine-tuned on the smaller clean dataset (Oquab et al., 2014). Recently, some approaches have been proposed to handle noise in a deep learning setting. These approaches aim at modeling the label noise by proposing either an additional loss function (Mnih and Hinton, 2012) or an additional noise layer that adapts the network outputs to match the noisy label distribution (Sukhbaatar and Fergus, 2014).

For machine translation, the majority of the noisy samples correspond to mismatched sentence pairs, where the degree of semantic equivalence between the source and target pairs is low. We provide a detailed discussion of this noise category along with examples in Section 5.3. For phrase-based MT, it has been shown that filtering or cleaning the

bitext prior to training can provide significant boosts in SMT performance (Vogel, 2003). The majority of the proposed methods for training SMT systems on noisy data focus either on the selection of high-quality parallel sentences from comparable corpora (Fung and Cheung, 2004; Axelrod et al., 2015) or on cleaning the noise in the aligned sentence pairs (Formiga and Fonollosa, 2012; Mermer et al., 2007). In filtering-based approaches, the noisy samples are usually identified using a classifier that is trained on features based on bilingual lexicons. A slightly different variant of the filtering approach is proposed by Hunsicker et al. (2012). They use additional heuristics to trim the search space by imposing thresholds on features such as length differences or dictionary scores. Instead of filtering out noisy sentences, another simple approach is to weigh the training samples according to their quality scores. As a result, the phrase pairs and other features learned from noisy data will have lower probability scores as compared to those learned from clean data. Zhang and Chiang (2014) use this approach through Kneser-Ney smoothing based on weighted counts of n-grams. Although all these approaches proposed for phrase-based MT have shown significant improvements, it is important to note that in the case of phrase-based MT, the effect of training noise is not as problematic as it has been observed for NMT (Khayrallah and Koehn, 2018).

Similar to phrase-based MT, there has been a large body of research for domain adaptation for NMT. The majority of these approaches focus on the selection of data relevant to the intended domain. Cross entropy-based data selection is the most common variant of this approach (Axelrod et al., 2015; van der Wees et al., 2017). By treating noisy data as a different domain from clean data, data selection approaches proposed for NMT can also be applied to handling noisy data. However, as already stated in (Khayrallah and Koehn, 2018), the problem of noisy data is different from domain adaptation because it deals with the quality of training data across domains. For training NMT, an important aspect is the requirement for substantial amounts of training data. In order to obtain additional training data, a very successful approach is that of data augmentation, which aims to generate additional synthetic data either by back-translation of monolingual data (Sennrich et al., 2016a) or by alteration of sentences in available parallel data (Fadaee et al., 2017). From the success of these data augmentation methods for NMT, we can conclude that NMT systems can benefit even from non-natural synthetic training data if the synthetic data has good translation equivalence between the source and target sentences. Based on these observations, it is worthwhile to explore whether NMT can benefit from additional parallel data that is not of the same quality as the annotated clean data. In this regard, important recent work is (Khayrallah and Koehn, 2018). The authors report that NMT models could suffer substantial degradation by the addition of noisy bitexts when compared to a baseline model trained only on high-quality parallel text.

In the WMT-2018 “Parallel corpus filtering” shared task (Koehn et al., 2018), the participants were asked to filter out high-quality subsamples from the noisy Paracrawl corpus. The submissions were evaluated based on the performance of NMT systems trained only on the bitext filtered from Paracrawl. However, given that many language pairs have at least some small amount of high-quality parallel corpora, it is important to investigate whether a bitext filtered using these proposed techniques results in any additional improvements when used in conjunction with the high-quality data. Filtering techniques involve discarding a sentence pair with a low confidence score. However, a

sentence pair with a low score may still have fragments in the source and target sentences that can provide useful contexts. Our results show that for a recurrent NMT model, filtering the noisy bitext using one of the best techniques submitted to the filtering task (known as "dual conditional cross-entropy filtering" (Junczys-Dowmunt, 2018)) yields only small improvements.

Finally, we discuss some of the approaches used in other neural network tasks that can be directly applied to NMT. For example, as a variant of bootstrapping, a forward translation procedure can be applied to the noisy bitext. In this method, the target side of the noisy bitext can be replaced by synthetic target sentences obtained by decoding the source sentences with a model trained on high-quality data. However, a better alternative for NMT would be to use back-translation (Sennrich et al., 2016a), i.e., to replace the source sentences in the noisy bitext by synthetic source sentences obtained by decoding the given target sentences with a model trained in the reverse direction. Our experiments show that although back-translations of noisy data show some improvements as compared to the degradations caused by direct use of the noisy data, these improvements are quite moderate. Nevertheless, these results do suggest that, indeed, NMT performance can be improved if the target labels have better correspondence with the source tokens. However, there are several limitations to back-translation-based data cleaning. First, it is expensive in terms of time because back-translation requires running beam search based decoding for millions of training examples. Second, the back-translation procedure does not guarantee the correction of 'label noise'; instead, it changes the sample features according to the given labels.

As explained in Section 4.3.2, fine-tuning (Barone et al., 2017) is a well-known technique used for domain adaptation for NMT. This technique can also be used as a possible solution for training with noisy data. In fine-tuning, the idea is to first pre-train the model on noisy data and then continue training on high-quality data. Another line of research relates to data selection, which is a popular technique for domain adaptation. The idea in this approach is to extract samples from the noisy corpus that are similar to representative samples of high-quality data. The most common similarity measure used in this approach is the cross-entropy between the high-quality samples and noisy samples (Moore and Lewis, 2010). This is done by training two separate language models on the noisy and clean data and then scoring the samples in the noisy data based on differences in cross-entropy values from the two language models. Samples with scores above a chosen threshold are then selected for training.

Recently, Wang et al. (2018) have proposed an online data selection based strategy for de-noising training data. Their idea is to first train a model on clean data and then iteratively fine-tune the model on noisy data while selecting the samples based on the differences between the binary cross entropies of the models trained on clean and noisy data. In each iteration, a fixed number of samples with the highest cross-entropy differences are used for fine-tuning. This solution is based on cross-entropy based data selection strategies for domain adaptation (Axelrod et al., 2015; van der Wees et al., 2017). Wang et al. (2018) apply this idea to handle noisy data. The success of their approach suggests that entropy of noisy samples with respect to a model trained on clean data could be an effective criterion to decide the quality of the samples. Our proposal for using knowledge distillation is based on the idea of generating a pseudo-label, which is a combination of a given target label (which can be noisy) and a label predicted by a

teacher model. We will discuss the details in Section 5.4.3.

Although knowledge distillation has been used as a solution to other problems of NMT such as domain adaptation (as discussed in Chapter 4) or transfer learning for low-resource languages (Chen et al., 2017) and for leveraging noisy data for image recognition (Li et al., 2017), our approach is the first attempt to exploit distillation for training NMT systems with noisy data.

5.3 Background

5.3.1 Noise in the training corpora

As discussed in Section 5.1, the majority of related research in machine learning deals with handling label noise. Label noise is a sample instance in which the target labels do not correspond to the given input. For machine translation, training samples are sentence pairs, which are sequences of tokens. However, in the traditional NMT approach, the model is trained using a loss function with respect to a single output token, and the overall loss from all tokens in a sentence or a batch of sentences is accumulated for back-propagation. For classification or other labeling tasks, it can be beneficial to either completely discard a sample or clean the label with one of the approaches discussed in Section 5.2. For NMT, discarding a sample means discarding a full-sentence pair. However, in the context of bitexts, noise in the samples occurs in different varieties and may not necessarily be considered label noise. Therefore, before proposing any solution, it is important to study the noise types in the NMT training data and identify which of the categories can be considered as "label noise." In this regard, (Khayrallah and Koehn, 2018) is an important work for NMT. The authors analyze the Paracrawl German→English bitext, identifying various types of noise in this corpus. They categorize the noise into the following four main types:

- **Misaligned sentences:** This type of noise forms the majority of samples in the Paracrawl corpus. This noise occurs when the aligned sentences are not correct translations of each other. Khayrallah and Koehn (2018) note that this type of noise is frequent in corpora extracted and aligned from the web but is less frequent in annotated corpora such as Europarl. For NMT, this type of noise corresponds to what is considered as "label noise" in the machine learning literature. In this chapter, our focus will be on handling this type of noise.

- **Wrong language:** This situation also occurs in the case of web-aligned corpora when some of the sections on either the source or target side are exchanged or texts from a third language is wrongly aligned. This type of noise can very easily be identified and filtered out by using language identification tools such as *lang-id* (Lui and Baldwin, 2012).

- **Untranslated sentences:** In web aligned corpora, often, some of the sentences may remain untranslated. These sentences may correspond to elements such as copyright information or markup tags. Again, it is very easy to filter out this type of noise. Simple pre-processing techniques can be used to identify untranslated sentences in a bitext.

Table 5.1: Categorisation of noise type in Paracrawl corpus.

Noise type	Count
Misaligned sentences	41%
Wrong language	23%
Untranslated Sentences	4%
Short segments	6%
Non linguistic characters	2%
Good sentence pairs	23%

- **Short segments:** In some cases, large bitexts may contain translations of small phrases or bilingual dictionaries. Since in NMT, the aim is to learn the translations of sentences, these short segments provide little significant information. They can be filtered using constraints based on sentence length.

Table 5.1 summarizes the findings of Khayrallah and Koehn (2018). The majority of noisy samples fall into the category of misaligned sentences. This type of noise is not easy to identify and can not be corrected with simple pre-processing. Therefore, similar to most of the previous work, in this chapter, our aim is to propose a solution for this type of noise.

Khayrallah and Koehn (2018) conduct their analysis for one language pair on the Paracrawl corpus. However, other well-known noisy corpora that are traditionally used for many language pairs are the automatically aligned parallel datasets known as comparable bitext. A well-known instance of a comparable bitext for Arabic→English and Chinese→English is the ISI bitext created by automatically aligning sentences from monolingual corpora such as the AFP and Xinhua news portals (Munteanu and Marcu, 2005). No detailed study has been conducted on the categorization of noise types in comparable corpora. However, from the description of the extraction technique, it becomes evident that most of the noise in such corpora would be due to poor translation equivalence. The alignment method used for the creation of these bitexts first searches for articles representing similar articles in two separate monolingual corpora using cross-lingual information retrieval with the help of a dictionary (Munteanu and Marcu, 2005). Then, parallel sentences are aligned by calculating the word overlap between each candidate sentence pair followed by a maximum entropy classifier. Since the bitexts are extracted from monolingual corpora in the source and target language, there is rarely any noise due to misspelling, wrong re-ordering, or non-linguistic characters. The majority of noise in the resulting aligned bitext is due to limitations of the sentence alignment techniques, often resulting in sentence pairs that are partial translations of each other.

Table 5.2 shows some examples of noisy sentence pairs for Arabic→English (from the ISI bitext) and German→English (from the Paracrawl corpus). The fragments marked red in either the source sentence or the target sentence have no equivalence on the corresponding aligned side. For instance, in the Arabic→English sentence pair, the aligned target sentence contains fragments like *'Iraqi Kurd'*, which are missing in the human annotated translation. This implies that there is no corresponding fragment in

the source sentence for these target fragments. During training, these fragments will result in incorrect parameter updates and should be considered label noise. However, the other fragments in the target sentence can still provide correct feedback.

Table 5.2: Noisy sentence pair example from ISI bitext (Arabic→English) and Paracrawl (German→English). Fragments in red in the aligned target translation have no corresponding fragment in the source sentence and vice versa. **Src** = Source sentence in the bitext, **Tgt** = Target sentence in the bitext, and **Ann** = Human annotated translation of the given source sentence.

Arabic→English (ISI bitext)	
Src:	وقررت وزارة العدل الهولندية ابعاده رغم طلب الاردن تسليمه في اطار قضية تهريب مخدرات.
Tgt:	The Dutch justice ministry decided to expel the Iraqi Kurd despite Amman's demand that he be handed over to Jordanian authorities.
Ann:	The Dutch Justice Ministry decided to deport him, despite Jordan's request to hand him over as part of a drug smuggling case.
German→English (Paracrawl)	
Src:	Der Elektroden Schalter KARI EL22 dient zur Füllstandserfassung und -regelung von elektrisch leitfähigen Flüssigkeiten .
Tgt:	The KARI EL22 electrode switch is designed for the control of conductive liquids .
Ann:	The electrode switch KARI EL22 is used for level detection and control of electrically conductive liquids.

5.4 De-noising training data

5.4.1 Problem definition

Let's assume we have a combined bitext $D_{cmb} = D_{clean} \cup D_Q$, where D_{clean} is a clean annotated bitext and D_Q is a quasi-parallel comparable bitext. $D_{cmb} = [X_1, Y_1], \ldots, [X_n, Y_n]$ where $X_i = [x_1, x_2, \ldots, x_t]$ is a source sequence and $Y_i = [y_1, y_2, \ldots, y_m]$ is the corresponding target sequence where x_i and y_j are tokens. When represented as indices in the target vocabulary, y_j can be considered as label. Since the training data includes the comparable data, some of the labels y_j corresponding to it are noisy or unreliable. As stated in (Li et al., 2017), the noisy label y_i can be considered as corrupted from the true label y_j^*. The goal is to train an optimal model on the entire dataset D_{cmb} so as to minimize the risk on unseen test data as defined in Equation 5.1:

$$g^* = \underset{g}{\operatorname{argmin}}\, R_{D_t}(g) = \underset{g}{\operatorname{argmin}}\, R_{D_t}\Big\{L[y^*, g(x)]\Big\}. \tag{5.1}$$

Here, D_t is the unseen test data and g is the classifier function. Li et al. (2017) define risk R in terms of loss with respect to the true label y^*.

5.4.2 De-noising by back-translation

As discussed in Section 5.2, bootstrapping is a strategy that avoids fitting model parameters to the noisy data by first achieving a reasonable model performance and relying more on the model predictions in the later stages. A simple way of applying bootstrapping to NMT would be to first train an initial model on the available clean data and then to obtain better translations by decoding the source side of the bitext (known as forward translations). However, for NMT fluency and correctness of the target side of the training data is more important than that of the source side.

A well-known technique called back-translation (Sennrich et al., 2016a) has been proposed for NMT, where the aim is to augment the training corpus by generating pseudo-source sentences for additional monolingual data using a model trained in the reverse direction of the intended translation direction. By using back-translation, the additional context in the fluent target sentences can be easily exploited regardless of the synthetic quality of the source sentences. We apply the back-translation strategy as a

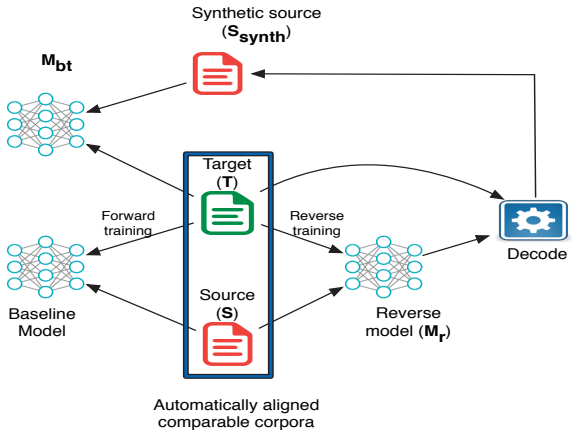


Figure 5.1: De-noising by back-translation. **S** = Source side of the training bitext, **T** = Target side of the training bitext, $\mathbf{M}_r$ = NMT model trained in revere direction (T $\rightarrow$ S), $\mathbf{S}_{synth}$ = Pseudo source sentences obtained by decoding **S** with model **Mr**, and $\mathbf{M}_{bt}$ = Final model trained on dataset $\mathbf{S}_{synth} \rightarrow \mathbf{T}$.

bootstrapping method in order to obtain synthetic source sentences for the automatically aligned comparable corpora. Similar to the bootstrapping method, an NMT model can be trained in reverse direction on the available clean data to achieve a model with reasonable generalization. Since the attention distribution in NMT highly correlates with traditional word alignments (Ghader and Monz, 2017), during decoding, most target words are generated corresponding only to one of the words/tokens in the source sequence, thus mostly avoiding over-generation. Therefore, decoding target sentences with a reverse model trained on clean data can produce back-translated source sentences with reasonably improved levels of correspondence to the input target sentences. The proposed process is illustrated in Figure 5.1.

Given a corpus (S, T), the baseline model will be trained directly on this noisy corpus. Simultaneously, we train a reverse model M_r on the same corpus in reverse direction (T, S). Then we use the model M_r to decode the corpus on which it was trained. This gives us the back-translated source side S_{synth}. A final model M_{bt} is then trained on the corpus (S_{synth}, T).

Formally, the "De-noising by back-translation" approach implies that in Equation 5.1, risk is minimized with respect to y (given noisy label) instead of actual label y^*. Moreover, the input sequence X_i is replaced by a pseudo source sequence $X_i^* = [x_1^*, x_2^*, \ldots, x_t^*]$. Now, risk is minimized with respect to given target label and pseudo-input x^*:

$$g^* = \underset{g}{\operatorname{argmin}}\, R_{D_t}(g) = \underset{g}{\operatorname{argmin}}\, R_{D_t}\Big\{L[y, g(x^*)]\Big\}. \tag{5.2}$$

Table 5.3 shows an example of a sample generated by the back-translation process. *Source* and *Target* are the original sentences in the noisy bitext. A pseudo translation $Source_{pseudo}$ is obtained by back-translation of *Target*. $Human - Source_{pseudo}$ is the back-translation of *'Target'* obtained by a human translator. We can observe that the back-translated source sentence $Source_{pseudo}$ is almost identical to the one obtained by a human translator.

Table 5.3: Example of de-noising by data cleaning.

Arabic→English (ISI bitext)	
Source	وقررت وزارة العدل الهولندية ابعاده رغم طلب الاردن تسليمه في اطار قضية تهريب مخدرات.
Target	The Dutch justice ministry decided to expel the Iraqi Kurd despite Amman's demand that he be handed over to Jordanian authorities.
$Source_{pseudo}$	قررت وزارة العدل الهولندية طرد الاكراد العراقيين ب الرغم من مطالبة عمان ب تسليم ه الي السلطات الاردنية
$Human - Source_{pseudo}$	The Dutch Justice Ministry has decided to expel the Iraqi **Kurds** despite Amman's demand that **it** be handed over to Jordanian authorities

5.4.3 Knowledge distillation for noisy data

Although de-noising by back-translation is a simple solution, obtaining back-translations for a large noisy corpus is expensive in terms of time. A better alternative would be to correct the noisy labels instead of modifying the source sentence. Moreover, the bootstrapping approach generates hard targets using beam search. Li et al. (2017) pointed out that soft distillation scores (based on predictions of another model) are better than hard labels when guiding the training process of a model. Motivated by these observations, we apply "knowledge distillation" to guide training of a student network by soft scores generated by a teacher model. As discussed in Section 2.4, knowledge distillation is a framework proposed by Hinton et al. (2014) for training compressed "student" networks by using supervision from a large teacher network. Here, we discuss

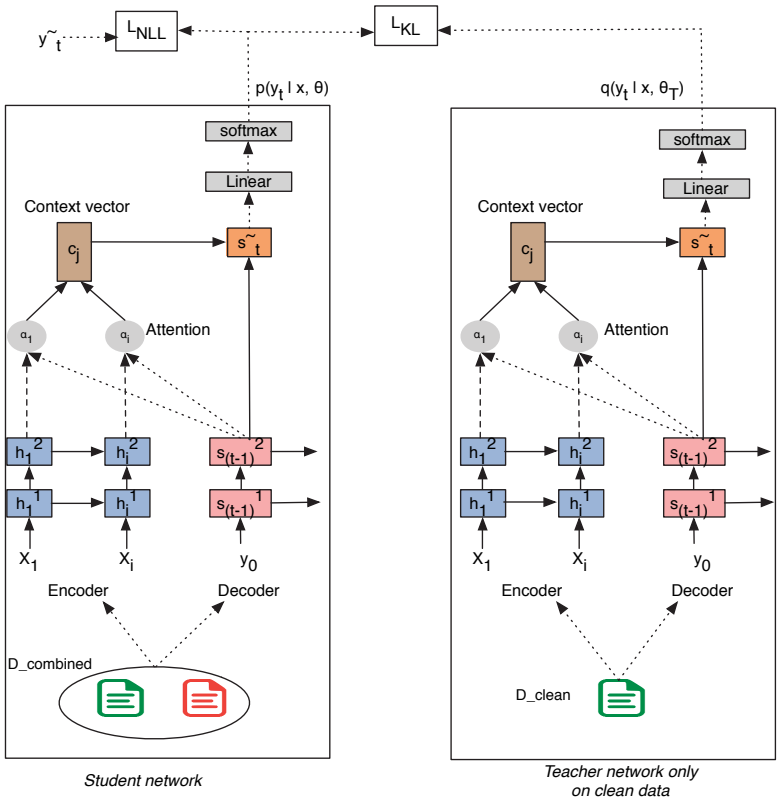


Figure 5.2: **Distillation for noisy data**. Both the teacher and student network have the same architecture. The teacher network is trained only on the clean data and the student network is trained for two losses: L_{NLL} with respect to target labels and L_{KL} with respect to the output distribution of the teacher network.

the main intuition and idea behind using knowledge distillation for noisy labels. A detailed analysis is given by Li et al. (2017). As shown in Figure 5.2, the idea is to first train the teacher model f on the clean data D_{clean} and then transfer the knowledge from the teacher to a student network that is trained on the entire dataset D_{cmb} by optimizing the following loss:

$$\mathrm{L}_{\mathrm{D_{cmb}}}\Big(y_i, f(x_i)\Big) = \lambda \times \mathrm{L}\Big(y_i, f(x_i)\Big) + (1-\lambda) \times \mathrm{L}\Big(s_i, f(x_i)\Big), \tag{5.3}$$

where $s_i = f_{D_{clean}}(x_i)/\tau$ and τ is the temperature of the softmax. In Equation 5.3, the student model is trained on the combination of two loss functions, the first term is the cross-entropy loss l between the prediction of the student model and the ground truth y_i, while the second term is the cross-entropy or KL-divergence between the output distributions of the student model and the teacher model. Also, λ is a parameter to

balance the weight between the two losses. Assuming the second loss to be cross-entropy, Equation 5.3 can be re-written as:

$$\mathrm{L}_{\mathrm{D}_{\mathrm{cmb}}}\Big(y_i, f(x_i)\Big) = L\bigg(\lambda \times y_i + (1-\lambda) \times \Big(s_i, f(x_i)\Big)\bigg). \tag{5.4}$$

We can define $y_i^{\lambda} = \lambda y_i + (1-\lambda)(s_i, f(x_i))$ as a pseudo-label that is a combination of the given noisy label y_i and the prediction s_i from the teacher model. Therefore Equation 5.4 can be re-written as:

$$\mathrm{L}_{\mathrm{D}_{\mathrm{cmb}}}\Big(y_i, f(x_i)\Big) = L(y_i^{\lambda}). \tag{5.5}$$

Li et al. (2017) provide an analysis based on a comparison between the risks involved in training directly on the noisy labels as compared to training on the pseudo label as defined above. They show that training on the pseudo-labels, for some values of λ defined through distillation, involves lower risks than direct training or bootstrapping. Therefore, a better model can be trained by driving the pseudo-labels closer to the ground truth labels. In the case of comparable corpora for NMT, instead of learning only from uncertain ground truth labels, the student model also benefits from the predictions of the teacher model while learning to imitate it.

5.5 Experimental setting for noisy data distillation

As discussed in Section 5.1, our first sub-research question RQ3.1 is to evaluate the effect of noisy data on the performance of NMT when used in conjunction with clean data. For this purpose, we conduct experiments with three language pairs: Arabic→English, Chinese→English, and German→English. To this end, we first establish baseline experiments where either only clean data or only noisy data or concatenations of both are used as training data. Further, to answer the second sub-research question RQ3.2, we evaluate our proposed distillation strategy in terms of translation performance as compared to other filtering and cleaning strategies.

First, we describe the baselines followed by the description of data and model parameters for recurrent NMT. Next, we discuss our experimental results in detail. As already discussed, the last research question RQ3.3 concerns a comparison of recurrent and non-recurrent NMT architectures. We will discuss the motivation, experimental settings and results for RQ3.3 in Section 5.7, 5.8 and 5.9, respectively.

5.5.1 Comparisons

We compare our knowledge distillation technique to baseline training on clean and noisy data. Further, we evaluate fine-tuning and data cleaning by back-translation. We carry out the following experimental comparisons:

- **Training on parallel data only:** Training only on high-quality parallel data. This experiment is also the primary baseline for comparison with the proposed method.

- **Training on comparable/noisy data only:** We conduct this experiment to demonstrate a substantial difference between the performance of the models trained only on noisy data and the models trained only on clean data.

- **Training on combined comparable and parallel data:** This experiment demonstrates the effect of adding noisy training data to the baseline training data pool.

- **Back-translation:** Here, we use de-noising by back-translation as described in Section 5.4.2. This is done by training an NMT system in reverse of the desired direction on the clean data and obtaining pseudo-source sentences for the noisy training data. After applying back-translation, we discard the original source sentences in the comparable data and replace them with the pseudo-source sentences. The back-translated comparable data is then added to the training data pool.

- **Fine-tuning:** Fine-tuning is the standard practice commonly used for domain adaptation. For noisy data, the idea is to first train the model on noisy data and then continue training on clean data.

- **Dual cross-entropy filtering:** As discussed in the introduction, Junczys-Dowmunt (2018) report the best results for the Parallel Corpus Filtering task for WMT-18. They use the dual cross-entropy method in which sentence pairs in the noisy corpus are ranked based on forward and backward losses for each sentence pair with respect to NMT models trained on clean data in forward and reverse directions. We consider this filtering method as a competitive baseline for our approach.

Note that back-translation requires beam-search based decoding, which is quite expensive for large amounts of comparable data.

5.5.2 Datasets

Noisy data: Two well-known automatically aligned corpora for Arabic→English and Chinese→English are the ISI bitexts that are commonly used as representatives of comparable data. These corpora are built by aligning similar stories in multilingual news portals such as *AFP* for Arabic→English $(LDC2007T08)$ and *Xinhua news agency* for Chinese→English $(LDC2007T09)$. We consider all AFP sources from the ISI Arabic→English bitext with a size of 1.1 million sentence pairs and Xinhua news sources for the Chinese→English bitext with a size of 550K sentence pairs. Both corpora are created by automatically aligning sentences from monolingual corpora (Munteanu and Marcu, 2005). For the German→English task, we randomly sample a bitext of equal size from the raw Paracrawl corpus ("very noisy" 1 billion English tokens) similar to (Khayrallah and Koehn, 2018). To be able to compare with the best filtering method, we also use a bitext of 100 million target tokens submitted by Junczys-Dowmunt (2018) (available from the shared task website using a score file), which is filtered using their proposed dual cross-entropy score.

Table 5.4: Datasets and statistics. Para$_{rn}$ = Randomly sampled subset of Paracrawl after langid filtering. Filt$_{toks=100M}$ = 100 million target token subsample submitted by Junczys-Dowmunt (2018).

	Clean		Noisy	
	Source	Size	Source	Size
Arabic→English	LDC	300k	ISI bitext	1.1m
Chinese→English	LDC	550k	ISI bitext	550k
German→English	WMT-17	5.1M	Para$_{rn}$	5.1M
German→English	WMT-17	5.1M	Filt$_{toks=100M}$	4.6M

Clean data: For Arabic→English, we compose the parallel data consisting of 325k sentence pairs from various LDC catalogues.[2] For Chinese→English, a parallel corpus of 550k parallel sentence pairs from LDC catalogues[3] is used. Note that for Arabic→English, the size of the comparable corpus is approximately four times that of the parallel data while for Chinese→English, the comparable corpus size is almost the same as that of the parallel corpus. For German→English, we use high-quality data from the training corpus provided for WMT-17 (Bojar et al., 2017). Table 5.4 summarizes clean and noisy training data for all language pairs. NIST MT05 is used as development set for both Arabic→English and Chinese→English. We use NIST MT08 and MT09 as test sets for Arabic→English and MT-06 and MT-08 for Chinese→English. A byte pair encoding (BPE) of size 20k is trained on the parallel data for both language pairs. For German→English, a BPE of 32k is trained on the WMT-17 training data, newstest15 is used as development set and newstest16 and newstest17 are used as test sets. Translation quality is measured in terms of case-sensitive 4-gram BLEU (Papineni et al., 2002). Approximate randomization (Noreen, 1989; Riezler and Maxwell, 2005) is used to detect statistically significant differences.

5.5.3 Model parameters

We train an LSTM-based bidirectional encoder-decoder model as described in (Luong et al., 2015) using the Open-NMT-python toolkit (Klein et al., 2017), with both embeddings and hidden layers of size 1000. The maximum sentence length is restricted to 80 tokens. Parameters are optimized using the *Adam* optimizer with an initial learning rate of 0.001, a decay rate of 0.5 (after every 10k steps), a dropout probability of 0.2 and label smoothing of 0.1. A fixed batch size of 64 sentences is used. Model weights are initialized uniformly within range $[-0.02, 0.02]$. We train for a maximum of $200k$ steps and select the model with the highest BLEU score on the development set for the final evaluation and decode with a beam size of 5.

[2]LDC2006E25, LDC2004T18, several Gale corpora, LDC2004T17, LDC2005E46 and LDC2004E13.
[3]LDC2003E14, LDC2005T10 and LDC2002E18.

5.6 Results

Table 5.5 shows all experimental results for Arabic→English. The performance of the model trained only on noisy (comparable) data is substantially lower than that of the model trained only on clean (parallel) data (a difference of up to -13.9). Further, combining the noisy data with clean data degrades the performance significantly (up to -2 BLEU). De-noising the comparable data through back-translation sightly improves the performance (up to $+1.4$) as compared to the parallel-only baseline. Fine-tuning shows slight improvement for two test sets (up to $+0.5$ for MT08) but results in a significant drop for the development set (MT05). To evaluate the best performing filtering technique, for Arabic→English, we apply dual cross-entropy filtering of Junczys-Dowmunt (2018) by ranking the sentence pairs in the comparable bitext based on to the dual cross-entropy score. Then we select subsamples from the top 25%, 50%, and 75% of the full comparable bitext. Filtering at 25% and at 50% shows significant improvements (up to $+1.4$ and $+1$, respectively). However, filtering 75% again degrades the baseline performance (up to -1.2) as compared to the parallel-only baseline. Our proposed knowledge distillation approach, which utilizes all comparable data, consistently outperforms fine-tuning, back-translation, and filtering with an improvement of up to $+2.4$ over the parallel-only baseline.

Table 5.5: Performance of various training strategies for Arabic→English. Comparable$_{bck}$ = Back-translated comparable corpora. KD = Knowledge distillation. Boldfaced represent Significant differences at p < 0.01; * represents baseline experiment.

	Arabic→English		
	MT05	MT08	MT09
Parallel only	57.7	46.1*	49.9*
Comparable only	48.9 (-8.8)	**32.7** (-13.4)	**36.0** (-13.9)
Combined (Parallel + Comparable)	55.2 (-2.5)	44.2 (-1.9)	**47.9** (-2)
Parallel + Comparable$_{bck}$	**60.4** (+2.7)	47.5 (+1.4)	**51.0** (+1.1)
Fine-tuning	**56.1** (-1.6)	46.6 (+0.5)	**50.3** (+0.4)
Dual cross entropy filtering			
Parallel + Comparable$_{filt-25\%}$	**59.9** (+2.2)	**47.4** (+1.4)	51.1 (+1.2)
Parallel + Comparable$_{filt-50\%}$	59.2 (+1.5)	46.8 (+0.7)	**50.9** (+1)
Parallel + Comparable$_{filt-75\%}$	56.7 (-1)	**44.9** (-1.2)	49.1 (-0.8)
Knowledge distillation			
KD	**62.3** (+4.6)	**48.4** (+2.3)	**52.3** (+2.4)

Similarly Table 5.6 reports the results on the Chinese→English experiments. The performance of training on noisy data is significantly lower than training on parallel data only (a difference of up to -17.5). Again, adding comparable data to the clean data degrades baseline performance by up to -2. Unlike Arabic→English, neither de-noising by back-translation nor fine-tuning shows any significant improvements over the baseline for Chinese→English. The dual cross-entropy filtering is applied similarly

to Arabic→English, as described above. However, unlike Arabic→English, filtering shows no improvement at all for any amount of the filtered comparable data. For all settings, filtering still degrades the performance significantly as compared to the parallel-only baseline. On the other hand, the knowledge distillation method consistently yields improvements over the parallel-only baseline (up to $+0.8$).

Table 5.6: Performance of various training strategies for Chinese→English. Comparable$_{bck}$ = Back-translated comparable corpora. KD = Knowledge distillation. Boldfaced represent Significant differences at $p < 0.01$; * represents baseline experiment.

	Chinese→English		
	MT05	MT06	MT08
Parallel only	28.8	27.5*	20.3*
Comparable only	11.3 (-12.1)	**10.2** (-17.5)	**5.20** (-15.1)
Combined (Parallel + Comparable)	27.7 (-1.1)	26.7 (-0.8)	**18.3** (-2)
Parallel + Comparable$_{bck}$	**29.1** (+0.3)	27.2 (-0.3)	19.8 (-0.5)
Fine-tuning	**25.1** (-3.7)	**23.5** (-4)	**17.2** (-3.1)
Dual cross Entropy Filtering			
Parallel + Comparable$_{filt-25\%}$	**19.7** (-9.1)	20.9 (-6.6)	**16.8** (-3.5)
Parallel + Comparable$_{filt-50\%}$	**20.4** (-8.4)	**21.8** (-5.7)	**17.0** (-3.3)
Parallel + Comparable$_{filt-75\%}$	**21.5** (-7.3)	**22.3** (-5.2)	**17.5** (-2.8)
Knowledge Distillation			
KD	**29.4** (+0.6)	**28.2** (+0.5)	**21.1** (+0.8)

For German→English, as shown in Table 5.7, the performance of a model trained on randomly sampled noisy Paracrawl data is significantly lower (a difference of up to -16.4) than training on high-quality WMT data. Khayrallah and Koehn (2018) reported degradation of up to -9 BLEU when combining clean and noisy data for German→English. However, we observe only -1 BLEU drop for the same setting. Nevertheless, directly adding noisy data seems to provide no additional improvement in performance. For the randomly sampled Paracrawl data, fine-tuning shows no improvement over the baseline; instead, a drop of up to -5.6 is observed. On the other hand, our knowledge distillation method with randomly sampled Paracrawl instances performs better than the baseline (up to $+0.3$).

For German→English, for comparison with dual cross-entropy filtering, instead of filtering repeatedly, we directly use the filtered bitext submitted by Junczys-Dowmunt (2018), which is available from the web portal of the shared task, and add it to the training data. As seen in Table 5.7, adding this filtered data also degrades BLEU by -1. Junczys-Dowmunt (2018) reported substantial gains after filtering as compared to the use of randomly sampled noisy bitext. However, we observe that neither training on the filtered bitext alone, nor combining it with the clean data (WMT) provide any additional improvement over a parallel data baseline. Instead, the BLEU score is worse than the baseline in both settings (-1.4 for filtered-only and -1 for combined WMT + filtered bitext). This implies that when used in conjunction with clean data, even this

Table 5.7: German→English results. WMT = Only clean Data, Para$_{rn}$ = Randomly sampled 5.1 million sentence pairs from Paracrawl. Filt$_{toks=100M}$ = 100 million target tokens filtered; Boldfaced represent Significant differences at p $<$ 0.01; * represents baseline experiment.

	test15	test16	test17
WMT (Parallel only)	25.2	30.0*	26.0*
Randomly sampled Paracrawl			
Para$_{rn}$	**14.6** (-10.6)	**10.2** (-19.8)	9.6 (-16.4)
WMT (Parallel) + Para$_{rn}$	**24.1** (-1.1)	**29.0** (-1)	25.0 (-1)
Fine-tuning (Para$_{rn}$)	**21.8** (-3.4)	**24.4** (-5.6)	**21.1** (-4.9)
KD (WMT + Para$_{rn}$)	**25.6** (+0.4)	**30.3** (+0.3)	**26.3** (+0.3)
Parcrawl filtered with dual cross entropy			
Filt$_{toks=100M}$ only	24.0 (-1.2)	28.8 (-1.2)	24.6 (-1.4)
WMT + Filt$_{toks=100M}$	24.1 (-1.1)	28.7 (-0.3)	25.0 (-1)
Fine-tuning (Filt$_{toks=100M}$)	23.9 (-1.3)	29.1 (-0.9)	25.1 (-0.9)
Knowledge Distillation			
KD (WMT + Filt$_{toks=100M}$)	**26.1** (+1.1)	**31.3** (+0.3)	**26.9** (+0.9)

high-performing filtering technique fails to provide any significant gains. On the other hand, applying knowledge distillation over this filtered bitext yields an improvement of up to $+0.9$ over the baseline. Based on the above results, we can now answer our research questions in this chapter. Our first subquestion in this chapter is:

RQ3.1 *What is the effect of comparable or noisy training data on the performance of a recurrent NMT system?*

As is evident from the above results, the performance of an NMT system trained only on noisy data is substantially worse than training on clean parallel data for all language pairs. Further, adding noisy data to the training data pool does not provide any additional improvements, instead it degrades the baseline performance significantly. The relative difference between the performance drop for all three language pairs can be attributed to the different sizes of clean and noisy data. While the findings of Khayrallah and Koehn (2018) were based on only one language pair, our experiments for two additional language pairs support the argument that low-quality or noisy data indeed harms the performance of NMT models, however, the relative performance degradation depends on the source and the quantity of noisy data as well as the clean data.

Finally, to answer research question RQ3.2, we evaluate the performance of our proposed distillation approach for all three language pairs.

RQ3.2 *To what extent can we leverage noisy comparable data to improve NMT performance through the transfer of knowledge from a high-quality model?*

Our proposed distillation strategy outperforms filtering as well as the back-translation replacement for all language pairs. The improvements for Arabic→English are substantially bigger, while only small improvements for Chinese→English are observed.

Nevertheless, distillation provides significant improvements as compared to the direct addition of noisy data. For German→English, applying the proposed distillation over the randomly sampled bitext combined with clean data shows slight improvements over the clean data baseline. Similarly, applying distillation on the filtered bitext (using "dual cross-entropy") combined with the clean data also shows significant improvements over the clean data baseline.

The improvements reported with knowledge distillation shown in Table 5.5, 5.6 and 5.7 correspond to the highest BLEU scores with respect to different values for λ. In Figure 5.3, we show the effect of varying values of λ (between 0.1 and 0.9) on the translation performance over the development sets. For all the language pairs, $\lambda = 0.5$ shows the best performance. In conclusion, the improvements observed for all three language pairs using distillation provide an answer to our second sub-research question RQ3.2. We have proposed distillation as a technique to efficiently leverage all noisy data and demonstrated that it outperforms a strong data filtering technique.

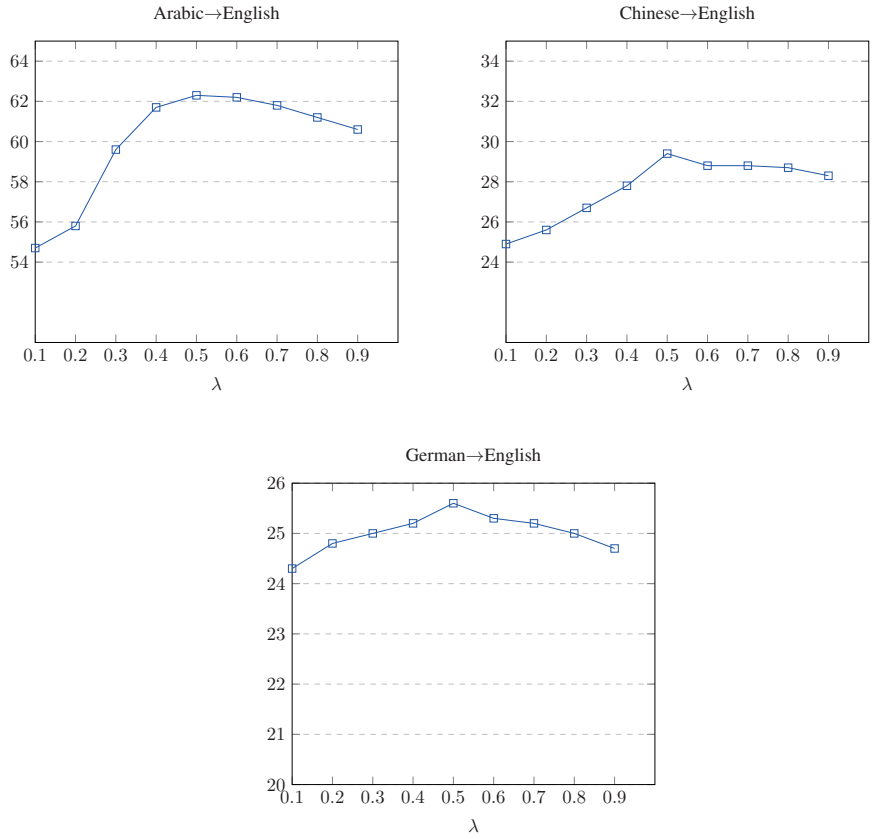


Figure 5.3: Variation in BLEU (y-axis) score for different values of λ (x-axis) for Knowledge distillation method. (a) Arabic→English, (b) Chinese→English and (c) German→English.

5.7 Recurrent vs non-recurrent architectures

Until now, all experiments discussed in this chapter used a recurrent NMT architecture. Moreover, the majority of the related work discussed in Section 5.2 is also focused on the exploration of noisy data with recurrent NMT architectures. The possible effect of the model architecture on robustness against training noise has not yet caught the attention of the NMT research community. However, the architectures of recently proposed non-recurrent NMT models suggest that these models may be more capable of handling noise caused by poor translation equivalence between the sentence pairs in the training data. In this section, we briefly discuss our intuition behind this hypothesis. In Section 5.8 and Section 5.9, we describe the experimental settings and conduct experiments to compare the performance of recurrent and non-recurrent NMT models when trained in the presence of noisy data.

As discussed in Section 5.3, Khayrallah and Koehn (2018) analyzed the Paracrawal corpus and categorized the noise into different categories and reported that around 41% of noisy samples are misaligned sentences due to faulty document or sentence alignment. Re-consider the example in Table 5.8 taken from the German→English Paracrawl corpus. The fragments or words marked red in the source sentence have no correspondence on the target side. While optimizing for the generation of a given target, these fragments or words do not contribute any features or information towards the generation of the targets and hence can be considered noise.

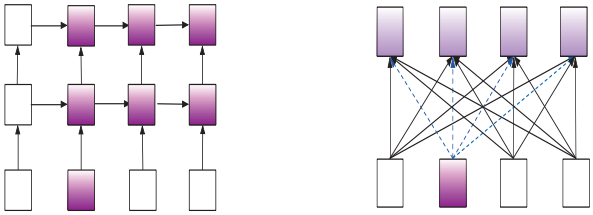
Table 5.8: Noisy sentence pair example from Paracrawl (De-En). Fragments in red in the aligned source translation have no corresponding fragment in the target sentence. **Src** = Source sentence in bitext, **Tgt** = Target sentence in bitext. **Human** = Human annotated translation of source sentence **Src**.

German→English (Paracrawl)	
Src:	Der Elektroden Schalter KARI EL22 dient zur Füllstandserfassung und -regelung von elektrisch leitfähigen Flüssigkeiten .
Tgt:	The KARI EL22 electrode switch is designed for the control of conductive liquids .
Human:	The electrode switch KARI EL22 is used for level detection and control of electrically conductive liquids.

Due to the sequential processing within recurrent architectures, the hidden state representation corresponding to each input word is directly affected by the representation of all previous words. Hence, the presence of noisy words or fragments in the sentence may equally affect the representation corresponding to each subsequent word, as shown in Figure 5.4a. Although the decoder attention weighs the importance of each hidden layer output (of the encoder) for the generation of a target word, these representations themselves may be noisy due to the presence of an incorrect fragment in previous positions and thus may result in possibly incorrect feedback.

On the other hand, in the non-recurrent Transformer model (Vaswani et al., 2017), the encoder processes the input *non-sequentially* through a self-attention mechanism by computing a weighted average of all surrounding input representations. That is,

every hidden layer node has direct access to the vector representation of all the previous positions individually, and the self-attention mechanism can learn to assign higher weights to words or segments according to their relatedness to the current position. As a result, an uninformative or noisy segment in previous positions would be assigned a low attention weight and thus would be less affected by noisy segments, as shown in Figure 5.4b.



(a) Recurrent NMT (b) Transformer NMT

Figure 5.4: Recurrent NMT vs Transformer. Purple boxes represent a noisy word or fragment.

Tran et al. (2018) compared recurrent and non-recurrent architectures with regard to their ability to model hierarchical structures. They found that recurrent architectures are better at capturing hierarchical information required for natural language understanding tasks such as subject-verb agreement due to the sequential processing of the input. As an alternative observation, Tang et al. (2018) demonstrated that Transformers perform better at extracting semantic features and hence excel at word sense disambiguation tasks. Nevertheless, Transformers have demonstrated comparable or even better performance for machine translation than recurrent networks (Vaswani et al., 2017). This suggests that for the translation task, an attention-based composition of important features could provide better performance as compared to a sequential carry-over of information through each time-step. This attention-based direct composition may reduce the influence of an uninformative word. However, this reasoning is only intuitive, and in this chapter, our goal is not to provide a detailed investigation of the hidden state representation of recurrent and non-recurrent models but to focus on the empirical comparison of the performance of the architectures when trained with noisy bitexts.

As already discussed in Section 5.1, the "Parallel corpus filtering" shared task was proposed at WMT-2018 aiming at filtering of sentence pairs of high quality from a noisy German→English web-crawled corpus. Submitted filtered bitexts were evaluated by the organizers according to the performance of SMT and RNN-based NMT systems trained on these filtered bitexts. The best performing systems showed performance equivalent to models trained on high-quality bitext. However, to the best of our knowledge, the impact of noisy training data on non-recurrent architectures has not been evaluated yet. In addition, only two ((Ash et al., 2018) and Sánchez-Cartagena et al. (2018)) of the 18 submissions in the shared task reported their own results for a Transformer model trained on the filtered bitext from Paracrawl. Therefore, there is an important question

that must be investigated in the context of the noisy data: "Does noisy training data have a degrading effect on non-recurrent NMT architectures similar to the effect observed for recurrent NMT?"

In the next section, we evaluate the impact of translation performance of recurrent NMT models as compared to Transformer models for three language pairs, when trained on combined high-quality and noisy bitexts.

5.8 Experimental setup for comparison of recurrent and non-recurrent NMT

5.8.1 Datasets

For the comparison of recurrent NMT and Transformer models, we use the clean and noisy training data sets in the same way as described in Section 5.5.2. We evaluate the comparison experiments on NIST-MT08 and MT09 for Arabic→English, MT06 for Chinese→English, WMT-16, and WMT17 test sets for German→English.

5.8.2 Model and parameters

To compare RNN and Transformer performance, we train the models using the OpenNMT-python toolkit[4] which provides implementations of RNN based sequence-to-sequence NMT as well as the Transformer architecture. For RNN based NMT, the encoder is a bidirectional LSTM, and the decoder is a unidirectional LSTM with input-feeding (Luong et al., 2015). The encoder and decoder both have 2 layers of dimension 1024. Gradients are optimized using *Adam* with an initial learning rate of 0.001 and a decay rate of 0.5 and label smoothing of 0.1. We use a fixed batch size of 64. For Transformer, we use 6 hidden layers, and 8 self-attention heads along with positional encodings and embedding and hidden size are each 512. We use the *Adam* optimizer with an initial learning rate of 2, a second moment decay rate (beta2) of 0.998, decay method *Noam*, label smoothing of 0.1 and 8,000 warmup steps. Each batch includes a maximum of 4096 tokens. Both RNN and Transformer are trained for a maximum of 400k steps and perplexity and BLEU on dev-sets are evaluated after every 20k steps.

5.9 Evaluation and results for comparison experiments

In this chapter, we asked the following third sub-question:

RQ3.3 *What is the relative variation in performance of recurrent vs. non-recurrent NMT models when noisy data is added to the training pool?*

As shown in Table 5.9 and 5.10, for most test sets, there is a substantial difference between the performance of RNN and Transformer models for both Arabic→English and

[4]https://github.com/OpenNMT/OpenNMT-py

Chinese→English. Adding the noisy comparable data to the training data pool degrades the performance for recurrent NMT up to -2 BLEU for Arabic→English and -1.1 BLEU for Chinese→English. Note that the size of noisy data for Chinese→English is equal to that of clean data, whereas, for Arabic→English, it is approximately four times that of the clean data. However, the drop is significant for both languages for recurrent NMT.

On the other hand, for both language pairs, adding the same noisy data for training Transformer improves the performance significantly: up to $+1.3$ BLEU for Arabic→English and $+1.9$ BLEU for Chinese→English.

Table 5.9: RNN vs Transformer experiments for **Arabic→English**; **Prl** = LDC clean data, **Noisy** = Comparable data from ISI bitext; **RN** = Recurrent NMT, **TF** = Transformers. ▲/▼ represent statistically significant differences compared to baseline at $p < 0.01$.

	MT05		MT08		MT09	
	RN	TF	RN	TF	RN	TF
Prl	57.7	59.1	46.0	46.0	49.9	52.7
Prl+Noisy	55.2▼ (-2.2)	60.2▲ (+1.1)	44.2▼ (-1.8)	**48.6**▲ (+2.6)	47.9▼ (-2.0)	**54.0**▲ (+1.3)

Table 5.10: RNN vs Transformer experiments for **Chinese→English**; **Prl** = LDC clean data, **Noisy** = Comparable data from ISI bitext; **RN** = Recurrent NMT, **TF** = Transformers. ▲/▼ represent statistically significant differences compared to baseline at $p < 0.01$.

	MT05		MT06	
	RN	TF	RN	TF
Prl	28.8	36.7	27.5	31.6
Prl+Noisy	27.7▼ (-1.1)	39.4▲ (+2.7)	26.7▼ (-0.8)	**33.5**▲ (+2.9)

In Table 5.11, we observe that for German→English, adding either the randomly sampled bitext from Paracrawl or the filtered bitext subsampled by one of the best filtering methods (Junczys-Dowmunt, 2018) lowers the performance by -1 BLEU for recurrent NMT. The results in Table 5.11 show that noisy corpora, even when sampled with the best-reported filtering technique, fail to provide any additional gains over the RNN-based baseline trained on high-quality data. On the other hand, Transformer models seem to benefit slightly ($+0.2$) from the randomly sampled bitext and substantially from the filtered bitext ($+1.2$).

Our observations for the three language pairs also suggest that the relative drops in performance of recurrent NMT and the gains of the Transformer model are correlated with the amount and the quality of the added noisy data as well as the size of the clean data. The respective gains and drops for RNN and Transformer are higher for Arabic→English and Chinese→English, which have a much smaller amount of clean

Table 5.11: RNN vs Transformer experiments for German→English; **Prl** = Clean WMT data, **Para**$_{rn}$ = Randomly sampled noisy Paracrawl bitext, **MSR**$_{filt}$ = Filtered bitext as explained in Table 5.4.

	newstest13		newstest16		newstest17	
	RN	TF	RN	TF	RN	TF
Prl	25.8	27.8	30.0	33.4	26.0	30.0
Prl + Para$_{rn}$	24.8$^{\blacktriangledown}$	27.9$^{\blacktriangle}$	29.0$^{\blacktriangledown}$ (-1.0)	33.8$^{\blacktriangle}$ (+0.4)	25.0 (-1.0)	30.2$^{\blacktriangle}$ (+0.2)
Prl + MSR$_{filt}$	24.5$^{\blacktriangledown}$	28.5$^{\blacktriangle}$	28.7$^{\blacktriangledown}$ (-1.3)	**35.8**$^{\blacktriangle}$ (+2.0)	25.0 (-1.0)	**31.2**$^{\blacktriangle}$ (+1.0)

data as compared to that of German→English. The above results do indicate that there are significant differences between the relative performances for recurrent NMT and Transformer models when the training corpus includes low-quality bitext.

5.10 Conclusion

In this chapter, we explored the utility and effect of noisy training data for neural machine translation. Our experiments show that depending on the size of the noisy data, the performance of a recurrent NMT model may suffer significant degradations. We evaluated the effect of noisy data for three language pairs with different varieties of noisy data. Further, most of the previous research on using noisy data for NMT has been limited to data selection or filtering techniques. Besides being expensive in terms of filtering time, these methods are limited due to their assumption that a sentence pair ranked lower with respect to various comparability metrics has no benefit for learning any important features and should be completely discarded. Our experiments demonstrated that as a result, the best filtering technique is limited in terms of the additional improvements when noisy data is used in conjunction with clean data. In view of the evaluation of these previous research, we addressed our main research question in this chapter:

RQ3 *How do noisy training corpora affect the performance of NMT and how can we leverage them to improve NMT performance?*

To answer this question, we first proposed a data cleaning technique whereby we clean the noisy training corpus through a back-translation procedure. In this approach, instead of changing the target labels, we regenerated the source sentences using an NMT model trained on the clean data in reverse of the intended direction. Our experiments demonstrated that the back-translation strategy provides a cleaner version of the training corpus as compared to the original noisy data. However, it is expensive due to beam search based decoding of large amounts of the training corpus. Moreover, it provides only moderate gains. In view of the above findings, we proposed distillation as a remedy to efficiently leverage the noisy data for NMT, where we train a primary NMT model on the combined training data with knowledge distillation from the teacher network trained on the clean data only. Our experiments show that distillation can successfully

utilize low-quality comparable data resulting in significant improvements as compared to training directly on the noisy data. Moreover, it outperforms not only the data filtering techniques but also fine-tuning based regularisation. We observe consistent improvements for all three language pairs with different quantities and sources of the noisy data, which suggests that distillation can be beneficial regardless of the data size and source of the noise.

Finally, since all previous research for using noisy data for NMT is limited by only using recurrent NMT models, we proposed to evaluate a more recently proposed non-recurrent architecture, namely Transformers, in this context. Based on our intuition that due to recurrent carry forward of information through the sequence, recurrent NMT models are more susceptible to training noise, we aimed to empirically evaluate the performance of recurrent NMT vs. non-recurrent Transformers in terms of the impact of noisy data. Primary experiments for three language pairs show that recurrent NMT models suffer significant degradation when noisy data is added to the training data pool, whereas the Transformer model is considerably more robust against training noise and shows improvements over a baseline that is only trained on clean data.

In the next chapter, we shift the focus from improving the training methods to the improvement of model architectures for NMT. We propose modifications to the standard NMT model in order to enhance its capacity to capture important features.

6

Convolutional over Recurrent Encoder for Neural Machine Translation

6.1 Introduction

In the previous chapters, we have addressed problems that concern the quality and robustness of machine translation models and proposed solutions based on the transfer of knowledge in incremental steps. All previous chapters address the variation of source, domain, or quality of the training data. Chapter 3 addresses the robustness of training procedure for phrase-based MT, Chapter 4 addresses the variation of domains, while Chapter 5 addresses variation of data quality. However, one other way of exploring the transferability of models for NMT is to combine the capacity of different model architectures. Therefore, diverging slightly from the research topics discussed in the previous chapters, in this chapter, we propose a new model architecture for NMT to improve the capacity of the model to reliably learn the relevant features in order to improve NMT performance. Specifically, we propose a combination of two well-known neural network architectures, i.e., recurrent neural networks (RNNs) and convolutional neural networks (CNNs), and investigate whether such a combination can model the relationship between source and target tokens more reliably than a vanilla RNN based NMT architecture. This addresses our final research question RQ4:

RQ4: *What are the limitations of using only recurrent layers towards effective modeling of source sentences in NMT and can NMT performance be improved by the addition of convolutional layers?*

We have discussed the established architectures for NMT in Chapter 2. However, the convolutional sequence-to-sequence model (Gehring et al., 2017b) and fully attentional model (Vaswani et al., 2017) have been proposed concurrently to the research conducted in this chapter. Prior to the invention of these two models, the dominating architecture in NMT research has been the recurrent neural network-based encoder-decoder framework of Bahdanau et al. (2015) in which a recurrent neural network (RNN) called 'encoder' converts a source sequence into a high-dimensional representation. Then another RNN called 'decoder' generates a target sentence word-by-word based on the source representation and target history. Besides machine translation, RNNs have shown

promising results in modeling various other NLP tasks such as language modeling (Mikolov et al., 2010) and text similarity (Mueller and Thyagarajan, 2016). The strength of using RNNs for language processing lies in their ability to recurrently maintain a history for a long input sequence, thus capturing the long-distance dependency, which is an important phenomenon of natural language texts.

Although modeling sequences using the recurrence property is important for most NLP tasks, there is a critical limitation in relying solely on the strengths of an RNN. In an RNN, at each timestep, the encoder output is a global representation in which the information about the current word and the previous history is represented compactly. Although RNNs effectively model interdependence of words, they cannot capture phrases without prefix context and are often biased towards last words in the final vector (Wen et al., 2016).

In this chapter, we propose to modify the RNN encoder-decoder framework by adding multiple convolutional layers on top of an RNN encoder output. Since CNNs apply to a fixed-size window of the input sentence, at each layer, each output represents a relatively uniform composition of information from multiple words. This provides effective guidance to the network to focus on the relevant parts of the source sentence. At the same time, sequence-to-sequence modeling, as in RNNs, is necessary to capture the long-distance dependencies between segments of the source sentence itself. Thus, in our model, a convolutional encoder complements the standard RNN encoder. We evaluate the performance of our proposed model in order to answer sub-research question RQ4.1:

RQ4.1 *Can a combination of recurrent and convolutional layers for the encoder provide the model with improved effective guidance to focus on relevant information in the source sequence?*

We conduct experiments on multiple language pairs to show that the addition of convolutional layers on top of recurrent layers of the encoder provides additional improvements in NMT performance as compared to a vanilla model with only recurrent layers. However, it can be argued that the observed improvements might not be due to the capability of convolutional layers but merely due to the increased number of parameters. This is also the concern of our second sub-research question RQ4.2:

RQ4.2 *Are the improvements observed with additional convolutional layers due to properties of convolutions or merely due to the increased number of parameters?*

In order to empirically validate the improved capacity of the proposed model, we train a deep RNN model by simply increasing the number of recurrent layers in the baseline model and compare its performance with the proposed model.

We first briefly discuss the properties of RNNs, the neural MT framework of Bahdanau et al. (2015), and convolutional neural networks in Section 6.2 and subsequently discuss the related work on convolutional neural networks in machine translation in Section 6.3. We introduce our model in Section 6.4 and discuss its details. Experiments and results are discussed in Sections 6.5 and 6.6, respectively.

6.2 Background

6.2.1 Recurrent neural network

Given a sequence $[x_1, x_2, \ldots, x_n]$ of length ns, at any timestep i, an RNN represents the hidden state output as function of the previous hidden state h_{i-1} output and the current input x_i:

$$h_i = f(h_{i-1}, x_i). \tag{6.1}$$

Here, f is commonly a nonlinear function. Thus RNNs represent a sequence as a vector by a function of the previous history and current input. It is this recurrence property of RNNs that makes them capable of capturing larger contexts and, therefore, long-distance dependencies commonly observed in variable-length texts.

A common problem observed while training RNNs is the decay of gradients over long sequences. To resolve this problem, Hochreiter and Schmidhuber (1997) proposed long short-term memory networks (LSTMs), which use input, output, and forget gates to control the amount of information that can pass through a cell unit in the RNN.

6.2.2 Baseline model

We have discussed various NMT architectures in detail in Chapter 2. In order to explain our proposed model, we briefly revisit the baseline model that we use for the experiments in this chapter.

We employ an NMT system based on (Luong et al., 2015) as discussed in Section 2.3. The encoder is a multi-layer recurrent network that converts an input sentence $[x_1, x_2, \ldots, x_n]$ into a sequence of hidden states $[h_1, h_2, \ldots, h_n]$:

$$h_i = f_{enc}(x_i, h_{i-1}). \tag{6.2}$$

Here, f_{enc} is an LSTM unit. The decoder is another multi-layer recurrent network which predicts a target sequence $y = [y_1, y_2, \ldots, y_m]$. Each word y_j in the sequence is predicted based on the last target word y_j, the current hidden state of the decoder s_j and the context vector c_j. The probability of the sentence is modeled as the product of the probability of each target word:

$$p(\mathbf{y}) = \prod_{j}^{m} p(y_j|y_1, \ldots, y_{j-1}, \mathbf{x}) = \prod_{j}^{m} g(y_j, s_j, c_j). \tag{6.3}$$

where g is a multi-layer feed forward neural network with nonlinear transformation and a softmax layer that generates the probability of each word in the target vocabulary. The end-to-end network is trained by maximizing the log-likelihood over the training data. In Equation 6.3, s_j is the decoder hidden state generated by LSTM units similar to the encoder:

$$s_j = f_{dec}(s_{j-1}, y_{j-1}, c_j). \tag{6.4}$$

The context vector c_j in turn is calculated using an attention mechanism (Luong et al., 2015) as the weighted sum of annotations of the encoder states h_i:

$$c_j = \sum_{i=1}^{n} \alpha_{ji} h_i, \tag{6.5}$$

where α_{ji} are attention weights corresponding to each encoder hidden state output h_i calculated as follows :

$$\alpha_{ji} = \frac{\exp(z_i)}{\sum_{k=1}^{n} \exp(z_k)}. \tag{6.6}$$

Activations $z_k = a(s_{j-1}, h_k)$ are calculated by using a context function such as the dot product between the current decoder state s_{j-1} and each of the hidden states of the encoder h_k.

In order to reduce the memory requirement for the softmax operation for a large number of words, the source and target vocabularies are usually limited to a fixed number of most frequent words. The translation is performed by a simple left-to-right beam search algorithm, which maintains a small set of the b best hypotheses for each target word. A hypothesis is complete as soon as the end of sentence <EOS> symbol is produced or the maximum number of timesteps has been reached. A more detailed description of the decoding algorithm can be found in (Sutskever et al., 2014).

6.2.3 Convolutional neural networks

Unlike recurrent neural networks, which are applied to a sequence of inputs, feeding the hidden layer from one time step to the next, convolutional neural networks apply filters of a fixed length over a window of inputs and generate outputs of fixed size. As discussed by Kim (2014), a narrow convolution operation involves applying a filter θ over a window of w inputs in order to generate a new feature; w is known as the width of the filter or kernel size. The new feature CN_i applied to an input window around x_i to x_{i+w} is then defined as:

$$CN_i = \sigma(\theta \cdot x_{i-[(w-1)/2]:i+[(w-1)/2]} + b). \tag{6.7}$$

This feature extraction capability of CNNs makes them suitable for image processing. In NLP, CNNs have been used for tasks such as sentence classification (Kim, 2014).

6.3 Related work

Although recurrent neural networks are very popular for many NLP tasks, CNNs have also been used for tasks such as text or sentence classification (Kim, 2014), sentiment analysis (dos Santos and Gatti, 2014), document modeling (Tang et al., 2015) and sentence modeling (Kalchbrenner et al., 2014), where specific features such as n-grams and phrases are more important than location-specific or grammatical features of the sentence.

Similarly, the standard approach to neural machine translation is the RNN-based encoder-decoder network. However, there have been various attempts towards using convolutional networks in neural MT. The first attempt to use convolutional networks for generating sentence representation for MT is (Kalchbrenner and Blunsom, 2013). The authors proposed to condition a recurrent neural language model of the target sentence on a source sentence vector generated by a convolutional neural network. However, their experiments were limited to rescoring candidate hypotheses generated by a phrase-based MT system. Moreover, the reported results were later significantly

outperformed by recurrent models (Cho et al., 2014). The first attempt to train a CNN based end-to-end NMT framework is due to Cho et al. (2014). They fully replace the recurrent encoder with a gated convolutional network that is recursively applied to the input sequence until it outputs a single fixed-length vector. However, their experiments demonstrate that the translation performance of such a network cannot surpass that of a fully recurrent encoder.

Gehring et al. (2017a) also proposed a similar architecture where the recurrent encoder is again fully replaced by a deep convolutional neural network. An important feature in their architecture is the use of a position embedding that encodes the position of each word in the source sentence. Their experiments demonstrate that while translation performance of the network is improved by using a very deep convolutional network, without the position embeddings, quality drops substantially below the standard RNN/LSTM encoder baseline. This implies that a CNN encoder by itself with simple word embeddings alone cannot encode position-dependent features which are otherwise efficiently captured by an RNN encoder. Another approach using convolutional networks in neural MT is the *ByteNet* system by Kalchbrenner et al. (2016). They replace both the encoder and decoder with dilated convolutional networks stacked on each other.

All of the above approaches either aim to fully replace the recurrent encoders with convolutional encoders with which they aim to reduce the complexity of the network and the training speed or to address the variable lengths of input sequences. In order to achieve performance comparable to RNN encoders, these approaches have to employ different mechanisms such as position embeddings to effectively capture the long-distance dependencies and position-dependent features.

Another approach which has shown the strength of convolutional networks as an additional feature for phrase-based MT is by Meng et al. (2015). They show improvements over a standard phrase-based MT by encoding the source sentence with a convolutional network and using it in a neural language model as an additional feature.

A related line of research that combined CNN and RNN is the character-level NMT (Lee et al., 2017a). However, their main idea is to model words as a combination of characters using convolutions and then feed the output as word embeddings to the RNN encoder. Their aim is to avoid having to limit the vocabulary by character modeling. However, in this chapter, our aim is to capture higher-level features on the source side; therefore, we apply the convolutional layers on top of the recurrent layer output.

Such a combination of RNNs and CNNs has successfully been used in various tasks such as saliency detection for image recognition (Tang et al., 2016), document modeling (Tang et al., 2015) and music classification (Choi et al., 2016).

6.4 Convolutional over recurrent (CoveR) model

As discussed in Section 6.2.2, when using the standard RNN framework, the context vector is a weighted sum of the encoder hidden states h_i. The attention weights as in Equation 6.6, are also calculated by a similarity function between the decoder state s_j and encoder states h_i. The attention weights mainly score how well the inputs around position *i* and the output at position *j* match (Bahdanau et al., 2015). Since in a

unidirectional encoder, each of these vectors h_i is a compact summary of the source sentence up to word *i*, the previous or future context available to the alignment function is only given by these compact global representations. We propose that instead of relying only on these single recurrent outputs, a composition of multiple hidden state outputs of the encoder can provide the attention function with additional context about the relevant features of the source sentence.

In order to do this, we apply multiple layers of fixed-size convolution filters over the output of the RNN encoder at each time step. As shown in Figure 6.1, for our model the input to the first convolution layer is the hidden state output of the RNN encoder. Thus CN_i^1 is defined as:

$$CN_i^1 = \sigma(\theta \cdot h_{i-[(w-1)/2]:i+[(w-1)/2]} + b). \tag{6.8}$$

At each layer, we apply filters repeatedly to the original input sentence length. Each filter is of a fixed width k. Note that the length of the output of the convolution filters reduces depending on the input length and the kernel width. In order to retain the original sequence length of the source sentence we apply padding at each layer. That is, for each convolutional layer, the input is zero-padded so that the output length remains the same. The output of the final convolution layer L is a set of vectors $[CN_1^L, CN_2^L, \ldots, CN_n^L]$ generated by multiple convolution operations. The modified context vectors c_i' are then computed similar to c_i using an attention mechanism:

$$\alpha_{ji} = \frac{\exp(a(s_{j-1}, CN_i^L))}{\sum_{i=1}^{n} \exp(a(s_{j-1}, CN_i^L))}, \tag{6.9}$$

$$c_j' = \sum_{i=1}^{n} \alpha_{ji} CN_i. \tag{6.10}$$

Finally, the decoder is provided with the context vectors c_i' as follows:

$$p(y_i|y_1, \ldots, y_{i-1}, \mathbf{x}) = g(y_i, s_i, c_i'). \tag{6.11}$$

Note that each of the vectors CN_i^L now represents a feature produced by multiple kernels over h_i. Thus each CN_i represents a wider context as compared to h_i.

It is a common practice to use pooling along with convolutional filters in order to down-sample the features (Kim, 2014). However, since in the proposed model, we want to widen the context of the encoder output while still retaining the information represented in the RNN output h_i, and also retaining the original sequence length, we do not apply pooling in our model.

With increasing depth of the network, training of the network becomes unstable. In order to ease and stabilize training with multiple layers, we use residual connections (He et al., 2016) between the input and output of each convolutional layer.

6.5 Experimental set-up

6.5.1 Data

The goal of the experiments in this chapter is to evaluate the performance of our proposed model against a standard recurrent NMT system in which both the encoder and decoder

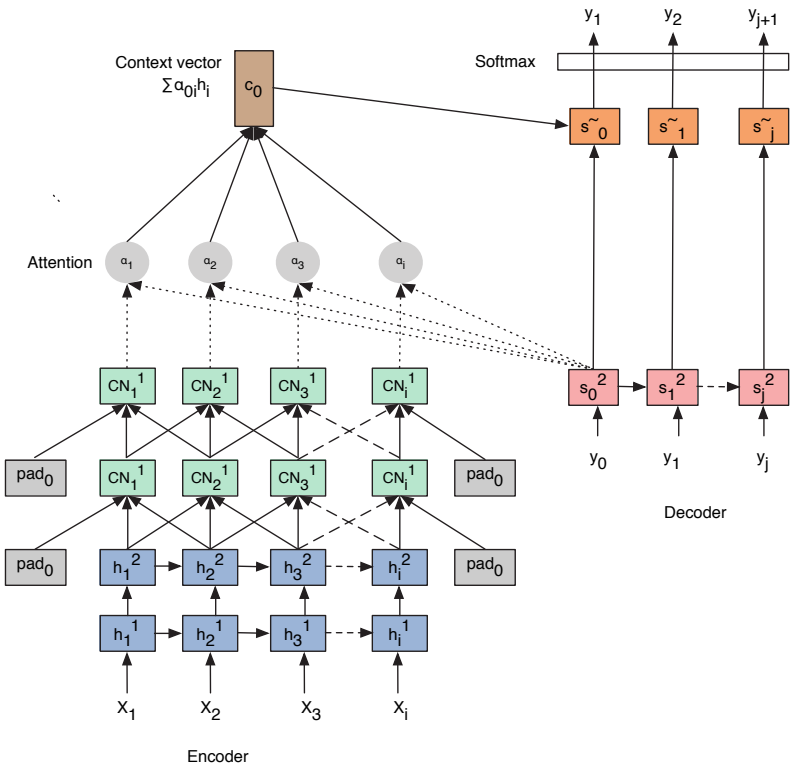


Figure 6.1: Convolution over Recurrent model.

are recurrent neural networks. The proposed model uses additional convolutional layers over the recurrent layers of the baseline NMT model. However, the overall performance of the convolutional networks is usually dependent on two hyper-parameters: the optimal number of convolutional layers and the kernel-width of the convolutions. Therefore, we first experiment by varying these two parameters on a small English→German corpus known as the TED-talks bitext (IWSLT 2013) (Cettolo et al., 2012). We use a combination of the official test sets for IWSLT 2011, 2012, and 2013 as an evaluation set for these experiments.

After determining the optimal number of parameters, in the second set of experiments, we evaluate the proposed model against the baseline for four language pairs/directions: English→German, German→English, Arabic→English, Chinese→English and Romanian→English. For the English↔German experiments, we use the training data provided for WMT-2015 (Bojar et al., 2015). The training data provided for the task consists of approximately 4.2 million sentence pairs.
For Arabic→English, we use a collection of parallel sentences taken from the following data sources released by the Linguistic Data Consortium: LDC2006E25, LDC2004T18, several GALE corpora, LDC2004T17, LDC2005E46, LDC2007T08, and LDC2004E13.

Table 6.1: Training and evaluation sets for all language pairs. Training size = Number of sentence pairs in training data.

Language Pair	Training size	Development set	Test set
English→German	4M	newstest13	newstest14, newstest15
German→English	4M	newstest13	newstest14, newstest15
Arabic→English	1.25M	NIST-MT05	NIST-MT09
Chinese→English	2.15M	NIST-MT-05	NIST-MT08
Romanian→English	550K	Random (2000)	newstest16

This results in a parallel corpus of approximately 1.2 million sentence pairs. Similarly, for Chinese→English, we use sentence pairs from the LDC catalog (LDC2003E14, LDC2005T10, and LDC2002E18, LDC2007T09), resulting in approximately 2.1 million sentence pairs.

For Romanian→English, we use the training data released for the news translation shared task at WMT-2016 (Bojar et al., 2016), which is mainly Europarl version-8 data (approximately 550K sentence pairs). We reserve a random sample of 2000 sentences as a development set and use the test set provided by WMT for evaluation.

For English↔German, we use WMT-newstest2013 as a development set and WMT-newstest2014 and WMT-newstest2015 as test sets. For Arabic→English, we use NIST MT05 as the development set and NIST MT09 as the test set. Similarly, for Chinese→English, NIST MT05 is used as a development set and NIST MT08 as the test set. Table 6.1 summarizes the training, development, and test sets for all language pairs. We keep sentences with a maximum sequence length of 80 tokens on both the source and target side. Results are reported in terms of case-insensitive BLEU-4 (Papineni et al., 2002).

6.5.2 Baselines

We train a baseline NMT system based on (Luong et al., 2015) implemented using the Torch deep learning framework. It is a two-layer unidirectional LSTM encoder-decoder with an attention (dot product) mechanism. Both the encoder and decoder have input embedding and hidden layer sizes of 1000. As is common practice, we limit the vocabulary sizes to 60k for the source and 40k for the target side. Parameters are optimized using stochastic gradient descent. We set the initial learning rate as 1 with a decay rate of 0.5 for each epoch after the 5th epoch. Model weights are initialized uniformly within $[-0.02, 0.02]$. A dropout value of 0.2 is applied to each layer. We train for a maximum of 20 epochs and decode with a standard beam search with a beam size of 10. All models are trained on NVIDIA Titan-X (Pascal) GPU devices.

6.5.3 CoveR model

As discussed in Section 6.5.1, we first experiment with varying the number of convolutional layers and kernel width. We add multiple convolution layers on top of the output

of the second RNN layer of the encoder. Note that similar to the baseline system, the RNN decoder has the same number of layers as the RNN encoder, i.e., two. The number of filters at each layer is the same as the input sequence length. Each filter operates on a window of (k =kernel width) consecutive inputs and generates a single output with a dimension equal to the input. Thus at each layer, the output sequence length is reduced as compared to the input, as shown in Figure 6.1. In order to retain the full sequence length, we apply $k - 1$ zero-paddings on both sides of the input. All other optimization parameters are the same as for the baseline.

6.5.4 Deep RNN encoder

In order to verify that the improvements achieved by the proposed model are due to the convolutions and not just because of the increased number of parameters, we also compare our model to another RNN baseline with an increased number of recurrent layers for the encoder. For the second set of experiments, we train a deeper RNN model with additional layers, which equals the number of optimal convolutional layers that are determined in the first set of experiments. For this deep NMT system, the number of layers in the decoder remains the same as for the baseline, i.e., 2. The initial states of the decoder layers are initialized through a non-linear transformation of all layers of the encoder RNN. This is done by concatenation of the final states of all n layers of the encoder resulting in a vector of size $n \times D$ (D is the dimension of the hidden layer) and then projecting it down to size $2 \times D$ by a simple non-linear transformation and finally splitting it in two vectors of size D that is used to initialize each of the layers of the decoder.

6.6 Results

As discussed in Section 6.5.1, in the first set of experiments, we determine the optimal number of convolutional layers and kernel-widths. In order to do this, we fix the kernel-width as 3 and vary the number of convolutional layers from 1 to 4. Table 6.2 shows the performance variation in terms of BLEU for a varying number of convolutional layers. The reported results correspond to the best epoch on the validation set. As can be seen in Table 6.2, the performance of the model increases with an increase in the number of layers up to $n = 3$. For $n = 4$, we observe that the performance starts to decrease. Thus we consider $n = 3$ as the optimal number of layers.

After setting the number of layers to $n = 3$, we experiment with kernel-width k. Since, $k = 1$ implies only a window of one word, we start with $k = 3$. Moreover, since a square kernel operation allows for only odd-sized kernels, we experiment with values $k = 3$, $k = 5$, $k = 7$. As shown in Table 6.3, we observe that the best performance is achieved for $k = 3$ and increasing the kernel-width does not improve model performance. Therefore we consider number of layers $n = 3$ and kernel-width $k = 3$ to be optimal values for the rest of the experiments.

Tables 6.4 and 6.5 show the results for our English→German and German→English translations experiments respectively. The first column indicates the best BLEU scores on the development set (newstest13) for all three models after 20 epochs. Results are

Table 6.2: BLEU variation with respect to the number of convolutional layers. Kernel-width is fixed to 3. Results are reported on IWSLT combined test, as explained in Section 6.5.1. Results marked with ▲ are statistically significant at $p < 0.05$ over baseline.

Number of convolutional layers	BLEU (IWSLT-test)
$n = 0$ (Baseline)	16.3
$n = 1$	**17.1**▲
$n = 2$	**18.0**▲
$n = 3$	**18.9**▲
$n = 4$	**18.4**▲
$n = 5$	**18.1**▲

Table 6.3: BLEU variation with respect to kernel-width. The number of layers is fixed to 3. Results are reported on IWSLT combined test as explained in Section 6.5.1.

Kernel-width	BLEU (IWSLT-test)
$k = 3$	18.9
$k = 5$	18.8
$k = 7$	17.6

reported on the newstest14 and newstest15 test sets. For English→German, our CoveR model shows improvements of $+1.1$ and $+0.5$ BLEU points, respectively for both test sets. Similarly for German→English, we observe up to $+0.5$ BLEU improvements for the deep RNN over the baseline and $+0.9$ improvement over baseline on the development set. Although the deep RNN encoder performs better than the baseline, the improvements achieved are lower than that of the CoveR model.

Table 6.4: English-German results for Baseline, Deep RNN and CoveR (proposed model). Results marked with ▲ are statistically significant at $p < 0.05$ over baseline.

	newstest13 (dev)	newstest14	newstest15
Baseline	17.9	15.8	18.5
Deep RNN encoder	18.3 +0.4	16.2 +0.4	18.7 +0.2
CoveR	**18.5**▲ +0.6	**16.9**▲ +1.1	**19.0**▲ +0.5

For Arabic→English, as shown in Table 6.6, we observe up to $+1.3$ BLEU improvements on the evaluation set for the CoveR model while only up to $+0.4$ for the deep RNN. Similarly, in Table 6.7, for Chinese-English, we observe an improvement of $+0.5$ with the deep RNN model, and an improvement of up to $+2.1$ with the CoveR model. Finally, as shown in Table 6.8 for Romanian→English, the deep RNN yields up to $+0.3$ improvements while up to $+0.8$ BLEU is achieved for CoveR model.

In conclusion, we can see that the proposed CoveR model shows significant im-

Table 6.5: German-English results for Baseline, Deep RNN and CoveR (proposed model).

	newstest13 (dev)	newstest14	newstest15
Baseline	22.4	20.5	21.0
Deep RNN encoder	22.6 +0.2	20.9 +0.4	21.5 +0.5
CoveR	**22.9**▲ +0.5	21.3▲ +0.8	**21.9**▲ +0.9

Table 6.6: Arabic-English results for Baseline, Deep RNN and CoveR (proposed model).

	MT05	MT09
Baseline	44.4	34.8
Deep RNN encoder	44.8 +0.4	35.0 +0.2
CoveR	**46.2**▲ +1.8	**36.1**▲ +1.3

Table 6.7: Chinese-English results for Baseline, Deep RNN and CoveR (proposed model).

	MT05	MT08
Baseline	12.9	8.9
Deep RNN encoder	15.5△ +2.6	9.4 +0.5
CoveR	**16.0**▲ +3.1	**10.9**▲ +2.1

Table 6.8: Romanian-English results for Baseline, Deep RNN and CoveR (proposed model).

	(development set)	newstest16
Baseline	49.2	13.4
Deep RNN encoder	47.1	**13.6** +0.3
CoveR	**49.9** +0.7	**14.2**▲ +0.8

provements over the baseline. Moreover, it also outperforms a deep RNN model with an equivalent number of recurrent layers. This implies that a combination of recurrent and convolutional layers performs better than recurrent layers only.

6.6.1 Qualitative analysis and discussion

Table 6.3 shows some translation examples produced by the baseline systems and our CoveR model for three language pairs. A general observation is the improved translations by our model over the baseline with regard to the reference translations. More specifically, Example 1 shows instances where the baseline suffers from incomplete coverage of the source sentence. One reason for such incomplete translations is the lack

Table 6.9: Translation examples. Words in bold show correct translations produced by our model as compared to the baseline. **Src** = Source sentence, **Ref** = Reference sentence, **Base** = Translation produced by baseline model. **Cover** = Translation produced by proposed (Cover) model.

1:	**English-German**
Src:	as the reverend martin luther king jr. said fifty years ago
Ref:	wie pastor martin luther king jr. vor fünfzig jahren sagte :
Base:	wie der martin luther king jr. sagte
Cover :	wie der martin luther king jr. sagte **vor fünfzig jahren** :
2:	**English-German**
Src:	he said the itinerary is still being worked out .
Ref:	er sagte , das genaue reiseroute werde noch ausgearbeitet .
Base:	er sagte , dass die strecke noch <unk> ist .
Cover:	er sagte , die **reiseroute** wird noch **ausgearbeitet** .
3:	**Arabic-English**
Src:	كما لقي ٥٢ شخصا مصرع هم و اصيب ١٤٢ ب جروح في انفجار سيارة مفخخة في اليوم ذات ه في مدينة النجف التي تبعد ٧٠ كلم عن كربلاء .
Ref:	a total of 52 people were killed and 142 wounded when a car bomb exploded in the same day in the holy city of najaf, 70 km from karbala.
Base:	on the same day , 52 people were killed and 142 injured when a car bomb exploded in the same day in the southern city of najaf .
Cover:	another 52 people were killed and 142 wounded in a car bomb attack on the same day in najaf , some **70 km** (60 miles) south of karbala .
4:	**Romanian-English**
Src:	însă un grup de oameni care au trăit pe străzi în copilărie au găsit un mod de a învăța o meserie și de a-și câștiga traiul .
Ref:	but one group of former street children have found a way to learn a skill and make a living .
Base:	but a group of people who lived on the streets have found a way to learn and win .
Cover:	however , a group of people who lived in the streets have found a way to learn a wealth and to **make a living .**

of coverage modelling which has been addressed by Tu et al. (2016) using coverage embeddings. We observe this problem frequently with the baseline model in instances where a specific word can signal completion of a sentence despite more words in the sequence remain to be be translated.

These words can cause the premature generation of the end-of-sentence *EOS* symbol. Since the beam search decoding algorithm considers a hypothesis complete when the end of sentence is generated. In such instances search stops, aborting further expansions, ignoring the remaining words. For instance in Example 1 in Table 6.3, by relying on the attention mechanism, the baseline system generates the translation of *said* in English as *sagte* in German. This is possibly due to the reason that the model might give a preference to the generation of an end-of-sentence *EOS* symbol immediately following

the verb. On the other hand, for our CoveR model, a wider context is available to the model through convolutional layers from both directions signalling the presence of other words remaining in the input sentence, thus producing a more complete translation.

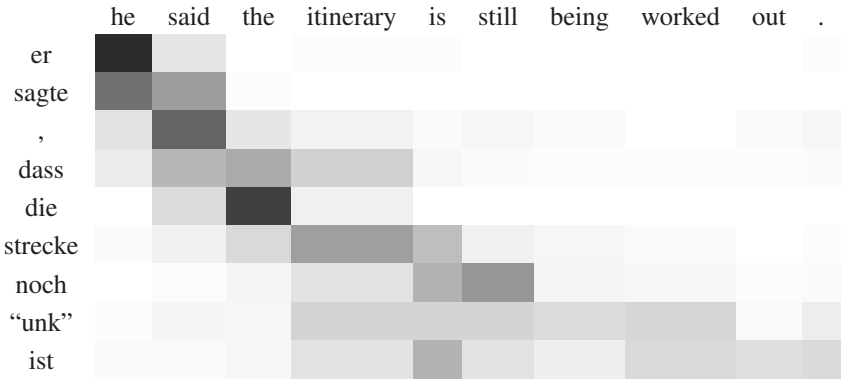


Figure 6.2: Attention distribution for Baseline

Figure 6.3: Attention distribution for CoveR model for Example 2 in Table 6.3

Another difference between the baseline model and our CoveR model can be observed in Example 2. Here, attention weights are distributed more uniformly among the source words. Specifically, for target position 6, as shown in Figure 6.3, the baseline model pays attention mainly to *'itinerary'* and *'is'* resulting in the generation of target word *'strecke'* which is a more common translation for the English word *'route'*. On the other hand, as shown in Figure 6.3, for the same position, the CoveR model pays attention to *'itinerary'* as well as the last three words *'being worked out'*. This allows for the generation of the correct target word *'reiseroute'*. Similarly, for both Arabic→English and Chinese→English, as shown in Example 3 and 4, respectively, the CoveR model generates more correct translations as compared to the baseline by

translating fragments which are otherwise dropped by the baseline model.

6.7 Conclusion

Motivated by the limitations of recurrent layers to capture the source context, we proposed a model that combines recurrent and convolutional layers for neural machine translation. Our main research question was to evaluate whether adding convolutional layers on top of the recurrent layers can increase the ability to capture features of the source sequence:

RQ4: *What are the limitations of using only recurrent layers towards effective modeling of source sentences in NMT and can NMT performance be improved by the addition of convolutional layers?*

The proposed model involves feeding outputs of the RNN encoder to multiple convolutional layers of fixed kernel size. We first experimented with varying the number of convolutional layers and kernel-width over a small training set and determined the optimal value of both parameters. After fixing these hyper-parameter values, we compared the proposed model to the baseline (recurrent only) model for five language pairs. Our experimental results demonstrated that the proposed model consistently performs better than the standard RNN encoder for all language pairs. This answers our first sub-research question:

RQ4.1 *Can a combination of recurrent and convolutional layers for the encoder provide the model with improved effective guidance to focus on relevant information in the source sequence?*

Further, we aimed to empirically verify whether the improvements demonstrated by the proposed model are a result of the increased number of parameters or due to the capacity of convolutional layers to better capture the source features. This is the premise of our second sub-research question:

RQ4.2 *Are the improvements observed with additional convolutional layers due to properties of convolutions or merely due to the increased number of parameters?*

To answer this question, we experimented with a model where the number of recurrent layers is equivalent to the total number of layers (recurrent and convolutional) in the proposed CoveR model. Our experimental results demonstrated that although for all language pairs, the deep-RNN model with the increased number of recurrent layers performs slightly better than the baseline model (with fewer layers), the CoveR model still outperforms the deep-RNN model. This implies that in the proposed CoveR model, the improvements are not merely due to the increased number of parameters. Further, a qualitative analysis of the translations of our model shows that CNNs capture the local context corresponding to each word more effectively while RNNs model the global information, thus capturing grammaticality and dependencies within the source sentence.

In the next chapter, we revisit all our research problems and discuss our solutions to these problems, along with the main findings of this thesis and possible future directions.

7 Conclusions

In this chapter, we summarize our discussions and the main findings of this thesis. We start with our main findings, followed by a review and discussions of our main research questions in Section 7.1. In Section 7.2, we discuss the proposed methods, experiments, and possible future directions.

7.1 Main findings

As discussed in Chapter 1, research in data-driven machine translation has to consider various problems such as model training, data selection, and domain adaptation. One of the most important challenges for data-driven machine translation is the limited availability of good-quality data in substantial amounts for various language pairs. However, training of high-performing MT systems is not only dependent on the quality and quantity of the data but also on the effectiveness of training strategies to learn robust models from all available training data. Standard training techniques assume that the training data instances are from identical domains and of high-quality. However, this is not always the case for different application scenarios, where the training data is often compiled from multiple sources with different domains and degrees of quality. This can result in low performance of the MT system due to domain bias, induction of low-quality features, etc. Therefore, there is a need to explore techniques that can improve the quality of models while making effective use of all training data. In this thesis, we addressed various problems such as compactness of translation models, domain bias, and performance degradation due to training noise. All these problems are sub-problems of a more general problem, which is the "ineffectiveness of standard training techniques to make the best use of training data." Therefore, while addressing different research questions, the underlying theme in this thesis is to address the question: "to what extent can incremental transfer of knowledge help improve the performance of machine translation models?"

In this thesis, we addressed four research questions that target various problems in phrase-based MT as well as neural MT. The solutions proposed to these problems revolve around techniques such as knowledge transfer across models and incremental refinement of model quality, which ultimately aim to improve the reliability of trained models. Below we revisit these research questions, proposed solutions, and the main findings from the experiments.

In Chapter 3, we addressed the problem of unreliable phrase segmentations obtained through heuristic extractions from word alignments across source-target sentence pairs. The main research question **RQ1** that we addressed in Chapter 1 is as follows:

RQ1: *How does heuristic training of translation models affect the performance of phrase-based MT and to what extent can alternative training strategies based on re-estimation of models improve phrase-based MT performance?*

The heuristic extraction technique for phrase-based MTs is solely based on an abstract definition of the consistency of phrases. It is devoid of any intuition about which phrases are more likely to be observed in high-quality translations when scores from different models are combined. As a result, a large number of unintuitive phrase pairs are extracted, that affect the size and quality of the models. We provided an analysis of how the forced decoding method leads to the re-estimation of incorrect phrase segmentations from noisy samples in the training data. We also discussed the limitations of the forced decoding method, which is incapable of re-estimating re-ordering models. Moreover, our final results in Chapter 3 demonstrated that the over-estimation of phrases in the heuristic method results in a very large size of the phrase translation model, thus increasing the memory requirements of the MT system.

As a remedy to heuristic extraction, we investigated if it is more beneficial to re-estimate the phrase segmentations from the best scoring translations of the training bitext (in terms of BLEU) instead of exact references. We proposed oracle-BLEU re-estimation as an alternative to heuristic extraction and forced decoding based re-estimation. Our proposal is based on the idea that due to the constraints of phrase-based decoding on hyper-parameters such as distortion-limit, phrase length, beam threshold, it can be very difficult and computationally expensive for forced decoding to generate the exact target sentence, especially in the case of noisy samples. Instead, it would be easier to reach the best possible translation in terms of BLEU.

We conducted experiments for Arabic→English translation in order to evaluate the performance of a model trained by the proposed oracle-BLEU re-estimation in comparison to standard heuristic training as well as forced decoding based re-estimation. Our results demonstrate that the proposed re-estimation technique provides better results than heuristic extraction and forced decoding. Moreover, it also reduces the phrase table size to just 3% of the initial size.

Further, the proposed method can also enable the re-estimation of re-ordering models, which is not possible with forced decoding. Therefore, it is critical to evaluate whether the re-estimation of models other than the phrase translation model can provide additional benefits to the overall translation performance. We conducted experiments for the re-estimation of re-ordering and Bilingual Language models (BiLm) and evaluated the performance of re-estimated models as compared to the baseline models. Our results demonstrated that the re-estimation of re-ordering models, especially the BiLm leads to additional improvements in translation performance. This implies that reinforcing those re-ordering decisions that lead to a better translation of the training data is beneficial for improving the translation performance of the SMT system.

Phrase-based MT is based on training each specialized model individually. Therefore, in this paradigm, the quality and domain of the training data can have isolated

effects on the reliability of the models trained from this data. As discussed in Chapter 3, various methods have been proposed to improve the quality of the trained models. In Chapter 3, we proposed one such method, which is a refinement of forced decoding.

However, these observations also raise questions about the reliability of neural machine translation models. As already discussed in Chapter 2, NMT architectures are based on end-to-end training of a single neural network, and the model parameters are the weights of the individual layers. Therefore, it is important to ask whether the reliability of neural network models can also be affected due to domain bias and low-quality of the training data, and if so, what are the possible remedies to improve the quality and reliability of these models? In Chapter 4, we addressed one specific problem related to the domain bias of the NMT models. We asked:

RQ2: *How can we apply fine-tuning to adapt NMT models to new domains while retaining the performance on the source (original) domain?*

It has been shown in the literature that NMT models are prone to domain bias (Koehn and Knowles, 2017). The performance of NMT is good if the domains of the train and test data are the same, but it tends to be significantly lower if the two domains are different. However, it is not always feasible to obtain large amounts of training data for all intended domains. Therefore it is important to explore techniques that can transfer knowledge across domains. *Fine-tuning* has been proposed in the NMT literature as a straightforward way to transfer models trained on large general domain data to specific application domains (Freitag and Al-Onaizan, 2016; Chu et al., 2017). However, *fine-tuning* suffers from a specific problem which is undesirable in practical settings. This problem is the degradation of NMT performance on the original or source domain after fine-tuning on the target domain. In Chapter 4, we addressed the question of retaining a consistent performance of the NMT model across the source and target domains when adapting the general domain model to specific domains.

We answered RQ2, in three sets of experiments. First, we evaluated the performance of a model trained on the general domain (news data) on three different target domains. Our experiments demonstrated that the performance of the general domain model is significantly lower than that of the in-domain model when evaluated on the in-domain test set. Moreover, given a reasonable size of in-domain data, the performance of a model trained on general domain data is substantially lower than training only on the in-domain data.

In the second step, our experiments had two evaluation aspects: First, we evaluated whether fine-tuning provides any significant improvements over the in-domain baselines when adapting the models to multiple domains. Second, we assessed whether the model adapted through fine-tuning can retain its performance on the general domain task. From the results observed for all domains, we concluded that fine-tuning does provide significant improvements as compared to training only on in-domain data, but the performance of the adapted model on the general domain tasks degrades drastically.

To address the problem of severe degradation, we proposed two modifications to the fine-tuning method for NMT in order to retain the performance on the source domain. Both methods are based on the idea of "knowledge distillation" (Hinton et al., 2014) where supervision from a teacher model prevents the fine-tuned model from forgetting

the information learned about the source domain. Based on experimental results for all domains, we concluded that both proposed approaches achieve performance comparable to vanilla fine-tuning while retaining performance on the source (general) domain. In conclusion, our experiments show that while domain bias is indeed a challenge for NMT, knowledge transfer techniques such as distillation can result in models that are robust across multiple domains.

Besides domain dissimilarity, one other aspect of the training data that can affect the performance of the NMT models is its quality. Training high-performing NMT systems require large amounts of high-quality parallel data. However, for many low resource languages, high-quality data is rarely available. For such language pairs, an easy alternative is to utilize slightly low-quality or noisy corpora. However, noisy corpora can have a negative effect on the performance of NMT systems. Therefore, in Chapter 5, we asked:

RQ3: *How do noisy training corpora affect the performance of NMT and how can we leverage them to improve NMT performance?*

In order to answer RQ3, we first experimented by training NMT systems in different settings, which include: training only on high-quality data, training only on noisy data, and training on a combination of noisy and high-quality data. We sampled from two different types of noisy bitexts (Paracrawl and ISI bitext) for three different language pairs and used the noisy data along with high-quality parallel data. Our experiments for three language pairs (Arabic-English, Chinese-English, and German-English) demonstrated that noisy data does negatively affect NMT performance as compared to the use of high-quality data only.

In the next step, we focused on possible remedies to reduce this negative effect. We discussed some well-known techniques, such as data selection (Axelrod et al., 2015; van der Wees et al., 2017), fine-tuning (Barone et al., 2017), and bootstrapping (Reed et al., 2014), which have been adapted from either phrase-based NMT or from noise handling techniques in the deep learning literature. We aimed to devise a method that can learn from all training data without any reliance on an external oracle to decide the quality of the training samples. Drawing inspirations again from the idea of knowledge distillation, we proposed that using supervision from a strong teacher model can help utilize all available data for training NMT models. In our technique, we first train a strong teacher model on available high-quality data. Next, we train a final model on the combined high-quality and noisy data while using supervision from the strong model. Our experiments demonstrated that distillation yields significantly better results than training directly on noisy data. Moreover, the proposed method significantly outperforms alternative techniques, such as data selection and fine-tuning. In conclusion, our experiments show that it is more beneficial to rely on the judgments of a strong model instead of an external oracle to decide on the quality of the samples.

Finally, we extended our analysis of the effect of noisy data for non-recurrent NMT architectures. Specifically, we proposed to evaluate what kind of effect does noisy training data has on performance of a non-recurrent architecture, namely Transformer models (Vaswani et al., 2017). Our experimental results for three language pairs show that Transformer models tend to be more robust against training noise. In our

experiments, transformer models not only resist degradation due to noisy data but even displayed slight improvements when noisy data is used in addition to high-quality data.

In conclusion, our experiments show that while noisy data has a negative effect on the performance of recurrent models, it can be utilized through knowledge transfer techniques such as distillation from a strong model. Moreover, more recent non-recurrent NMT architectures such as Transformer models already seem to be robust against training noise.

In Chapters 3–5, we explored the idea of improving the reliability of MT by transfer of knowledge across models, domains, and data sources. However, another interesting way of combining the capabilities of multiple models is the combination of different NMT architectures. We wanted to explore the effect of combining two different types of neural network typologies on NMT performance. We focused our experiments on a combination of recurrent and convolutional architectures for NMT. In Chapter 6, we asked:

RQ4: *What are the limitations of using only recurrent layers towards effective modeling of source sentences in NMT and can NMT performance be improved by the addition of convolutional layers?*

We discussed the limitations of recurrent processing of the input and proposed that combining convolutional and recurrent architectures is more beneficial to model local as well as global features in NMT. Our experiments showed that combining recurrent and convolutional layers can enable the encoder to capture relevant features of the source sentences, which leads to improved NMT performance.

7.2 Future work

In the previous section, we revisited the research questions addressed in this thesis and discussed our findings towards answering these questions. In this section, we list some of the limitations of the proposed approaches and also discuss possible future extensions of our solutions. We divide the different aspects of the limitations of the experiments into three groups: the use of datasets, baselines, and comparisons, and additional experiments, all of which are discussed below.

7.2.1 Additional language pairs and datasets

In all our experiments, the choices of language pairs and training datasets have been based on factors such as public availability of datasets, source or origin of the datasets, domains, complexity of experiments, etc. However, for some of the experiments, there is always the possibility of replicating the experiments with other language pairs and datasets to further verify our observations. For example, in Chapter 3, due to the time and space required for decoding the training bitext, we only conducted experiments for Arabic→English translation and used samples from publicly available datasets. However, it would be helpful to verify whether the oracle-BLEU experiments show similar improvements for additional language pairs such as Chinese→English, where

re-ordering of words is an important problem. We believe that for such language pairs, the re-estimation of re-ordering models could achieve better performance improvements. Moreover, since the proposed re-estimation techniques aim at improving the reliability of models even from noisy data, it would be interesting to explore re-estimation using datasets that are known to be noisy such as web-crawled datasets. Similarly, in Chapter 6, it would be beneficial to evaluate the proposed convolutional over a recurrent model for additional language pairs.

Further, in Chapter 5, we ran experiments for three specialized domains but limited the experiments only to one language pair, namely English→German. Although the three domains that we selected in Chapter 5 provide sufficient insights into the problem of domain bias, the training datasets for these domains are significantly smaller than the general domain dataset. Therefore, it would be interesting to explore domain adaptation in a setting where the in-domain training data size is comparable to that of the general domain data and evaluate whether the observed results are consistent with such a setting. In Chapter 6, we experimented with noisy data for three different language pairs. Recently additional training data has been released for many language pairs, which are known to be noisy or low-quality, for example, the Paracrawl[1] resource provides noisy data for around 23 languages paired with English. Therefore, it would be interesting to experiment with these additional language pairs to further verify our observations.

7.2.2 Baselines and comparisons

In all our experiments, we compared our proposed solutions with standard training methods and architectures as well as with state-of-the-art techniques aimed at solving a similar problem. However, as already discussed, there is a possibility of alternative solutions for the same problem. Therefore, it could be beneficial to additionally evaluate such solutions in comparison to our techniques. For example, in Chapter 3, we compared the proposed oracle-BLEU re-estimation with heuristic training and forced-decoding based re-estimation. The re-estimated phrase translation model yields translation performance comparable to the original phrase table. A similar approach for reducing the phrase table size while retaining the model performance is phrase table pruning (Zens et al., 2012). Therefore, it would be worth comparing the results from our re-estimation technique with phrase table pruning techniques. However, it is important to note that the goal of our proposed training strategy is not to prune the model size but to modify the training itself in order to improve the model reliability. Nevertheless, a comparison with pruning techniques can provide additional insights towards reaching similar end goals.

In Chapter 4, we modified the fine-tuning method in order to reduce degradation during domain adaptation. Although fine-tuning is a straightforward approach, it is not the only possible method for domain adaptation. There are various other techniques that have recently been proposed for domain adaptation for NMT (Chu and Wang, 2018; Hu et al., 2019). It would be interesting to evaluate whether other domain adaptation techniques suffer from similar degradation on the source domains and, if so, what

[1] https://paracrawl.eu/releases.html

remedies are possible.

In Chapter 5, we proposed distillation as a solution for degradation in NMT performance due to noisy training data and compared our experiments with state-of-the-art data filtering techniques. However, there has been a growing interest in training NMT models using only monolingual datasets in source and target languages. Although the aim of our proposed method is the optimal utilization of noisy data for training NMT models, it is important to explore whether models trained only on monolingual data can perform better than those trained using noisy data (Artetxe et al., 2018; Yang et al., 2018). Moreover, we evaluated the effect of noisy data on one non-recurrent architecture, i.e., the Transformer NMT model. However, the convolutional sequence-to-sequence model has also been proposed as a non-recurrent NMT architecture. Therefore, it would be interesting to explore whether fully convolutional NMT models would demonstrate similar resistance against training noise.

Finally, in Chapter 6, we experimented by augmenting the standard recurrent NMT architecture of Bahdanau et al. (2015) with additional convolutional layers. Additional enhancements, including newer architectures, have been proposed for NMT, such as fully convolutional (Gehring et al., 2017a) and fully attentional models (Vaswani et al., 2017), which have outperformed recurrent NMT models by using various optimization heuristics. Therefore, it is important to evaluate the performance of these non-recurrent architectures in comparison to a combined recurrent and non-recurrent architecture similar to our proposed model. One limitation of our proposed model is the use of a unidirectional encoder for the recurrent NMT baseline. However, bi-directional encoders have become the state-of-art in recurrent NMT models (Bahdanau et al., 2015). Using bi-directional encoders could have a different effect on the baseline model, and a comparison with the proposed model can provide further insights.

7.2.3 Additional experiments

Phrase-based machine translation and neural machine translation both have a very large number of hyper-parameters, and different settings can affect the performance of the MT systems, and some of these have been discussed in Chapter 2. However, we only experimented using some standard values of these parameters for both baselines and the proposed solutions. Evaluation with different values of these parameters, especially for the baselines, may have alternative effects on some of our observations. For example, in Chapter 3, we fixed the value of μ in Equation 3.5 to be 0.5. This may not necessarily be the optimal value for μ. However, rescoring the lattices for different values of μ over all training data is very expensive. Therefore, based on the previous literature, we fixed this value to be 0.5. Further, to obtain the oracle-BLEU segmentations, we decode the bitext with standard hyper-parameters identical to that of the baseline. However, we believe that relaxing some of the decoding parameters can further improve the overall performance of the oracle-BLEU re-estimation. As explained in Section 3.3.5, to avoid overfitting, we drop the re-estimated phrase pairs with a frequency below a threshold. However, a better approach could be to apply the leave-one-out strategy similar to forced decoding, as explained in (Wuebker et al., 2010).

Similarly, in Chapter 4, we evaluated domain adaptation for a general domain model to a specific in-domain dataset. In real-life industrial settings, it would be desirable

to train a model that can be adapted to multiple domains iteratively but retains its performance for all domains. Therefore one extension of our work would be to evaluate our distillation based strategy for multiple domains and further explore modifications that can enable the model to retain its performance for all the domains. In addition, it would be beneficial to evaluate the proposed solutions along with preprocessing techniques such as byte-pair encoding (Sennrich et al., 2016b).

In Chapter 5, we provide distillation as a remedy against noisy training data for recurrent NMT. Additionally, we showed that non-recurrent NMT models are relatively robust against training noise. However, it would be interesting to investigate whether non-recurrent models can additionally benefit from noisy data and yield even better performance through distillation. Further, we empirically compared the effect of noise on recurrent NMT and Transformer models based on the intuition that the self-attention mechanism provides the model with the capability to avoid noise propagation in transformer models (as discussed in Section 5.7). However, it is important to theoretically analyze and verify our intuition and evaluate the model under varying parameter settings.

In Chapter 6, we only evaluated the number of layers as a hyper-parameter for the proposed convolutional over the recurrent model. Tuning other hyper-parameters such as the number of dimensions for embeddings and hidden layers, learning rate, attention functions can further affect the overall translation performance of the model.

In conclusion, all experiments conducted in this thesis suggest new research directions towards answering the research questions that we asked. However, the applicability and effectiveness of the proposed solutions can be further verified and enhanced by conducting the experiments suggested in this section.

Bibliography

W. C. Abraham and A. J. Robins. Memory retention – the synaptic stability versus plasticity dilemma. *Trends in Neurosciences*, 28:73–78, 2005. (Cited on page 59.)

D. Arnold, L. Balkan, R. Humphreys, S. Meijer, and L. Sadler. *Machine Translation: An Introductory Guide*. NCC Blackwell, London, 1994. (Cited on page 1.)

M. Artetxe, G. Labaka, E. Agirre, and K. Cho. Unsupervised neural machine translation. In *International Conference on Learning Representations*, 2018. (Cited on page 121.)

T. Ash, R. Francis, and W. Williams. The speechmatics parallel corpus filtering system for WMT18. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 866–872, Belgium, Brussels, October 2018. Association for Computational Linguistics. (Cited on page 93.)

A. Axelrod, R. B. Mayne, C. Callison-burch, M. Osborne, and D. Talbot. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation*, 2005. (Cited on page 18.)

A. Axelrod, X. He, and J. Gao. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 355–362. Association for Computational Linguistics, Jul 2011. (Cited on pages 2 and 56.)

A. Axelrod, P. Resnik, X. He, and M. Ostendorf. Data selection with fewer words. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 58–65, Lisbon, Portugal, Sept. 2015. Association for Computational Linguistics. (Cited on pages 77, 78, and 118.)

W. Aziz, M. Dymetman, and L. Specia. Exact decoding for phrase-based statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1237–1249, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. (Cited on page 1.)

D. Bahdanau, K. Cho, and Y. Bengio. 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*, 2015. (Cited on pages 4, 7, 22, 23, 24, 27, 99, 100, 103, and 121.)

P. Banerjee, J. Du, B. Li, S. Naskar, A. Way, and J. Genabith. Combining multi-domain statistical machine translation models using automatic classifiers. In *Proceedings of the 9th Conference of the Association for Machine Translation in the Americas*, May 2010. (Cited on page 56.)

S. Banerjee and A. Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. (Cited on page 28.)

E. Barbu and V. Barbu Mititelu. A hybrid pipeline of rules and machine learning to filter web-crawled parallel corpora. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 880–884, Belgium, Brussels, October 2018. Association for Computational Linguistics. (Cited on pages 73 and 74.)

M. A. V. Barone, B. Haddow, U. Germann, and R. Sennrich. Regularization techniques for fine-tuning in neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1489–1494. Association for Computational Linguistics, 2017. (Cited on pages 78 and 118.)

E. Beigman and B. B. Klebanov. Learning with annotation noise. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 280–287, Suntec, Singapore, Aug. 2009. Association for Computational Linguistics. (Cited on page 76.)

A. Bisazza and M. Federico. Surveys: A survey of word reordering in statistical machine translation: Computational models and language phenomena. *Computational Linguistics*, 42(2):163–205, June 2016. (Cited on page 3.)

A. Bisazza, N. Ruiz, and M. Federico. Fill-up versus interpolation methods for phrase-based SMT adaptation. In *International Workshop on Spoken Language Translation*, 2011. (Cited on page 56.)

O. Bojar, R. Chatterjee, C. Federmann, B. Haddow, M. Huck, C. Hokamp, P. Koehn, V. Logacheva, C. Monz, M. Negri, M. Post, C. Scarton, L. Specia, and M. Turchi. Findings of the 2015 workshop on statistical machine translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, Lisbon, Portugal, September 2015. Association for Computational Linguistics. (Cited on pages 63 and 105.)

O. Bojar, R. Chatterjee, C. Federmann, Y. Graham, B. Haddow, M. Huck, A. Jimeno Yepes, P. Koehn, V. Logacheva, C. Monz, M. Negri, A. Neveol, M. Neves, M. Popel, M. Post, R. Rubino, C. Scarton, L. Specia, M. Turchi, K. Verspoor, and M. Zampieri. Findings of the 2016 conference on machine

translation. In *Proceedings of the First Conference on Machine Translation*, pages 131–198, Berlin, Germany, August 2016. Association for Computational Linguistics. (Cited on pages 63 and 106.)

O. Bojar, R. Chatterjee, C. Federmann, Y. Graham, B. Haddow, S. Huang, M. Huck, P. Koehn, Q. Liu, V. Logacheva, C. Monz, M. Negri, M. Post, R. Rubino, L. Specia, and M. Turchi. Findings of the 2017 conference on machine translation (WMT17). In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 169–214, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. (Cited on page 87.)

O. Bojar, R. Chatterjee, C. Federmann, M. Fishel, Y. Graham, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, A. Martins, C. Monz, M. Negri, A. Névéol, M. Neves, M. Post, M. Turchi, and K. Verspoor, editors. *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, Florence, Italy, Aug. 2019. Association for Computational Linguistics. (Cited on page 53.)

P. Brown, J. Cocke, S. D. Pietra, V. D. Pietra, F. Jelinek, R. Mercer, and P. Roossin. A statistical approach to language translation. In *Proceedings of the 12th Conference on Computational Linguistics - Volume 1*, page 71–76, USA, 1988. Association for Computational Linguistics. (Cited on page 1.)

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993. (Cited on pages 1, 11, 12, and 13.)

M. Cettolo, C. Girardi, and M. Federico. Wit^3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, pages 261–268, Trento, Italy, May 2012. (Cited on pages 57, 63, and 105.)

S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, pages 310–318, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics. (Cited on page 17.)

Y. Chen, Y. Liu, Y. Cheng, and V. O. Li. A teacher-student framework for zero-resource neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1925–1935. Association for Computational Linguistics, 2017. (Cited on page 79.)

Z. Chen and B. Liu. *Lifelong Machine Learning*. Morgan & Claypool Publishers, 2016. (Cited on page 59.)

J. Cheng, L. Dong, and M. Lapata. Long short-term memory-networks for machine reading. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 551–561, Austin, Texas, Nov. 2016. Association for Computational Linguistics. (Cited on page 7.)

D. Chiang, Y. Marton, and P. Resnik. Online large-margin training of syntactic and structural translation features. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 224–233, Honolulu, Hawaii, Oct. 2008. Association for Computational Linguistics. (Cited on pages 35, 38, and 39.)

K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. (Cited on page 103.)

K. Choi, G. Fazekas, M. B. Sandler, and K. Cho. Convolutional recurrent neural networks for music classification. *CoRR*, abs/1609.04243, 2016. URL `http://arxiv.org/abs/1609.04243`. (Cited on page 103.)

C. Chu and R. Wang. A survey of domain adaptation for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319, Santa Fe, New Mexico, USA, Aug. 2018. Association for Computational Linguistics. (Cited on pages 2, 53, and 120.)

C. Chu, R. Dabre, and S. Kurohashi. An empirical comparison of domain adaptation methods for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vancouver, Canada, July 2017. Association for Computational Linguistics. (Cited on pages 54, 57, and 117.)

H. Cuong and K. Sima'an. A survey of domain adaptation for statistical machine translation. *Machine Translation*, 31(4):187–224, Dec. 2017. (Cited on pages 1, 53, and 56.)

P. Dakwale and C. Monz. Improving statistical machine translation performance by oracle-BLEU model re-estimation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 38–44. Association for Computational Linguistics, 2016. (Cited on page 10.)

P. Dakwale and C. Monz. Fine-tuning for neural machine translation with limited degradation across in-and out-of-domain data. In *Proceedings of the 16th Machine Translation Summit*, Nagoya, Japan, September 2017a. (Cited on page 10.)

P. Dakwale and C. Monz. Convolutional over recurrent encoder for neural machine translation. In *Proceedings of The 20th Annual Conference of the European Association for Machine Translation*, Prague, Czech Republic, 2017b. European Association for Machine Translation. (Cited on page 10.)

P. Dakwale and C. Monz. Improving neural machine translation using noisy parallel data through distillation. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pages 118–127, Dublin, Ireland, 19–23 Aug. 2019. European Association for Machine Translation. (Cited on page 10.)

Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier. Language modeling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, page 933–941. JMLR.org, 2017. (Cited on page 26.)

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal statistical society, series B*, 39(1):1–38, 1977. (Cited on pages 12 and 14.)

J. DeNero and D. Klein. The complexity of phrase alignment problems. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics-08: HLT, Short Papers*, pages 25–28, Columbus, Ohio, June 2008. Association for Computational Linguistics. (Cited on pages 1 and 3.)

J. DeNero, D. Gillick, J. Zhang, and D. Klein. Why generative phrase models underperform surface heuristics. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 31–38, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. (Cited on pages 1 and 34.)

G. Devy. *In Another Tongue: Essays on Indian English Literature*. P. Lang, 1993.

B. J. Dorr, J. Olive, J. McCary, and C. Christianson. *Machine Translation Evaluation and Optimization*, pages 745 – 843. Springer New York, 2011. (Cited on page 1.)

C. dos Santos and M. Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers*, pages 69–78, Dublin, Ireland, August 2014. Dublin City University and Association for Computational Linguistics. (Cited on page 102.)

M. Dreyer, K. Hall, and S. Khudanpur. Comparing reordering constraints for SMT using efficient BLEU oracle computation. In *Proceedings of SSST, NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 103–110. Association for Computational Linguistics, 2007. (Cited on pages 38 and 41.)

S. Eetemadi, W. Lewis, K. Toutanova, and H. Radha. Survey of data-selection methods in statistical machine translation. *Machine Translation*, 29(3–4):189–223, Dec. 2015. (Cited on page 1.)

T. Etchegoyhen, A. Fernández Torné, A. Azpeitia, E. Martínez Garcia, and A. Matamala. Evaluating domain adaptation for machine translation across scenarios. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). (Cited on page 56.)

M. Fadaee, A. Bisazza, and C. Monz. Data augmentation for low-resource neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vancouver, Canada, July 2017. Association for Computational Linguistics. (Cited on page 77.)

J. R. Firth. The technique of semantics. *Transactions of the Philological Society*, 34(1):36–73, 1935. (Cited on page 13.)

M. L. Forcada, M. Ginestí-Rosell, J. Nordfalk, J. O'Regan, S. Ortiz-Rojas, J. A. Pérez-Ortiz, F. Sánchez-Martínez, G. Ramírez-Sánchez, and F. M. Tyers. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144, 2011. (Cited on page 1.)

L. Formiga and J. A. R. Fonollosa. Dealing with input noise in statistical machine translation. In *Proceedings of International Conference on Computational Linguistics, 2012: Posters*, pages 319–328, Mumbai, India, December 2012. The COLING 2012 Organizing Committee. (Cited on page 77.)

G. Foster and R. Kuhn. Mixture-model adaptation for SMT. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 128–135, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. (Cited on page 56.)

G. Foster and R. Kuhn. Forced decoding for phrase extraction. Technical report, University of Montreal, 2012. (Cited on page 46.)

G. Foster, R. Kuhn, and H. Johnson. Phrasetable smoothing for statistical machine translation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2006. (Cited on page 32.)

G. Foster, C. Goutte, and R. Kuhn. Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 451–459, Cambridge, MA, October 2010. Association for Computational Linguistics. (Cited on page 56.)

M. Freitag and Y. Al-Onaizan. Fast domain adaptation for neural machine translation. *CoRR*, abs/1612.06897,

2016. URL http://arxiv.org/abs/1612.06897. (Cited on pages 5, 54, 57, 64, 66, and 117.)
B. Frénay and A. Kabán. A comprehensive introduction to label noise. In *Proceedings of the 2014 European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, Bruges, Belgium, 2014. (Cited on page 76.)
B. Frénay and M. Verleysen. Classification in the presence of label noise: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5):845–869, May 2014. (Cited on page 76.)
P. Fung and P. Cheung. Multi-level bootstrapping for extracting parallel sentences from a quasi-comparable corpus. In *Proceedings of the 20th International Conference on Computational Linguistics*, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics. (Cited on page 77.)
M. Galley and C. D. Manning. A simple and effective hierarchical phrase reordering model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, Honolulu, Hawaii, Oct. 2008. Association for Computational Linguistics. (Cited on pages 18 and 40.)
Q. Gao, W. Lewis, C. Quirk, and M.-Y. Hwang. Incremental training and intentional over-fitting of word alignment. In *Proceedings of Machine Translation Summit XIII*. Asia-Pacific Association for Machine Translation, September 2011. (Cited on page 2.)
J. Gehring, M. Auli, D. Grangier, and Y. Dauphin. A convolutional encoder model for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada, July 2017a. Association for Computational Linguistics. (Cited on pages 103 and 121.)
J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 1243–1252, International Convention Centre, Sydney, Australia, 06–11 Aug 2017b. PMLR. (Cited on pages 25, 26, 27, and 99.)
X. Geng and K. Smith-Miles. *Incremental Learning*, pages 731–735. Springer US, Boston, MA, 2009. (Cited on page 2.)
U. Germann, M. Jahr, K. Knight, D. Marcu, and K. Yamada. Fast decoding and optimal decoding for machine translation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 228–235, Toulouse, France, July 2001. Association for Computational Linguistics. (Cited on page 1.)
H. Ghader and C. Monz. What does attention in neural machine translation pay attention to? In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 30–39, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing. (Cited on page 82.)
Y. Goldberg and G. Hirst. *Neural Network Methods in Natural Language Processing*. Morgan & Claypool Publishers, 2017. (Cited on page 59.)
I. J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40:16–264, 1953. (Cited on page 17.)
B. Haddow and P. Koehn. Analysing the effect of out-of-domain data on SMT systems. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, Montréal, Canada, June 2012. Association for Computational Linguistics. (Cited on page 53.)
K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. (Cited on pages 26 and 104.)
G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science (New York, N.Y.)*, 313:504–7, 08 2006. (Cited on page 58.)
G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *NIPS 2014 Deep Learning Workshop*, 2014. (Cited on pages 6, 27, 28, 55, 83, and 117.)
C. Hoang and K. Sima'an. Latent domain translation models in mix-of-domains haystack. In *Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1928–1939, Dublin, Ireland, 2014. Dublin City University and Association for Computational Linguistics. (Cited on page 56.)
S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, Nov. 1997. (Cited on pages 21 and 101.)
M. Hopkins and J. May. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics. (Cited on pages 19, 29, and 45.)
J. Hu, M. Xia, G. Neubig, and J. Carbonell. Domain adaptation of neural machine translation by lexicon induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, July 2019. Association for Computational Linguistics. (Cited on page 120.)

S. Hunsicker, R. Ion, and D. Stefanescu. Hybrid parallel sentence mining from comparable corpora. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation*, Trento, Italy, May 2012. (Cited on page 77.)

W. J. Hutchins. *Machine Translation: Past, Present, Future*. John Wiley and Sons, Inc., USA, 1986. (Cited on page 1.)

A. Irvine and C. Callison-Burch. Combining bilingual and comparable corpora for low resource machine translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 262–270, Sofia, Bulgaria, Aug. 2013. Association for Computational Linguistics. (Cited on page 1.)

M. Junczys-Dowmunt. Dual conditional cross-entropy filtering of noisy parallel corpora. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, Belgium, Brussels, Oct. 2018. Association for Computational Linguistics. (Cited on pages 73, 74, 78, 86, 87, 88, 89, and 95.)

N. Kalchbrenner and P. Blunsom. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA, Oct. 2013. Association for Computational Linguistics. (Cited on page 102.)

N. Kalchbrenner, E. Grefenstette, and P. Blunsom. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, Maryland, June 2014. Association for Computational Linguistics. (Cited on page 102.)

N. Kalchbrenner, L. Espeholt, K. Simonyan, A. van den Oord, A. Graves, and K. Kavukcuoglu. Neural machine translation in linear time. *CoRR*, abs/1610.10099, 2016. URL `http://arxiv.org/abs/1610.10099`. (Cited on page 103.)

S. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401, 1987. (Cited on page 17.)

R. Kemker, M. McClure, A. Abitino, T. L. Hayes, and C. Kanan. Measuring catastrophic forgetting in neural networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 3390–3398, New Orleans, Louisiana, USA, February 2018. (Cited on page 54.)

H. Khayrallah and P. Koehn. On the impact of various types of noise on neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, Melbourne, Australia, 2018. Association for Computational Linguistics. (Cited on pages 2, 6, 73, 74, 75, 77, 79, 80, 86, 89, 90, and 92.)

H. Khayrallah, H. Xu, and P. Koehn. The JHU parallel corpus filtering systems for WMT 2018. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 909–912, Belgium, Brussels, October 2018. Association for Computational Linguistics. (Cited on page 73.)

Y. Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. (Cited on pages 102 and 104.)

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. (Cited on page 24.)

J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13): 3521–3526, 2017. (Cited on page 54.)

G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (System Demonstrations)*, 2017. (Cited on pages 29 and 87.)

R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184 vol.1, May 1995. (Cited on page 17.)

K. Knight. Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4): 607–615, 1999. (Cited on page 20.)

P. Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand, 2005. AAMT. (Cited on page 11.)

P. Koehn. *Statistical Machine Translation*. Cambridge University Press, New York, NY, USA, 1st edition, 2010. (Cited on pages 1, 2, 3, 12, 14, 17, 20, 21, 32, and 40.)

P. Koehn. *Neural Machine Translation*. Cambridge University Press, 2020. (Cited on pages 2, 3, and 24.)

P. Koehn and R. Knowles. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver, Aug. 2017. Association for Computational

Linguistics. (Cited on pages 1, 4, 53, and 117.)
P. Koehn and J. Schroeder. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 224–227, Prague, Czech Republic, 2007. Association for Computational Linguistics. (Cited on pages 2 and 56.)
P. Koehn, F. J. Och, and D. Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*. Association for Computational Linguistics, 2003. (Cited on pages 13, 14, 16, 18, 40, and 45.)
P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics, 2007. (Cited on page 45.)
P. Koehn, H. Khayrallah, K. Heafield, and M. L. Forcada. Findings of the WMT 2018 shared task on parallel corpus filtering. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 739–752, Belgium, Brussels, October 2018. Association for Computational Linguistics. (Cited on pages 6, 73, 74, 75, and 77.)
J. Lee, K. Cho, and T. Hofmann. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378, 2017a. (Cited on page 103.)
S.-W. Lee, J.-H. Kim, J. Jun, J.-W. Ha, and B.-T. Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in Neural Information Processing Systems 30*, pages 4652–4662. Curran Associates, Inc., 2017b. (Cited on page 54.)
Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L. Li. Learning from noisy labels with distillation. In *2017 IEEE International Conference on Computer Vision*, 2017. (Cited on pages 79, 81, 83, 84, and 85.)
Z. Li and D. Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018. (Cited on pages 54, 55, 57, 58, 59, 60, and 61.)
Z. Li and S. Khudanpur. Efficient extraction of oracle-best translations from hypergraphs. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 9–12. Association for Computational Linguistics, 2009. (Cited on pages 41 and 43.)
P. Liang, A. Bouchard-Côté, D. Klein, and B. Taskar. An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 761–768, Sydney, Australia, 2006. Association for Computational Linguistics. (Cited on pages 35 and 36.)
T. Liu and D. Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38:447–461, 2016. (Cited on page 76.)
Y. Lü, J. Huang, and Q. Liu. Improving statistical machine translation performance by training data selection and optimization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 343–350, Prague, Czech Republic, June 2007. Association for Computational Linguistics. (Cited on pages 1 and 2.)
M. Lui and T. Baldwin. langid.py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*, pages 25–30. Association for Computational Linguistics, 2012. (Cited on page 79.)
M.-T. Luong and C. D. Manning. Stanford neural machine translation systems for spoken language domain. In *International Workshop on Spoken Language Translation*, 2015. (Cited on page 57.)
M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015. Association for Computational Linguistics. (Cited on pages 4, 7, 25, 29, 63, 87, 94, 101, and 106.)
N. Manwani and P. Sastry. Noise tolerance under risk minimization. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : A publication of the IEEE Systems, Man, and Cybernetics Society*, 43, 11 2012. (Cited on page 76.)
D. Marcu and W. Wong. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2002. (Cited on pages 1 and 32.)
S. Matsoukas, A.-V. I. Rosti, and B. Zhang. Discriminative corpus weight estimation for machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 708–717, Singapore, August 2009. Association for Computational Linguistics. (Cited on page 56.)
F. Meng, Z. Lu, M. Wang, H. Li, W. Jiang, and Q. Liu. Encoding source language with convolutional

neural network for machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Beijing, China, July 2015. Association for Computational Linguistics. (Cited on page 103.)

C. Mermer, H. Kaya, and M. U. Dogan. The TÜBITAK-UEKAE statistical machine translation system for IWSLT 2007. In *Proceedings of the International Workshop on Spoken Language Translation*, 2007. (Cited on page 77.)

L. Miculicich Werlen, N. Pappas, D. Ram, and A. Popescu-Belis. Self-attentive residual decoder for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1366–1379, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. (Cited on page 7.)

T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur. Recurrent neural network based language model. In T. Kobayashi, K. Hirose, and S. Nakamura, editors, *INTERSPEECH*, pages 1045–1048. ISCA, 2010. (Cited on pages 21, 53, and 100.)

V. Mnih and G. Hinton. Learning to label aerial images from noisy data. In *Proceedings of the 29th International Conference on International Conference on Machine Learning*, pages 203–210, USA, 2012. Omnipress. (Cited on page 76.)

R. C. Moore and W. Lewis. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. (Cited on pages 56 and 78.)

J. Mueller and A. Thyagarajan. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, page 2786–2792. AAAI Press, 2016. (Cited on page 100.)

D. S. Munteanu and D. Marcu. Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, pages 477–504, Dec. 2005. (Cited on pages 74, 80, and 86.)

D. F. Nettleton, A. Orriols-Puig, and A. Fornells. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial Intelligence Review*, 33(4), 2010. (Cited on page 76.)

G. Neubig and T. Watanabe. Optimization for statistical machine translation: A survey. *Computational Linguistics*, 42(1):1–54, Mar. 2016. (Cited on page 3.)

J. Niehues, T. Herrmann, S. Vogel, and A. Waibel. Wider context by using bilingual language models in machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 198–206, Edinburgh, Scotland, July 2011. Association for Computational Linguistics. (Cited on pages 19 and 40.)

E. W. Noreen. *Computer Intensive Methods for Testing Hypotheses. An Introduction*. Wiley- Interscience, 1989. (Cited on pages 45, 64, and 87.)

F. J. Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, pages 160–167, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. (Cited on page 19.)

F. J. Och and H. Ney. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2000. (Cited on pages 4 and 31.)

F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003. (Cited on pages 29 and 45.)

M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1717–1724, June 2014. (Cited on pages 58 and 76.)

S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. (Cited on pages 2 and 58.)

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2002. (Cited on pages 28, 45, 64, 87, and 106.)

A. Poncelas and A. Way. Selecting artificially-generated sentences for fine-tuning neural machine translation. In *Proceedings of the 12th International Conference on Natural Language Generation*, Tokyo, Japan, Oct.–Nov. 2019. Association for Computational Linguistics. (Cited on page 57.)

F. Radenović, G. Tolias, and O. Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(7):1655–1668, 2019. (Cited on page 58.)

M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. Sequence level training with recurrent neural networks. In

4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings, 2016. (Cited on page 24.)

S. A. Rauf and H. Schwenk. Parallel sentence generation from comparable corpora for improved SMT. *Machine Translation*, 25(4):341–375, December 2011. (Cited on page 2.)

S. E. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *CoRR*, abs/1412.6596, 2014. URL `http://arxiv.org/abs/1412.6596`. (Cited on pages 76 and 118.)

S. Riezler and J. T. Maxwell. On some pitfalls in automatic evaluation and significance testing for mt. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Ma- chine Translation and/or Summarization*, 2005. (Cited on pages 45, 64, and 87.)

N. Rossenbach, J. Rosendahl, Y. Kim, M. Graça, A. Gokrani, and H. Ney. The RWTH Aachen university filtering system for the WMT 2018 parallel corpus filtering task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, Belgium, Brussels, Oct. 2018. Association for Computational Linguistics. (Cited on page 73.)

S. Ruder, P. Ghaffari, and J. G. Breslin. Knowledge adaptation: Teaching to adapt. *CoRR*, abs/1702.02052, 2017. URL `http://arxiv.org/abs/1702.02052`. (Cited on page 57.)

H. Schwenk. Investigations on large-scale lightly-supervised training for statistical machine translation. In *In Proceedings of the International Workshop on Spoken Language Translation*, pages 182–189, 2008. (Cited on page 56.)

R. Sennrich. Perplexity minimization for translation model domain adaptation in statistical machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 539–549, Avignon, France, Apr. 2012. Association for Computational Linguistics. (Cited on page 56.)

R. Sennrich, B. Haddow, and A. Birch. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96. Association for Computational Linguistics, 2016a. (Cited on pages 57, 75, 77, 78, and 82.)

R. Sennrich, B. Haddow, and A. Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, Aug. 2016b. Association for Computational Linguistics. (Cited on page 122.)

C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, July 1948. (Cited on page 11.)

C. C. Silva, C.-H. Liu, A. Poncelas, and A. Way. Extracting in-domain training corpora for neural machine translation using data selection methods. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, Brussels, Belgium, Oct. 2018. Association for Computational Linguistics. (Cited on page 56.)

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and R. Weischedel. A study of translation error rate with targeted human annotation. In *In Proceedings of the Association for Machine Translation in the Americas*, 2006. (Cited on page 28.)

A. Sokolov, G. Wisniewski, and F. Yvon. Computing lattice BLEU oracle scores for machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 120–129, Avignon, France, 2012. Association for Computational Linguistics. (Cited on page 41.)

A. K. Srivastava, Y. Ma, and A. Way. Oracle-based training for phrase-based statistical machine translation. In *Proceedings of the 15th annual meeting of the European Association for Machine Translation*, 2011. (Cited on page 38.)

A. Stolcke, J. Zheng, W. Wang, and V. Abrash. Srilm at sixteen: Update and outlook. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop*. IEEE SPS, December 2011. (Cited on pages 29 and 45.)

S. Sukhbaatar and R. Fergus. Learning from noisy labels with deep neural networks. *CoRR*, abs/1406.2080, 2014. URL `http://arxiv.org/abs/1406.2080`. (Cited on page 76.)

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112, 2014. (Cited on pages 21, 22, and 102.)

V. M. Sánchez-Cartagena, M. Bañón, S. Ortiz Rojas, and G. Ramírez. Prompsit's submission to WMT 2018 parallel corpus filtering shared task. In *Proceedings of the Third Conference on Machine Translation,*

Volume 2: Shared Task Papers, pages 968–975, Belgium, Brussels, October 2018. Association for Computational Linguistics. (Cited on page 93.)

D. Tang, B. Qin, and T. Liu. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1432, Lisbon, Portugal, September 2015. Association for Computational Linguistics. (Cited on pages 102 and 103.)

G. Tang, M. Müller, A. Rios, and R. Sennrich. Why Self-Attention? A Targeted Evaluation of Neural Machine Translation Architectures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4263–4272. Association for Computational Linguistics, 2018. (Cited on page 93.)

Y. Tang, X. Wu, and W. Bu. Deeply-supervised recurrent convolutional neural network for saliency detection. In *Proceedings of the 2016 ACM Conference on Multimedia Conference, MM 2016, Amsterdam, The Netherlands, October 15-19, 2016*, pages 397–401. ACM, 2016. (Cited on page 103.)

J. Tiedemann. News from opus - a collection of multilingual parallel corpora with tools and interfaces. In *Recent Advances in Natural Language Processing*, 2009. (Cited on page 63.)

J. Tiedemann. *Bitext Alignment*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2011. (Cited on page 11.)

C. Tillman. A unigram orientation model for statistical machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*. Association for Computational Linguistics, 2004. (Cited on pages 18 and 40.)

A. Toral and V. M. Sánchez-Cartagena. A multifaceted evaluation of neural versus phrase-based machine translation for 9 language directions. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1063–1073, Valencia, Spain, Apr. 2017. Association for Computational Linguistics. (Cited on pages 3 and 53.)

K. Tran, A. Bisazza, and C. Monz. The importance of being recurrent for modeling hierarchical structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4731–4736. Association for Computational Linguistics, 2018. (Cited on page 93.)

A. Trujillo. *Translation engines: techniques for machine translation*. Springer-Verlag, 1999. (Cited on page 1.)

Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li. Coverage-based neural machine translation. *CoRR*, abs/1601.04811, 2016. (Cited on page 110.)

M. Turchi, T. De Bie, and N. Cristianini. Learning performance of a machine translation system: a statistical and computational analysis. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 35–43, Columbus, Ohio, June 2008. Association for Computational Linguistics. (Cited on page 1.)

E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *International Conference in Computer Vision (ICCV)*, 2015. (Cited on page 57.)

M. van der Wees. *What's in a Domain? Towards Fine-Grained Adaptation for Machine Translation*. PhD thesis, University of Amsterdam, 2017. (Cited on pages 54 and 56.)

M. van der Wees, A. Bisazza, W. Weerkamp, and C. Monz. What's in a domain? analyzing genre and topic differences in statistical machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 560–566, Beijing, China, July 2015. Association for Computational Linguistics. (Cited on page 53.)

M. van der Wees, A. Bisazza, and C. Monz. Dynamic data selection for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. (Cited on pages 1, 2, 4, 56, 77, 78, and 118.)

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. (Cited on pages 25, 26, 75, 92, 93, 99, 118, and 121.)

S. Vogel. Using noisy bilingual data for statistical machine translation. In *10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 175–178, 2003. (Cited on page 77.)

W. Wang, K. Macherey, W. Macherey, F. J. Och, and P. Xu. Improved domain adaptation for statistical machine translation. In *Proceedings of the Tenth Biennial Conference of the Association for Machine Translation in the Americas*, San Diego, California, October 2012. (Cited on page 56.)

W. Wang, T. Watanabe, M. Hughes, T. Nakagawa, and C. Chelba. Denoising neural machine translation training with trusted data and online data selection. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 133–143, Belgium, Brussels, Oct. 2018. Association for

Computational Linguistics. (Cited on page 78.)

Y.-Y. Wang and A. Waibel. Decoding algorithm in statistical machine translation. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, page 366–372, USA, 1997. Association for Computational Linguistics. (Cited on pages 1 and 20.)

T. Watanabe, J. Suzuki, H. Tsukada, and H. Isozaki. Online large-margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 764–773, Prague, Czech Republic, June 2007. Association for Computational Linguistics. (Cited on pages 19 and 39.)

Y. Wen, W. Zhang, R. Luo, and J. Wang. Learning text representation using recurrent convolutional neural network with highway layers. *CoRR*, abs/1606.06905, 2016. URL `http://arxiv.org/abs/1606.06905`. (Cited on page 100.)

J. S. White and T. A. O'Connell. Evaluation of machine translation. In *Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*, 1993. (Cited on page 1.)

G. Wisniewski, A. Allauzen, and F. Yvon. Assessing phrase-based translation models with oracle decoding. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2010. (Cited on page 38.)

I. H. Witten and T. C. Bell. The zero-frequency problem: estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094, July 1991. (Cited on page 17.)

J. Wuebker, A. Mauser, and H. Ney. Training phrase translation models with leaving-one-out. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, July 2010. (Cited on pages 1, 2, 3, 4, 34, 35, 36, 44, 47, 48, and 121.)

Z. Yang, W. Chen, F. Wang, and B. Xu. Unsupervised neural machine translation with weight sharing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia, July 2018. Association for Computational Linguistics. (Cited on page 121.)

J. Yen. An algorithm for finding shortest routes from all source nodes to a given destination in general networks. *Quarterly of Applied Mathematics*, 27, jan 1970. (Cited on page 44.)

J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems 27*, pages 3320–3328. Curran Associates, Inc., 2014. (Cited on page 58.)

R. Zens, D. Stanton, and P. Xu. A systematic comparison of phrase table pruning techniques. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 972–983, Jeju Island, Korea, 2012. (Cited on page 120.)

H. Zhang and D. Chiang. Kneser-ney smoothing on expected counts. In *52nd Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 765–774, 06 2014. (Cited on page 77.)

Z. Zhou, J. Shin, L. Zhang, S. Gurudu, M. Gotway, and J. Liang. Fine-tuning convolutional neural networks for biomedical image analysis: Actively and incrementally. In *2017 IEEE Conference on Computer Vision and Pattern Recognition*, pages 4761–4772, July 2017. (Cited on page 58.)

B. Zoph, D. Yuret, J. May, and K. Knight. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas, November 2016. Association for Computational Linguistics. (Cited on pages 2 and 54.)

Summary

Machine Translation (MT) systems are developed with the aim to automatically translate text from one language (the source language) to another language (the target language). The vast majority of present-day MT systems are data-driven, that is, they learn how to translate between the two languages of interest by modelling the probabilities of translation patterns from large amounts of parallel corpora. For many language pairs, the training data is compiled from multiple sources that often come from different domains and can have varying degrees of translation equivalence.

One limitation of standard training mechanisms is that they assume training data to be homogeneous with respect to domain and quality. As a result, these training mechanisms may learn noisy or undesirable translation patterns. This thesis explores training strategies that can refine the quality of the translation model and minimize the effect of training data variations. The majority of the strategies proposed in this thesis are based on the idea of iterative knowledge transfer, in which the quality of a model is refined through predictions of other models that are trained in earlier steps. The main idea underlying the solutions proposed in this thesis is that knowledge gathered by the models during initial training steps can be refined and used for solving varied problems such as domain adaptation and better utilization of noisy training data. Based on this idea, this thesis proposes solutions to four different problems.

First, this thesis demonstrates that the quality of models used in phrase-based machine translation can be improved by re-estimating them from their own predictions on the training data.

Second, this thesis investigates the problem of low performance of neural MT models when used to translate sentences from domains that are underrepresented in the training data. This thesis proposes a knowledge transfer strategy for training neural MT models with stable performance across multiple domains.

Third, we address the problem of the negative effect of low-quality training data on the performance of neural MT systems. Our research demonstrates that the performance degradation due to noise in the training data can be reduced using knowledge transfer from a strong model trained on small high-quality data.

Fourth, this thesis discusses the limitations of single architectures for neural MT. We propose a combination of two different neural network architectures to effectively model important features of the source sentences.

Samenvatting

Automatische Vertaalsystemen (afgekort MT, naar het Engelse Machine Translation) zijn ontwikkeld met als doel het automatisch vertalen van tekst van de ene taal (de brontaal) naar een andere taal (de doeltaal). De overgrote meerderheid van de huidige MT systemen zijn datagestuurd, dat wil zeggen ze leren hoe ze moeten vertalen tussen de twee talen door het modelleren van de waarschijnlijkheid van vertaalpatronen van een grote hoeveelheid parallele corpora. Voor veel taalparen wordt de trainingsdata samengesteld uit meerdere bronnen die vaak afkomstig zijn uit verschillende domeinen en die verschillende niveaus van vertalingsequivalentie kunnen hebben.

Een beperking van de standaard trainingsmechanismen is dat ze ervan uitgaan dat trainingsdata homogeen zijn met betrekking tot het domein en de kwaliteit. Als gevolg daarvan kunnen deze trainingsmechanismen ruis of ongewenste vertaalpatronen aanleren. Dit proefschrift richt zich op het onderzoeken van trainingsstrategieën die de kwaliteit van het vertaalmodel kunnen verfijnen en het effect van variaties in de trainingsdata kunnen minimaliseren. Het merendeel van de in dit proefschrift voorgestelde strategieën is gebaseerd op het idee van iteratieve kennisoverdracht, waarbij de kwaliteit van een model wordt verfijnd door middel van voorspellingen van andere modellen die in eerdere stappen worden getraind. Het hoofdidee dat ten grondslag ligt aan de oplossingen die in dit proefschrift worden voorgesteld, is dat de kennis die de modellen tijdens de eerste trainingsstappen hebben verzameld, kan worden verfijnd en gebruikt voor het oplossen van uiteenlopende problemen zoals domeinaanpassing en een beter gebruik van trainingsdata met veel ruis. Op basis van dit idee worden in dit proefschrift oplossingen voorgesteld voor vier verschillende problemen.

Ten eerste toont dit proefschrift aan dat de kwaliteit van de modellen die gebruikt worden bij het vertalen van frasen verbeterd kan worden door ze opnieuw te beoordelen op basis van hun eigen voorspellingen op de trainingsdata.

Ten tweede onderzoekt dit proefschrift het probleem van lage prestaties van neurale MT-modellen bij het vertalen van zinnen uit domeinen die ondervertegenwoordigd zijn in de trainingsdata. Dit proefschrift stelt een kennisoverdrachtstrategie voor voor het trainen van neurale MT-modellen met stabiele prestaties over meerdere domeinen.

Ten derde gaat dit proefschrift in op het probleem van het negatieve effect van laagwaardige trainingsdata op de prestaties van neurale MT-systemen. Ons onderzoek toont aan dat de prestatievermindering door ruis in de trainingsdata kan worden gereduceerd door gebruik te maken van kennisoverdracht vanuit een sterk model dat getraind is op weinig kwaliteitsdata.

Ten vierde bespreekt dit proefschrift de beperkingen van enkelvoudige architecturen voor neurale MT. We stellen een combinatie voor van twee verschillende neurale netwerkarchitecturen om belangrijke kenmerken van de bronzinnen effectief te modelleren.