An Empirical Analysis of Phrase-based and Neural Machine Translation

Hamidreza Ghader

An Empirical Analysis of Phrase-based and Neural 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 donderdag 29 oktober 2020, te 16:00 uur

door

Hamidreza Ghader

geboren te Karaj

Promotiecommissie

Promotor:		
	Dr. C. Monz	Universiteit van Amsterdam
Co-promotor:		
	Prof. dr. M. de Rijke	Universiteit van Amsterdam
Overige leden:		
	Dr. G. Chrupala	Tilburg University
	Prof. dr. J. van Genabith	German Center for AI (DFKI)
	Prof. dr. T. Gevers	Universiteit van Amsterdam
	Prof. dr. P. Groth	Universiteit van Amsterdam
	Dr. E. Shutova	Universiteit van Amsterdam

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

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

Cover by Maryam Kamranfar
Printed by Ipskamp Printing
ISBN: 978-94-6421-039-2

To my parents
whose dedication was the most inspiring

Acknowledgements

It was amazing and exciting to do a PhD. However, to be honest, it was the most stressful. I have never faced the number of new things that I faced during this time, and I have never walked this far out of my comfort zone. As a result, I have learned more than any stage in my life. None of these was possible without the support and the love I have received from the people around me.

Foremost, I would like to thank Christof for trusting me not only when he gave me this opportunity to start as his PhD student, but also during these years. I remember you asked me in one of the interviews that what I would do if I burn out during my PhD. That actually happened. I went through a serious burnout, but I managed to overcome it just as I explained in response to your question. However, this was not possible without you trusting me and without you being so patient. I am so grateful, and I feel indebted to you for all the support. Thanks for all the discussions, critical feedbacks and useful pieces of advice.

Next, I would like to thank Maarten for his support and for creating such a great environment at ILPS for the researchers to flourish. Thanks for your leadership, for your insightful bits of advice, all the feedback on my thesis and the presentations.

I am thankful to my colleagues at SMT group for all the pleasant moments we had together. Arianna, Ivan, Katya, Ke, Marlies, Marzieh, Praveen thanks for the supportive comments at times of failure, sharing happiness at times of success, interesting discussions, direct feedbacks, and proofreadings of the papers. I enjoyed every moment of working with you and I learned a lot from you.

I am happy to have Marlies and Bob as my paranymphs. Thank you for proofreading my thesis, answering my questions and more importantly accepting to fight on my side. I would like to thank my other friends who also helped with proofreading the thesis. Cristina, Livia and Maartje, thank you for helping me with that.

Ekaterina, Grzegorz, Josef, Paul and Theo, thanks for accepting to be my committee members, and for your valuable time that you have spent on it.

I enjoyed being part of ILPS thank to all members of this amazing group. You all contributed to making these years of doing the PhD such an incredible experience to me. Thank you, Adith, Aleksandr, Alexey, Ali A, Ali V, Amir, Ana, Anna, Anne, Arezoo, Arianna, Artem, Bob, Boris, Chang, Christof, Christophe, Chuan, Cristina, Daan, Dan, David, David, Dilek, Eva, Evangelos, Evgeny, Fei, Harrie, Hendra, Hendrik, Hendrike, Hinda, Hosein, Isaac, Ilya, Ivan, Jiahuan, Jie, Julia, Julien, Kaspar, Katya, Ke, Lars, Maarten, Maarten, Maartje, Mahsa, Manos, Marc, Marlies, Mariya, Marzieh, Maurits, Mozhdeh, Mostafa, Nikos, Olivier, Pengjie, Petra, Praveen, Richard, Ridho, Rolf, Shangsong, Shaojie, Spyretta, Svitlana, Thorsten, Tobias, Tom, Trond, Vera, Xiaohui, Xiaojuan, Xinyi, Yaser, Yangjun, Yifan, Zhaochun and Ziming. Thanks to Ana, Maartje, Marlies and Maurits once more for being my nicest officemates. A special thank to Anne, Ivan and Marlies for their wise bits of advice and supportive words at

the desperate times of failure. And last, but not least, a big thank to Isaac for always asking questions, even nonsense ones.

Thanks to Maryam for the design of the cover, which I really like.

I would also like to thank my friends who made life in the Netherlands easier for me by being there at the times of fun and failure. Thanks to Ali, Ali, Amin, Amirhosein, Aylar, Azad, Azadeh, Behrouz, Bob, Cristina, Danial, Fahimeh, Fatemeh, Hadi, Hester, Hoda, Hoda, Hojat, Hoora, Hosein, Irene, Jurre, Keyvan, Livia, Maartje, Mahdi, Mahdieh, Marlies, Maryam, Marzieh, Masoud, Mehran, Mehran, Mohammad, Mohammad Hosein, Mojtaba, Mostafa, Narges, Naser, Nasrin, Parisa, Samira, Samira, Samy, Sara, Shayan, Shima, Tim, Tom, Vahid, Vivianne, Zahra, Zoheir.

I did an internship at the National Research Institute of Japan during which I enjoyed working with Satoshi sensei, Anna, Aron, Ben, Dianna, Hassan, Jerome, Kentaro, Koki, Thomas, Tiphaine.

I thank my siblings, Mobina and Omid for encouraging me and supporting me to take this path of doing a PhD, and three of my cousins, Mansour, Masoud and Mona who are as close as my siblings to me and have always supported me and inspired me with their kindness and their dedication. Special thanks to Masoud for being my role model.

I would like to express my utmost gratitude to my mom and dad. This was impossible without you, your love and unconditional support. You are the most hard-working people I have ever seen. This has always been inspiring to me and was a source of motivation to me. I know and deeply appreciate all the sacrifices you have made to make this dream of mine comes true.

My deepest gratitude and appreciation goes to my beautiful wife, Marzieh, who is my closest friend before anything else. Thank you so much for all the support and love you gave to me during these years. You played the role of a colleague at work and the best friend at life. No one could help me as you did during the hard times of failure and disappointment. You were always there by my side with your deepest feelings at life and the smartest ideas at work. Thank you for your presence during all we have been through.

Hamidreza
September 2020

Contents

1 Introduction

Until recently, phrase-based machine translation was the state-of-the-art for more than a decade (Och et al., 1999, Och and Ney, 2002, Och, 2003, Koehn et al., 2003, Tillmann, 2004, Chiang, 2007, Galley and Manning, 2008, Cherry and Foster, 2012). Phrase-based machine translation models have recently been challenged and outperformed by several neural machine translation (NMT) models (Sutskever et al., 2014, Bahdanau et al., 2015, Luong et al., 2015b, Jean et al., 2015, Wu et al., 2016, Vaswani et al., 2017). Despite their differences, both phrase-based and neural machine translation models are sophisticated statistical approaches that use a large amount of bilingual data to learn translation models.

Phrase-based models are a combination of different models, each responsible for capturing a specific aspect of the translation task. These models mainly consist of a *translation model*, a *phrase reordering model* and a *language model* (Koehn et al., 2003). This composition makes phrase-based models more modular than neural machine translation models. However, the phrase reordering model in particular is complex. It is responsible for the difficult task of translating phrases in the right order and is still not fully understood. To give an example, it is not clear which words inside a phrase-pair are the most important features for a phrase reordering model. Interpretability of the phrase reordering model is important for two reasons. First, it helps natural language processing (NLP) engineers and practitioners to better adapt the model to their specific use cases. Second, it can benefit our understanding of how the even more complex neural machine translation systems handle similar linguistic phenomena.

Neural machine translation models are end-to-end models that do not split the responsibilities between multiple models as is the case for phrase-based machine translation. This end-to-end nature makes neural machine translation models less interpretable than phrase-based models (Belinkov, 2018, Stahlberg et al., 2018).

Both phrase-based and neural machine translation systems are complex systems that aim to capture complex linguistic phenomena in natural language. In this thesis, we study how we can contribute to the interpretability of both machine translation frameworks. For this study, we follow an empirical probing approach. To this end,

we choose three important aspects of these models and investigate their behavior for capturing different syntactic phenomena. We study phrase reordering models in phrase-based systems and attention modeling and hidden state representations in neural machine translation systems.

The two most successful phrase reordering models in phrase-based machine translation are the lexicalized reordering model (LRM) (Tillmann, 2004, Koehn et al., 2005) and the hierarchical reordering model (HRM) (Galley and Manning, 2008). These models use the full lexical form of phrase-pairs to decide on the correct reordering of the phrase-pairs. However, it is not clear whether it is necessary to use the full lexical form to predict the reordering distribution with reasonable accuracy. LRM and HRM both suffer from the problem of learning unreliable distributions for infrequent phrase-pairs. This problem could be mitigated if it were possible to estimate the reordering distributions using some generalization of the phrase-pairs.

In Chapter 3, we investigate the influence of the words in a phrase on the reordering behavior of the phrase. We reduce the problem of infrequent phrase-pairs by eliminating or generalizing less essential words.

In the next chapters, we shift our study towards neural machine translation models, since these models are shown to better model different aspects of translation including lexical translation and word reordering (Bentivogli et al., 2016).

Attention-based neural machine translation models (Bahdanau et al., 2015, Luong et al., 2015a) have become popular due to their ability to use the most relevant parts of the source sentence for each translation step. In the next part of our study, after having studied phrase reordering models, we investigate what the relevant parts are that the attention model attends to for different linguistic phenomena. Additionally, we compare the attention model with the traditional alignment model to find the differences and similarities between these two models. Next, we examine the distributional behavior of an attention model for different syntactic phenomena within our empirical probing methodology. Understanding attention behavior is important for the interpretability of neural machine translation models since it defines the importance of source hidden states for generating a target word.

We expand our study of interpretability to the hidden states of the encoder of neural machine translation models. These are the hidden states encoding the source side information that the attention models attend to while generating target words. Previous studies use an extrinsic approach of feeding the hidden state representations into different classifiers to find out what information is encoded in those representations (Shi et al., 2016, Belinkov et al., 2017a). Other work has also used such extrinsic approaches to compare different neural sequence-to-sequence architectures in terms of capturing syntactic structure and lexical semantics by the encoder hidden state representations (Tran et al., 2018, Tang et al., 2018a). However, to the best of our knowledge, there is no work that uses an intrinsic approach to study the information captured by the hidden state representations and investigates the difference of these representations with the underlying word embedding representations. The advantage of adopting an intrinsic

approach compared to an extrinsic approach is that it enables us to understand *what* information is captured by the hidden states, as well as *how* this information is captured.

In Chapter 5, we analyse the hidden state representations of different sequence-to-sequence models from a nearest neighbor perspective. We investigate the information captured by the hidden state representations that goes beyond what is transferred from word embeddings. Additionally, we compare two common neural machine translation architectures with respect to the extent to which they capture syntactic and lexical semantic information in their hidden states. This comparison is especially important to our study of interpretability as contrasting two different, yet closely related models allows us to focus our study on the aspects in which both models differ from each other.

1.1 Research Outline and Questions

In this thesis, we study phrase-based and neural machine translation models and shed light on how different syntactic and semantic phenomena in natural language are captured by these models. This research is an attempt to increase the interpretability of the complex machine translation systems. Here, we study phrase reordering models in phrase-based machine translation, attention models in neural machine translation and the internal hidden states also in neural machine translation models. We structure our studies by asking three main research questions and further divide them to subquestions. We ask the following research questions:

RQ1 *To what extent do the orderings of phrases in a phrase-based machine translation model depend on the full lexical forms of the phrases?*

LRMs and HRMs both use the relative frequency of the orientations for phrases in the training corpus to estimate the distributions over orientations conditioned on phrase-pairs. In the case of infrequent phrase-pairs, both models suffer from the problem of insufficient observations to allow for a reliable estimate of the corresponding distributions. Originally, these models rely on the full lexical forms of the phrases when they count the orientations given the phrases. RQ1 investigates whether it is necessary to always use the full lexical form of the phrases when estimating the corresponding distributions.

In the subquestions below, we elaborate more on the primary research question. We approach the problem by first asking which words from the inside of a phrase have most influence on defining the reordering behavior of the phrase. This may result in ignoring less important words, allowing us to use shorter phrase-pairs that decrease the chance of being rarely observed. To this end, we ask the following subquestion:

RQ1.1 *How does the importance of the words of a phrase change in defining the ordering of the phrase?*

We answer this subquestion by experimenting with different patterns for backing-off or marginalizing to shorter sub-phrase-pairs.

In our backing-off experiments, we assume that the importance of internal words of a phrase for reordering changes with their proximity to the borders of the phrase. In these cases, we consider border words as the most important words for defining reordering following Nagata et al. (2006) and Cherry (2013).

In our marginalizing experiments, we consider *exposed heads* (Chelba and Jelinek, 2000, Li et al., 2012) as the important words for reordering, where exposed heads are defined by dependency parse relations. Exposed heads are words from inside a sequence that are in dependency relations with words outside of the sequence and are dominated by those words (Li et al., 2012).

Next, we investigate to what extent we can estimate the reordering distribution of phrases by removing less important words and use only part of the phrase for estimation, analogous to back-off smoothing in n-gram language models. To this end, we ask the following subquestion:

RQ1.2 *How helpful is it to estimate the reordering distribution of phrases by backing off to shorter forms of the phrases and removing less influential words?*

To answer this question, we use different back-off strategies to estimate the reordering distributions of phrase-pairs. We use the resulting reordering model in a phrase-based machine translation system and compare the results with systems based on commonly used reordering models.

Furthermore, we investigate to what extent we can better estimate the distribution of the infrequent phrases by applying class-based generalization for less influential words (RQ1.3):

RQ1.3 *How accurately can the orderings of phrases be predicted by using class-based generalizations of the words in phrases when keeping the most influential words in their original form?*

We answer this question by following the same approach used to answer subquestion RQ1.2. We create multiple phrase reordering models using different phrase generalization patterns and compare the results of using them in the translation system with other reordering models.

In summary, we answer the questions in Chapter 3 by (i) investigating two backing off strategies to use sub-phrase-pairs of the original phrase-pairs to smooth the orientation distribution of the original phrase-pairs, and (ii) experimenting with a phrase generalization approach that uses the linguistic notion of exposed heads (Chelba and Jelinek, 2000, Li et al., 2012) to define which words to generalize and which words to keep in their original forms.

In Chapter 4, we continue investigating the importance of words, but this time from a neural machine translation perspective. As mentioned above, attention-based neural machine translation models (Bahdanau et al., 2015, Luong et al., 2015a) have achieved popularity due to their capability to use the most relevant parts of a source sentence for

each translation step. However, it is unclear what exactly the definition of relevance should be. So the following research question aims to shed light on the criteria of relevance.

RQ2 *What are the important words on the source side that attention-based neural models attend to while translating different syntactic phenomena?*

Earlier machine translation research stipulates that attention models in neural machine translation are similar to traditional alignment models in phrase-based machine translation (Alkhouli et al., 2016, Cohn et al., 2016, Liu et al., 2016, Chen et al., 2016). As a result, there have been some attempts to train attention models with explicit signals from existing traditional alignment models (Alkhouli et al., 2016, Liu et al., 2016, Chen et al., 2016). However, these attempts yield improvements in some domains and no improvements or even drops in performance in other domains. To answer RQ2, we subdivide it into the following subquestions:

RQ2.1 *To what extent does an attention model agree with the traditional alignment model in capturing translation equivalent words?*

In RQ2.1, we question the similarity of traditional alignments in phrase-based machine translation with attention models in neural machine translation. Of course, there are some similarities between these models (Cohn et al., 2016), but we aim to test the extent of this similarity empirically.

We answer RQ2.1 by comparing attention and alignment models using both pre-existing and new measures introduced in our research.

Attention models and traditional alignment models are essentially different, since attention models learn distributions resulting in a soft alignment, whereas traditional alignments are hard alignments models. However, the question is what advantages are brought about by this difference. So we ask the following subquestion:

RQ2.2 *Are the differences between attention and traditional alignments due to errors in attention models or do attention models capture additional information compared to traditional alignment models?*

We answer RQ2.2 by comparing the distributional behavior of attention and the alignment model and their correlation with translation quality.

We observe that attention is not as static as the alignment model. This motivates us to study this phenomena more closely as it could be an advantage of attention model over the alignment model. So we ask the following research subquestion:

RQ2.3 *How does the distribution of the attention model change for different syntactic phenomena?*

RQ2.3 requires a more in-depth investigation of the attention model by analyzing its distributional behavior for different syntactic phenomena and its impact on translation

quality. We define an entropy-based metric to measure how focused or dispersed attention weights are for different syntactic phenomena. We also investigate the correlation of this metric with translation quality and alignment quality to further analyze the distributional behavior of the attention model.

RQ2.4 *What types of words are attended to by the attention model while translating different syntactic phenomena?*

RQ2.4 investigates what word types the attention model attends to on the source side while generating different syntactic types on the target side. For example, what types of words are attended to on the source side when the model generates a noun on the target side? Such an analysis can help explain the differences between the attention model and the alignment model if there are any.

In Chapter 4, we answer these questions by defining metrics to compare attention models to traditional alignment models. We empirically show that these models are different and attention models capture more information than alignment models. We study the behavior of the attention model for different syntactic phenomena and provide an in-depth analysis of what information the attention model attends to during translation. Neural machine translation models follow a general encoder-decoder architecture that encodes the source sentence into distributed representations and then decodes these representations into a sentence in the target language. At each target word generation step, the attention model computes a weighted sum over the encoder hidden representations. This weighted sum is used by the decoder to generate the next word in the target sentence. So far, we asked questions about the behavior of the attention model and the relevant words that the attention model attends to while generating different syntactic phenomena. That means we have assumed that the corresponding hidden state of a source word is representative of that word. However, it is not precisely clear what linguistic information from the source words is encoded in these hidden states. Assuming the corresponding hidden states of a source word to be representative of the source word means that the lexical information encoded in the word embeddings should be transferred to corresponding hidden states. To investigate what information is captured by the hidden states, we ask the following research question:

RQ3 *What information is captured by the hidden states of the encoder of a neural machine translation model?*

RQ3 generally asks about the information encoded in the hidden states. In the subquestions, we break this investigation into more fine-grained steps by questioning the similarity and difference to the underlying word embeddings.

RQ3.1 *How does the captured information in the hidden states differ from the information captured by the word embeddings?*

In RQ3.1, we question the similarity and the difference of the hidden states with the corresponding word embeddings. Word embeddings are distributed representations

of words, encoding the general lexical information about the corresponding words. Hence, if no similarity existed between the hidden states and the corresponding word embeddings, then taking the hidden states as representative of the corresponding source word would not be a correct assumption.

We answer this question by comparing the lists of the nearest neighbors of hidden states with the nearest neighbors of the corresponding word embeddings. We define a quantitative measure for this comparison. This is the first measure used by our intrinsic study.

In the next step, we ask to what extent the difference between the hidden states and the word embeddings is due to the syntactic and lexical semantic information captured by the hidden states:

RQ3.2 *To what extent do the hidden states of the encoder capture different types of information, such as syntactic or semantic information?*

Earlier work has fed the hidden state representations into diagnostic classifiers to reveal the syntactic and semantic information captured by the hidden states (Shi et al., 2016, Belinkov et al., 2017a). However, we take a more intrinsic approach to answer research question RQ3.2. We investigate the nearest neighbors of the hidden states in terms of the captured syntactic structure and the coverage of WordNet (Fellbaum, 1998, Miller, 1995) connections to answer this question. We define two intrinsic measures to quantify the captured information.

Once we have defined our intrinsic measures to interpret the hidden state representation of neural machine translation models, we are able to compare different neural machine translation architectures considering the information they encode from the source side. So, to study how the captured information differs in different NMT architectures we ask the following subquestion:

RQ3.3 *How different is the captured information in the encoder hidden states for different NMT architectures?*

RQ3.3 investigates whether different NMT architectures capture the same information to the same extent. In particular, we investigate how the hidden states in a recurrent neural model (Bahdanau et al., 2015, Luong et al., 2015b) are different from the hidden states in a transformer model (Vaswani et al., 2017).

In Chapter 5, we answer these questions by looking into the nearest neighbor list of the hidden states. We compare these lists to the lists of the nearest neighbors of the corresponding word embeddings to find differences. We also look up the nearest neighbors in the WordNet connections of the corresponding source word of a hidden state to define a metric to measure lexical semantics. We use this metric to compare different models in terms of their capability to capture lexical semantics. We also compare different models by computing the similarity of the local syntactic structures of the nearest neighbors of a hidden state to the local syntactic structure of the corresponding

word of the hidden state. We show that transformer models are better in capturing lexical semantics, whereas recurrent models are superior in capturing syntactic structure. Our observation is in line with the observed results of the extrinsic comparison of these models by Tran et al. (2018) and Tang et al. (2018a).

1.2 Main Contributions

In this thesis, we mainly contribute to the interpretability of phrase-based and neural machine translation models. We provide algorithmic contributions as well as metrics to study the behavior of neural models. We also make empirical contributions by providing a more detailed analysis of the models based on our metrics.

1.2.1 Algorithmic Contributions

1. In Chapter 3, we propose two algorithms to use shortened forms of a phrase-pair to smooth the orientation distribution of the original phrase-pair following back-off smoothing in n-gram language modelling (Chen and Goodman, 1999). These algorithms are applicable to lexicalized (Tillmann, 2004, Koehn et al., 2005) and hierarchical reordering models (Galley and Manning, 2008) yielding slight improvements for these models. Our algorithms use linear interpolation and recursive MAP (Maximum a Posteriori) smoothing (Cherry, 2013, Chen et al., 2013) to combine the shortened forms of the phrase-pairs.

2. In Chapter 3, we also propose four methods to use generalized forms of a phrase-pair to smooth the orientation distribution of the original phrase-pair. These generalized forms are produced by keeping essential words and marginalizing other words. Our proposed methods are applicable to both LRM and HRM models and improve the reordering performance of these models.

3. In Chapter 4, we propose an approach to compare attention with traditional alignment. We define various metrics to measure the agreement between attention and alignment and to investigate the relationship of attention and alignment agreement with translation quality.

4. In Chapter 5, we propose an intrinsic approach to study the syntactic and lexical semantic information captured by the hidden state representations based on their nearest neighbors.

5. In Chapter 5, we define metrics that provide interpretable representations of the information captured by the hidden states in NMT systems, highlighting the differences between hidden state representations and word embeddings.

1.2.2 Empirical Contributions

1. In Chapter 3, for the phrase reordering models, we show how keeping essential words of a phrase-pair and marginalizing out the other words lead to generalized forms of the phrase-pair that improves the performance of the reordering model when being used to smooth the reordering distribution.

2. In Chapter 3, we provide an in-depth analysis showing that orientation distributions conditioned on long phrase-pairs typically depend on a few words within phrase-pairs and not the whole lexicalized form. As a result, using generalized forms of the phrase-pairs to smooth the original reordering distribution leads to performance improvements of the reordering models.

3. In Chapter 4, we provide a detailed comparison of an attention model in neural machine translation against a word alignment model. We show that the attention and the alignment models have different behaviors and attention is not necessarily an alignment model.

4. In Chapter 4, we also show that while different attention mechanisms can lead to different degrees of compliance with respect to word alignments, a full compliance is not always helpful for word prediction.

5. In Chapter 4, we additionally show that attention follows different patterns depending on the type of word being generated. We contribute to the interpretability of the attention model by providing a detailed analysis of the attention model. Our analysis explains the mixed observations of earlier works that have used alignments as an explicit signal to train attention models (Chen et al., 2016, Alkhouli et al., 2016, Liu et al., 2016).

6. In Chapter 4, we provide evidence showing that the difference between attention and alignment is due to the capability of the attention model to attend to the context words influencing the current word translation. We show that the attention model attends not only to the translation equivalent of the word being generated, but also other source words that may have some effect on the form of the target word. For example, while generating a verb on the target side, the attention model often attends to the preposition, subject or object of the translation equivalent of the verb on the source side.

7. In Chapter 5, we use our proposed intrinsic approach for the analysis of syntactic and lexical semantic information captured by the hidden states to compare transformer and recurrent models in terms of their capabilities to capture this kind of information. We show that transformer models are superior in capturing lexical semantic information, whereas recurrent neural models are better in capturing syntactic structures.

8. In Chapter 5, we additionally provide analyses of the behavior of the hidden states for each directional layer and the concatenation of the states from the directional layers in a recurrent neural model. We show that the reverse recurrent layer captures more lexical semantic information. In contrast, the forward recurrent layer captures more extended context information.

1.3 Thesis Overview

The rest of this thesis is organized as follows. Chapter 2 discusses background and related work. The main research is presented in the following three chapters (Chapters 3–5). Chapter 6 summarizes the main findings of this thesis. Below, we provide a high-level summary of the content of each chapter.

- **Chapter 2 (Background)** gives an introduction to phrase-based and neural machine translation, which are the main machine translation approaches of recent years and are used in the research presented in this thesis. We discuss phrase reordering models including the lexicalized reordering model (LRM) (Tillmann, 2004, Koehn et al., 2005) and hierarchical reordering model (HRM) (Galley and Manning, 2008), which are the focus of the research presented in Chapter 3. Next, we provide an introduction to neural machine translation and attention models in neural machine translation, which are the focus of Chapter 4. We also briefly talk about alignment models to be referred back to in Chapter 4, where we show its contrast with attention models. Subsequently, we discuss hidden state representations and word embeddings in neural machine translation. We also discuss the transformer neural machine translation model, since we compare this model with recurrent machine translation model using their hidden representations in Chapter 5.

- **Chapter 3 (The Impact of Internal Words on Phrase Reorderings)** introduces the problem of reliable estimation of the phrase reordering distribution for infrequent phrase-pairs. We briefly discuss previous methods proposed to improve estimation. We answer RQ1 by experimenting with different ways of shortening phrase-pairs following the idea of backing off in n-gram language models. We experiment with keeping essential words of the phrase-pairs and removing or using generalized forms of the remaining words. We show that the latter approach achieves improvements over strong models based on LRM and HRM.

- **Chapter 4 (What does Attention in Neural Machine Translation Pay Attention to?)** provides a detailed analysis of the attention model in neural machine translation. We answer RQ2 by comparing the attention and alignment models. We define metrics to analyze the behavior of the attention model for different syntactic phenomena. We use part of speech (POS) tags to provide a more fine-grained analysis of the attention behavior. In this chapter, we show that the

discrepancies between attention and alignment are not due to errors in attention for several specific syntactic phenomena. We also show what word types are being attended to when generating specific syntactic word forms and discuss why these attentions are intuitive.

- **Chapter 5 (Interpreting Hidden Encoder States)** takes the research presented in Chapter 4 further to study the information captured by the hidden states of the encoder side of a neural machine translation model. We compare hidden states and word embeddings to reveal the information captured by the hidden states on top of the information captured by embeddings. We investigate what share of the nearest neighbors list of the hidden states are covered by the direct connections of the corresponding source words in WordNet to measure how much of the capacity of the hidden states is capturing lexical semantics. We also look into the similarities of the syntactic structures captured by the nearest neighbors and the hidden states. We use these intrinsic methods to compare recurrent and transformer architectures in terms of capturing syntactic and lexical semantic information. This chapter answers RQ3.

- **Chapter 6 (Conclusion)** concludes this thesis by reviewing the research questions and our answers as the result of this research. We also provide a brief discussion of how this thesis has contributed to the interpretability of phrase-based and neural machine translation models. Finally, we discuss problems that remain open for further studies.

1.4 Origins

The research presented in the Chapters 3–5 is based on the following peer reviewed publications.

- **Chapter 3** is based on Ghader and Monz (2016), *Which Words Matter in Defining Phrase Reorderings in Statistical Machine Translation?*, published in the Proceedings of the Twelfth Conference of the Association for Machine Translation in the Americas (AMTA 2016), Volume 1: MT Researchers' Track, pages 149–162, Austin, TX, USA, Association for Machine Translation in the Americas.

  The back-off model was proposed by Monz and the generalization model was proposed by Ghader. Experiments and analysis were performed by Ghader. Both authors contributed to the writing of the article. Ghader did most of the writing.

- **Chapter 4** is based on Ghader and Monz (2017), *What does Attention in Neural Machine Translation Pay Attention to?*, published in the Proceedings of the Eighth International Joint Conference on Natural Language Processing (IJCNLP2017), Volume 1: Long Papers, pages 30–39, Taipei, Taiwan, Asian Federation of Natural Language Processing.

Ghader proposed the analysis of attention and carried out experiments and analysis. The article was mostly written by Ghader. Monz contributed to the article and discussion.

- **Chapter 5** is based on Ghader and Monz (2019), *An Intrinsic Nearest Neighbor Analysis of Neural Machine Translation Architectures*, published in the Proceedings of Machine Translation Summit XVII, Volume 1: Research Track, pages 107–117, Dublin, Ireland, European Association for Machine Translation.

  Monz suggested the analysis of hidden states. Experiments and analysis were performed by Ghader. Both authors contributed to the article. Ghader did most of the writing.

2
Background

In this chapter, we introduce the two main machine translation paradigms that are used in this thesis. Phrase-based machine translation is used in Chapter 3 and two different types of neural machine translation architecture are used in Chapters 4 and 5. We discuss the components of the models in detail, focusing on the components that are relevant to this thesis.

2.1 Phrase-Based Machine Translation

In phrase-based machine translation (Koehn et al., 2005, 2003, 2007), continuous sequences of words (phrases) play the role of the translation unit. These translation units come in pairs of source and target language phrases. The phrase-pairs are learned from bilingual parallel data using unsupervised learning algorithms. In the next sections, we explain how these translation units are learned from data and are used to translate from the source language into the target language.

2.1.1 Word Alignment

The translation units (phrase-pairs) for phrase-based machine translation are learned from a parallel corpus consisting of a large body of text in a source and a target language which is parallel at the sentence level. This means that each sentence in the target language is the translation of the corresponding sentence in the source language (Koehn et al., 2003). Parallel corpora are typically comprised of hundreds of thousands or millions of sentence pairs.

In order to learn the phrase-pairs from the parallel corpus, the parallel corpus should be word aligned first. This means that words from the source side are aligned with their corresponding word or words from the target side. It can also happen that a word from one side has no corresponding word on the other side to be aligned with. For example, in Figure 2.1, the word “wieder” on the German side and the comma on the english side have no corresponding word on the other side and hence are left unaligned. It can

also happen that one word from the target side is aligned with multiple words from the source side. Therefore, commonly used word alignments are many-to-many (Koehn et al., 2003).

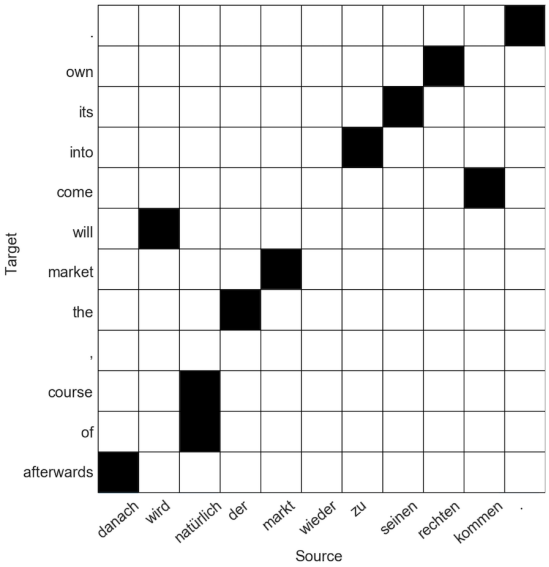


Figure 2.1: An example of a word alignment matrix. As shown, words from the source and target can remain unaligned or can have more than one aligned words on the other side.

Figure 2.1 shows an example of a word alignment matrix. Word alignments were first introduced as part of the IBM models (Brown et al., 1993). IBM models produce one-to-many word alignments. To produce a many-to-many alignment, two alignments are produced by running IBM models on the source-to-target and the inverse direction. Then, the two alignments are merged into one many-to-many alignment in a process called symmetrization. The extreme cases of symmetrization are the union and intersection of the two alignments. However, it has been shown that heuristics that explore the space between the intersection and the union of the two alignments and expand the intersection with the most reliable alignments from the union result in alignments of a better quality (Koehn et al., 2003). One of the commonly used heuristics is called *grow-diag-final-and* (Koehn et al., 2005). Starting from the intersection of the two alignments, the heuristic adds neighboring alignment points that are part of the union but not the intersection.

The quality of an automatic word alignment tool is commonly measured by comparing its output on a test set with a ground truth alignment done by human experts (Och and Ney, 2000, 2003, Fraser and Marcu, 2007). Human experts have to follow some guidelines to solve the ambiguities that come up during the annotation process. One

of these guidelines is to use *sure alignment points* and *possible alignment points* (Och and Ney, 2000, Koehn, 2010). Possible alignments are alignment points where human annotators are not in agreement. For example, function words that do not have a clear equivalent in the other language or two equivalent idiomatic expressions that are not a word-level translation of one another are aligned using *possible alignments*. A commonly used measure of alignment quality is alignment error rate (AER). AER is computed using the following formula:

$$\text{AER}(S, P; A) = 1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|}. \tag{2.1}$$

Here, A is the set of alignment points that is being evaluated. S and P are the sets of sure alignment points and possible alignment points from a gold standard alignment, respectively.

GIZA++ (Och and Ney, 2003) and fast_align (Dyer et al., 2013) are two commonly used tools to compute automatic word alignments. Throughout this thesis, we use GIZA++, which produces alignments for each direction using the IBM models and symmetrizes the alignments by performing grow-diag-final-and heuristic.

Next, we discuss the most basic model of a phrase-based machine translation system. We explain how the model works and how it is created using the output of a word alignment system.

2.1.2 Translation Models

The translation model can be considered the core model of a phrase-based machine translation system. In a word-based model the translation units are pairs of words, but in phrase-based models each item of the translation model or phrase table can be either a pair of words or a pair of phrases in the source and target languages.

Each item in a phrase table has four corresponding scores. Two of these are conditional phrase probability scores and the other two are lexical weightings. Let $\bar{f}$ and $\bar{e}$ stand for a source and a target phrase, respectively, then the phrase translation probabilities are $p(\bar{e} \mid \bar{f})$ and $p(\bar{f} \mid \bar{e})$. Phrase translation probabilities are computed using the counts of the phrase-pairs in the bilingual training corpus:

$$p(\bar{f} \mid \bar{e}) = \frac{C(\bar{e}, \bar{f})}{\sum_{\bar{f}_i} C(\bar{e}, \bar{f}_i)}. \tag{2.2}$$

Here, $C(\bar{e}, \bar{f})$ is the number of times that phrases $\bar{e}$ and $\bar{f}$ are extracted together as a phrase-pair from the bilingual training corpus.

The noisy channel model (Shannon, 1956) forms the basis for phrase-based machine

translation:

$$\begin{aligned} \arg\max_e \ p(e \mid f) &= \arg\max_e \ \frac{p(f \mid e)p(e)}{p(f)} \\ &= \arg\max_e \ p(f \mid e)p(e). \end{aligned} \tag{2.3}$$

Here, f is source sentence, e is target sentence and $p(f \mid e)$ is the likelihood that f is a translation of e. In phrase-based machine translation, the likelihood $p(f \mid e)$ is estimated by breaking f and e into phrases and using the phrase translation probabilities.

Based on the noisy channel definition of translation probability estimation, the probability of source phrases conditioned on target phrases, $p(\bar{f} \mid \bar{e})$, are needed for translation probability estimation. However, if a target phrase is infrequent then the probability will not be a reliable estimate. To improve estimation in such a situations, it can be helpful to use the inverse conditional probability as well (Koehn et al., 2005).

As mentioned before, the other two scores of each item in a phrase table are the lexical weighting scores. These scores are meant to help estimate phrase translation probabilities if a phrase-pair is infrequent in the training corpus. The lexical weightings are also computed in two directions as the conditional phrase probabilities. Lexical weighting is basically a back-off smoothing method to improve phrase probability estimations. In this case, we back off to the probabilities of the words in a phrase, since they are more frequent than the phrase itself and the probability estimates are more reliable. Given the alignment of the words in a phrase-pair, we compute the lexical weighting of a phrase-pair as follows:

$$p_l(\bar{e} \mid \bar{f}, a) = \prod_{i=1}^{\text{length}(\bar{e})} \frac{1}{|\{j \mid (i,j) \in a\}|} \sum_{\forall (i,j) \in a} w(e_i \mid f_j), \tag{2.4}$$

where a refers to the alignment and $w(e_i \mid f_j)$ is the probability of generating word e_i given its aligned word f_j on the source side (Koehn et al., 2003).

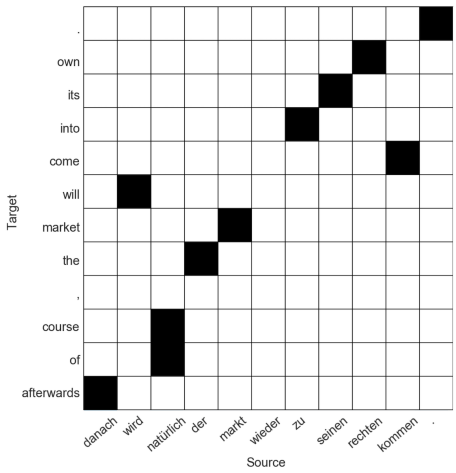
Phrase Extraction

Next, we describe the standard phrase extraction algorithm (Och et al., 1999, Koehn et al., 2003), which we use in Chapter 3. As mentioned in Section 2.1.1, phrases in a phrase table are extracted from word alignments and have to be consistent with the word alignment. Consistency with the alignment is ensured by following three conditions:

1. each phrase-pair includes at least one alignment link;
2. phrases are continuous sequences of words without any gaps; and
3. no word in any of the phrases of a phrase-pair has alignment links to words outside of the phrase-pair.

The phrase extraction algorithm loops over target side sequences and matches all source phrases that satisfy the consistency restrictions. If the matched source phrase has

unaligned words at its borders, it can be expanded by those unaligned words and adds the new phrase-pairs to the extracted phrase-pairs set. There is a limit to the length of extracted phrase-pairs. This limit is typically set to 7 for each of the phrases in a phrase-pair (Koehn et al., 2003). Therefore, the phrase extraction algorithm continues to expand a phrase-pair until either the consistency condition is violated or the length of one of the source or target phrases exceeds the limit.



afterwards <—> danach
of course <—> natürlich
of course , <—> natürlich
of course , the <—> natürlich der
into its own <—> zu seinen rechten
into its own <—> wieder zu seinen rechten
of course , the market <—> natürlich der markt
come into its own <—> zu seinen rechten kommen
come into its own . <—> zu seinen rechten kommen .
of course , the market <—> natürlich der markt wieder
of course , the market will <—> wird natürlich der markt
come into its own <—> wieder zu seinen rechten kommen
come into its own . <—> wieder zu seinen rechten kommen .
of course , the market will <—> wird natürlich der markt wieder

Figure 2.2: Extracted phrase-pairs by the phrase-pair extraction algorithm from the alignment in Figure 2.1.

Figure 2.2 shows extracted phrase-pairs by the phrase extraction algorithm from the alignment given in Figure 2.1.

2.1.3 Log-Linear Model and Parameter Estimation

The feature functions of a phrase-based machine translation are combined in a log-linear model with each feature function having its own weight (Och and Ney, 2002). The formal definition of the model is as follows:

$$p(e \mid a, f) = \exp \sum_{i=1}^{n} \lambda_i h_i(e, f, a), \tag{2.5}$$

where e is a target sentence, f is a source sentence and a is a latent variable representing the phrasal alignment between f and e; $h_i(e, f, a)$ are the feature functions and λ_i are their corresponding weights in the log-linear model. The feature functions are the log-values of the features including phrase probability, inverse phrase probability, and lexical weightings, as introduced in Section 2.1.2, as well as the language model and reordering model probabilities that will be discussed in Sections 2.1.4 and 2.1.5, respectively.

The optimal values for the weight parameters of the log-linear model, i.e., the values for λ_i, are learned on a held-out data set called development set. The process of finding these optimal values is called "tuning." The development set should be close to the target translation task in terms of vocabulary and the domain of the data. The optimal values are learned using a parameter optimization algorithm such as minimum error rate training (MERT) (Och, 2003) or pairwise ranking optimization (PRO) (Hopkins and May, 2011). MERT performs a grid-based optimization by generating translations of the development set for different values of the parameters of the model and compares the generated translations with the corresponding reference translations. Both MERT and PRO can optimize any evaluation metric that provides a comparison of generated translation and reference translation. PRO converts the parameter optimization problem into a binary classification problem by using a pairwise ranking of translation hypotheses. It samples the parameter values from the search space and finds the best value vector by pairwise ranking. PRO can optimize systems with a large number of features whereas MERT is not scalable to handle more than a dozen parameters. For tuning the phrase-based model we use PRO, see Section 2.3.

2.1.4 N-Gram Language Models

To ensure the fluency of the generated translations, phrase-based machine translation leverages n-gram language models (Och and Ney, 2004). An n-gram language model computes the probability of the next word given the history of the previously generated words. However, the length of the history is limited by n which is the order of the language model. The order of the language model needs to be large enough to capture the specificity of different sequences and small enough that the learned distributions are not too sparse. An n-gram language model is trained on a large monolingual corpus in the target language. The language model is trained using the relative frequency of sequences of words with length of n. For example, for a language model of order 3, this is defined as:

$$P(w_3 \mid w_1, w_2) = \frac{C(w_1 w_2 w_3)}{\sum_{w_i} C(w_1 w_2 w_i)}, \tag{2.6}$$

where $C(w_1 w_2 w_3)$ is the count of string "$w_1 w_2 w_3$" in the training data. To compute the probability score of a sentence, the probabilities of all words in the sentence given their $n-1$ words histories are multiplied:

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

It may happen that while scoring a sentence, sequences of words occur that have never been encountered during training. A maximum likelihood n-gram language model would assign a zero probability score to such a sequence, and hence a zero probability to the full sentence. To avoid this, the language model needs to be smoothed.

There are various smoothing methods including Kneser-Ney (Kneser and Ney, 1995, Chen and Goodman, 1999), Add-k smoothing (Gale and Church, 1994), and Backoff and Interpolation (Jelinek and Mercer, 1980). We briefly discuss the Backoff and Interpolation method, since we refer back to it in Chapter 3. In the back-off method, we back off to a lower order n-gram $(n-1)$ whenever the count for the n-gram in the training data is zero. We continue backing off until we reach a shorter history that has a non-zero count. The formal definition of the back-off method is:

$$P_b(w_i \mid w_{i-n+1}^{i-1}) = \begin{cases} \hat{P}(w_i \mid w_{i-n+1}^{i-1}), & \text{if } C(w_{i-n+1}^{i}) > 0 \\ \alpha_{w_{i-n+1}^{i-1}} P_b(w_i \mid w_{i-n+2}^{i-1}) & \text{otherwise.} \end{cases} \tag{2.8}$$

Here, w_{i-n+1}^{i-1} is the n-gram history sequence, $C(w_{i-n+1}^{i})$ is the count of n-gram in training data, $\hat{P}(w_i \mid w_{i-n+1}^{i-1})$ is the discounted probability distribution which leaves some probability mass for unseen n-gram sequences and $\alpha_{w_{i-n+1}^{i-1}}$ is the back-off weight which specifies how much of the left-over probability mass is assigned to the current unseen sequence. The discounted probability distribution $\hat{P}(w_i \mid w_{i-n+1}^{i-1})$ and the back-off weight $\alpha_{w_{i-n+1}^{i-1}}$ can be computed by the Good-Turing method (Katz, 1987).

Another method to use lower order n-gram probabilities to estimate the probability of unseen n-grams is interpolation. Here, the estimated probability for an n-gram sequence is a linear interpolation of the maximum likelihood probabilities for the n-gram sequences with varying values for n. For example, for a 3-gram sequence, the estimated probability is computed as follows:

$$\hat{P}(w_n \mid w_{n-2}^{n-1}) = \lambda_1 P(w_n \mid w_{n-2}^{n-1}) + \lambda_2 P(w_n \mid w_{n-1}) + \lambda_3 P(w_n), \tag{2.9}$$

where the λs sum to 1. Here, λ_1, λ_2 and λ_3 are weights that can be learned from data using expectation maximization (Dempster et al., 1977, Federico, 1996). As can be seen, in this method, we include shorter histories, which are more frequent, to obtain more reliable estimates of the probability of the n-gram.

2.1.5 Reordering Models

Different languages have different word orderings. To translate from one language to an other by using phrases as the unit of translation, a model needs to select source phrases for translation in the order that is natural in the target language. Multiple reordering models have been proposed for phrase-based machine translation. We describe three that are used in Chapter 3.

Linear Reordering

The simplest reordering model for phrase-based machine translation is the linear reordering model or linear distortion model (Koehn et al., 2003). The linear reordering

model is a simple function that penalizes long jumps in the source side while translating. The penalty depends on the distance between the last translated phrase and the current one. The distance is computed by using the position of the last word of the previously translated phrase and the first word of the current phrase. This function basically encourages translating the source phrases in the same order as they appear on the source side assuming that monotone translation is the best strategy in most cases. The formal definition of linear distortion is:

$$d(x) = \alpha^{|start_i - end_{i-1} - 1|}, \tag{2.10}$$

which is an exponential decay function with $0 < \alpha < 1$. The only parameter controlling the reordering cost in linear reordering is the distance and therefore does not depend on any lexical information. In this case, the language model controls the lexical aspect of the reordering to some extent by computing the fluency score of the resulting translations based on different reorderings.

Lexicalized Reordering

As described above, the linear distortion model does not leverage any lexical information to control reordering of the phrases. However, there are many situations where the reordering of a phrase has some relation with the phrase and its context (neighboring phrase). The lexicalized reordering model (LRM) (Tillmann, 2004, Koehn et al., 2005) addresses these kinds of situation by conditioning the reordering movements on the lexical form of a phrase and its neighbors.

Since many phrase-pairs are not frequent enough to observe all possible reorderings during translation, the lexicalized reordering model defines a limited number of possible reordering moves to compensate for this sparsity. These reordering moves are referred to as "orientations." The original lexicalized reordering model originally includes three orientations: (i) monotone (M), (ii) swap with previous phrase (S), and (iii) discontinuous jumps (D). Some variants of the lexicalized reordering model split discontinuous jumps into two types of orientations: discontinuous left (DL) and discontinuous right (DR), since it is more linguistically motivated (Galley and Manning, 2008). In this thesis, we use the variant with four orientations (M, S, DL and DR).

In a lexicalized reordering model we use two probability distributions to decide about the reordering: (i) the probability of the orientations given the phrase-pair being translated and (ii) the probability of the orientations given the last translated phrase-pair. The probability of the orientations given the phrase-pair being translated is called *the reordering probability with respect to the left of a phrase-pair*. Similarly, the probability of the orientations given the last translated phrase-pair is called *the reordering probability with respect to the right of a phrase-pair*.

Lexicalized reordering probability distributions are learned from word aligned data. To this end, we count the number of times each phrase-pair appears with different orientations in the word-aligned training data. Then, the probability distribution is

estimated using the relative frequency of the counts:

$$P(o \mid \bar{f}, \bar{e}) = \frac{C(o, \bar{f}, \bar{e})}{\sum_{o'} C(o', \bar{f}, \bar{e})}. \tag{2.11}$$

Here, $C(o, \bar{f}, \bar{e})$ refers to the number of times a phrase-pair co-occurs with orientation o.

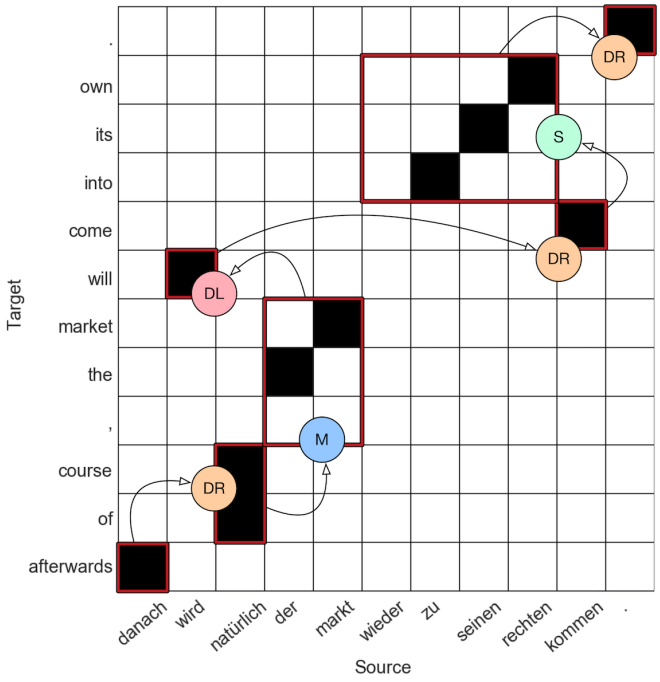


Figure 2.3: An example of phrase-pairs with their corresponding reordering orientations for a given word alignment. The highlighted boxes are the phrase-pairs that are used for translation. Note that other sequences of phrase-pairs that are consistent with the word alignments could also be used.

Figure 2.3 shows an example of how orientations are identified.

Hierarchical Reordering

The hierarchical reordering model (Galley and Manning, 2008) has the same orientations as a lexicalized reordering model. It also uses the conditional probability distributions of the orientations given phrase-pairs. However, the way the orientations are counted during training of the model and the way the right orientation is identified during translation are different.

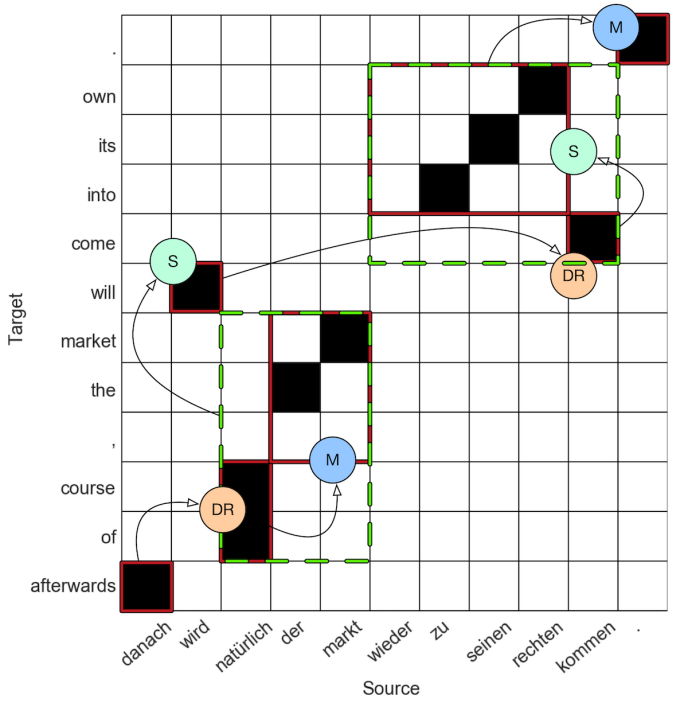


Figure 2.4: The changes to the orientations of Figure 2.3 if a hierarchical reordering model is used in place of the lexicalized reordering model. See the orientations at *(wird, will)* and *(., .)* phrase-pairs.

During training, the hierarchical reordering model uses the longest possible phrase-pairs that the phrase extraction algorithm could have extracted if there was no limit for the phrase-pair length. This considerably reduces the number of discontinuous orientations and increases the number of swap and monotone orientations. During translation, the algorithm uses a shift-reduce parser to merge the currently translated phrase-pairs to the longest possible phrase-pairs and identifies the right orientation for the next phrase-pair with respect to the long phrase-pair formed in the neighborhood. The shift-reduce parser can be replaced by an approximation method that uses the source coverage vector to approximate the top block in the stack of the shift-reduce parser (Cherry et al., 2012).

Figure 2.4 shows how the orientations in Figure 2.3 change if a hierarchical reordering model is used in place of a lexicalized reordering model.

The dashed groupings are adjacent phrase-pairs grouped by the hierarchical reordering model. We have not shown all groupings that a hierarchical reordering model would create in this example, but those that result in a change in the orientations shown in Figure 2.3.

For almost more than a decade phrase-based machine translation was the stat-

of-the-art model. However, it was suffering from some of the problems including but not limited to reordering problems (Cherry et al., 2012, Cherry, 2013, Wuebker et al., 2013, Chahuneau et al., 2013, Li et al., 2014). Neural machine translation showed great potential in the early stages by improving different aspects including reorderings (Bentivogli et al., 2016). Next, we describe neural machine translation as we use it in Chapters 4 and 5.

2.2 Neural Machine Translation

Devlin et al. (2014) showed that a feed-forward neural language model (Bengio et al., 2003) with its conditioning history expanded to the source phrase in a machine task can outperform strong phrase-based machine translation models by a large margin. This showed the great potential of neural language models for machine translation with additional research adapting neural language models to end-to-end machine translation (Kalchbrenner and Blunsom, 2013, Sutskever et al., 2014, Bahdanau et al., 2015, Cho et al., 2014b, Luong et al., 2015b, Jean et al., 2015, Wu et al., 2016, Vaswani et al., 2017).

In this section, we describe neural machine translation and two commonly used model architectures.

2.2.1 Neural Language Models

Like n-gram language models, neural language models also compute the conditional probability of the next word, given a sequence of previous words. However, neural language models are more powerful when it comes to computing probabilities for unseen sequences. Therefore, there is no need for smoothing in neural language models as it is the case for n-gram language models.

Feed-Forward Language Models

Bengio et al. (2003) made it possible to effectively learn the conditional probability of the next word given a limited history of the previous words by introducing a distributed representation of words. This distributed representation which is nowadays called *word embedding* was the key concept for neural networks to open their way into language modeling and machine translation (Mikolov et al., 2010, Zou et al., 2013, Cho et al., 2014a,b, Vaswani et al., 2013).

In the feed-forward neural language model by Bengio et al. (2003), the history of the n-gram is given as the input and the probability score of each of the words from a predefined vocabulary appearing as the next word is returned as the output. The input words are encoded in one-hot vectors and are fed to a dictionary-like lookup layer that returns a high dimensional real-number vector representation for each word. The same high dimensional vector is returned for a word independent of the position of the word

in a sentence:

$$C(w_i) = Cw_i. \tag{2.12}$$

Here, C is the word embeddings weight matrix and $C(w_i)$ is the word embedding of word w which is shown by its index w_i.

The output of this layer is then fed to a hidden layer with a non-linear activation function such as the hyperbolic tangent or $\tanh$.

$$h = \tanh(b_h + \sum_i H_i C(w_i)). \tag{2.13}$$

Here, H_i are the weights of the i-th unit in the hidden layer and b_h is the bias of the hidden layer.

Finally, the output of the hidden layer is fed to a softmax function to ensure the characteristics of a probability distribution:

$$\begin{aligned} s &= Wh, \\ p_i &= \text{softmax}(s_i, s) = \frac{e^{s_i}}{\sum_j e^{s_j}}. \end{aligned} \tag{2.14}$$

Here, W is the weights of the output embedding layer. In the final layer, a softmax function converts the output to a probability distribution over the vocabulary. This formulation of the feed-forward language model is based on (Koehn, 2020) and eliminates the skip layer connections in the version by Bengio et al. (2003).

Recurrent Neural Language Models

The history that the feed-forward language model can capture is limited in the same way as n-gram language models. To expand this to longer histories, allowing one to capture longer dependencies, recurrent neural language models were introduced (Mikolov et al., 2010). The basic idea in these models is to feed back the state of the hidden layer in the previous time step to the current time step to keep track of the previous decisions of the network. The hidden layer plays the role of a representation for the whole history observed up to a given point in time. Since the history can become very long and the effect of earlier observations by the network vanishes in the hidden layer representation, long short-term memory models (LSTM) (Hochreiter and Schmidhuber, 1997) are used for modeling sentences (Sundermeyer et al., 2012). LSTM units have a memory state allowing them to store information for longer periods of time. Additionally, the gates of LSTM units make it possible to control the information in the memory state to be fully or partially changed at each time step. These properties of an LSTM unit help recurrent neural networks with LSTM units avoid the problem of vanishing gradients for longer histories (Le et al., 2015, Salehinejad et al., 2018).

Figure 2.5 shows the general flow of information in a recurrent neural language model. The model digests one word at a time from the sequence and outputs the

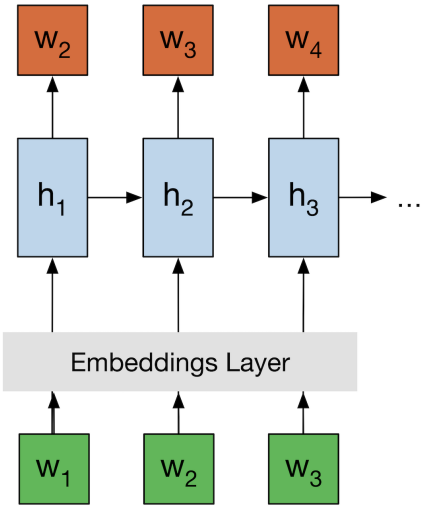


Figure 2.5: The flow of information in a recurrent neural language model. The model reads a word from the input sequence at each time step and predicts the next word. The predicted word is not necessarily the same as the next word in the input sequence. This is reflected by the difference in the colors.

probability score for each word of a predefined vocabulary to appear as the next word of the sequence given the entire history the network has observed so far. The highest scoring word is the predicted next word.

2.2.2 Word Embeddings

Word embeddings are high dimensional vector representations of words based on the context in which they appear. The capability of representing words in their context by high dimensional vectors of real numbers is a key concept to the success of the neural networks in language modeling, machine translation and other NLP tasks (Bengio et al., 2003, Mikolov et al., 2013b, Pennington et al., 2014, Zou et al., 2013). The core idea behind word embeddings is that words that occur in similar contexts are semantically similar and therefore should have similar representations (Firth, 1957, Miller and Charles, 1991). The word embedding layer in neural networks is typically a look-up table matrix of weights that is learned during training of the neural network for a language generation task (typically language modeling). An interesting and useful characteristic of word embeddings is that the similarity between words can be quantified by cosine distance (or similarity) between their respective embedding representations. Word embeddings also offer the possibility of adding and subtracting word vectors to achieve some sort of semantic inference (Mikolov et al., 2013b).

Word embeddings capture a general representation of all senses of a multi-sense

word. Hence, the most frequent sense of a word in the training data is the dominant sense captured by an embedding representation (Faruqui et al., 2016, Fadaee et al., 2017).

In neural machine translation, it is common to learn the word embedding during training of the machine translation model (Qi et al., 2018, Bahdanau et al., 2015, Luong et al., 2015b, Jean et al., 2015, Wu et al., 2016, Vaswani et al., 2017), in contrast to most other language-specific tasks that often use pre-trained word embeddings (Qi et al., 2018, Ma and Hovy, 2016, Lample et al., 2016). Nevertheless, it is also possible to use pre-trained word embeddings in neural machine translation (Qi et al., 2018).

2.2.3 Recurrent Neural Translation Models

With long dependencies and long reorderings being very difficult phenomena for phrase-based machine translation to capture (Galley and Manning, 2008, Green et al., 2010, Durrani et al., 2011, Bisazza and Federico, 2016), recurrent neural machine translation appeared to be able to solve these shortcomings of phrase-based machine translation (Bentivogli et al., 2016, Bahdanau et al., 2015). Recurrent neural translation models are end-to-end models that encode source sentences as high dimensional distributed representations and then decode the representations into a translation in the target language. Both the encoder and decoder can be LSTM (Hochreiter and Schmidhuber, 1997) or GRU (Cho et al., 2014b) networks.

Recurrent neural translation models are extensions to the recurrent neural language models. In recurrent language models, we predict the next word given the words generated so far in a sequence. In recurrent neural translation model, we predict one word at a time, given the translated words generated so far and the high dimensional representation of the source sentence.

Formally, the source sentence can be seen as a sequence of embedding vectors:

$$S = (v_1, \ldots, v_{T_s}). \tag{2.15}$$

The encoder reads the source sentence one vector at a time into a hidden recurrent layer:

$$h_t = f(v_t, h_{t-1}), \tag{2.16}$$

where h_t is the output of the hidden layer at time t and $f(v_t, h_{t-1})$ can be an LSTM (Hochreiter and Schmidhuber, 1997) or GRU (Cho et al., 2014b) recurrent layer applied to the previous hidden state and the current input word.

Then, the sequence of the hidden states generated by the observation of the input word at each time step is used to generate a hidden state representation of the input sentence which is referred to as context vector:

$$c = q(h_1, \ldots, h_{T_s}) \tag{2.17}$$

In the simplest recurrent neural machine translation model $q(h_1, \ldots, h_{T_s})$ returns the last hidden state of the encoder as the output. This function can be more complex including various types of attention functions, which are discussed in Section 2.2.4.

The decoder predicts the next translation word given the context vector $c = q(h_1, \ldots, h_{T_s})$ and all predicted target words so far.

$$r_t = f'(r_{t-1}, y_{t-1}, c) \tag{2.18}$$

$$p(y_t \mid y_1, \ldots, y_{t-1}, c) = g(y_{t-1}, r_{t-1}, c). \tag{2.19}$$

Here, f' can be an LSTM or GRU recurrent layer, r_t is the hidden state of the decoder recurrent layer at time t, g is a linear transformation followed by a softmax function, and $y_1, \ldots, y_{t-1}$ are the target words predicted so far.

Some variants of encoders use bidirectional recurrent layers to generate the encoder hidden states at each time step (Bahdanau et al., 2015). In these models, one recurrent layer encodes the source sentence from left to right and the other layer encodes it from right to left. Then, the output hidden states of each layer at time step t are concatenated to form the output hidden state of the encoder at time step t. The bidirectional encoding causes the model to learn more temporal dependencies in the source sequence and has been shown to achieve performance improvements (Zhou et al., 2016).

2.2.4 Attention Models

In the case of long source sentences, the capacity of a single encoder hidden state is not enough to capture all information from the source side that is needed for the decoder to generate a correct translation (Bahdanau et al., 2015). Cho et al. (2014a) show that the performance of the vanilla recurrent neural model indeed deteriorates for long sentences as a result of the limited capacity of the output representation of the encoder. To address this problem, the context vector c in Equation 2.17 is computed in a more complex way that learns to choose the required information from the source side needed for target word generation at each time step.

These so-called attention models define the context vector c to be a weighted average of encoder hidden states from all time steps:

$$c_i = \sum_{i=1}^{|S|} \alpha_{t,i} h_i. \tag{2.20}$$

The weights $\alpha_{t,i}$ are computed by a softmax over the similarity of hidden states from the encoder and the current hidden state of the decoder:

$$e_{t,i} = h_i^T r_t \tag{2.21}$$

$$\alpha_{t,i} = \frac{\exp(e_{t,i})}{\sum_{j=1}^{|S|} \exp(e_{t,j})}. \tag{2.22}$$

Instead of using a single vector to generate the entire target translation as is the case in earlier neural machine translation models (Sutskever et al., 2014, Kalchbrenner and Blunsom, 2013, Cho et al., 2014a), the context vector changes at each time step during the generation of the target sentence and assigns more weight to the relevant parts of the source sentence.

There are more complex types of attention that keep track of the amount of attention payed to different parts of the source side at previous time steps (Bahdanau et al., 2015, Luong et al., 2015a, Feng et al., 2016). We introduce one of these attention models in more detail in Section 4.3.

2.2.5 Transformer Model

Vaswani et al. (2017) showed that attention can be used to replace the capabilities that recurrent neural networks provide for machine translation. Transformers are also encoder-decoder models in which both the encoder and decoder use fully attentional networks.

Attention is a core component in the transformer architecture. Here, we introduce the notation used to describe attention in the transformer model. We use general notation that can describe all different attentions in a transformer architecture.

In the transformer model, each word embedding is mapped to three types of vector: query, key and value, using three different linear transformations for which the weights are learned during training.

In the attention model, query vectors are compared to key vectors by using the dot product. The result of the dot product is scaled by the square root of the dimension size of the vectors. The result of the scaled dot product of a query and the keys is then passed through a softmax function, and the result is used as weights for the value vectors to form the output of the attention function. This is shown by the following formula:

$$A(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{2.23}$$

Here, Q stands for queries, K stands for keys, V stands for values and d_k is the size of query and key vectors.

The difference between the attention described here and the attention model described in Section 2.2.4 is the scaling factor and the linear mappings to query, key and value vectors. Otherwise, the attention model used in recurrent neural machine translation can also be described by using the current general notation.

Multi-Head Attention

Attention is the only mechanism that gathers all information from the source sentence as well as the prefix of the target sequence in the transformer architecture. This is in contrast to the usage of recurrence and attention in recurrent neural models. Vaswani

et al. (2017) also introduce multi-head attention which allows attention to attend to information from different representation subspaces. The difference in the functionality of each head is realized by different linear projections for each head.

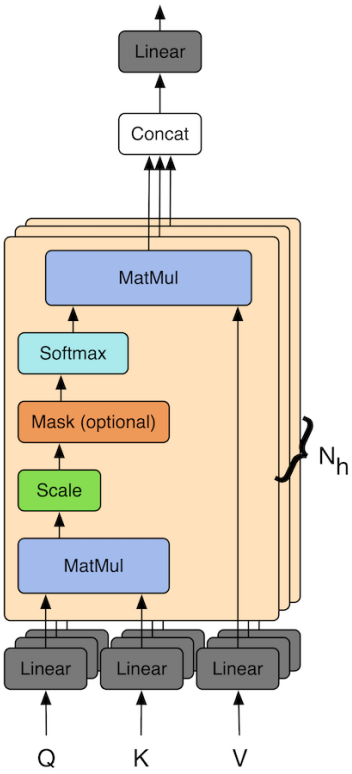


Figure 2.6: Multi-head attention.

Figure 2.6 shows how multi-head attention works. Typically the number of heads being used is set to 8 following Vaswani et al. (2017). The output vectors of each head are concatenated and passed through another linear transformation to form the ultimate output vector of the attention layer. The formal definition of multi-head attention is as follows:

$$A_m(Q, K, V) = \text{concat}(h_1, \ldots, h_n)W^o \text{ where } h_i = A(QW_i^Q, KW_i^K, VW_i^V). \tag{2.24}$$

Here, W_i^Q, W_i^K and W_i^V are the weights for linear transformation of queries, keys and values for ith attention head, respectively. W^o are the weights of the output linear transformation.

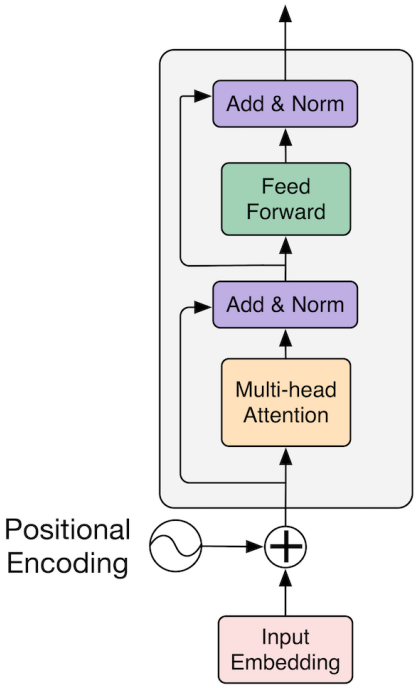


Figure 2.7: Encoder of a transformer.

Encoder

The encoder in a transformer architecture is composed of stacked network layers each of which contains a multi-head self-attention (Parikh et al., 2016) and a feed-forward sub-layer preceded by layer normalization (Ba et al., 2016). Figure 2.7 shows the architecture of one of these layers. This layer is stacked n times to form the encoder.

Words from the input sequence are looked up in an embedding layer just as in a feed-forward or recurrent language model. Since the input is read one word at a time and there is no recurrence to encode and transfer the order of information, we need some type of position information for each input word. To this end, a positional encoding (Gehring et al., 2017) is added to each word embedding. A positional encoding is a vector of the same dimensionality as the word embeddings to allow for vector addition. Positional encodings can be embeddings that are learned during training just as the word embeddings, or it can be a fixed vector.

Word embeddings together with the positional encodings go into a multi-head self-attention layer. In this self-attention layer all queries, keys and values are computed from the word embeddings and positional encodings. This layer learns to attend to different parts of the source sentence depending on which source word is considered as

the query. Then, the output of self-attention added with the positional encoded word embedding goes through a layer normalization component. The positional encoded word embeddings are connected by high-way or residual connections (He et al., 2016) to the output of the self-attention layer, see Figure 2.7. The result is passed through a two-layer simple feedforward network with the following formal definition:

$$f(x) = \max(0, xW_1 + b_1)W_2 + b_2. \tag{2.25}$$

A ReLU activation function, the $\max(0, y)$ function in Equation 2.25, is used between the two linear transformation layers of the feed-forward layer. W_1 and W_2 are the weights of the linear layers and b_1, b_2 are the respective biases.

The size of the first layer of the feed-forward network is commonly set to 4 times the model size while the input and the output size are equal to the model size (Vaswani et al., 2017). For the feed-forward layer also the output and the input are added and the result goes through layer normalization. The process of going through self-attention and then a feed-forward network is repeated n times using identical architectures but different weights.

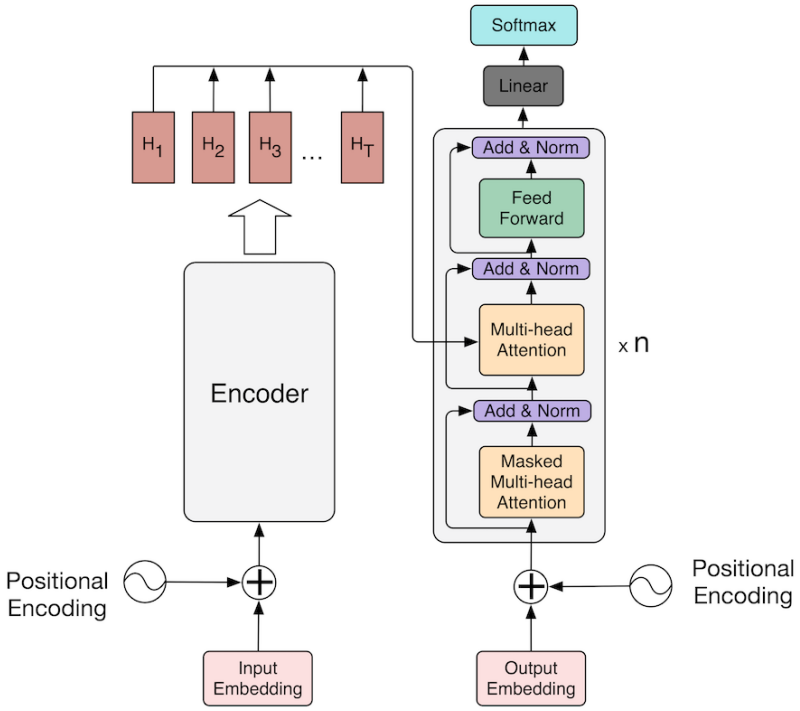


Figure 2.8: Decoder of a transformer as part of the whole architecture.

Decoder

The decoder in the transformer architecture has three main sub-layers: a multi-head self-attention, a multi-head attention which attends to hidden state outputs of the encoder, and a feed-forward network similar to the one in the encoder.

Figure 2.8 shows the decoder as part of the full architecture of the transformer model. As can be seen, the architecture of a decoder layer is very similar to the encoder layer, see Figure 2.7, with an extra multi-head attention sub-layer which attends to the encoder outputs.

The first multi-head attention sub-layer in the decoder is masked for target words that occur after the current position in the target sentence. This masking is required to make sure that the model learns only to depend on the previously generated target words when predicting the next word.

The second multi-head attention performs a similar task to the attention layer in a recurrent neural machine translation model. This sub-layer compares the current state in the decoder, which is the output of the masked attention added with its input through a residual connection, to all the output hidden states from the encoder stack. Based on this comparison, it computes the attention weights. The attention weights are then used to compute a weighted average over the output hidden states of the encoder.

The rest of the decoder layer is identical to an encoder layer. The ultimate output of the decoder stack is transformed linearly and goes through a softmax layer to be converted to a probability distribution over the entire target vocabulary.

2.3 Evaluation

In this thesis, we use automated metrics to evaluate the translation quality of our systems. We use BLEU (Papineni et al., 2002) as the most common evaluation metrics for all our systems. BLEU is a precision-based evaluation metric that matches n-grams in the translation output with (multiple) human reference translations. The BLEU metric computes precisions for different n-grams (usually up to 4-gram) and combines the precisions by multiplying them. Since precision encourages short translations, BLEU uses a *brevity penalty* to compensate for this preference.

We also use two other evaluation metrics that are more sensitive to reorderings in Chapter 3. We use translation error rate (TER) (Snover et al., 2006), which computes a word-based edit distance between a translation and corresponding reference translation inspired by the Levenshtein distance. TER has one more editing step in addition to the insertion, deletion and substitution editing steps from the Levenshtein distance. This extra edit is a block movement called jump. Using its list of editing steps, TER finds the shortest sequence of editing steps to convert a translation output into the corresponding human reference translation. The last evaluation metric we use is RIBES (Isozaki et al., 2010). RIBES is a rank correlation based evaluation metric designed for translation tasks with language pairs that require long-distance reorderings. RIBES aligns words

between the translation output and the human reference translation and computes the rank correlation of the aligned word positions.

In the next chapter, we will discuss phrase reordering models and how different internal words of phrase-pairs can have a different impact on phrase reordering.

3

The Impact of Internal Words on Phrase Reorderings

3.1 Introduction

Structural differences between languages make the correct reordering of phrases an important and complex aspect of phrase-based machine translation systems (Bisazza and Federico, 2016). To this end, there is a lot of work on modeling reordering for phrase-based machine translation (Tillmann, 2004, Galley and Manning, 2008, Green et al., 2010, Durrani et al., 2011, Cherry et al., 2012, Cherry, 2013).

The introduction of lexicalized reordering models (LRMs) (Tillmann, 2004) was a significant step towards better reordering, outperforming the distance-based reordering model (Koehn et al., 2005), by modeling the orientation of the current phrase-pair with respect to the previously translated phrase; see Section 2.1.5 for a detailed introduction. LRMs score the order in which phrases are translated by using a distribution over orientations conditioned on phrase-pairs. Typically, the set of orientations consists of: *monotone* (M), *swap* (S) and *discontinuous* (D), which is often split into *discontinuous left* (DL) and *discontinuous right* (DR). However, LRMs are limited to reorderings of neighboring phrases only. Galley and Manning (2008) proposed a hierarchical phrase reordering model (HRM); see Section 2.1.5, for more global reorderings. HRMs handle long-distance reorderings better than LRMs by grouping adjacent phrase-pairs into longer phrase-pairs and determining the orientations with respect to these longer phrase-pairs.

LRMs and HRMs both use relative frequencies observed in a parallel corpus to estimate the distribution of orientations conditioned on phrase-pairs. As a result, both suffer estimating unreliable distributions for cases that rarely occur during training and therefore have to resort to smoothing methods to alleviate sparsity issues.

Table 3.1: Examples of similar phrase-pairs and their orientation probabilities using Dirichlet (Equation 3.1) and Recursive MAP (Equation 3.2) smoothing. M = monotone, S = swap, DL = discontinous left, and DR = discontinous right.

				Dirichlet Smoothed			
	Source	Target	Freq	M	S	DL	DR
a	中国 政府	chinese government	2834	0.216	0.034	0.315	0.433
b	日本 政府	japanese government	580	0.157	0.039	0.299	0.503
c	尼泊尔 政府	nepalese government	11	0.525	0.001	0.101	0.370

				Recursive Map Smoothed			
	Source	Target	Freq	M	S	DL	DR
a	中国 政府	chinese government	2834	0.216	0.034	0.315	0.432
b	日本 政府	japanese government	580	0.158	0.039	0.300	0.501
c	尼泊尔 政府	nepalese government	11	0.400	0.009	0.202	0.388

3.1.1 Problem Definition

In order to smooth the original maximum likelihood estimation, LRMs originally back off to the general distribution over orientations:

$$P(o \mid \bar{f}, \bar{e}) = \frac{C(o, \bar{f}, \bar{e}) + \sigma P(o)}{\sum_{o'} C(o', \bar{f}, \bar{e}) + \sigma}, \tag{3.1}$$

which is also known as Dirichlet smoothing, where $\sigma P(o)$ denotes the parameters of the Dirichlet prior that maximizes the likelihood of the observed data. $C(o, \bar{f}, \bar{e})$ refers to the number of times a phrase-pair co-occurs with orientation o, and σ is the *equivalent sample size*, i.e., the number of samples required from $P(o)$ to reflect the observed data (Smucker and Allan, 2005). Cherry (2013) and Chen et al. (2013) introduce recursive Maximum a Posteriori (MAP) smoothing, which makes use of more specific priors by recursively backing off to orientation priors, see Equation 3.2:

$$\begin{aligned}
P(o \mid \bar{f}, \bar{e}) &= \frac{C(o, \bar{e}, \bar{f}) + \alpha_s P_s(o \mid \bar{f}) + \alpha_t P_t(o \mid \bar{e})}{\sum_{o'} C(o', \bar{e}, \bar{f}) + \alpha_s + \alpha_t} \\
P_s(o \mid \bar{f}) &= \frac{\sum_{\bar{e}} C(o, \bar{f}, \bar{e}) + \alpha_g P_g(o)}{\sum_{o', \bar{e}} C(o', \bar{f}, \bar{e}) + \alpha_g} \\
P_t(o \mid \bar{e}) &= \frac{\sum_{\bar{f}} C(o, \bar{f}, \bar{e}) + \alpha_g P_g(o)}{\sum_{o', \bar{f}} C(o', \bar{f}, \bar{e}) + \alpha_g} \\
P_g(o) &= \frac{\sum_{\bar{f}, \bar{e}} C(o, \bar{f}, \bar{e}) + \alpha_u / 3}{\sum_{o', \bar{f}, \bar{e}} C(o', \bar{f}, \bar{e}) + \alpha_u}
\end{aligned} \tag{3.2}$$

While recursive MAP smoothing factorizes phrase-pairs into source and target phrases, it still considers the phrases themselves as fixed units.

To better understand what kind of information is ignored by both of the aforementioned smoothing methods, consider the phrase-pairs and their corresponding orientations distributions given in Table 3.1. The phrase-pairs in rows (a) and (b) are frequently observed during training, resulting in reliable estimates. On the other hand, the phrase-pair in row (c) is infrequent, leading to a very different distribution due to the smoothing prior, while being semantically and syntactically close to (a) and (b). In Table 3.1, we can also see that recursive MAP smoothing results in slightly more similar distributions compared to plain Dirichlet smoothing but the overall differences remain noticeable.

In this chapter, we argue that in order to obtain smoother reordering distributions for infrequent phrase-pairs such as the ones in Table 3.1, one has to take phrase-internal information into account.

3.1.2 Research Questions

In earlier work, Cherry (2013) builds on top of HRMs and proposes a sparse feature approach that uses word clusters instead of fully lexicalized forms for infrequent words to decrease the effect of sparsity on the estimated model.

Nagata et al. (2006) and Durrani et al. (2014) have used part of speech (POS) tags and different types of word classes instead of the original forms of words to address the unreliable estimation of the reordering distribution for infrequent phrases. In particular, Nagata et al. (2006) have used the first and the last words of the phrases as the most influential words of the phrase-pairs and have introduced them as separate features to their model. However, they have not provided any argument or evidence for their choice.

In this chapter, we investigate the importance of the internal words in defining reordering behavior of the phrases. This helps to better understand the linguistic phenomena captured by phrase reordering models. Additionally, it also helps improve the estimation of the reordering distribution in case of infrequent phrase-pairs by backing off to the most influential words. We ask the following general research question:

RQ1 *To what extent do the orderings of phrases in a phrase-based machine translation model depend on the full lexical forms of the phrases?*

Examples like those in Table 3.1 already show that the ordering of phrase-pairs does not entirely depend on the complete lexical form of the phrase-pairs. Intuitively, those phrase-pairs should follow the same reordering distribution. To achieve this, one has to know to what extent it is required to use the full lexical form of a phrase-pair to be able to estimate the reordering distribution for it. This is important because if we could substitute the full form with a shorter or more abstract form of a phrase-pair to estimate

its reordering distribution then the problem of estimating the reordering distribution for infrequent phrase-pairs can be alleviated by finding the right shortened or abstract form.

RQ1 is subdivided into the following three subquestions:

RQ1.1 *How does the importance of the words of a phrase change in defining the ordering of the phrase?*

Understanding the impact of the internal structure of the phrase-pairs on their reordering behavior can help find more effective representations of phrases to improve the estimation of reordering distributions for infrequent phrase-pairs. We answer this question with two types of experiment. We use a back-off approach following the back-off idea in n-gram language modelling as one approach, and a generalized form approach that keeps *exposed heads* lexicalized and generalizes the remaining words. Exposed heads are words inside a subsequence that are in a dependency relation with a word outside of the subsequence and are dominated by that word (Li et al., 2012). Based on these experiments, we also answer RQ1.2 and RQ1.3.

RQ1.2 *How helpful is it to estimate the reordering distribution of phrases by backing off to shorter forms of the phrases and removing less influential words?*

The use of the first and the last word of the phrases as a separate feature of the reordering model, as proposed by Nagata et al. (2006) and Cherry (2013), focuses on the border words of phrases as the most influential words on the ordering of phrases. We experiment with backing off towards the border words of phrases following the back-off idea in smoothing n-gram language models to validate whether the influence of words on reordering increases with their proximity to the borders:

RQ1.3 *How accurately can the orderings of phrases be predicted by using class-based generalizations of the words in phrases when keeping the most influential words in their original form?*

Some work has already used class-based generalized forms using POS tags and automatically learned word clusters to estimate the reordering distribution of infrequent phrase-pairs (Nagata et al., 2006). However, the authors have done this for all words in a phrase-pair without accounting for potentially different impacts of the internal words on the reordering behavior of the phrase-pair. Here, we experiment with generalizing the words that are less influential on reordering behavior of a phrase-pair and leave the more influential words in their original form to avoid over-generalization.

In this chapter, we propose two types of approach to use the most influential words from inside the original phrase-pairs to estimate better orientation distributions for infrequent phrase-pairs. These approaches perform better at taking phrase-pair similarities into account than the LRM and HRM which use the full lexical form of the phrase-pair. In the first approach, we define a back-off model to shorten towards important words inside the original phrase-pairs following the idea of back-off models in

language model smoothing. This is to some extent complementary to the HRM by using smaller phrase-pairs to achieve better predictions. The difference is that within HRMs smaller phrase-pairs are merged into longer blocks when possible, while we propose to use shorter forms of phrase-pairs when possible. In the second approach, we propose to produce generalized forms of fully lexicalized phrase-pairs by including important words and marginalizing the remaining words allowing for smoothed distributions that better capture the true distributions over orientations. Here, we use syntactic dependencies from the original phrase-pair to generalize and shorten in a more linguistically informed way.

This chapter makes the following contributions:

1. We introduce new methods that use shortened and generalized forms of a phrase-pair to smooth the original phrase orientation distributions. We show that our smoothing approaches result in improvements in a phrase-based machine translation system, even when compared against a strong baseline using both LRM and HRM together. These methods do not require any changes to the decoder and do not lead to any additional computational costs during decoding.

2. As our second contribution, we provide a detailed analysis showing that orientation distributions conditioned on long phrase-pairs typically depend on a few words within phrase-pairs and not the whole lexicalized form. These words are syntactically important words based on the dependency parse of the sentences. Our analysis contributes to the interpretability of reordering models and shows that capturing syntax is an important aspect for better machine translation quality. In Chapter 5, we investigate how syntactic information is captured by neural machine translation architectures which are currently the state-of-the-art models. Moreover, in Chapter 4, we show how attention in recurrent neural machine translation model learns to attend to modifiers of a word, as defined by a dependency parse, while generating the translation of the word. Note that exposed heads used in the current chapter are modifiers that are modifying a word outside of their containing phrases. Our analysis of reordering behavior, as described in this chapter, supports and adds to the sparse reordering features of Cherry (2013).

3.2 Related Work

The problem of data sparsity of training LRMs has first been addressed by Nagata et al. (2006), who propose to use POS tags and word clustering methods and distinguish the first or last word of a phrase as the head of the phrase.

Complementary to our work, Galley and Manning (2008) introduced hierarchical reordering models that group phrases occurring next to the current phrase into blocks, ignoring the internal phrase composition within a block, which biases orientations more towards monotone and swap. At the same time, orientations are still conditioned on

entire phrase-pairs, which means that their approach suffers from the same sparsity problems as LRMs. This problem has been more directly addressed by Cherry (2013), who uses unsupervised word classes for infrequent words in phrase-pairs in the form of sparse features. Like Nagata et al. (2006), the first and last words of a phrase-pair are used as features in this model. Unfortunately, this approach also introduces thousands of additional sparse features, many of which have to be extracted *during* decoding, requiring changes to the decoder as well as a sizable tuning set.

Durrani et al. (2014) investigate the effect of generalized word forms on reordering in an n-gram-based operation sequence model, where they use different generalized representations including POS tags and unsupervised word classes to generalize reordering rules to similar cases with unobserved lexical operations.

While the approaches above use discrete representations, Li et al. (2014) propose a discriminative model using continuous space representations of phrase-pairs to address data sparsity problems. They train a neural network classifier based on recursive auto-encoders to generate vector space representations of phrase-pairs and base reordering decisions on those representations. They apply their model as an additional hypergraph reranking step since direct integration into the decoder would make hypothesis recombination more complex and substantially increase the size of the search space.

In addition to reordering models, several approaches have used word classes to improve other models within a statistical machine translation system, including translation (Wuebker et al., 2013) and language models, where the problem of data sparsity is particularly exacerbated for morphologically rich target languages (Chahuneau et al., 2013, Bisazza and Monz, 2014).

3.3 Model Definition

In this section, we propose two different types of model that use different words as a source of information to better estimate reordering distributions for sparse phrase-pairs. Each model uses a different generalization scheme to obtain less sparse but still informative representations.

Additionally, each model follows a different notion of what is defined to be an important word. For the first type of models, we follow Nagata et al. (2006) assuming the border words of a phrase-pair as the most important words. However, instead of just using the border words as Nagata et al. (2006) do, we use all words, but assume that the importance increases by getting closer to the borders of a phrase-pair. For the second type, we assume the exposed heads to be the most important words, following (Chelba and Jelinek, 2000, Garmash and Monz, 2015, Li et al., 2012).

3.3.1 Interpolated Back-Off Sub-Phrases

In n-gram language modeling shorter n-grams have been used to smooth the probability distributions of higher-order, sparse n-grams. Lower-order n-grams form the basis of

Jelinek-Mercer, Katz, Witten-Bell and absolute discount smoothing methods (Chen and Goodman, 1999). For instance, Jelinek-Mercer smoothing linearly interpolates distributions of lower orders to smooth the distributions of higher-order n-grams.

We use this as a motivation that shorter phrase-pairs in lexicalized reordering models could play the role of lower-order n-grams in language model smoothing. But while backing off is obvious in language modeling, it is not straightforward in the context of lexicalized reordering models as there are several plausible ways to shorten a phrase-pair, which is further complicated by the internal word alignments of the phrase-pairs.

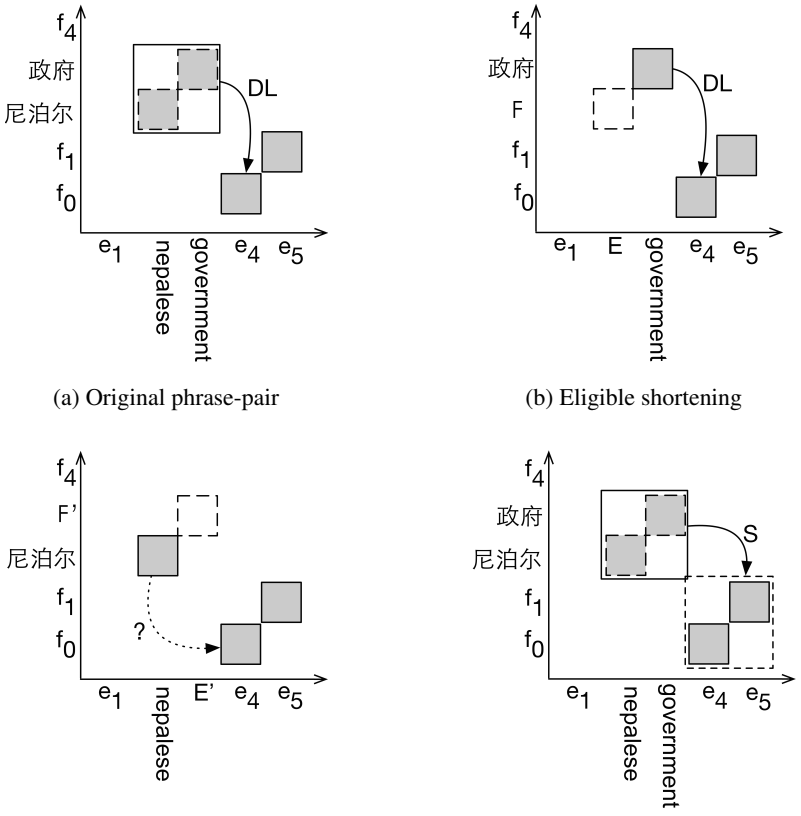


Figure 3.1: An example of an original phrase-pair (a) and backing off to shorter phrase-pairs using eligible sub-phrase-pairs (b). For comparison, we also include an example of a grouping of phrases as done in HRM (d). We have also included an invalid example of shortening (c) to provide a comprehensive case.

The example in Figure 3.1 illustrates how sub-phrase-pairs (Figure 3.1b and 3.1c) of the longer phrase-pair (Figure 3.1a) can be used to estimate the *discontinuous left* orientation for a longer, infrequent phrase-pair. Following the strategy within language

modeling to back off to shorter n-grams, we back off to the sub-phrase-pairs that are consistent with the inside alignment of the longer phrase-pair and provide a shorter and less sparse history. In this example, the number of times that sub-phrase-pair (政府, government), see Figure 3.1b, appears with a *discontinuous left* jump of the length of the previous phrase-pair for the next translation is considered when estimating the *discontinuous left* orientation for the longer phrase-pair. On the other hand, the sub-phrase-pair (尼泊尔, nepalese), see Figure 3.1c, cannot be used to predict a future *discontinuous left* of the long phrase-pair, as there is no direct way to connect it to (f_0, e_4). The difference between this model and HRM can be seen in Figure 3.1d. HRM groups small phrase-pairs from the context into longer blocks and determines the orientation with respect to the grouped block, while our model looks into the phrase-pair itself and uses possible shortenings to better estimate the orientation distribution conditioned on the original phrase-pair. Our model can be applied to HRMs as well as LRMs to estimate a better distribution for long and infrequent phrase-pairs. This can be done without requiring to retrain the models and by only using the statistics resulting from training.

In order to provide a formal definition of which sub-phrase-pairs to consider when backing off, let us assume that A is the set of alignment connections between the source $\bar{f}$ and target $\bar{e}$ side of a longer phrase-pair. The set of eligible sub-phrase-pairs, $E_{\bar{f},\bar{e}}$, is defined as follows:

$$E_{\bar{f},\bar{e}} = \{(\bar{f}^{[l,k]}, \bar{e}^{[l',n]}) \mid 1 \leqslant k \leqslant m,\ 0 \leqslant l \leqslant k, 0 \leqslant l' \leqslant n \text{ and } (\bar{f}^{[l,k]}, \bar{e}^{[l',n]}) \text{ is consistent with } A \text{ if } l > 0 \wedge l' > 0\ \}, \tag{3.3}$$

where $\bar{f}^{[l,k]}$ is a sub-phrase of $\bar{f}$ with length l that ends at the k-th word of $\bar{f}$, m and n are the lengths of $\bar{f}$ and $\bar{e}$ respectively and the consistency with the alignment is ensured by the following three conditions (Koehn et al., 2005):

1. $\exists e_i \in \bar{e}^{[l',n]}, f_j \in \bar{f}^{[l,k]} : (i,j) \in A$
2. $\forall e_i \in \bar{e}^{[l',n]} : (i,j) \in A \Rightarrow f_j \in \bar{f}^{[l,k]}$
3. $\forall f_i \in \bar{f}^{[l,k]} : (i,j) \in A \Rightarrow e_i \in \bar{e}^{[l',n]}$

Considering Figure 3.2, it is clear why ($\bar{f}^{[1,2]}$, $\bar{e}^{[1,3]}$) and ($\bar{f}^{[2,3]}$, $\bar{e}^{[2,3]}$) are considered eligible shortenings. Other possible shortenings such as ($\bar{f}^{[2,2]}$, $\bar{e}^{[3,3]}$) and ($\bar{f}^{[1,3]}$, $\bar{e}^{[1,2]}$) either violate the consistency conditions or do not run up to the end of the target side of the original phrase-pair as in the definition above. Note that n is a constant here and $\bar{e}^{[l',n]}$ means that all sub-phrase-pairs must finish at the end of the target side of the original phrase-pair. This is necessary as otherwise one cannot directly determine the orientation with respect to the next phrase-pair. This condition makes sure that cases such as the one in Figure 3.1c, are identified as an invalid way of shortening.

In our first model, we compute the smoothed orientation distribution conditioned on a phrase-pair by linearly interpolating the distribution of all eligible sub-phrases as

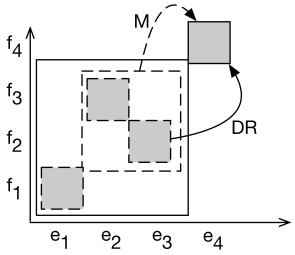


Figure 3.2: The distributions needed to estimate the conditional probability $\hat{p}(M \mid f_1 f_2 f_3, e_1 e_2 e_3)$ include $p(DR \mid f_2, e_3)$ and $p(M \mid f_2 f_3, e_2 e_3)$

follows:

$$\hat{P}(o \mid \bar{f}, \bar{e}) = \sum_{(\bar{f}^{[l,k]}, \bar{e}^{[l',n]}) \in E_{\bar{f},\bar{e}}} \lambda_{l,l'} P(\Omega(\bar{f}^{[l,k]}, \bar{f}, o) \mid \bar{f}^{[l,k]}, \bar{e}^{[l',n]}). \quad (3.4)$$

Here $E_{\bar{f},\bar{e}}$ is the set of eligible sub-phrase-pairs and $\bar{f}^{[l,k]}$ indicates a sub-phrase of $\bar{f}$ with length l ending at the k-th word of $\bar{f}$, $\bar{e}^{[l',n]}$ is a sub-phrase of $\bar{e}$ with the length of l' that ends at the last word of $\bar{e}$, and the function $\Omega(\bar{f}^{[l,k]}, \bar{f}, o)$ returns the correct orientation considering the position of source sub-phrase $\bar{f}^{[l,k]}$ with respect to either end of the source phrase $\bar{f}$ and orientation o.

In order to compute the linear interpolation over the conditional distributions of the sub-phrase-pairs, we have to determine the weight of each term in the linear interpolation (Equation 3.4). Here, we use expectation-maximization (EM) to find a set of weights that maximizes the likelihood over a held-out data set, which is word-aligned using GIZA++ (Och and Ney, 2003). The alignments are refined using the grow-diag-final-and (Koehn et al., 2003) heuristic. We extract phrase-pairs using a common phrase extraction algorithm (Koehn et al., 2005) and count the number of occurrences of orientations for each phrase-pair. These counts are used with unsmoothed reordering probabilities learned over the training data to compute the likelihood over the held-out data. We designed the EM algorithm to learn a set of lambda parameters for each length combination of the original phrase-pairs. To reduce the number of parameters, we assume that all sub-phrases with the same length on the source and target side share the same weight. This model is referred to as the Back-off model in the remainder of this chapter.

Next, we introduce a model based on the same scheme of shortening into sub-phrase-pairs, but a different way of combining them to smooth the target distribution.

3.3.2 Recursive Back-Off MAP Smoothing

Above we used linear interpolation to estimate the orientation distributions of shorter sub-phrase-pairs. Here we investigate another method which aims to affect the distributions

of frequent phrase-pairs to a lesser extent than those of non-frequent ones.

To this end, we use recursive Maximum a Posteriori (MAP) smoothing (Cherry, 2013) to estimate the distribution of the original phrase-pair. In our interpolated back-off model, see Section 3.3.1, all phrase-pairs with the same length will get the same fraction of their estimated distribution from their sub-phrase-pairs of the same length since the interpolation parameters are the same for these phrase-pairs and sub-phrase-pairs. On the other hand, for more frequent phrase-pairs, the maximum likelihood distribution of the phrase-pair itself is more reliable than the distributions of its sub-phrase-pairs. Thus, a model relying more on the distribution of the original phrase-pairs for frequent phrase-pairs would be desirable.

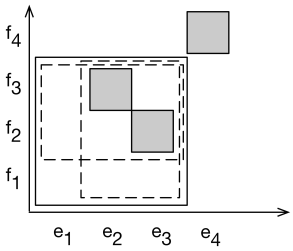


Figure 3.3: Illustration of sub-phrases where neither of the two pairs $(f_1 f_2 f_3, e_2 e_3)$ and $(f_2 f_3, e_1 e_2 e_3)$ of the original phrase-pair $(f_1 f_2 f_3, e_1 e_2 e_3)$ is a sub-phrase of the other.

To achieve this, we use a formulation similar to recursive MAP smoothing (Equation 3.2) with recursively backing off to the distributions of shorter sub-phrase-pairs. At each recursion step we use the distribution of the longest sub-phrase-pair as the prior distribution. Taking our definition for eligible sub-phrase-pairs into account (Equation 3.3), for every two sub-phrase-pairs one is always a sub-phrase-pair of the other, if the original phrase-pair does not include unaligned words. For original phrase-pairs that include unaligned words, see the example in Figure 3.3, there could be pairs of sub-phrases where neither of them is a sub-phrase of the other. In these cases, we include the distributions of all sub-phrases with the same *equivalent sample size* as prior distributions, see Section 3.1.1. The estimated probability distribution of a phrase-pair $(\bar{f},\bar{e})$ is defined as follows:

$$\hat{P}(o \mid \bar{f}, \bar{e}) = \frac{C(o, \bar{f}, \bar{e}) + \sum_{(\bar{f}_L, \bar{e}_L) \in L_{\bar{f},\bar{e}}} \alpha \hat{P}(\Omega(\bar{f}_L, \bar{f}, o) \mid \bar{f}_L, \bar{e}_L)}{\sum_{o \in O} C(o, \bar{f}, \bar{e}) + \sum_{(\bar{f}_L, \bar{e}_L) \in L_{\bar{f},\bar{e}}} \alpha}, \tag{3.5}$$

where $L_{\bar{f},\bar{e}}$ refers to the set of eligible sub-phrase-pairs of $(\bar{f}, \bar{e})$ that are not sub-phrase-

pairs of each other and which is defined as follows:

$$L_{\bar{f},\bar{e}} = \Big\{ (\bar{f}', \bar{e}') \in E_{\bar{f},\bar{e}} \setminus \{(\bar{f}, \bar{e})\} \mid \neg\exists (\bar{f}'', \bar{e}'') \in E_{\bar{f},\bar{e}} \setminus \{(\bar{f}, \bar{e}), (\bar{f}', \bar{e}')\} : \bar{f}' \sqsubseteq \bar{f}'' \wedge \bar{e}' \sqsubseteq \bar{e}'' \Big\} \tag{3.6}$$

Here, $\bar{f}' \sqsubseteq \bar{f}''$ means that $\bar{f}'$ is a sub-phrase of or equal to $\bar{f}''$. As a result, $L_{\bar{f},\bar{e}}$ is the set of longest eligible sub-phrase-pairs of the original phrase-pair where none of them is a sub-phrase of the others. For $E_{\bar{f},\bar{e}}$, see Equation 3.3. O is the set of possible orientations. We refer to this model as RecursiveBackOff model from now on. We also refer to both this model and the BackOff model described in the previous section as back-off models.

3.3.3 Dependency Based Generalization

The methods described so far generalize the original phrase-pairs by shortening towards the last aligned words as the most important words to define the reordering behavior of a phrase-pair. In the remainder of this section, we use dependency parses to define how to generalize the original phrase-pair and shorten towards important words.

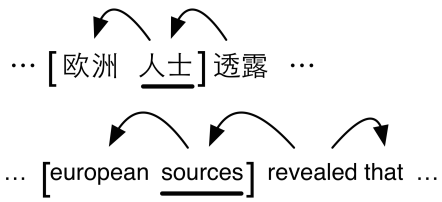


Figure 3.4: Examples of exposed heads in a Chinese-English phrase-pair (between square brackets). The underlined words are the exposed heads since they have an incoming dependency originating outside of the phrase.

Head-driven hierarchical phrase-based translation (Li et al., 2012) argues that using heads of phrases can be beneficial for better reordering. In our work, we define the heads of a phrase to be its *exposed heads*. Given a dependency parse, the exposed heads are all words inside a subsequence that play the role of a modifier in a dependency relation with a word outside of the subsequence. Figure 3.4 shows an example of exposed heads in a phrase-pair. The underlined words are the exposed heads of the phrase-pair. Exposed heads have been used in multiple linguistically motivated approaches as strong predictors of the next word in structured language models (Chelba and Jelinek, 2000, Garmash and Monz, 2015) and the next translation rule to be applied in a hierarchical

phrase-based translation model (Chiang, 2007, Li et al., 2012). Figure 3.5 shows a sentence pair with the extracted phrase-pairs and their corresponding exposed heads.

In our model, besides training a regular lexicalized or hierarchical reordering model on surface forms of phrases, we train another reordering model which keeps the exposed heads lexicalized and replaces the remaining words in a phrase-pair by a generalized representation.

Assume that RE is the set of dependency relations in the dependency parse tree of sentence S. We consider each relation as an ordered pair (w'_l, w_k) which means that word w at index k of sentence S modifies word w' at index l. In addition, assuming f_i^j is a phrase in S, starting from the i-th and ending with the j-th word in sentence S, then the generalization w_G of a word is:

$$w_G = \begin{cases} w_k & \text{if } w_k \in f_i^j, \exists (w'_l, w_k) \in RE : \\ & l < i \text{ or } l > j \text{ or } w' = ROOT \\ Gen(w_k) & \text{otherwise.} \end{cases}$$

Here, k and l are indices of words w and w' in sentence S and $ROOT$ is the root of the dependency parse of sentence S. The function $Gen(w)$ returns a generalization form for word w. We define this function in three different ways to create three different models.

Definition 3.3.1. $Gen(w_k)$ = POS_tag(w_k)

Definition 3.3.2. $Gen(w_k)$ = <mod> if w_{k-1} is not equal to <mod> and nothing otherwise (replaces all consecutive modifiers with a single <mod> symbol)

Definition 3.3.3. $Gen(w_k)$ remove w_k.

Here, <mod> is a designated symbol that replaces the modifiers of the exposed heads. The question is how to use these generalizations to improve the estimation of reordering distribution for each phrase-pair. Our first model applies a generalization to the bilingual training data and creates a reordering model similar to the regular lexicalized reordering model, but based on the relative frequency of the generalized phrase-pairs. In practice, a phrase-pair can have multiple generalizations due to different dependency parses in different contexts. Since it is difficult to produce a dependency parse for the target side during decoding, we assume that a phrase-pair will always have one possible generalization. Under this assumption, we can approximate the orientation distribution of a phrase-pair to be the orientation distribution of its generalized form:

$$\hat{P}(o \mid \bar{f}, \bar{e}) = P(o \mid \bar{f_G}, \bar{e_G}). \tag{3.7}$$

Here, $\bar{f_G}$ and $\bar{e_G}$ are word by word generalizations of $\bar{f}$ and $\bar{e}$. In the case of multiple generalizations we use the one maximizing $P(\bar{f_G}, \bar{e_G} \mid \bar{f}, \bar{e})$. Depending on which of the three definitions for $Gen(w_k)$ we use, we name our models PMLH (POSed Modi-

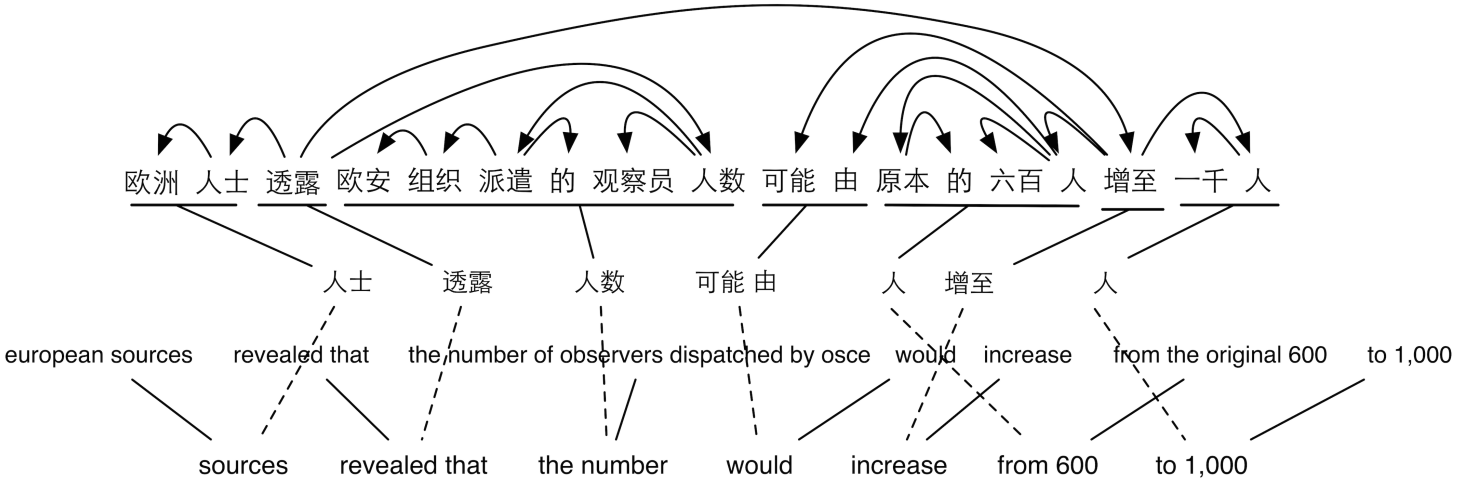


Figure 3.5: A source (Chinese) and the target (English) sentence pair with the extracted phrases and their corresponding exposed heads of each phrase from the source and the target side. Solid lines connect a phrase to its exposed heads and dashed lines connect aligned words. The target side dependency parse is not shown for visualization purposes.

Table 3.2: Statistics for the Chinese-English bilingual corpora used in all experiments. Token counts for the English side of dev and test sets are averaged across all references.

Corpus	Lines	Tokens (ch)	Tokens (en)
train	937K	22.3M	25,9M
MT04 (dev)	1,788	49.6K	59.2K
MT05 (test)	1,082	30.3K	35.8K
MT06 (test)	1,181	29.7K	33.5K

fiers, Lexicalized Heads), see Definition 3.3.1, MMLH (Merged Modifiers, Lexicalized Heads), see Definition 3.3.2, and LH (Lexicalized Heads), see Definition 3.3.3.

As an alternative model, we propose to use the generalized distributions as a prior distribution in Dirichlet smoothing, where the distribution of each phrase-pair is smoothed with the distribution of its generalized form. This results in the following smoothing formulation:

$$P(o \mid \bar{f}, \bar{e}) = \frac{C(o, \bar{f}, \bar{e}) + \sigma P(o \mid \bar{f_G}, \bar{e_G})}{\sum_{o'} C(o', \bar{f}, \bar{e}) + \sigma}. \tag{3.8}$$

Here, $P(o \mid \bar{f_G}, \bar{e_G})$ is the generalized distribution, see Equation 3.7, and $C(o, \bar{f}, \bar{e})$ refers to the number of times a phrase-pair co-occurs with orientation o, and σ is the equivalent sample size, as defined in Section 3.1.1. Our assumption is that this should result in more accurate distributions since it affects the distributions of frequent phrase-pairs to a lesser extent.

3.4 Experiments

We evaluate our models for Chinese-to-English translation. Our training data consists of the parallel and monolingual data released by NIST's OpenMT campaign, with MT04 used for tuning and news data from MT05 and MT06 for testing, see Table 3.2. Case-insensitive BLEU (Papineni et al., 2002), RIBES (Isozaki et al., 2010) and translation error rate (TER) (Snover et al., 2006) are used as evaluation metrics.

3.4.1 Baseline

We use an in-house implementation of a phrase-based statistical machine translation system similar to Moses (Koehn et al., 2007). Our system includes all the commonly used translation, lexical weighting, language, lexicalized reordering, and hierarchical reordering models as described in Chapter 2. We use both lexicalized and hierarchical reordering models together, since this is the best model reported by Galley and Manning (2008) and our smoothing methods can be easily applied to both models. Word alignments are produced with GIZA++ (Och and Ney, 2003) and the grow-diag-final-and

alignment refinement heuristic (Koehn et al., 2003). A 5-gram language model is trained on the English Gigaword corpus containing 1.6B tokens using interpolated, modified Kneser-Ney smoothing. The lexicalized and the hierarchical reordering models are trained with relative and smoothed frequencies using Dirichlet smoothing (Equation 3.1), for both left-to-right and right-to-left directions distinguishing four orientations: monotone (M), swap (S), discontinuous left (DL), and discontinuous right (DR). Feature weights are tuned using PRO (Hopkins and May, 2011) and statistical significance of the differences are computed using approximate randomization (Riezler and Maxwell, 2005).

In addition to the baseline, we reimplemented the 2POS model by Nagata et al. (2006), which uses the POS tag of the first and last words of a phrase-pair to smooth the reordering distributions. The 2POS model is used in combination with the baseline lexicalized and hierarchical reordering models. Comparing our models to the 2POS model allows us to see whether backed-off sub-phrases and exposed heads of phrase-pairs yield better performance than simply using the first and last words.

Table 3.3: Model comparison using BLEU, TER (lower is better), and RIBES over news data, which is a combination of newswire and broadcast_news for MT06 and just newswire for MT05. Scores better than the baseline are in italics. ▲ and △ indicate statistically significant improvements at $p < 0.01$ and $p < 0.05$, respectively. The left hand side ▲ or △ is with respect to Lex+Hrc and the right hand side with respect to Nagata's 2POS. PMLH refers to the model using POS tags for modifiers and keeps exposed heads lexicalized. MMLH merges modifiers and keeps exposed heads lexicalized. LH removes modifiers. LHSmoothed uses the LH model with Dirichlet smoothing.

Model	MT05		MT06		MT05 + MT06		
	BLEU↑	TER↓	BLEU↑	TER↓	BLEU↑	TER↓	RIBES↑
Lex+Hrc	32.25	60.13	33.00	57.17	32.84	58.62	79.24
Nagata's 2POS	32.20	60.31	33.13	57.07	32.87	58.66	79.11
BackOff	*32.40*	60.45	*33.10*	*57.00*	*33.00*	58.69	79.07
RecursiveBackOff	*32.37*	60.29	*33.34*$^{\triangle,-}$	*57.08*	*33.05*$^{\triangle,-}$	58.65	*79.28*
PMLH	*32.41*	*60.08*	*33.26*$^{\triangle,-}$	*57.05*	*33.04*$^{\triangle,-}$	*58.53*	*79.26*
MMLH	*32.62*$^{\blacktriangle,\triangle}$	*59.84*$^{-,\blacktriangle}$	*33.18*	*56.91*$^{\triangle,-}$	*33.04*$^{\triangle,-}$	*58.35*$^{\blacktriangle,\triangle}$	**79.43**
LH	*32.64*$^{\blacktriangle,\triangle}$	*59.85*$^{\triangle,\blacktriangle}$	*33.26*$^{\triangle,-}$	*56.85*$^{\blacktriangle,-}$	*33.11*$^{\blacktriangle,\triangle}$	*58.32*$^{\blacktriangle,\blacktriangle}$	*79.34*
LHSmoothed	**32.65**$^{\blacktriangle,\blacktriangle}$	**59.80**$^{\triangle,\blacktriangle}$	**33.38**$^{\triangle,-}$	**56.77**$^{\blacktriangle,\triangle}$	**33.20**$^{\blacktriangle,\triangle}$	**58.25**$^{\blacktriangle,\blacktriangle}$	*79.35*

3.4.2 Experimental Results

We compare the baseline to the models described in Section 3.3. For all systems other than the baseline, the lexicalized and the hierarchical reordering models are replaced by the corresponding smoothed models. When computing the RecursiveBackOff model (Section 3.3.2), using Equation 3.5, we set the value of α to 10, following (Cherry,

2013) and (Chen et al., 2013).

For the model using exposed heads, we use the dependency parses of the source and the target side of the training corpus. The Stanford Neural-network dependency parser (Chen and Manning, 2014) is used to generate parses for both sides of the training corpus. From a dependency parse, we extract the smallest subtree that includes all incoming and outgoing relations of the words of a phrase. This is done for both the source and the target side phrases. Considering these subtrees, all words with an incoming connection from outside are exposed heads. Having identified the exposed heads, we apply the generalization methods introduced in Section 3.3.3 to produce the estimated generalized distributions.

The experimental results for all models are shown in Table 3.3. As one can see, all our models achieve improvements over the baselines in terms of BLEU. The improvements for our back-off models are statistically significant only for RecursiveBackOff for MT06 and MT05+MT06. The improvements for MT05 by our dependency-based shortenings are statistically significant for all models except PMLH. In the case of MT06, only the improvements resulting from MMLH are not statistically significant. However, both the PMLH and the MMLH model achieve the same improvements for MT05 and MT06 combined, and both are statistically significant. The LH model performs better than these models and also achieves higher improvements on MT05+MT06. This model generalizes much more than the other models and is the only model that changes the distributions of single word phrases which are among the most frequently used phrase-pairs. However, for frequent phrase-pairs, being mapped to the same generalization form can be potentially harmful, since it may change the reordering distribution for phrase-pairs which are frequent enough for reliable estimation in their original form. In order to be able to control the effect of the model on phrase-pairs based on their frequency, we use the distributions in LH as a prior distribution with Dirichlet smoothing (Equation 3.8). Using this formulation, frequent phrase-pairs are smoothed to a lesser degree. This results in the LHSmoothed model shown in the last row of Table 3.3 which achieves the best improvements for both MT05 and MT06.

In addition to BLEU, we also report results using TER. The results for TER are in line with BLEU. BLEU and TER are general translation quality metrics, which are known to be not very sensitive to reordering changes (Birch et al., 2010). To this end we also include RIBES (Isozaki et al., 2010), a reordering-specific metric that is designed to directly address word-order differences between translation candidates and reference translations in translation tasks with language pairs often requiring long-distance reorderings. As the values of the RIBES metric shows, our dependency models improve reordering more than the other models.

3.4.3 Analysis

The improvements achieved by our BackOff and RecursiveBackOff methods show that these models capture useful generalizations by shortening the phrase-pairs towards the

last aligned words in the target side as the most important words. The difference between the two models indicates that shortening is less beneficial for frequent phrase-pairs and shorter phrase-pairs, which are less affected by lower-order distributions in the case of RecursiveBackOff. As a result, the RecursiveBackOff model achieves a slightly better performance.

The improvements achieved by our generalization models support our hypothesis that not all words inside a phrase-pair have the same impact on the reordering properties of the phrase-pair as a whole. The experimental results for the PMLH model show that the lexicalized form of modifier words inside a phrase-pair may increase data sparsity.

Observing the improvements achieved by MMLH, we can go further and say that even the number of modifiers of an exposed head in a phrase does not influence the reordering properties of a phrase-pair. The improvements achieved by our LH model show not only that the number of modifiers but also the mere presence or absence of them does not significantly influence the reordering properties of a phrase-pair.

Table 3.4: Number of times that phrase-pairs with different lengths and a frequency of less than 10 in the training data have been used by the baseline for MT06. Phrase-pairs occurring less than 10 times account for 72% of all phrases used during decoding of MT06.

Source Length	Target Length							
	1	2	3	4	5	6	7
1	8232	2719	879	269	89	18	7
2	2344	1777	1055	410	158	58	18
3	316	390	376	252	100	61	29
4	42	46	97	63	29	28	14
5	2	3	11	11	10	8	12
6	0	1	1	3	3	5	2
7	0	0	0	3	1	1	1

One thing to bear in mind is that the PMLH and MMLH models do not change the distribution of single word phrase-pairs that are mostly frequent phrase-pairs, while the LH model does change these distributions as well. With the LHSmoothed model we have controlled this effect and decreased it to a negligible degree for frequent single word phrase-pairs. However, it still changes the distributions of infrequent single word phrase-pairs. Comparing the results of LH and LHSmoothed in Table 3.3, we suspect that the difference between the models for MT06 is due to the effect of frequent single word phrase-pairs.[1] However, comparing the results of LHSmoothed with the other models we can say that even infrequent single word phrase-pairs benefit from the higher generalization level offered by this model. The statistics of infrequent phrase-pairs

[1] We also used the distributions from PMLH and MMLH as priors in Dirichlet smoothing, but it did not lead to any noticeable changes in the results.

used during testing and their lengths are shown in Table 3.4. This indicates why this model achieves the highest improvements. The table shows that 41% of the infrequent phrase-pairs (frequency < 10) used while translating MT06 have a length of one on both sides. Therefore, our models, apart from LH and LHSmoothed, do not affect almost half of the infrequent phrase-pairs. This probably explains why the back-off models achieve such small improvements when 75% of infrequent phrase-pairs have lengths of less than 3 on both sides.

Table 3.5: Examples from MT05 illustrating reordering improvements over the baseline.

Source	... 由于 东京 和 汉城 当局 均 寄望 能 于 二〇〇五年 底 前 签署 自由 贸易 协定 ...
Baseline	... tokyo and seoul authorities are to be placed in 2005 before the end of the signing of a free trade agreement ...
LHSmoothed	... tokyo and seoul authorities both in the hope of signing a free trade agreement before the end of 2005 ...
Ref	... tokyo and seoul both hoped to sign a fta agreement by the end of 2005 ...

Source	俄罗斯 多 次 指控 西方 插手 东欧 事务 ...
Baseline	russia has repeatedly accused of meddling in the affairs of the western and eastern europe ...
LHSmoothed	russia has repeatedly accused western intervention in the eastern european affairs ...
Ref	russia has been accusing the west of interfering in the affairs of eastern europe ...

In general, our dependency-based models change the distributions to a larger degree than our back-off models. In our back-off models, not all long phrase-pairs have eligible sub-phrase-pairs, while the dependency models more often result in shorter generalizations. Table 3.5 provides some examples where our LHSmoothed model has improved the translations by better modeling of reorderings. To further the understanding of how the reordering distributions of infrequent phrase-pairs, as used in the examples in Table 3.5, are affected by our models, we show some examples in Table 3.6. This table lists the examples before and after applying our model. As a result of our model, for the phrase-pair (寄望, in the hope) the monotone orientation probability is increased although it has a frequency of zero for this orientation. For the next phrase-pair (能 于, of) the discontinuous right probability is increased resulting in the correct orientation during translation. The most interesting case is the substantial decrease in the probability of discontinuous left and the increase for discontinuous right for the phrase-pair (签署 自由 贸易 协定, signing a free trade agreement), although it has frequency 1 for the former orientation and 0 for the latter. In the second example, the shifts in the distributions for the phrase-pairs (插手, meddling in the) and (插手, intervention in the) caused the first translation option to be dropped and replaced with the second translation option, leading to the correct reordering of the word *western*. These shifts within the probability distributions lead to the better translation generated by the LHSmoothed

Table 3.6: Changes to the orientation distributions for some infrequent phrase-pairs as a result of applying our LHSmoothed generalization. These infrequent phrase-pairs are used with the correct orientation in the translations by the LHSmoothed model in Table 3.5. The used orientations are also shown. Note the shifts between the LHSmoothed distributions and the corresponding baseline distributions. The original frequencies are also shown.

寄望	in the hope	M	S	DL	DR
Monotone with previous	Baseline	0.10	0.01	0.11	0.78
	LHSmoothed	0.28	0.01	0.01	0.70
	Counts in training	0	0	0	1
能 于	of				
Discontinuous right with next	Baseline	0.10	0.01	0.12	0.77
	LHSmoothed	0.06	0.01	0.07	0.87
	Counts in training	0	0	0	1
签署 自由 贸易 协定	signing a free trade agreement				
Discontinuous right with previous	Baseline	0.10	0.02	0.68	0.20
	LHSmoothed	0.21	0.03	0.30	0.46
	Counts in training	0	0	1	0

插手	meddling in the				
Discontinuous right with previous	Baseline	0.10	0.01	0.68	0.21
	LHSmoothed	0.42	0.01	0.51	0.06
(correctly replaced by the phrase-pair below)	Counts in training	0	0	1	0
插手	intervention in the				
Monotone with previous	Baseline	0.67	0.02	0.11	0.20
	LHSmoothed	0.75	0.01	0.03	0.21
	Counts in training	1	0	0	0

model for the first example in Table 3.5.

3.5 Conclusions

Our reimplementation of the 2POS method by Nagata et al. (2006) shows that the full lexical form does not play a significant role in estimating reliable reordering distributions. However, the improvements achieved by keeping important words in their surface form show that the lexical forms for the exposed heads still have an impact on estimating the reordering distributions more precisely. This provides the answer to our RQ1, which asks:

RQ1 *To what extent do the orderings of phrases in a phrase-based machine translation model depend on the full lexical forms of the phrases?*

Further, we have introduced a novel method that builds on the established idea of

backing off to shorter histories, commonly used in language model smoothing, and shown that it can be successfully applied to smoothing of lexicalized and hierarchical reordering models in statistical machine translation. We have also shown that not all sub-phrase-pairs are equally influential in determining the most likely phrase reordering. We experimented with different general forms of the phrase-pairs in which we keep the exposed heads in their original forms and generalize the remaining words by using word classes or simply removing them. The experimental findings show that sub-phrase-pairs consisting of just exposed heads tend to be the most important ones and most other words inside a phrase-pair have negligible influence on reordering behavior.

RQ1.1 *How does the importance of the words of a phrase change in defining the ordering of the phrase?*

Earlier approaches, such as (Nagata et al., 2006) and (Cherry, 2013), often assume that the last and the first word of a phrase-pair are the most important words for defining reordering behavior, but our experiments show that exposed heads tend to be stronger predictors. We experiment with both backing off towards the border words by assuming those as the important words, following Nagata et al. (2006) and Cherry (2013), and assuming exposed heads as the most important words, following Chelba and Jelinek (2000) and Garmash and Monz (2015). We showed that generalized representations of phrase-pairs based on exposed heads can help decrease sparsity and result in more reliable reordering distributions.

RQ1.2 *How helpful is it to estimate the reordering distribution of phrases by backing off to shorter forms of the phrases and removing less influential words?*

The results show that backing off to the shorter sub-phrase-pairs that include the ending or the starting word of the phrase helps estimate the reordering distributions. However, our back-off models do not achieve the best results even though they perform better than the baselines. Our generalized models achieve better results in comparison to our back-off models because exposed heads are in general more important for phrase reordering compared to border words.

RQ1.3 *How accurately can the orderings of phrases be predicted by using class-based generalizations of the words in phrases when keeping the most influential words in their original form?*

Our generalized representations of phrase-pairs based on exposed heads achieved the best results in our experiments. This is mostly due to the better choice of the most influential internal words rather than the approach we took for the generalization of less important words. However, among the models that generalize phrase-pairs by keeping exposed heads intact as the most influential words, the way less influential words are generalized has a negligible effect on performance. Interestingly, simply removing the less influential words yields the best performance improvements in our experiments. We

show that exposed heads and border words do provide good indicators for the reordering behavior of phrase-pairs and that exposed heads are more influential compared to border words on average.

Considering the analysis of the length of infrequent phrase-pairs used during translation, we also conclude that a smoothing model that would be able to further improve the distribution of single word phrase-pairs is crucial for achieving higher improvements during translation.

In this chapter, we showed how the internal words of phrase-pairs impact reordering behavior of phrase-pairs. This is a step towards better understanding how reordering models capture different reordering phenomena. This also helps us in interpreting the behavior of attention models in neural machine translation which is the focus of the next chapter.

4

What does Attention in Recurrent Neural Machine Translation Pay Attention to?

4.1 Introduction

In this chapter, we shift from phrase-based to neural machine translation. Neural machine translation has gained a lot of attention due to its substantial improvements in machine translation quality over phrase-based machine translation and achieving state-of-the-art performance for many languages (Luong et al., 2015b, Jean et al., 2015, Bentivogli et al., 2016, Wu et al., 2016, Vaswani et al., 2017, Wang et al., 2019b, Ott et al., 2018, Chen et al., 2018, Edunov et al., 2018, Wang et al., 2019a). The core architecture of neural machine translation models is based on the general encoder-decoder approach (Sutskever et al., 2014).

In Chapter 3, we proposed feature extraction algorithms that identify the words that effect reordering most, in order to improve reordering distributions in phrase-based machine translation. However, neural machine translation model learns to handle reorderings more indirectly. Neural machine translation is an end-to-end approach that learns to encode source sentences into distributed representations and decode these representations into sentences in the target language. Encoding sentence information into distributed representations makes neural machine translation models even harder to interpret, compared to phrase-based machine translation. However, attention-based neural machine translation (Bahdanau et al., 2015, Luong et al., 2015a) explicitly identifies the most relevant parts of the source sentence at each translation step. This capability also makes the attentional model superior in translating longer sentences (Bahdanau et al., 2015, Luong et al., 2015a).

In order to understand how neural machine translation captures different syntactic phenomena, especially better word ordering in comparison to phrase-based machine translation, it is important to understand the behavior of the attention model. Due to the capability of attending to the most relevant parts of the source sentence at each translation step, the attention model is often considered similar to reordering models

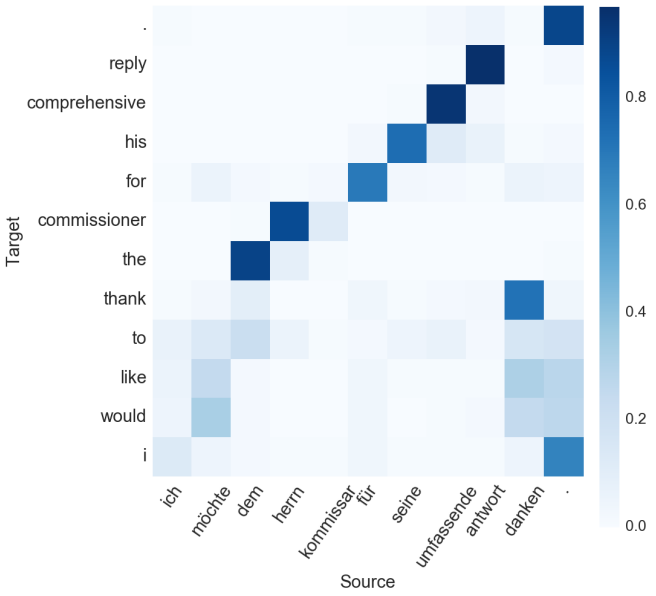


Figure 4.1: Visualization of the attention paid to the relevant parts of the source sentence for each generated word of a translation example. Note how the attention is 'smeared out' over multiple source words in the case of "would" and "like".

(Liu et al., 2016).

Figure 4.1 shows an example of how attention uses the most relevant source words to generate a target word at each step of the translation. In this chapter, we focus on studying the relevance of the attended parts, especially cases where attention is 'smeared out' over multiple source words and where their relevance is not entirely obvious, see, e.g., "would" and "like" in Figure 4.1. Here, we ask whether this is due to errors or noise of the attention mechanism or is a desired behavior of the model.

Since the introduction of attention models in neural machine translation (Bahdanau et al., 2015) various modifications such as integrating inductive biases from traditional alignments into attention (Cohn et al., 2016), training an attention model to follow traditional alignments (Liu et al., 2016, Chen et al., 2016), improving the architecture to be simpler and more general (Luong et al., 2015a), and integrating a coverage model into the attention model (Tu et al., 2016), have been proposed. However, to the best of our knowledge there is no study that provides an analysis of what kind of phenomena are being captured by attention. There is some work that has looked at attention as being similar to traditional word alignment (Alkhouli et al., 2016, Cohn et al., 2016, Liu et al., 2016, Chen et al., 2016). Some of these approaches also experimented with training the attention model using traditional alignments (Alkhouli et al., 2016, Liu et al., 2016, Chen et al., 2016). Liu et al. (2016) have shown that attention can be seen as a reordering model as well as an alignment model.

In this chapter, we focus on investigating the differences between attention and alignment and what is being captured by the attention mechanism in general. The general question that we are aiming to answer in this chapter is:

RQ2 *What are the important words on the source side that attention-based neural models attend to while translating different syntactic phenomena?*

We analyze the attention model behavior using two methods: (i) by looking into its difference with the behavior of traditional alignment and (ii) by studying the distributional behavior of the attention model for different syntactic phenomena. The following subquestions elaborate the analyses we propose in this chapter.

RQ2.1 *To what extent does an attention model agree with the traditional alignment model in capturing translation equivalent words?*

To answer this question, we first define a metric to compute the difference between the attention distribution and traditional alignment. Then, we study the relation between the quality of translation and the convergence of attention to alignment. We show that the relation differs for different syntactic phenomena, meaning that closer attention to alignments leads to a better translation for some cases, but hurts performance in some other cases. Our analysis suggests that attention and traditional alignment models are different and it is not a good idea to train an attention model to follow the traditional alignments under all circumstance.

RQ2.2 *Are the differences between attention and traditional alignments due to errors in attention models or do attention models capture additional information compared to traditional alignment models?*

Here, we analyze whether the differences between the attention model and the traditional alignment model are due to errors in the attention or whether they are side products of the attention model learning meaningful information in addition to what a traditional alignment model learns. We carry out these analyses by studying the distributional behavior of the attention model, especially for cases in which attention deviates from the alignment model. We also study the relation between attention distribution and translation quality to see whether there are meaningful correlations across different syntactic phenomena.

Furthermore, we study the word types that receive the most attention when it is paid to words other than alignment points. This also helps better understand whether the differences between the attention and the alignment models are meaningful.

RQ2.3 *How does the distribution of the attention model change for different syntactic phenomena?*

In order to achieve a more fine-grained analysis, we also look into the changes in the distributional behavior of the attention model for different syntactic phenomena. We

use POS tags as the basis for this analysis since they are established syntactic classes. Additionally, POS tags are simply computed using available tools. We show how the attention behavior changes with respect to the POS tag of the word being generated. We show that this is a key difference between attention and alignment models that causes the attention model to capture additional information compared to traditional alignment models. We show that this difference should not be reversed by training the attention model to closely follow the alignment model.

RQ2.4 *What types of words are attended to by the attention model while translating different syntactic phenomena?*

In order to understand what additional information the attention model captures compared to the traditional alignment model, it is useful to know what word types are attended to by the attention model based on the type of the generated words. Are the attended words always the same as the aligned words? Or is attention paid to words other than the alignment points. To answer these questions, we look into the average portion of attention that is paid to words other than the alignment points based on the POS tag of the generated word. The observations provide good evidence for our hypothesis that the attention model captures information that the traditional alignment model is unable to capture.

Our analysis shows that attention follows traditional alignment in some cases more closely while it captures information beyond alignment in others. For instance, attention agrees with traditional alignments to a high degree in the case of nouns. However, in the case of verbs, it captures information beyond just translational equivalence.

This chapter makes the following contributions:

1. We provide a detailed comparison of attention in neural machine translation and traditional word alignment. We show that attention and alignment have different behaviors and attention is not necessarily an alignment model.

2. We show that while different attention mechanisms can lead to different degrees of compliance with respect to word alignments, full compliance is not always helpful for word prediction.

3. We additionally show that attention follows different patterns depending on the type of word being generated. By providing a detailed analysis of the attention model, we contribute to its interpretability. Our analysis explains the mixed results of previous work that has used alignments as an explicit signal to train attention models (Chen et al., 2016, Alkhouli et al., 2016, Liu et al., 2016).

4. We provide evidence showing that the difference between attention and alignment is due to the capability of the attention model to attend to the context words influencing the current word translation. We show that the attention model attends not only to the translational equivalent of a word being generated, but also to

other source words that can affect the form of the target word. For example, while generating a verb on the target side, the attention model regularly attends to the preposition, subject or object of the translational equivalent of the verb on the source side.

4.2 Related Work

The attention model was an important addition to neural machine translation (Bahdanau et al., 2015) and it has been frequently used and adapted by other work since. Luong et al. (2015a) propose a global and a local attention model. They argue that their global attention model is similar to the original attention model by Bahdanau et al. (2015) while having a simpler architecture and requiring less computation. In their local model they predict an alignment point at each step and make the attention focus on a window with the aligned point at the centre. They also propose the input-feeding attention model which takes as input previous attention decisions to better maintain coverage of the source side.

Liu et al. (2016) investigate how training the attention model in a supervised manner can benefit machine translation quality. To this end, they use traditional alignments obtained by running automatic alignment tools (GIZA++ (Och and Ney, 2003) and fast_align (Dyer et al., 2013)) on the training data and feed it as ground truth to the attention network. They report some improvements in translation quality, arguing that the attention model has learned to better align source and target words. Training attention using traditional alignments has also been proposed by others (Mi et al., 2016, Chen et al., 2016, Alkhouli et al., 2016).

Chen et al. (2016) show that guiding attention with traditional alignments helps in the domain of e-commerce translation which includes many out-of-vocabulary (OOV) product names and placeholders. On the other hand, they did not report any notable improvements for other domains.

Alkhouli et al. (2016) have separated the alignment model and translation model, reasoning that this avoids propagation of errors from one model to the other as well as providing more flexibility with respect to model types and the training of models. They use a feed-forward neural network as their alignment model that learns to model jumps on the source side using HMM/IBM alignments obtained by running GIZA++.

Mi et al. (2016) also propose to train the attention model to minimize a distance metric between attention and alignments given by an automatic tool. They show that separate optimization of the distance and translation objectives leads to a performance drop, although joint optimization of the two objectives achieves small gains. They also experiment with first transforming the traditional alignments by using a Gaussian distribution followed by joint optimization which achieves larger performance improvements.

Shi et al. (2016) show that various kinds of syntactic information are being learned and encoded in the output hidden states of the encoder. The neural system for their experimental analysis is not an attention-based model and they argue that attention does not have any impact on learning syntactic information. However, performing the same analysis for morphological information, Belinkov et al. (2017a) show that attention also has some effect on the information that the encoder of a neural machine translation system encodes in its output hidden states. As part of their analysis, they show that a neural machine translation system that has an attention model can learn the POS tags of the source side more efficiently than a system without attention.

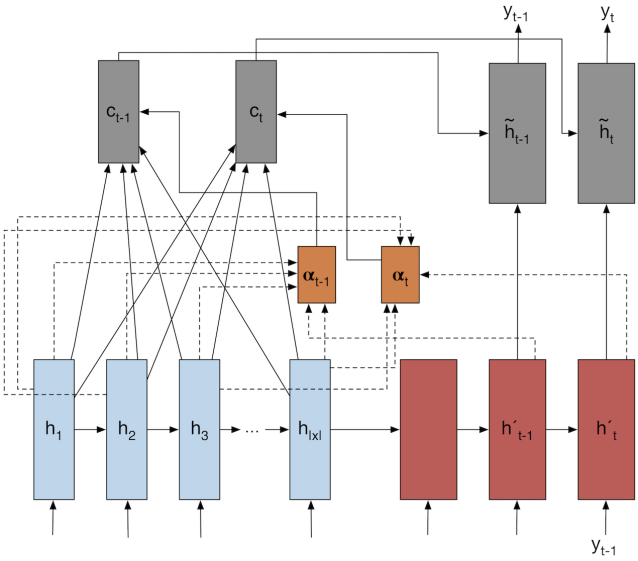
Koehn and Knowles (2017) carry out a simple analysis of the extent to which attention and alignment correlate with each other in different languages by measuring the probability mass that attention gives to alignments obtained from an automatic alignment tool. They also report differences based on the words that received the highest attention. They perform their experiments for both directions for three language pairs. They observe that close to half of the probability mass from the attention distribution is paid to words other than the alignment points in almost all experiments. However, the German-English experiment appears to be an outlier in their experiments, since the alignment points are one position off from the highest attention points. Although we also use German-English for our experiments in this chapter, we do not observe the same shift.

Tang et al. (2018b) investigate attention behavior when translating ambiguous words. They hypothesize that the attention model spreads out its weights to the context words that help disambiguate the correct sense of the ambiguous words. They carry out their experiments on word sense disambiguation datasets that involve the translation of ambiguous nouns. However, they observe the opposite behavior to their hypothesis. Interestingly, the attention model concentrates more of its weight on the source word while translating ambiguous nouns compared to the cases of translating other types of words.

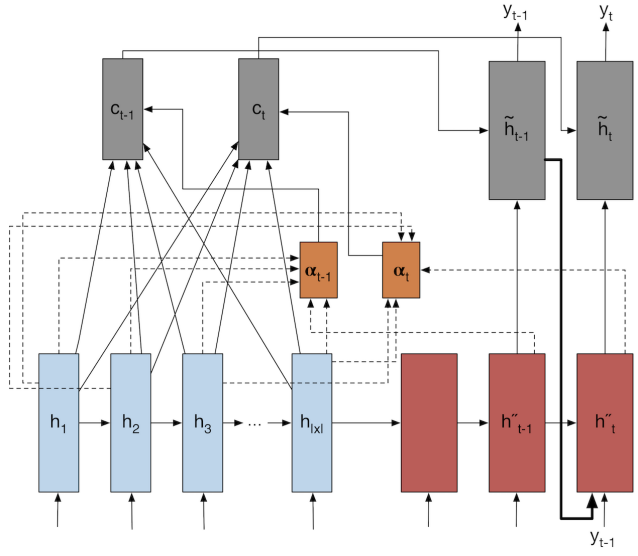
4.3 Attention Models

This section discusses the two popular attention models, which are also used in this chapter. The first model is the non-recurrent attention model, which is equivalent to the "global attention" method proposed by Luong et al. (2015a). The second attention model is *input-feeding* by Luong et al. (2015a). Figure 4.2a and 4.2b show the architecture for global and input-feeding attention models, respectively. The figures are drawn to present the architectures in a way that simply highlights the difference between the two attention models. The bold connection in Figure 4.2b highlights the architectural difference between the two models. Below we describe the details of both models.

Both global and input-feeding models compute a context vector c_i at each decoding time step. Subsequently, they concatenate the context vector to the hidden state of the



(a) Global attention model



(b) Input-feeding attention model

Figure 4.2: Comparison of global and input-feeding attention models. These figures show two time steps of decoding. The difference between the two models is shown by the highlighted feedback connection between $\tilde{h}_{t-1}$ and h''_t in the input-feeding model. This connection feeds the context-dependent output of the decoder from the previous time step back to the decoder as input to the current time step.

decoder and pass it through a non-linearity before it is fed into the softmax output layer of the translation network:

$$\tilde{h}_t = \tanh(W_c[c_t; h'_t]), \tag{4.1}$$

where h'_t and c_t are the hidden state of the decoder and the context vector at time t, respectively; h'_t is defined as follows:

$$h'_t = f(y_{t-1}, h'_{t-1}). \tag{4.2}$$

Here, y_{t-1} is the last generated target word, h'_{t-1} is the previous hidden state of the decoder and f can be an LSTM or GRU recurrent layer.

In the global model, the hidden state of the decoder is compared to each hidden state of the encoder. Often, this comparison is realized as the dot product of vectors, see Equation 4.3. Then the result is fed to a softmax layer to compute the attention weight distribution, see Equation 4.4:

$$e_{t,i} = h_i^T h'_t \tag{4.3}$$

$$\alpha_{t,i} = \frac{\exp(e_{t,i})}{\sum_{j=1}^{|x|} \exp(e_{t,j})}. \tag{4.4}$$

Here h'_t is the hidden state of the decoder at time t, h_i is the i-th hidden state of the encoder and $|x|$ is the length of the source sentence. The attention weights are used to calculate a weighted sum over the encoder hidden states, which results in the context vector mentioned above:

$$c_i = \sum_{i=1}^{|x|} \alpha_{t,i} h_i. \tag{4.5}$$

The difference between global and input-feeding models is based on how the context vector is computed. The input-feeding model changes the context vector computation such that at each step t the context vector is aware of the previously computed context c_{t-1}. To this end, the input-feeding model feeds back $\tilde{h}_{t-1}$ as input to the network and uses the resulting hidden state instead of the context-independent h'_t. This is defined in the following equations:

$$h''_t = f(W[\tilde{h}_{t-1}; y_{t-1}], h''_{t-1}), \tag{4.6}$$

where the corresponding hidden state in global attention is defined as in Equation 4.2. Here, f can be an LSTM or GRU recurrent layer, y_{t-1} is the last generated target word, and $\tilde{h}_{t-1}$ is the output of the previous time step of the input-feeding network itself. $e_{t,i}$ for the input-feeding model is defined as follows:

$$e_{t,i} = h_i^T h''_t. \tag{4.7}$$

Table 4.1: Statistics of manual alignments provided by RWTH German-English data.

	RWTH data
# sentences	508
# alignments	10534
% sure alignments	91%
% possible alignments	9%
% 1-to-1 alignments	74%
% 1-to-2 alignments	12%
% 2-to-1 alignments	7%
% 1-to-n[1] alignments	5%
% n[1]-to-1 alignments	2%

[1] where $n \geqslant 3$

The main difference between the input-feeding and global model is the feed-back in Equation 4.6. Global attention is a simpler model that is computationally more efficient than the input-feeding model (Luong et al., 2015a).

4.4 Comparing Attention with Alignment

As mentioned above, it is a commonly held assumption that attention corresponds to word alignment (Bahdanau et al., 2015, Liu et al., 2016, Cohn et al., 2016, Chen et al., 2016). To verify this assumption, we investigate whether higher consistency between attention and alignment leads to better translations. To this end, we define three metrics, one to measure the discrepancy between attention and alignment, one to measure attention concentration, and one to measure word prediction loss. Then, we measure the correlation between the two attention metrics and the word prediction loss.

4.4.1 Measuring Attention-Alignment Accuracy

In order to compare attentions of multiple systems as well as to measure the difference between attention and word alignment, we convert the hard word alignments into soft ones, following Mi et al. (2016), and use cross entropy between attention and soft alignment as a loss function. For this purpose, we use manual alignments provided by the RWTH German-English dataset (Vilar et al., 2006) as the hard alignments. The statistics of the data are shown in Table 4.1. Almost 10 percent of the alignments are possible alignments. As explained in Section 2.1.1, for possible alignments human experts are not in consensus.

We convert the hard alignments to soft alignments using Equation 4.8. For unaligned words, we first assume that they have been aligned to all words on the source side and

then convert them as follows:

$$Al(x_i, y_t) = \begin{cases} \frac{1}{|A_{y_t}|} & \text{if } x_i \in A_{y_t} \\ 0 & \text{otherwise.} \end{cases} \tag{4.8}$$

Here, A_{y_t} is the set of source words aligned to target word y_t and $|A_{y_t}|$ is the number of source words in the set.

After conversion of the hard alignments to soft ones, we compute the *attention loss* as follows:

$$L_{At}(y_t) = -\sum_{i=1}^{|x|} Al(x_i, y_t) \log(At(x_i, y_t)). \tag{4.9}$$

Here, x is the source sentence and $Al(x_i, y_t)$ is the weight of the alignment link between source word x_i and the target word y_t, see Equation 4.8. $At(x_i, y_t)$ is the attention weight $\alpha_{t,i}$ (see Equation 4.4) of the source word x_i, when generating the target word y_t .

In our analysis, we also look into the relationship between translation quality and the quality of the attention with respect to the alignments. For measuring the quality of attention, we use the attention loss as defined in Equation 4.9. As a measure of translation quality, we choose the loss between the output of our NMT system and the reference translation at each translation step, which we call *word prediction loss*. The word prediction loss for word y_t is the logarithm of the probability given in Equation 4.10.

$$p_{nmt}(y_t \mid y_{<t}, x) = \text{softmax}(W_o \tilde{h}_t). \tag{4.10}$$

Here, x is the source sentence, y_t is target word at time step t, $y_{<t}$ is the target history given by the reference translation and $\tilde{h}_t$ is given by Equation 4.1 for either global attention or input-feeding attention.

Spearman's rank correlation is used to compute the correlation between attention loss and word prediction loss:

$$\rho = \frac{\text{Cov}(R_{L_{At}}, R_{L_{WP}})}{\sigma_{R_{L_{At}}} \sigma_{R_{L_{WP}}}}, \tag{4.11}$$

where $R_{L_{At}}$ and $R_{L_{WP}}$ are the ranks of the attention losses and word prediction losses, respectively, Cov is the covariance between two input variables, and $\sigma_{R_{L_{At}}}$ and $\sigma_{R_{L_{WP}}}$ are the standard deviations of $R_{L_{At}}$ and $R_{L_{WP}}$.

If there is a close relationship between word prediction quality and consistency of attention versus alignment, then there should be a high correlation between word prediction loss and attention loss. Figure 4.3 shows an example with different levels of consistency between attention and word alignments. For the target words "will" and "come" the attention is not focused on the manually aligned word but distributed between the aligned word and other words. The focus of this chapter is to examine cases

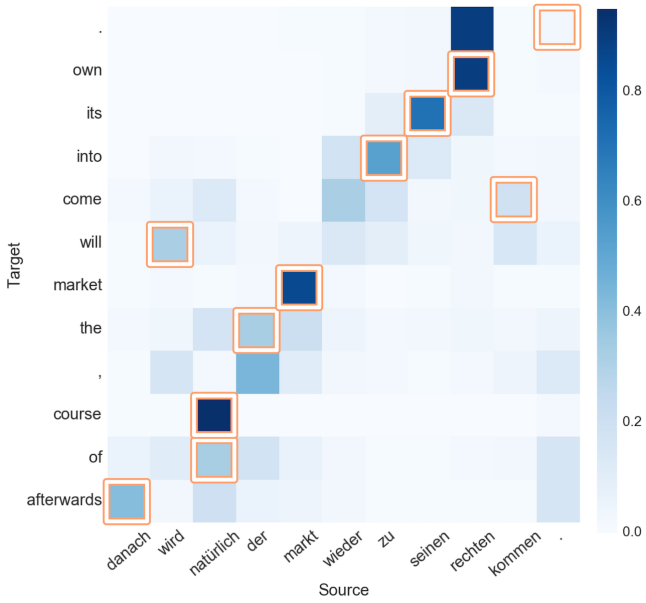


Figure 4.3: An example of inconsistent attention and alignment. The outlined cells show the manual alignments from the RWTH dataset (see Table 4.1). Note how attention deviates from alignment points in the case of "will" and "come".

where attention does not strictly follow alignment, answering the question whether those cases represent errors or are a desirable behavior of the attention model.

4.4.2 Measuring Attention Concentration

As another informative variable in our analysis, we look into the attention concentration. While most word alignments only involve one or a few words, attention can be distributed more freely. We measure the concentration of attention by computing the entropy of the attention distribution:

$$E_{At}(y_t) = -\sum_{i=1}^{|x|} At(x_i, y_t) \log(At(x_i, y_t)). \tag{4.12}$$

As in Equation 4.9, x is the source sentence and $At(x_i, y_t)$ is the attention weight $\alpha_{t,i}$ (see Equation 4.4) of the source word x_i, when generating the target word y_t.

By measuring attention concentration, we can closely trace shifts in the attention distribution and see for which syntactic phenomena the attention model learns to focus its weight and for which phenomena it learns to spread out the weights to other relevant source words. We also investigate the relationship between attention concentration and translation quality to see whether attention concentration has any direct effect on

translation quality for different syntactic phenomena.

Table 4.2: Statistics for the parallel corpus used to train our models. The length statistics are based on the source side.

Data	# of Sent	Min Len	Max Len	Average Len
WMT15	4,240,727	1	100	24.7

4.5 Empirical Analysis of Attention Behavior

We conduct our analysis using the two attention models described in Section 4.3. Our first attention model is the global model as introduced by Luong et al. (2015a). The second model is the input-feeding model (Luong et al., 2015a), which uses recurrent attention. Our NMT system is a unidirectional encoder-decoder system with 4 recurrent layers as described in Luong et al. (2015a).

We train the systems with a dimension size of 1,000 and a batch size of 80 sentences for 20 epochs. The vocabulary for both source and target side is set to be the 30K most common words. The learning rate is set to 1 and a maximum gradient norm of 5 has been used. We also use a dropout rate of 0.3 to avoid overfitting.

Table 4.3: Performance of our experimental system in BLEU on different standard WMT test sets.

System	test2014	test2015	test2016	RWTH
Global	17.80	18.89	22.25	23.85
Input-feeding	19.93	21.41	25.83	27.18

4.5.1 Impact of Attention Mechanism

We train both systems on the WMT15 German-to-English training data, see Table 4.2 for statistics of the data. Table 4.3 shows the BLEU scores (Papineni et al., 2002) for both systems on different test sets.

Since we use POS tags and dependency roles in our analysis, both of which are based on words, we chose not to use BPE (Sennrich et al., 2016), which operates at the sub-word level.

To compute AER over attentions, we follow Luong et al. (2015a) to produce hard alignments from attention distributions by choosing the most attended source word for each target word. We also use GIZA++ (Och and Ney, 2003) to produce automatic alignments over the data set to allow for a comparison between automatically generated

Table 4.4: Alignment error rate (AER) of the hard alignments produced from the output attentions of the systems with input-feeding and global attention models. We use the most attended source word for each target word as the aligned word. The last column shows the AER for the alignment generated by GIZA++.

	Global	Input-feeding	GIZA++
AER	0.60	0.37	0.31

Table 4.5: Average loss between attention generated by input-feeding and global systems and the manual alignment over RWTH German-English data.

	Global	Input-feeding
Attention loss	0.46	0.25

alignments and the attentions generated by our systems. GIZA++ is run in both directions and alignments are symmetrized using the grow-diag-final-and refined alignment heuristic (Och and Ney, 2003).

As shown in Table 4.4, the input-feeding system not only achieves a higher BLEU score, but also uses attentions that are closer to the human alignments.

Table 4.5 compares input-feeding and global attention in terms of attention loss computed using Equation 4.9. Here, the losses between the attentions produced by each system and the human alignments is reported. As expected, the difference in attention loss is in line with AER.

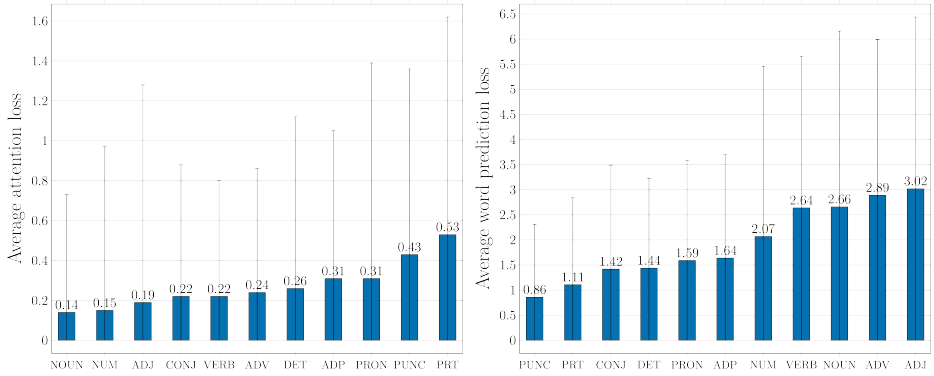
The difference between these comparisons is that AER only takes the most attended word into account while attention loss considers the entire attention distribution.

Table 4.6: List of the universal POS tags used in our analysis.

Tag	Meaning	Example
ADJ	Adjective	large, latest
ADP	Adposition	in, on, of
ADV	Adverb	only, whenever
CONJ	Conjunction	and, or
DET	Determiner	the, a
NOUN	Noun	market, system
NUM	Numeral	2, two
PRT	Particle	’s, off, up
PRON	Pronoun	she, they
PUNC	Punctuation	;, .
VERB	Verb	come, including

4.5.2 Alignment Quality Impact on Translation

Based on the results in Section 4.5.1, one might be inclined to conclude that the closer the attention is to the word alignments, the better the translation. However, earlier research (Chen et al., 2016, Alkhouli et al., 2016, Liu et al., 2016) reports mixed results by optimizing their NMT system with respect to word prediction and alignment quality. These findings warrant a more fine-grained analysis of attention. To this end, we include POS tags in our analysis and study the patterns of attention based on the POS tags of the target words. We choose POS tags because they exhibit simple syntactic characteristics. We use the coarse grained universal POS tags given in Table 4.6 (Petrov et al., 2012).



(a) Average attention loss based on the POS tags of the target side.

(b) Average word prediction loss based on the POS tags of the target side.

Figure 4.4: Average attention losses and word prediction losses from the input-feeding system.

To better understand how attention accuracy affects translation quality, we analyse the relationship between attention loss and word prediction loss for individual POS classes. Figure 4.4a shows how attention loss differs when generating different POS tags. One can see that attention loss varies substantially across different POS tags. In particular, we focus on the cases of NOUN and VERB which are the most frequent POS tags in the dataset. As shown, the attention of NOUN is the closest to alignments on average. But the average attention loss for VERB is almost two times larger than the loss for NOUN.

Considering this difference and the observations in Section 4.5.1, a natural follow-up would be to focus on forcing the attention of verbs to be closer to alignments. However, Figure 4.4b shows that the average word prediction loss for verbs is actually smaller compared to the average loss for nouns. In other words, although the attention for verbs is substantially more inconsistent with the word alignments than for nouns, on average, the NMT system translates verbs more accurately than nouns.

To formalize this relationship we compute Spearman's rank correlation between

word prediction loss and attention loss, based on the POS tags of the target side which is shown in Figure 4.5 for the input-feeding model.

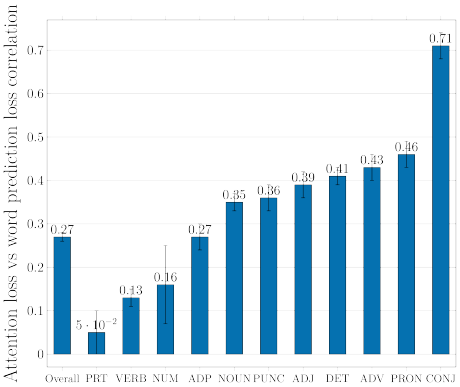


Figure 4.5: Correlation between word prediction loss and attention loss for the input-feeding model.

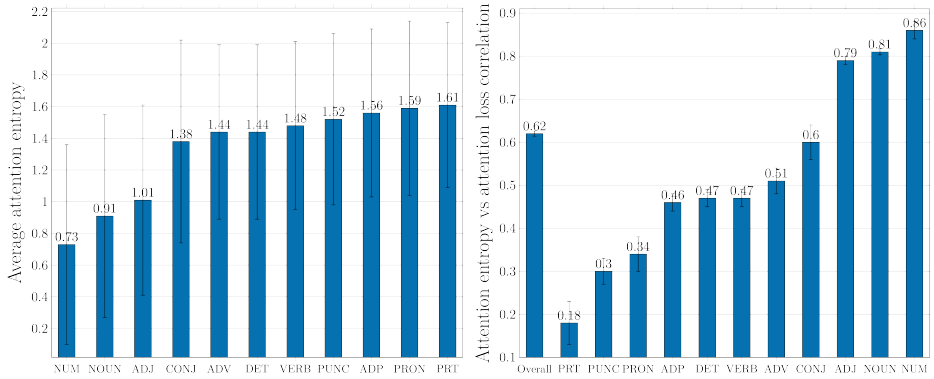
The low correlation for verbs confirms that attention to other parts of the source sentence rather than the aligned word is necessary for translating verbs correctly and that attention does not necessarily have to follow alignments. However, the higher correlation for nouns means that consistency of attention with alignments is more desirable. This can partly explain the mixed results reported for training attention using alignments (Chen et al., 2016, Liu et al., 2016, Alkhouli et al., 2016). This is especially apparent in the results by Chen et al. (2016), where large improvements are achieved for the e-commerce domain, which contains many OOV product names and placeholders, but no or only very small improvements are achieved for general domains.

4.5.3 Attention Concentration

In word alignment, most target words are aligned to one source word (Graça et al., 2010), see Table 4.2. The average number of source words aligned to nouns and verbs is 1.1 and 1.2, respectively. To investigate to what extent this also holds for attention, we measure the attention concentration by computing the entropy of the attention distribution, see Equation 4.12.

Figure 4.6a shows the average entropy of attention based on POS tags. As one can see, nouns have one of the lowest entropies, meaning that on average, the attention for nouns tends to be concentrated. This also explains the closeness of the attention to alignments for nouns. In addition, the correlation between attention entropy and attention loss in the case of nouns is high, as shown in Figure 4.6b. This means that attention entropy can be used as a measure of closeness of attention to alignment in the case of nouns.

The higher attention entropy for verbs, see Figure 4.6a, shows that the attention is



(a) Average attention entropy based on the POS tags.

(b) Correlation between attention entropy and attention loss.

Figure 4.6: Attention entropy and its correlation with attention loss for the input-feeding model.

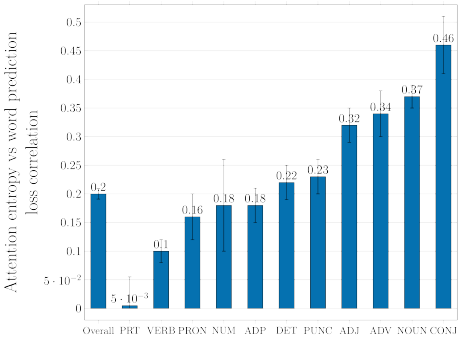


Figure 4.7: Correlation of attention entropy and word prediction loss for the input-feeding model.

more distributed than for nouns. The low correlation between attention entropy and word prediction loss, see Figure 4.7, shows that attention concentration is not required when translating into verbs. This also confirms that the correct translation of verbs requires the systems to pay attention to different parts of the source sentence.

Another interesting observation is the low correlation for pronouns (PRON) and particles (PRT), see Figure 4.7. For example, as can be seen in Figure 4.6a, these tags have a wider attention distribution compared to nouns. This could either mean that the attention model does not know where to focus or that it deliberately pays attention to multiple relevant places to be able to produce a better translation. The latter is supported by the relatively low word prediction losses, as shown in Figure 4.4b.

Table 4.7: Distribution of attention probability mass (in %) paid to alignment points and the mass paid to words other than alignment points for each POS tag.

POS tag	attention to alignment points %	attention to other words %
NUM	73	27
NOUN	68	32
ADJ	66	34
PUNC	55	45
ADV	50	50
CONJ	50	50
VERB	49	51
ADP	47	53
DET	45	55
PRON	45	55
PRT	36	64
Overall	54	46

4.5.4 Attention Distribution

To further understand under which conditions attention is paid to words other than the aligned words, we study the distribution of attention over the source words. First, we measure how much attention, on average, is paid to the aligned words for each POS tag. To this end, we compute the percentage of the probability mass that the attention model has assigned to aligned words for each POS tag, see Table 4.7.

It is interesting that numbers, nouns and adjectives are ranked at the top of the table with most of the attention paid to the alignment points. Intuitively, little attention has to be paid to other context words while translating these types of words. These are cases that benefit when the attention model is trained by explicit signals from traditional alignments. The higher gain in the e-commerce data compared to other domains reported by Chen et al. (2016) also confirms this. However, one can notice that less than half of the attention is paid to alignment points for most of the POS tags. To examine how attention has been distributed for the remaining cases, we measure the attention distribution over dependency roles on the source side. We first parse the source side of the RWTH data using the ParZu parser (Sennrich et al., 2013). Then, we compute how the attention probability mass assigned to words other than the alignment points is distributed over dependency roles. Table 4.8 lists the most attended roles for each POS tag. Here, we focus on the POS tags discussed earlier. One can see that the most attended roles when translating to nouns include adjectives and determiners and, in the case of translating to verbs, they include auxiliary verbs, adverbs (including negation), subjects, and objects.

Table 4.8: The most attended dependency roles with their received attention percentage from the attention probability mass paid to words other than the alignment points. Here, we focus on the POS tags discussed earlier.

POS tag	roles(attention %)	description
NOUN	punc(16%)	Punctuations[1]
	pn(12%)	Prepositional complements
	attr(10%)	Attributive adjectives or numbers
	det(10%)	Determiners
VERB	adv(16%)	Adverbial functions including negation
	punc(14%)	Punctuations
	aux(9%)	Auxiliary verbs
	obj(9%)	Objects[2]
	subj(9%)	Subjects
CONJ	punc(28%)	Punctuations
	adv(11%)	Adverbial functions including negation
	conj(10%)	All members in a coordination[3]

[1] Punctuations have the role "root" in the parse generated using ParZu. However, we use the POS tag to differentiate them from other tokens having the role "root".
[2] Attention mass for all different objects is summed up.
[3] Includes all different types of conjunctions and conjoined elements.

4.6 Attention as Hard Alignment

To further analyze how attention is paid to words other than the alignment points, we look to the cases where the most attended word on the source side is not the alignment point of the generated word of the target side. In other words, we convert (soft) attention to hard alignments by taking the most attended source word for each target word as the alignment point of the target word. Next, we compare the resulting alignments to the manual gold alignments. Almost half of the cases of the resulting alignments are not consistent with the gold alignments. Figure 4.8 shows how these alignments are distributed based on the POS tags on the source and target side. Note that the numbers in this figure are not attention weights. These are the average portions of cases where the highest attended word has the POS tag given on the horizontal axis while generating a word with the POS tag on the vertical axis.

A number of interesting observations emerge from Figure 4.8. Consider the first row that corresponds to generating nouns on the target side. Interestingly, the most attended cases are almost uniformly distributed over different POS tags. This indicates that these are the cases for which the attention has not properly learned what to attend to. For example, it is clear from the first cell from the top left that in 18 percent of the cases the attention model has put most mass on nouns while generating nouns on the target side. However, considering that these are the cases in which attention is inconsistent with the

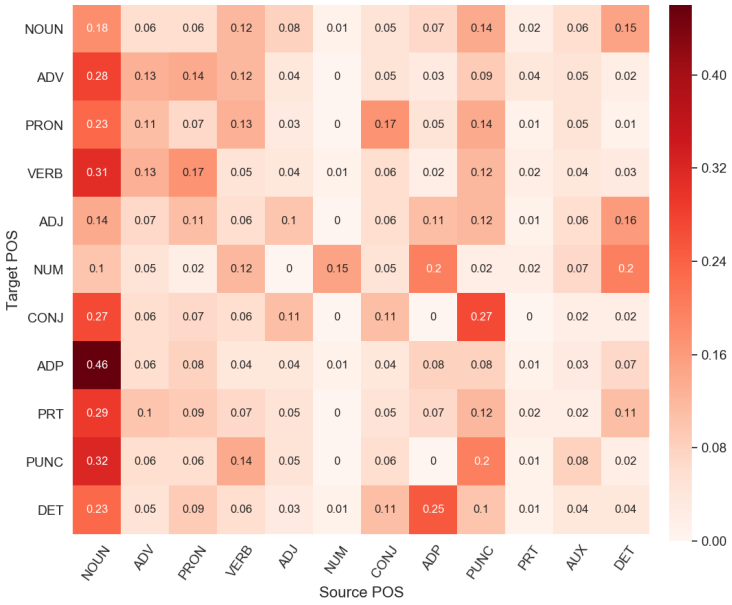


Figure 4.8: The distribution of the most attended words that are not consistent with the alignment points.

alignment, it means that the attention model has devoted its highest weight to another noun rather than the translational equivalent of the generated target word. These are the cases that benefit most from an explicit training signal based on traditional alignments.

Another interesting observation relates to verbs. As can be seen in the corresponding row of the target VERB POS tag, in 31, 13 and 17 percent of the cases the highest attended words on the source side are nouns, adverbs and pronouns, respectively. The attended nouns and pronouns are likely to be the subjects and the objects of the verbs. This is consistent with the results reported in Table 4.8, which shows the attention weights with respect to the dependency roles.

The fact that in almost all cases, the highest attended words on the source side are nouns may warrant further investigation in future research. However, this may also be a side product of the distribution of the word types, since nouns are more frequent than other POS types.

As well as investigating attention-based hard alignments inconsistent with the gold alignments, we also investigated those that are consistent. Figure 4.9 shows the distribution of the cases where the most attended word is the same as the alignment point of the generated target word.

The focus of weights on the same POS tags in the case of nouns, numbers and adjectives reflects that, in most cases, the translation of these words does not require paying attention to the context. These are also the cases that assign most attention to

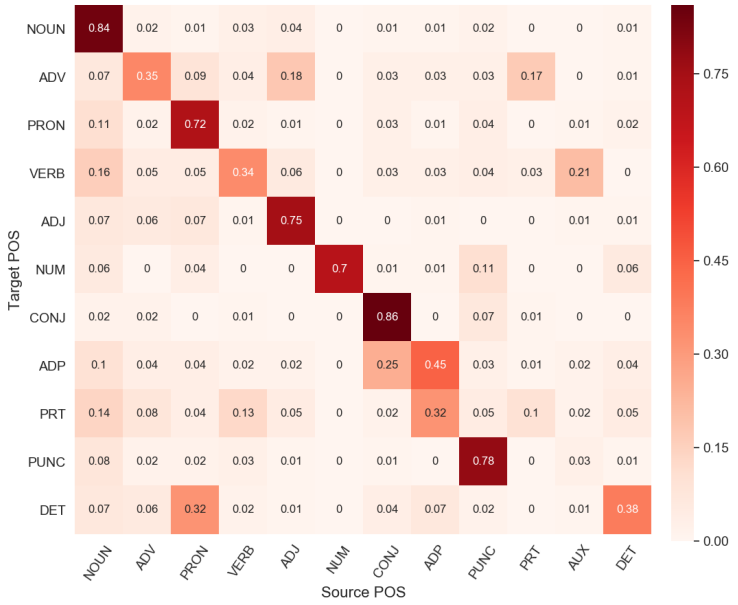


Figure 4.9: The distribution of the most attended words that are consistent with the alignment points.

the alignment points, as shown in Table 4.7.

Figure 4.9 shows that for verbs attention is not focused. This is because our POS tagger tools separate auxiliary verbs and main verbs on the source side and tag them as only verbs on the target side. However, this is different for verbs in Figure 4.8, which contains inconsistent attention to alignment points.

4.7 Conclusion

In this chapter, we have studied attention in neural machine translation and provided an analysis of the relationship between attention and word alignment. We have used pre-existing and new metrics to analyze the behavior of attention across different syntactic phenomena. Our research in this chapter answers the following sub-research questions:

RQ2.1 *To what extent does an attention model agree with the traditional alignment model in capturing translation equivalent words?*

We first checked the extent of agreement between two different attention models and traditional word alignments. We did this by measuring alignment error rate (AER) of global (Luong et al., 2015a) and input-feeding (Bahdanau et al., 2015) attention models and GIZA++, a frequently used automatic alignment tool, against word alignments produced by human annotators.

We have shown that attention agrees with traditional alignments to a limited degree. However, the level of agreement differs substantially by attention mechanism and the type of the word being generated. When generating nouns, attention mostly focuses on the translational equivalent of the target word on the source side. Here, attention behaves very similar to traditional alignments. However, its behavior diverges from that of the traditional alignment model when it comes to generating words that need contextual information to be correctly generated. This is especially the case for verbs that need to be in agreement with their subject or are affected by their auxiliary verb.

Next, we investigate whether the divergence from traditional alignment is due to errors in the attention model or whether it is evidence that attention captures more information than traditional alignments. We initiate this investigation with the following question:

RQ2.2 *Are the differences between attention and traditional alignments due to errors in attention models or do attention models capture additional information compared to traditional alignment models?*

To answer this question, we ensured that the divergence of attention from traditional alignments is not due to errors made by the attention model. To this end, we examine the correlation between translation quality and attention loss as well as attention concentration. The low correlations between these quantities, especially when generating verbs, show that the difference is not due to errors, but a side effect of capturing additional information compared to traditional alignments.

The concentration of attention when generating nouns and adjectives shows that for these syntactic phenomena, attention captures mostly translational equivalence, very similar to traditional alignments. However, while generating verbs more weight is assigned to other context words.

RQ2.3 *How does the distribution of the attention model change for different syntactic phenomena?*

We have shown that attention has different patterns based on the POS tag of the target word. The concentrated pattern of attention and the relatively high correlations for nouns show that training attention with explicit alignment labels is useful for generating nouns. However, this is not the case for verbs, since a large portion of attention is paid to words other than the alignment points capturing other relevant information. In this case, training attention with alignments will force the attention model to forget this useful information. This explains the mixed results reported when guiding attention to comply with alignments (Chen et al., 2016, Liu et al., 2016, Alkhouli et al., 2016).

RQ2.4 *What types of words are attended to by the attention model while translating different syntactic phenomena?*

To answer this question, we investigated the attention weights that are distributed to the source words other than the alignment point or points of the generated target word. We

studied the distribution of these weights with respect to the dependency relations of the alignment points.

We have shown that the attention model learns to assign some attention weights to the dependency relations of a word while translating it. This is specially true in the case of words that need additional context to be translated correctly. For example, verbs, adverbs, conjunctions and particles only receive half of the attention weight while being translated. The rest of the attention is assigned to their dependent words that affect their translations in the target language.

This allows us to answer the overall research question of this chapter:

RQ2 *What are the important words on the source side that attention-based neural models attend to while translating different syntactic phenomena?*

The distributional behavior of the attention model shows that the type of relevant information changes with the syntactic category of the word being generated. We can conclude that for nouns and adjectives, translational equivalents or alignment points are the most important words on the source side. However, for verbs, pronouns, particles, conjunctions and generally most of the other syntactic categories, contextual dependencies are as important as alignment points.

In this chapter, we have contributed to interpretability of an attention model by showing how attention behavior changes for different syntactic phenomena and what the important words for attention are. We have also shown that an attention model cannot interpreted as an alignment model.

In the next chapter, we investigate what information is contained in the encoder hidden states which attention models use to compute the context vectors.

5

Interpreting Hidden Encoder States

5.1 Introduction

It is straightforward to train an NMT system in an end-to-end fashion. This has been made possible by an encoder-decoder architecture that encodes the source sentence into a distributed representation and then decodes this representation into a sentence in the target language. While earlier work has investigated what information is captured by the attention mechanism (Belinkov et al., 2017a, Koehn and Knowles, 2017, Tang et al., 2018b, Voita et al., 2019, Moradi et al., 2019) and the hidden state representations of an NMT system (Shi et al., 2016, Belinkov et al., 2017a, Bisazza and Tump, 2018, Xu et al., 2020), more analyses are still required to better interpret what linguistic information from the source sentence is captured by the encoders' hidden distributed representations. To this end, the primary research question that we seek to answer in this chapter is as follows:

RQ3 *What information is captured by the hidden states of the encoder of a neural machine translation model?*

In the previous chapter, we investigated the attention model, which computes a weighted sum over the encoder hidden states at each decoding time step. The attention model decides which source words to attend to or ignore at each step. However, this interpretation of the attention model has the underlying assumption that each encoder hidden state represents the corresponding source word. In this chapter, we look deeper inside the hidden states to study what information from the source side is encoded in hidden states. In doing so, we investigate whether the assumption that encoder hidden states represent their underlying tokens is valid.

Recently, some attempts have been made to shed some light on the information that is being encoded in the intermediate distributed representations (Shi et al., 2016, Belinkov et al., 2017a). Feeding the hidden states of the encoder of different sequence-to-sequence systems, including multiple NMT systems, as the input to different classifiers, Shi et al. (2016) aim to investigate what syntactic information is encoded in the hidden

states. They provide evidence that syntactic information such as the voice and tense of a sentence and the part-of-speech (POS) tags of words are being learned with reasonable accuracy. They also provide evidence that more complex syntactic information such as the parse tree of a sentence is also learned, but with lower accuracy.

Belinkov et al. (2017a) follow the same approach as Shi et al. (2016) to conduct more analyses about how syntactic and morphological information is encoded in the hidden states of the encoder. They carry out experiments for POS tagging and morphological tagging. They study the effect of different word representations and different layers of the encoder on the accuracy of their classifiers to reveal the impact of these variables on the amount of syntactic information captured by the hidden states.

Despite the approaches discussed above, attempts to study the hidden states more intrinsically are still missing. For example, to the best of our knowledge, there is no work that studies the encoder hidden states from a nearest neighbor perspective to compare these distributed word representations with the underlying word embeddings. It seems intuitive to assume that the hidden state of the encoder corresponding to an input word conveys more contextual information compared to the embedding of the input word itself. But what type of information is captured and how does it differ from the word embeddings? Answering this question will help us understand what information is captured by the hidden states and find the answer to our general research question RQ3. We start by pursuing a more fine-grained research question:

RQ3.1 *How does the captured information in the hidden states differ from the information captured by the word embeddings?*

We investigate this research question by looking into the nearest neighbors of the hidden states and compare them with the nearest neighbors of the corresponding word embeddings. Understanding the similarities and differences between the lists of the nearest neighbors of embeddings and hidden states facilitate human interpretation of the encoder hidden state properties.

As mentioned earlier, we intuitively expect the hidden states to capture more contextual information than word embeddings. Additionally, we already know that word embeddings capture a general representation of all possible senses of a word and the most frequent sense of a word in the training data is dominating the embedding representation (Iacobacci et al., 2015, Faruqui et al., 2016). Observing that neural machine translation systems successfully translate most of the source words with their correct sense in specific contexts, we would expect that hidden states can capture the sense of the words in their context. We further investigate this in the next research question (RQ3.2).

RQ3.2 *To what extent do the hidden states of the encoder capture different types of information, such as syntactic or semantic information?*

Additionally, we use the similarities and differences between the embeddings and their corresponding hidden states to compare the recurrent neural network model and the

transformer model. We compare how much the hidden states of these two types of models contain similar information with their corresponding embeddings. This can help us to understand how much of the capacity of the hidden states in these two models is devoted to other types of information.

Comparing the nearest neighbors list of the word embeddings and the list of the nearest neighbors of their corresponding hidden states we see that there are some words appearing in both and some that only appear in one of the lists. In our investigations to answer research question RQ3.2, we focus on those words that appear in the list of the nearest neighbors of the hidden states and not in the list of the neighbors of the corresponding word embeddings.

Shi et al. (2016) and Belinkov et al. (2017a) have already shown that the hidden states capture syntactic, morphological and semantic information by feeding the hidden state into diagnostic classifiers. However, we aim to investigate how the captured syntactic and lexical semantic information is reflected in the nearest neighbors list. Knowing this is important as it makes the information captured by the hidden states more interpretable. Additionally, we seek to achieve an estimate of the capacity of the hidden states that has been devoted to capture this information. We also investigate the changes made to these capacities throughout the sentences. This is especially interesting in the case of recurrent models since they have to devote more capacity to capturing context information for the last hidden states in each direction when encoding long sentences. Later on, we can use these estimates for a comparison of the functionality of the hidden states in different neural architectures. This is addressed in the final sub research question:

RQ3.3 *How different is the captured information in the encoder hidden states for different NMT architectures?*

In addition to work that has investigated the captured information in hidden states by the use of extrinsic tasks of syntactic and semantic classifications, there are recent approaches that compare different state-of-the-art encoder-decoder architectures in terms of their capabilities to capture syntactic structures (Tran et al., 2018) and lexical semantics (Tang et al., 2018a). This work also uses extrinsic tasks for their comparisons. Tran et al. (2018) use subject-verb agreement and logical inference tasks to compare recurrent models with transformers. On the other hand, Tang et al. (2018a) use subject-verb agreement and word sense disambiguation for comparing those architectures in terms of capturing syntax and lexical semantics, respectively. In addition to these tasks, Lakew et al. (2018) compare recurrent models with transformers in a multilingual machine translation task.

Having defined human interpretable intrinsic measures to capture lexical semantics and syntactic information in the hidden states, we can compare different neural architectures intrinsically in the complex task of machine translation. This helps to understand how the information that is captured by different architectures differs. Here, we compare

recurrent and self-attention architectures, which use entirely different approaches to capture context.

In this chapter, we also try to shed light on the information encoded in the hidden states that goes beyond what is transferred from the word embeddings. To this end, we analyze to what extent the nearest neighbors of words based on their hidden state representations are covered by direct relations in WordNet (Fellbaum, 1998, Miller, 1995). For our German experiments, we use GermaNet (Hamp and Feldweg, 1997, Henrich and Hinrichs, 2010). From now on, we use *WordNet* to refer to either WordNet or GermaNet.

This chapter does not directly seek improvements to neural translation models, but to further our understanding of the behavior of these models. It explains what information is learned in addition to what is already captured by embeddings. This chapter makes the following contributions:

1. We provide an analysis of the representations of hidden states in NMT systems highlighting the differences between hidden state representations and word embeddings.

2. We propose an intrinsic approach to study the syntactic and lexical semantic information captured in the hidden state representations based on the nearest neighbors.

3. We show that the hidden states also capture a positional bias in addition to syntactic and lexical semantic information. The captured positional bias causes some words that do not have any syntactic or lexical semantic similarity to a word to end up in the list of the nearest neighbors of the word.

4. We compare transformer and recurrent models in a more intrinsic way in terms of capturing lexical semantics and syntactic structures. This is in contrast to previous work which focuses on extrinsic performance.

5. We provide analyses of the behavior of the hidden states for each direction layer and the concatenation of the states from the direction layers.

5.2 Related Work

Shi et al. (2016) were the first to use diagnostic classifiers to investigate the intermediate hidden states in sequence-to-sequence recurrent neural models. Using diagnostic classifiers based on the encoder hidden states, they verify that syntactic information is captured by the hidden states to some extent. They use two groups of syntactic features that cover word and sentence level syntactic labels. They use voice and tense of a sentence together with the top level syntactic sequence of the constituent tree as the sentence-level labels. As word-level labels, they use POS tags and the smallest

phrase constituent above each word. They show that the hidden states of a sequence-to-sequence model trained for machine translation capture the syntactic labels accurately without having to be retrained for the classification task. They also investigate whether the hidden states capture deeper syntactic information when being trained on a translation task. Interestingly, their reported results show that the hidden states can even capture the constituent structure of the source sentence to a reasonable degree missing only some subtle details.

Belinkov et al. (2017a) build upon the work by Shi et al. (2016) and expand the investigation to different variables, ranging from language specific features to architectural features. The authors use the extrinsic tasks of POS and morphological tagging as classification tasks into which they feed the intermediate hidden states to investigate how good a feature representation they are. They investigate four variables that possibly influence the information captured by the hidden state representations: (i) the richness of the morphology of the source and the target languages, (ii) whether a word-based or character-based architecture is used for the sequence-to-sequence model, (iii) the layer within the model from which the hidden states representations are captured, and (iv) the effect of attention on the hidden state representations from the decoder side of the model.

Using diagnostic classifiers to investigate hidden states has become a popular method. Giulianelli et al. (2018) and Hupkes et al. (2018) also use this method to inspect subject and verb agreement in the hidden states in language models and hierarchical structures learned by the hidden states in recurrent and recursive neural models. Giulianelli et al. (2018) show that letting diagnostic classifiers fix the intermediate representations can even lead to a boost in language modeling performance.

Wallace et al. (2018) take a nearest neighbor approach to investigating feature importance in deep neural networks. They adopt a nearest neighbors based conformity score introduced by Papernot and McDaniel (2018) to measure the impact of leaving features out on the classification performance. The nearest neighbor based conformity score is shown to be a robust decision measure which does not affect classification performance and also results in robustness and interpretability. This method uses the agreement of the nearest neighbors of the intermediate hidden states, saved during training with their corresponding labels, to make classification decisions during prediction. While this method is difficult to apply to sequence-to-sequence models, it shows the importance of the nearest neighbors of the intermediate hidden states for making robust predictions with neural networks.

Ding et al. (2017) use layer-wise relevance propagation (LRP) (Bach et al., 2015) to study the contributions of words to the intermediate hidden states in a neural machine translation model. The LRP method was originally developed to compute the relevance of a single pixel for the predictions of an image classifier. They modify the method to be able to compute the relevance between two arbitrary hidden states in a neural machine translation model. This way, they can compute the contribution of the source and target word embeddings to the internal hidden states and to the next word being

generated on the target side. The authors use the relevance computation to study the commonly observed errors made by neural machine translation systems including word omission, word repetition, unrelated word generation, negation reversion, and spurious word generation.

Kádár et al. (2017) introduce an omission score to measure the contribution of an input token to the predictions of a recurrent neural network for the two tasks of language modeling and multi-modal meaning representations, where they use textual and visual input. They show that the network in language modeling task learns to pay more attention to tokens with a syntactic function.

In addition to work that pursues interpretability of the hidden states in neural sequence-to-sequence models by either diagnostic classifiers or relevance computation, there is work that seeks interpretability of complex neural models by comparing different neural architectures on some extrinsic tasks (Tran et al., 2018, Tang et al., 2018a).

In this chapter, we use an intrinsic approach to both investigate the information captured by the hidden states and to compare two commonly used neural machine translation architectures. This contributes to the interpretability of hidden states in both architectures.

5.3 Datasets and Models

We conduct our analyses using recurrent (Bahdanau et al., 2015) and transformer (Vaswani et al., 2017) machine translation models. Our recurrent model is a two-layer bidirectional recurrent model with Long Short-Term Memory (LSTM) units (Hochreiter and Schmidhuber, 1997) and global attention (Luong et al., 2015a). The encoder consists of a two-layer unidirectional forward and a two-layer unidirectional backward pass. The corresponding output representations from each direction are concatenated to form the encoder hidden state representation for each token. A concatenation and down-projection of the last states of the encoder is used to initialize the first hidden state of the decoder. The decoder uses a two-layer unidirectional (forward) LSTM. We use no residual connections in our recurrent model as they have been shown to result in a performance drop if used on the encoder side of a recurrent model (Britz et al., 2017). Our transformer model is a 6-layer transformer with multi-head attention using 8 heads (Vaswani et al., 2017). We choose these settings to obtain competitive models with the relevant core components from each architecture.

Figure 5.1 and Figure 5.2 show the hidden states ($H_1, H_2, H_3, ..., H_T$) that are of interest for our study in this chapter. These are the encoder hidden states that are fed to the cross attention mechanism in each architecture. Each of the figures shows the encoder hidden states in their respective neural machine translation architecture. Figure 5.1 and Figure 5.2 show a recurrent neural architecture and a transformer architecture, respectively. Figure 5.1 also shows the direction-wise hidden states ($\overrightarrow{h}_1, \overleftarrow{h}_1, \overrightarrow{h}_2, \overleftarrow{h}_2, \overrightarrow{h}_3, \overleftarrow{h}_3, ..., \overrightarrow{h}_T, \overleftarrow{h}_T$) that are studied in Section 5.5.5.

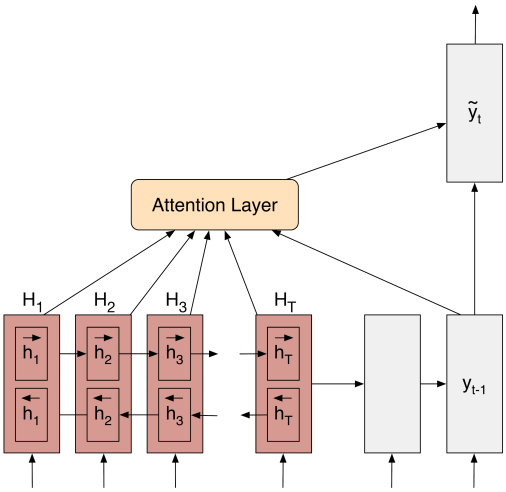


Figure 5.1: Encoder hidden states, $(H_1, H_2, H_3, ..., H_T)$, of a recurrent model that are studied in this chapter.

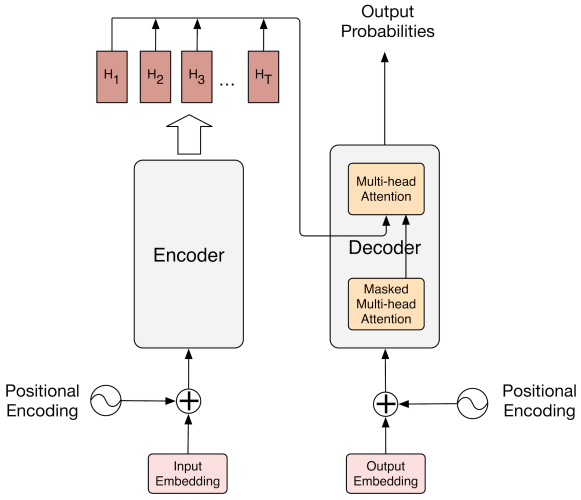


Figure 5.2: Encoder hidden states, $(H_1, H_2, H_3, ..., H_T)$, of a transformer model that are studied in this chapter.

We train our models for two translation directions, namely English-German and German-English, both of which use the WMT15 parallel training data (Bojar et al., 2015). We exclude 100k randomly chosen sentence pairs which are used as our held-out data. Our recurrent system has hidden state dimensions of size 1,024 (512 for each direction) and is trained using a batch size of 64 sentences. The learning rate is set

to 0.001 for the Adam optimizer (Kingma and Ba, 2015) with a maximum gradient norm of 5. A dropout rate of 0.3 is used to avoid overfitting. Our transformer model has hidden state dimensions of 512 and a batch size of 4096 tokens and uses layer normalization (Ba et al., 2016). A learning rate of 2, changed under a warm-up strategy with 8000 warm-up steps, is used for the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.998$ and $\epsilon = 10^{-9}$ (Vaswani et al., 2017). The dropout rate is set to 0.1, and no gradient clipping is used. The word embedding size of both models is 512. We apply Byte-Pair Encoding (BPE) (Sennrich et al., 2016) with 32K merge operations.

Table 5.1: Performance of our experimental systems in BLEU on WMT (Bojar et al., 2017) German-English and English-German standard test sets.

English-German

Model	test2014	test2015	test2016	test2017
Recurrent	24.65	26.75	30.53	25.51
Transformer	26.93	29.01	32.44	27.36

German-English

Model	test2014	test2015	test2016	test2017
Recurrent	28.40	29.61	34.28	29.64
Transformer	30.15	30.92	35.99	31.80

We train our models until convergence and then use the trained models to compute and log the hidden states for 100K sentences from a held-out dataset to use in our analyses. The remaining sentence pairs of the WMT15 parallel training data are used as our training data.

Table 5.1 summarizes the performance of our experimental models in terms of BLEU (Papineni et al., 2002) on different standard test sets.

5.4 Nearest Neighbors Analysis

Following earlier work on word embeddings (Mikolov et al., 2013a, Pelevina et al., 2016), we choose to look into the nearest neighbors of the hidden state representations to learn more about the information encoded in them. We treat each hidden state as the representation of the corresponding input token. This way, each *occurrence* of a word has its own representation. Based on this representation, we compute the list of n nearest neighbors of each word occurrence, using cosine similarity as the distance measure. We set n equal to 10 in our experiments.

In the case of our recurrent neural model, we use the concatenation of the corresponding output representations of our two-layer forward and two-layer backward passes as the hidden states of interest for our main experiments. We also use the output representations of the forward and the backward passes for our direction-wise experiments. In the case of our transformer model, we use the corresponding output of the top

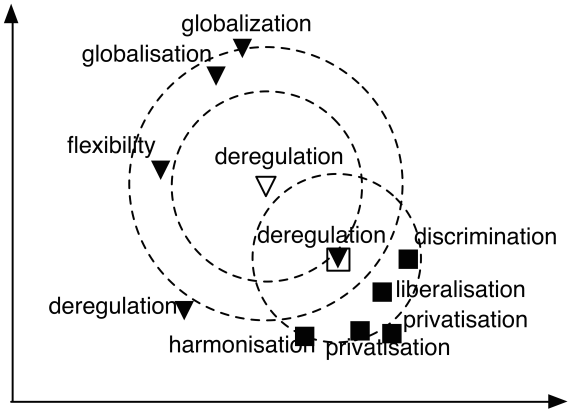


Figure 5.3: An example of 5 nearest neighbors of two different occurrences of the word "deregulation". Triangles are the nearest neighbors of one occurrence of "deregulation" shown with the empty triangle. Squares are the nearest neighbors of another occurrence of "deregulation" shown with the empty square.

layer of the encoder for each word occurrence as the hidden state representation of the word.

Figure 5.3 shows an example of 5 nearest neighbors for two different occurrences of the word "deregulation". Each item in this figure is a specific word occurrence, but we have removed their context information for the sake of simplicity. This figure shows how different occurrences of the same word can share no nearest neighbor in the hidden states space. It is interesting that one of the occurrences of the word "deregulation" is the nearest neighbor of the other, but not vice versa.

5.4.1 Hidden States vs. Embeddings

Here, we count how many of the words in the nearest neighbors lists of hidden states are covered by the nearest neighbors list based on the corresponding word embeddings. Just like the hidden states, the word embeddings used for computing the nearest neighbors are also from the same system and the same trained model for each experiment. The nearest neighbors of the word embeddings are also computed using cosine similarity. It should be noted that we generate the nearest neighbors lists for the embeddings and the hidden states separately and never compute cosine similarity between word embeddings and the hidden state representations.

Coverage is formally computed as follows:

$$cp^{H,E}_{w_{i,j}} = \frac{\left|C^{H,E}_{w_{i,j}}\right|}{\left|N^{H}_{w_{i,j}}\right|}, \tag{5.1}$$

where

$$C^{H,E}_{w_{i,j}} = N^{H}_{w_{i,j}} \cap N^{E}_{w} \tag{5.2}$$

and $N^{H}_{w_{i,j}}$ is the set of the n nearest neighbors of word w based on hidden state representations. Since there is a different hidden state for each occurrence of a word, we use i as the index of the sentence of occurrence and j as the index of the word in the sentence. Similarly, N^{E}_{w} is the set of the n nearest neighbors of word w, but based on the embeddings.

Word embeddings tend to capture the dominant sense of a word, even in the presence of significant support for other senses in the training corpus (Pelevina et al., 2016). Additionally, it is reasonable to assume that a hidden state corresponding to a word occurrence captures more of the context-specific sense of the word. Comparing the lists can provide useful insights as to which hidden state-based neighbors are not strongly related to the corresponding word embedding. Furthermore, it shows in what cases the dominant information encoded in the hidden states comes from the corresponding word embedding and to what extent other information has been encoded in the hidden state.

5.4.2 WordNet Coverage

In addition to comparisons with word embeddings, we also compute the coverage of the list of the nearest neighbors of hidden states with the directly related words from WordNet. This can shed further light on the capability of hidden states in terms of learning the sense of the word in the current context. Additionally, it constitutes an intrinsic measure to compare different architectures by their ability to learn lexical semantics. To this end, we check how many words from the nearest neighbors list of a word, based on hidden states, are in the list of related words of the word in WordNet. More formally, we define R_w to be the union of the sets of synonyms, antonyms, hyponyms and hypernyms of word w in WordNet:

$$cp^{H,W}_{w_{i,j}} = \frac{\left|C^{H,W}_{w_{i,j}}\right|}{\left|N^{H}_{w_{i,j}}\right|}, \tag{5.3}$$

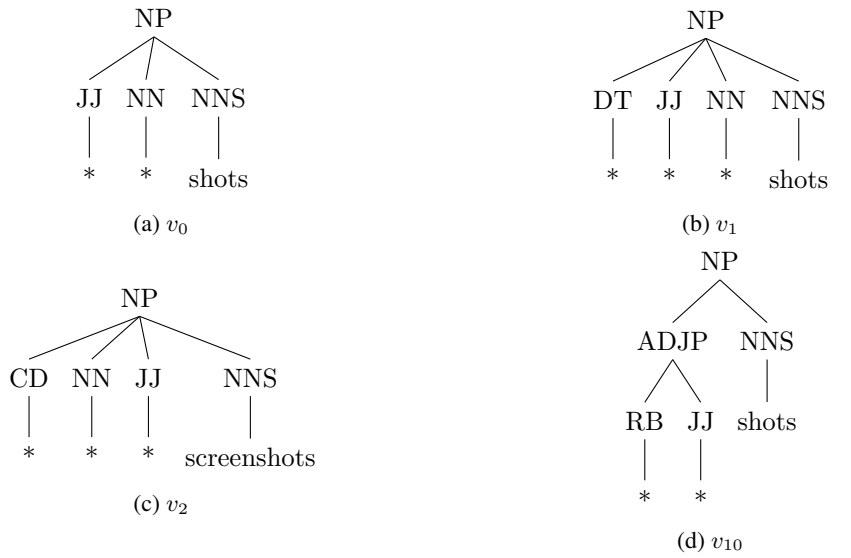
where

$$C^{H,W}_{w_{i,j}} = N^{H}_{w_{i,j}} \cap R_w \tag{5.4}$$

and $N^{H}_{w_{i,j}}$ is the set of the n nearest neighbors of word w based on hidden state representations.

5.4.3 Syntactic Similarity

Recent comparisons of recurrent and non-recurrent architectures for learning syntax (Tran et al., 2018, Tang et al., 2018a) also motivate a more intrinsic comparison of how much different syntactic and lexical semantic information they capture. To this end, we



(a) v_0 (b) v_1 (c) v_2 (d) v_{10}

Figure 5.4: The figure shows the corresponding word and constituent subtree of query hidden state (v_0) and the corresponding word and subtree of the first (v_1) and the second (v_2) and the last (v_{10}) nearest neighbors of it.

also study the nearest neighbors of hidden states in terms of syntactic similarities. For this purpose, we use the subtree rooting in the smallest phrase constituent above each word, following Shi et al. (2016). This way, we will have a corresponding parse tree for each word occurrence in our corpus. We parse our corpus using the Stanford constituent parser (Zhu et al., 2013). We tag our corpus with POS tags and parse its sentences prior to applying BPE segmentation. Then, after applying BPE, we use the same POS tag and the same subtree of a word for its BPE segments, following Sennrich and Haddow (2016).

To measure the syntactic similarity between a hidden state and its nearest neighbors, we use the PARSEVAL standard metric (Sekine and Collins, 1997) as similarity metric between the corresponding trees. PARSEVAL computes precision and recall by counting the correct constituents in a parse tree with respect to a ground truth tree and divides the count by the number of constituents in the candidate parse tree and the ground truth tree, respectively.

Figure 5.4a shows the corresponding word and subtree of a hidden state of interest. Figures 5.4(b–d) show the corresponding words and subtrees of its three neighbors. The leaves are substituted with dummy "*" labels to show that they do not influence the computed tree similarities. We compute the similarity score between the corresponding tree of each word and the corresponding trees of its nearest neighbors. For example, in Figure 5.4 we compute the similarity score between the tree in Figure 5.4a and each of the other trees.

5.4.4 Concentration of Nearest Neighbors

Each hidden state, together with its nearest neighbors, behaves like a cluster centered around the corresponding word occurrence of the hidden state, where the neighboring words give a clearer indication of the captured information in the hidden state. However, this evidence is more clearly observed in some cases rather than others, as in some cases the neighboring words appear unrelated.

The stronger the similarities that bring the neighbors close to a hidden state, the more focused the neighbors around the hidden state are. Bearing this in mind, we choose to study the relation between the concentration of the neighbors and the information encoded in the hidden states.

To make it simple but effective, we estimate the population variance of the neighbors' distances from a hidden state as the measure of the concentration of its neighbors. More formally, this is computed as follows:

$$v_{w_{i,j}} = \frac{1}{n} \sum_{k=1}^{n} (1 - x_{k,w_{i,j}})^2. \tag{5.5}$$

Here, n is the number of neighbors and $x_{k,w_{i,j}}$ is the cosine similarity score of the k-th neighbor of word w occurring as the j-th token of the i-th sentence.

5.4.5 Positional Distribution of the Nearest Neighbors

When inspecting many examples of the nearest neighbors of hidden states, we noticed that some positional information is being encoded in the hidden states. To test this hypothesis, we propose to estimate the distribution of the relative position of the nearest neighbors of the hidden states.

We define the relative position as the position of a hidden state, or corresponding token, with respect to the length of the sentence that the token appears in. The relative position of a token is computed as follows:

$$rp_{w_{i,j}} = \left\lfloor \frac{j * 10}{l(i)} \right\rfloor. \tag{5.6}$$

Here, $w_{i,j}$ is the j-th token of the i-th sentence and $l(i)$ is the length of the sentence.

We compute the positional distribution of the nearest neighbors for each relative position. Based on this definition, we have 10 bins of relative positions. For each relative position we count the number of neighbors in the same bin. Then we use these numbers to estimate a normal distribution for each relative position.

5.5 Empirical Analyses

We train our systems for English-German and German-English and use our trained model to compute the hidden state representations on held-out data of 100K sentences. We log the hidden state representations together with their corresponding source tokens and their sentence and token indices.

We use the logged hidden states to compute the nearest neighbors of the tokens with frequencies between 10 and 2000 in our held-out data. We compute cosine similarity to find the nearest neighbors.

In addition to the hidden states, we also log the word embeddings from the same system and the same trained model. Similar to the hidden states, we also use embedding representations to compute the nearest neighbors of words. We have to note that in the case of embedding representations we have one nearest neighbor list for each word whereas for hidden states there is one list for each occurrence of a word.

Table 5.2: Percentage of the nearest neighbors of hidden states covered by the list of the nearest neighbors of embeddings.

Model	POS	English-German	σ^2	German-English	σ^2
Recurrent	All POS	18%	4	24%	7
	VERB	29%	5	31%	5
	NOUN	14%	3	19%	8
	ADJ	19%	3	31%	7
	ADV	36%	5	48%	2
Transformer	All POS	37%	14	33%	10
	VERB	39%	8	36%	7
	NOUN	38%	16	31%	14
	ADJ	32%	11	36%	9
	ADV	33%	12	38%	3

5.5.1 Nearest Neighbors Coverage of Embedding

As a first experiment, we measure how many of the nearest neighbors based on the embedding representation are retained as nearest neighbors of the corresponding hidden state, as described in Section 5.4.1.
Table 5.2 shows the statistics of the coverage by the nearest neighbors based on embeddings in general and based on selected source POS tags for each of our models. To carry out an analysis based on POS tags, we tagged our training data using the Stanford POS tagger (Toutanova et al., 2003). We convert the POS tags to the universal POS tags and report only for POS tags available in WordNet. We use the same POS tag of a word for its BPE segments, as described in the Section 5.4.3.

The first row of Table 5.2 shows that for our recurrent model only 18% (for English) and 24% (for German) of the information encoded in the hidden states is already

captured by the word embeddings. Interestingly, in all cases except ADV, the similarity between the hidden states and the embeddings is much higher for the transformer model, and the increase for nouns is much higher than for the rest. This may be a product of not using recurrence in the transformer model which results in a simpler path from each embedding to the corresponding hidden state. We hypothesize that this means that the recurrent model uses the capacity of its hidden states to encode other information that is encoded to a lesser extent in the hidden states of the transformer model.

Table 5.3: Percentage of the nearest neighbors of hidden states covered by the list of the directly related words to the corresponding word of the hidden states in WordNet.

Model	POS	English-German	σ^2	German-English	σ^2
Recurrent	All POS	24%	6	51%	12
	VERB	49%	9	48%	10
	NOUN	19%	3	28%	8
	ADJ	15%	2	60%	12
	ADV	24%	4	23%	1
Transformer	All POS	67%	16	74%	10
	VERB	77%	9	70%	9
	NOUN	65%	18	63%	13
	ADJ	66%	14	81%	9
	ADV	74%	10	35%	5

5.5.2 WordNet Coverage

Having observed that a large portion of nearest neighbors of the hidden states are still not covered by the nearest neighbors of the corresponding word embeddings, we look for other sources of similarity that cause the neighbors to appear in the list. As our next step, we check to see how many of the neighbors are covered by directly related words of the corresponding word in WordNet.

This does not yield subsets of the nearest neighbors that are fully disjoint with the subset covered by the nearest neighbors from the embedding list. However, it still shows whether this source of similarity is fully covered by the embeddings or whether the hidden states capture information from this source that the embeddings miss.

Table 5.3 shows the general and the POS-based coverage for our English-German and German-English systems. The transformer model again has the lead by a large margin. The jump for nouns is again the highest, similar to what we have already seen in Table 5.2. This basically means that more words from the WordNet relations of the word of interest are present in the hidden state nearest neighbors of the word. A simple explanation for this is that the hidden states in the transformer model capture more word semantics than the hidden states in the recurrent model. Or in other words, the hidden states in the recurrent model capture some additional information that brings words beyond the WordNet relations to its neighborhood.

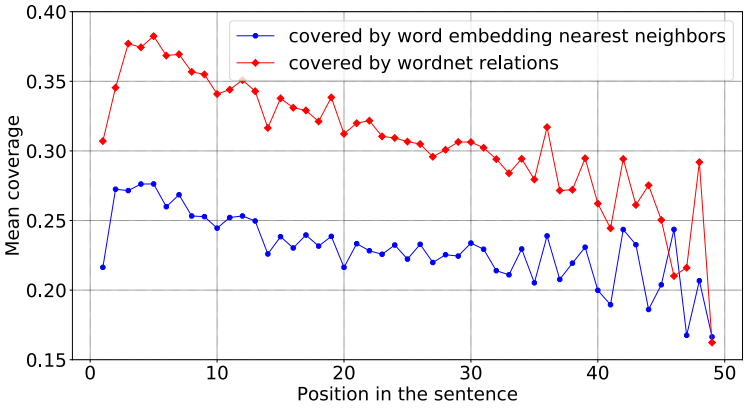


Figure 5.5: The mean coverage per position of the nearest neighbors of hidden states in the recurrent model; (i) by the nearest neighbors of the embedding of the corresponding word, and (ii) by WordNet related words of the corresponding word of the hidden state.

To investigate whether recurrency has any direct effect on the observed difference between the transformer and the recurrent model in Table 5.3, we compute the mean coverage by direct relations in WordNet per position. Similarly, we also compute the mean coverage by embedding neighbors per position. More formally, we write:

$$acp_j^{H,W} = \frac{\sum_{i=1}^{m}(cp_{w_{i,j}}^{H,W})}{\left|S_{l(s)\geqslant j}\right|} \tag{5.7}$$

and

$$acp_j^{H,E} = \frac{\sum_{i=1}^{m}(cp_{w_{i,j}}^{H,E})}{\left|S_{l(s)\geqslant j}\right|} \tag{5.8}$$

for the mean coverage by WordNet direct relations per position and the mean coverage by embedding neighbors per position, respectively.

Here $cp_{w_{i,j}}^{H,W}$ and $cp_{w_{i,j}}^{H,E}$ are the values computed in Equation 5.3 and 5.1, respectively. The function $l(s)$ returns the length of sentence s and $S_{l(s)\geqslant j}$ is the set of sentences that are longer than or equal to j.

Figure 5.5 shows that for the recurrent model the mean coverage for both embedding and WordNet is first increasing, but starts to decrease from position 5 onwards. This drop in coverage is surprising, considering that the model is a bidirectional recurrent model. However, this may be a reflection of the fact that the longer a sentence, the less the hidden states encode information about the corresponding word.

Figure 5.6 shows the same mean coverage for the hidden states in the transformer model. The absence of decreasing coverage with regard to position, confirms our hypothesis that the lower coverage for recurrent models indeed directly relates to the recurrency.

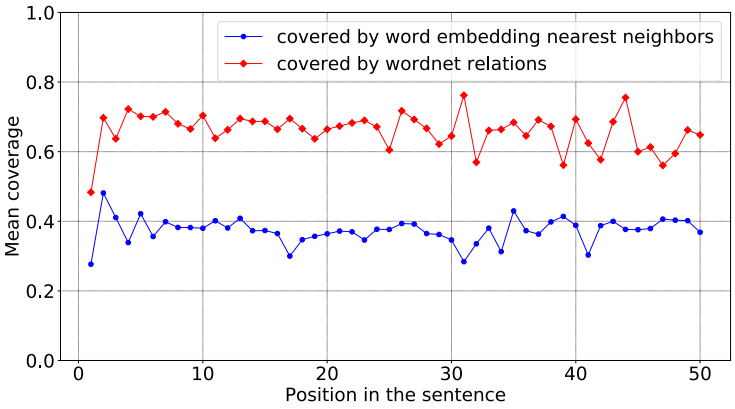


Figure 5.6: The mean coverage per position of the nearest neighbors of hidden states in the transformer model; (i) by the nearest neighbors of the embedding of the corresponding word, and (ii) by WordNet related words of the corresponding word of the hidden state.

In order to refine the analysis of the positional behavior of the hidden states, we compute the average variance per position of the cosine distance scores of the nearest neighbors based on the hidden states. To compute this value we use the following definition:

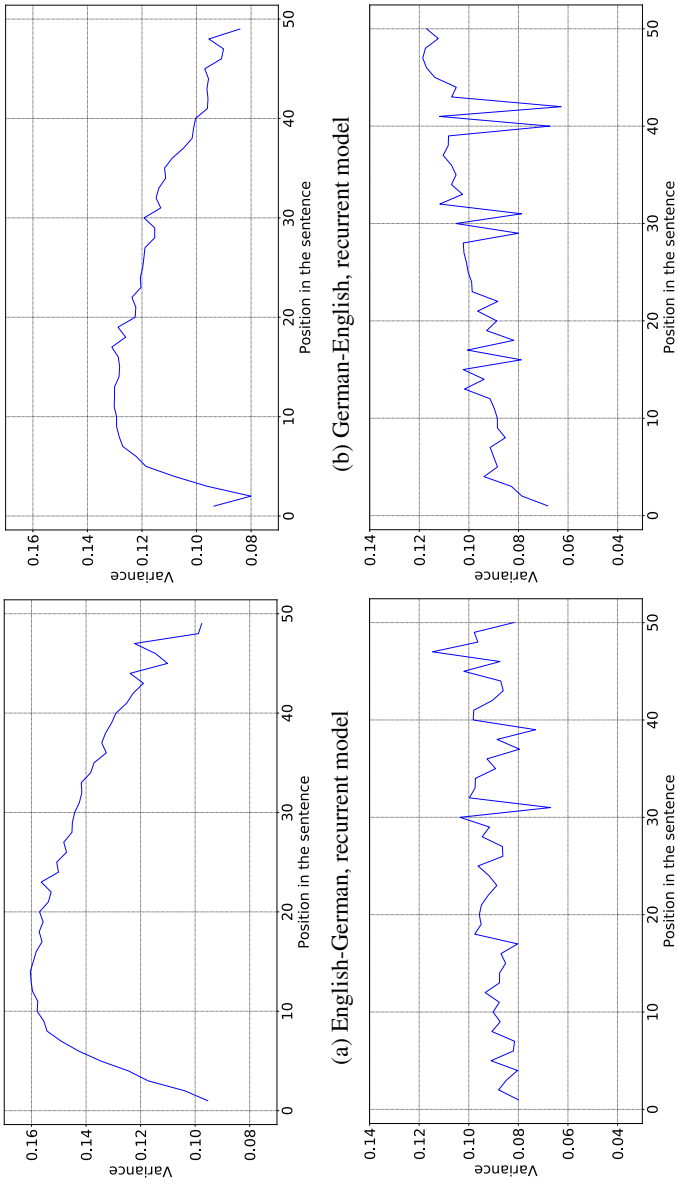
$$Av_j = \frac{\sum_{i=1}^{m} v_{w_{i,j}}}{\left|S_{l(s)\geqslant j}\right|}. \tag{5.9}$$

Here, $v_{w_{i,j}}$ is the variance estimate as defined in Equation 5.5, $l(s)$ is the function returning the length of sentence s and $S_{l(s)\geqslant j}$ is the set of sentences that are longer than or equal to j as mentioned earlier.

Figures 5.7a and 5.7b show the average variance per position for the recurrent model. One can see that the average variance close to the borders is lower than the variance in the middle. This means that the nearest neighbors of the words close to the borders of sentences are more concentrated in terms of similarity score in general. This could mean that the lexical meaning of those words plays less of a role compared to other information encoded in the corresponding hidden states, especially if we take the coverage per position into account. Interestingly, this does not happen for the transformer model; see Figures 5.7c and 5.7d.

5.5.3 Positional Bias

To test the hypothesis whether the nearest neighbors of the hidden states are sharing some positional bias, we estimate the position distribution of the nearest neighbors. This can also provide more evidence for the behavior of the hidden states close to the borders of sentences.



(a) English-German, recurrent model

(b) German-English, recurrent model

(c) English-German, transformer model

(d) German-English, transformer model

Figure 5.7: The average variance of cosine distance scores of the nearest neighbors of words per positions.

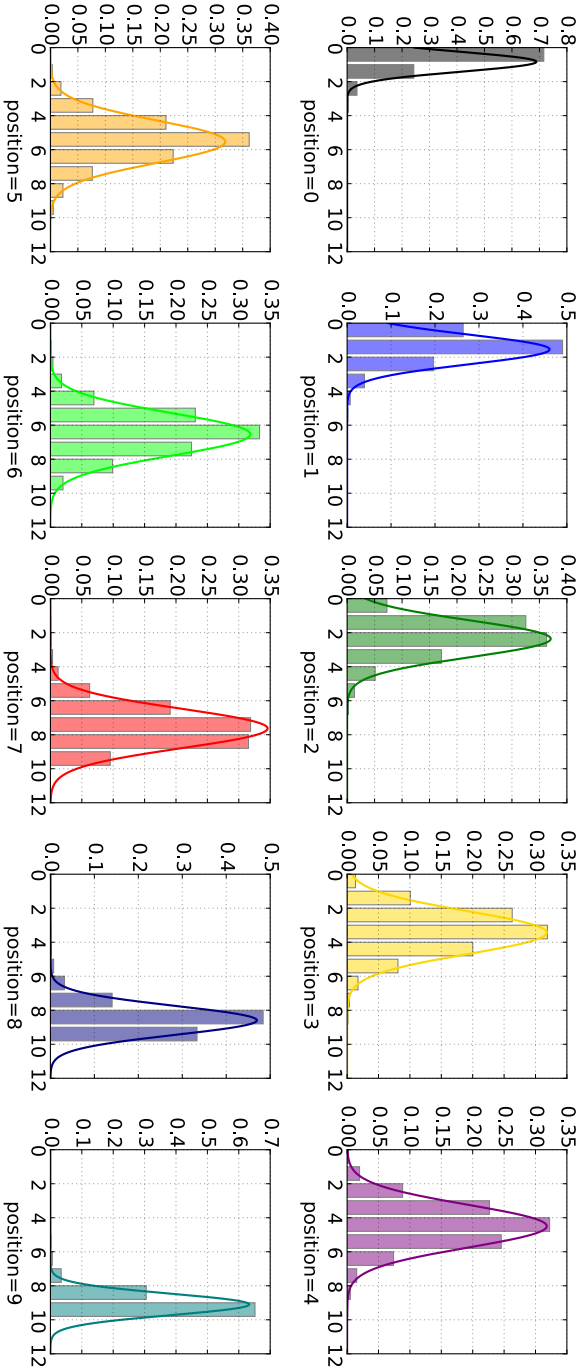


Figure 5.8: Estimated distributions of the relative positions of the nearest neighbors of hidden states, per bin of relative position. This is for the English-German experiment.

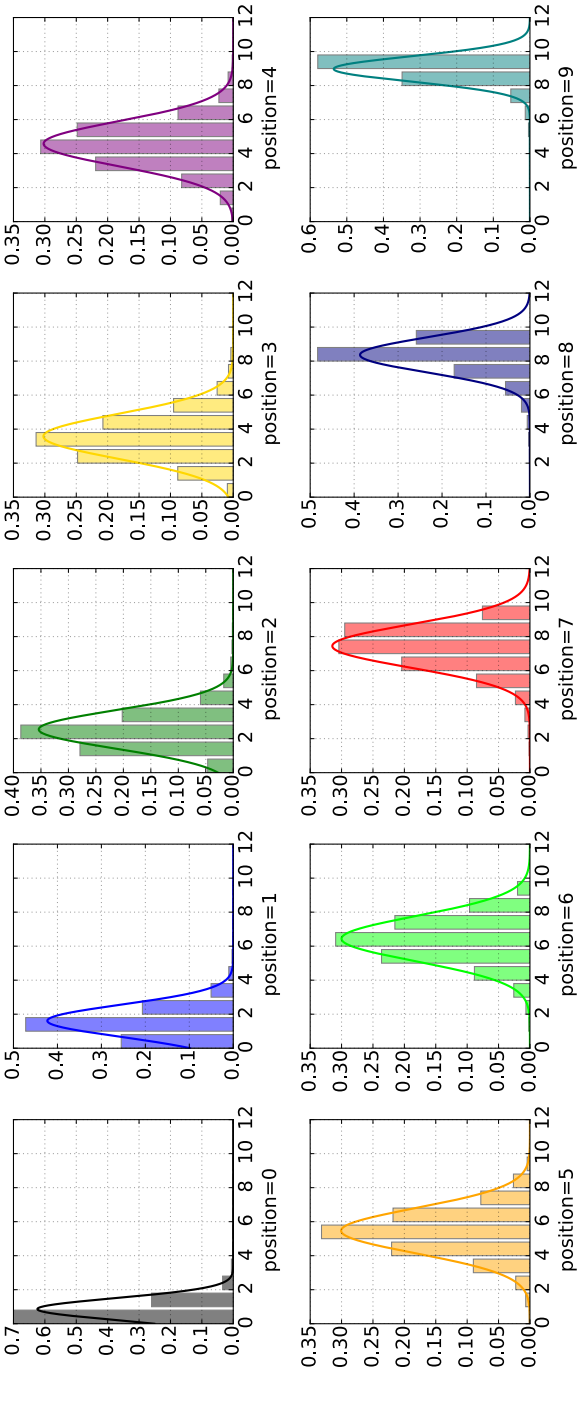
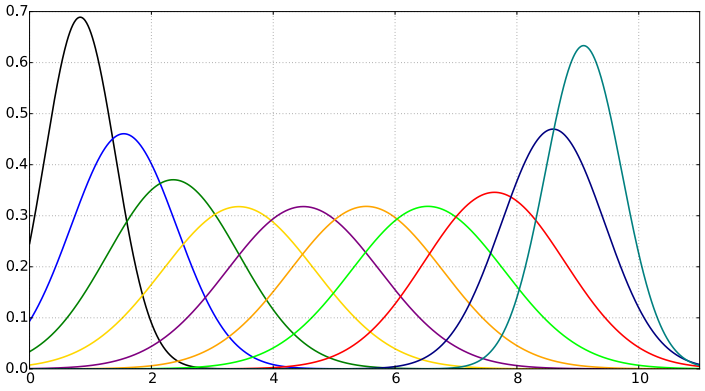
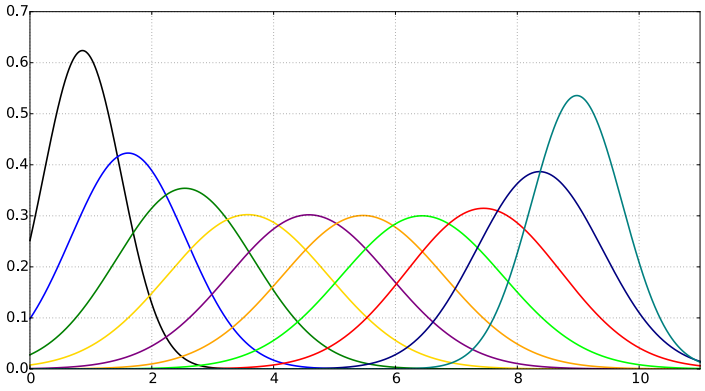


Figure 5.9: Estimated distributions of the relative positions of the nearest neighbors of hidden states, per bin of relative position. This is for the German-English experiment.



(a) English-German



(b) German-English

Figure 5.10: Estimated distributions of the relative positions of the nearest neighbors of hidden states per bin.

Figure 5.8 and Figure 5.9 show the distributions of relative position of nearest neighbors for the 10 bins of relative positions, as explained in Section 5.4.5 for the English and German source sides, respectively. The closer it is to the borders, the stronger the positional bias becomes and the more nearest neighbors result from closer positions.

Figure 5.10 combines the distributions from Figure 5.8 and Figure 5.9 in two respective plots to make it easier to compare. Note the higher concentration (lower variance) of the distributions near the borders in both languages. For the sake of readability, we have removed the histograms in this figure.

This bias causes some words that do not share lexical or syntactic similarities with the words close to the borders to end up in their nearest neighbors list. This could be interpreted as an example of where corpus-level discourse dependency (Wang and Cho, 2016) in recurrent neural networks may help improve language modelling performance.

Table 5.4: Average parse tree similarity (PARSEVAL scores) between word occurrences and their nearest neighbors. Note that the apparent identity of precision and recall values is due to rounding and the very close number of constituents in the candidate and gold parse trees.

English-German

Model	Recurrent	Transformer
Precision	0.38	0.31
Recall	0.38	0.31
Matched Brackets	0.42	0.35
Cross Brackets	0.31	0.28
Tag Accuracy	0.46	0.40

German-English

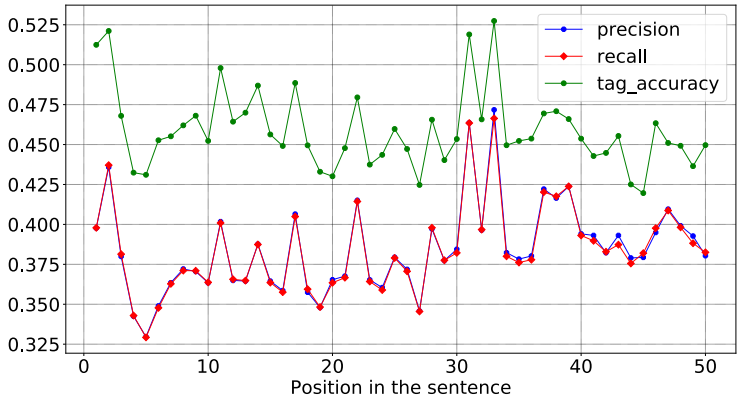
Model	Recurrent	Transformer
Precision	0.12	0.11
Recall	0.12	0.11
Matched Brackets	0.30	0.28
Cross Brackets	0.80	0.77
Tag Accuracy	0.32	0.31

5.5.4 Syntactic Similarity

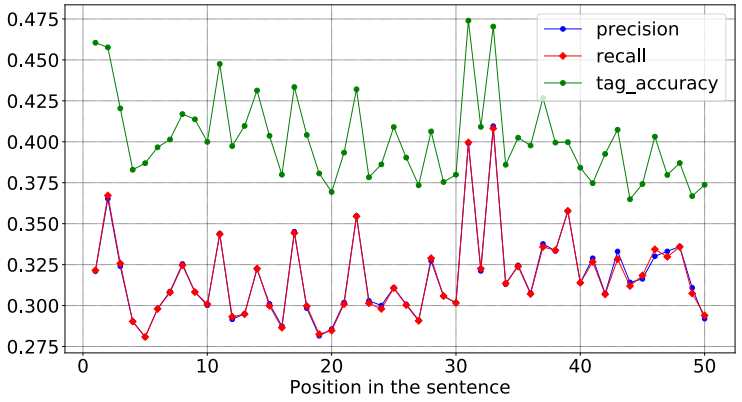
The differences between recurrent and transformer models, as we have seen for the coverage of embeddings and WordNet relations, motivate a comparison of these models based on the syntactic similarities between the hidden states and their nearest neighbors. Especially because it has been shown on extrinsic tasks that recurrent models are superior in capturing syntactic structures (Tran et al., 2018). To this end, we use the approach introduced in Section 5.4.3.

Table 5.4 shows the average similarity between corresponding constituent subtrees of hidden states and corresponding subtrees of their nearest neighbors, computed using PARSEVAL (Sekine and Collins, 1997). Interestingly, the recurrent model has the highest average syntactic similarity. This confirms our hypothesis that the recurrent model dedicates more of the capacity of its hidden states, compared to transformer, to capturing syntactic structures. It is also in agreement with the results reported on implicit learning of syntactic structures using extrinsic tasks (Tran et al., 2018). We should add that our approach may not fully explain the degree to which syntax in general is captured by each model, but only to the extent to which this is measurable by comparing syntactic structures using PARSEVAL.

Figures 5.11 and 5.12 show the average syntactic similarity scores per position in the sentences with length up to 50 for both German-English and English-German and for both recurrent and transformer models. We performed this analysis to check whether there is any trending increase or decrease similar to what we have observed for the mean coverage scores, see Section 5.4.2.



(a) Recurrent model.

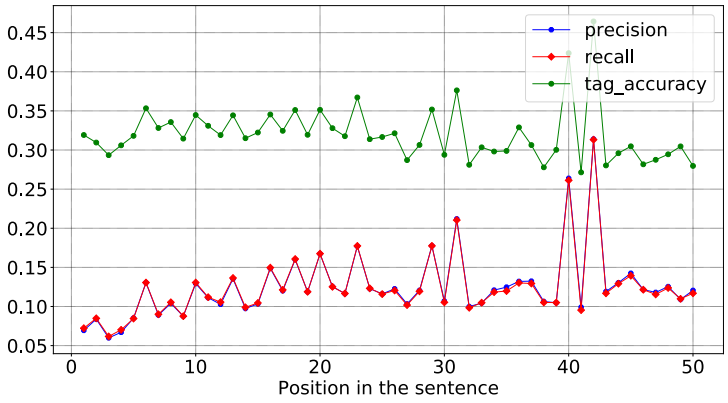


(b) Transformer.

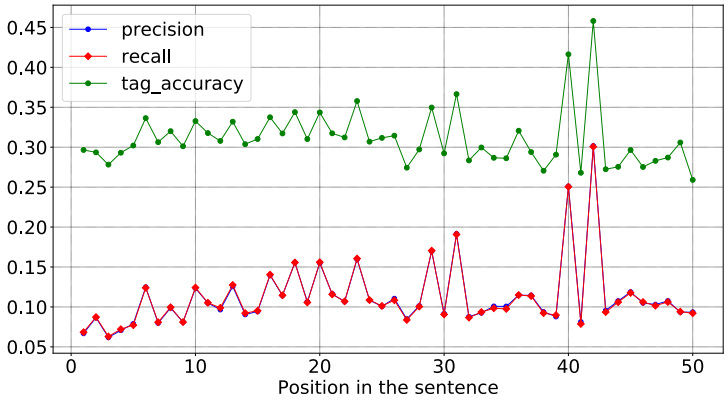
Figure 5.11: Average syntactic similarity for each sentence position for sentences with the length up to 50 tokens for English-German.

Interestingly, there are slightly increasing trends in the precision and the recall scores for both the transformer and the recurrent model in both translation directions. However, the figures show that the change is more pronounced for the recurrent model. The recurrent model not only generally achieves higher syntactic similarity scores compared to the transformer model, but the score difference also increases by observing more context.

Another interesting observation in Figures 5.11 and 5.12 is the similarity of the general patterns of changes in the diagrams for the recurrent and transformer model. This may mean that the difficulty of syntactic structures throughout the sentences is almost the same for both models. We attribute this difficulty to the depth of a word in the syntactic tree structure. However, future studies are required to confirm this suspicion.



(a) Recurrent model.



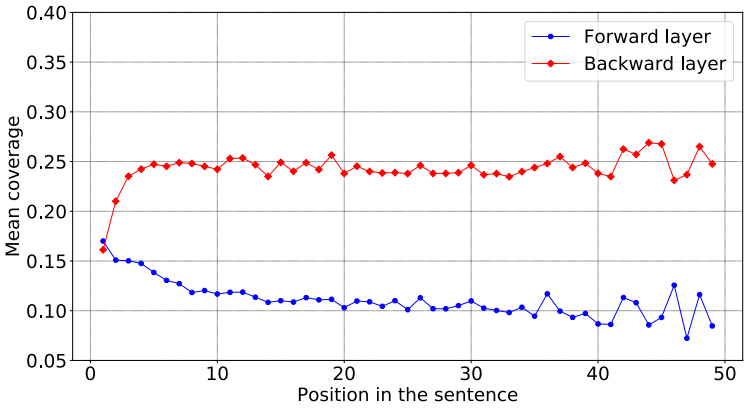
(b) Transformer.

Figure 5.12: Average syntactic similarity for each sentence position for sentences with the length up to 50 tokens for German-English.

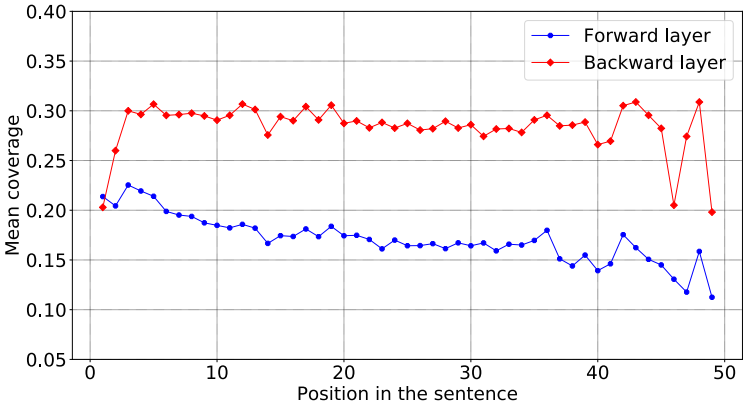
Although the syntactic similarity scores are higher for the recurrent model, the changing pattern of the average similarity scores throughout sentences is nearly identical for both models.

5.5.5 Direction-Wise Analyses

To gain a better understanding of the importance of directionality on the hidden states in the recurrent model, we repeat our experiments with hidden states from different directions. Note that so far the recurrent hidden states in our experiments were the concatenation of the hidden states from both directions of our encoder. In Section 5.3, Figure 5.1 shows both direction-wise hidden states and their concatenations. So far we have used the hidden states $H_1, H_2, H_3, ..., H_T$ from both the



(a) Covered by the nearest neighbors of the embedding of the corresponding word of the hidden state.



(b) Covered by the directly related words of the corresponding word of the hidden state in WordNet.

Figure 5.13: The mean coverage per position of the nearest neighbors of hidden states from the forward and backward recurrent layers.

recurrent and the transformer model. In the discussion below, we use the hidden states $\overrightarrow{h}_1, \overleftarrow{h}_1, \overrightarrow{h}_2, \overleftarrow{h}_2, \overrightarrow{h}_3, \overleftarrow{h}_3, ..., \overrightarrow{h}_T, \overleftarrow{h}_T$ from the recurrent model.

Table 5.5 shows the statistics of embedding coverage and WordNet coverage from the forward and the backward layers. As shown, the coverage of the nearest neighbors of the hidden states from the backward recurrent layer is higher than the nearest neighbors based on those from the forward layer.

Furthermore, Figure 5.13 shows the mean coverage per position of the nearest neighbors of hidden states from the forward and the backward recurrent layers. Figure 5.13a shows to what extent the nearest neighbors of a hidden state are on average covered by the nearest neighbors of its corresponding word embedding, see Equation 5.1 in

Table 5.5: Percentage of the nearest neighbors of hidden states, from the forward and backward layers, that are covered by the list of the nearest neighbors of embeddings and the list of the directly related words in WordNet.

POS	Embedding		WordNet	
	Forward	Backward	Forward	Backward
All	12%	24%	18%	29%
VERB	19%	36%	38%	52%
NOUN	9%	21%	14%	25%
ADJ	13%	22%	12%	17%
ADV	28%	34%	20%	23%

Section 5.4.1. As shown for the forward layer the coverage degrades towards the end of sentences. However, the coverage for the backward layer, except for the very beginning, almost stays constant. The coverage for the backward layer is much higher than the coverage for the forward layer indicating that it keeps more information from the embeddings compared to the forward layer. The decrease in the forward layer indicates that it captures more context information over time and "forgets" more of the corresponding embeddings.

Figure 5.13b shows the extent to which the direct WordNet relations of a word are on average covered by the nearest neighbors of its corresponding hidden state, see Equation 5.3 in Section 5.4.2. The difference between the coverage of the nearest neighbors of hidden states from the backward layer compared to those from the forward layer confirms that the lexical semantics of words is better captured in the backward layer. This may be indicative of a division of responsibilities between the forward and the backward layer. To this end, we investigate syntactic information in the direction-wise hidden states.

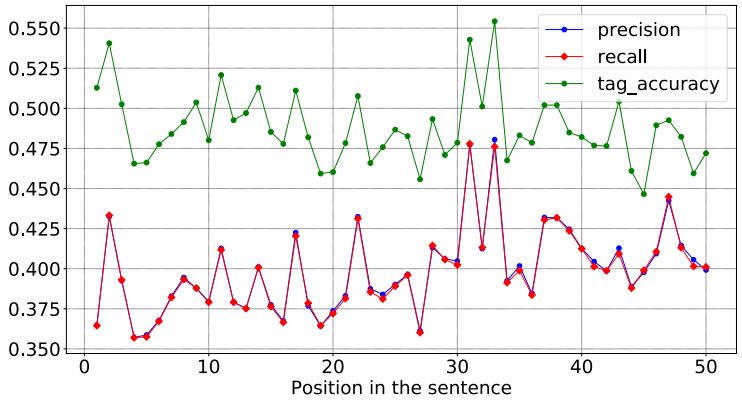
Direction-Wise Syntactic Similarity

In addition to investigating the hidden states from the backward and forward layers by computing and comparing the embedding and WordNet coverage, we also examine the syntactic similarities.

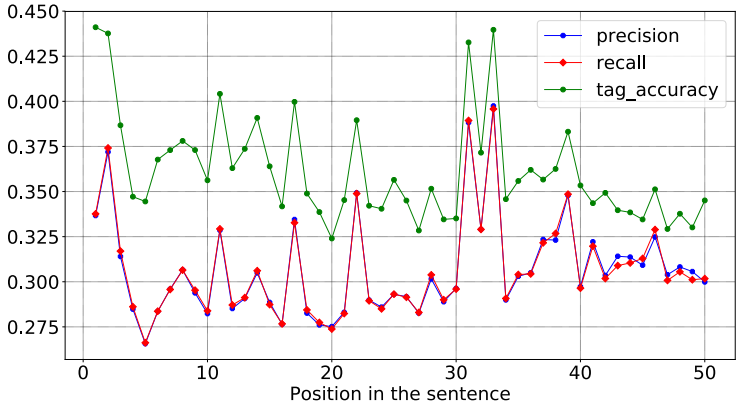
Table 5.6: Average parse tree similarity (PARSEVAL scores) between word occurrences and their nearest neighbors for forward and backward layers.

English-German

Model	Forward Layer	Backward Layer
Precision	0.39	0.30
Recall	0.39	0.30
Matched Brackets	0.43	0.35
Cross Brackets	0.29	0.30
Tag Accuracy	0.49	0.37



(a) Forward layer.



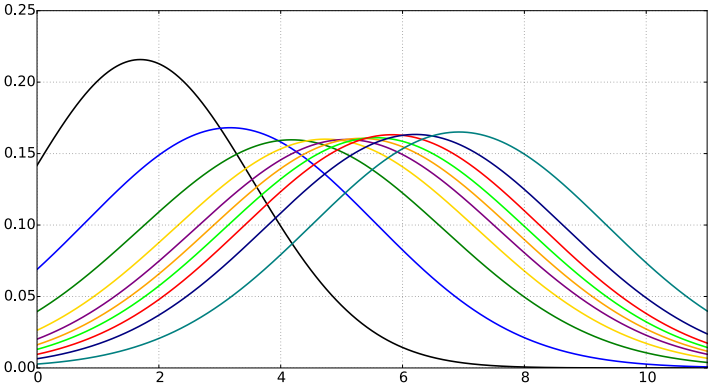
(b) Backward layer.

Figure 5.14: Average syntactic similarity for each sentence position throughout sentences with lengths of up to 50 tokens for the forward and the backward layers from the recurrent model.

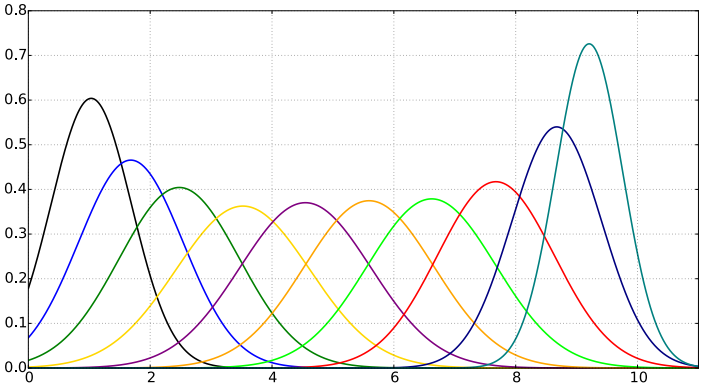
Table 5.6 reports the similarity scores for the hidden states for both directions. Interestingly, the syntactic similarity for the hidden states from the forward layer is higher than the similarity for the hidden states from the backward layer. This difference is evidence for our hypothesis that the forward layer uses more of its capacity to capture context information compared to the backward layer. This also explain the drop in the embedding and WordNet coverages in the forward layer as the hidden states from this layer devote more capacity to context information.

Figures 5.14a and 5.14b show the average syntactic similarity scores (precision, recall and tag accuracy) per position for sentences with length of up to 50 for the forward and the backward layers, respectively. Both figures show similarly increasing trends

over time. However, the scores for the forward layer are generally higher than for the backward layer, see also Table 5.6. This means that the forward layer learns more syntactic information compared to the backward layer. This confirms that there is a devision of responsibilities between both layers.



(a) Forward layer



(b) Backward layer

Figure 5.15: Estimated distributions of the relative positions of the nearest neighbors of hidden states, from the different layers, per bin of relative position.

Direction-Wise Positional Bias

Figure 5.15 visualizes the positional bias for the hidden states for the forward and backward directions. Interestingly, there is almost no positional bias for the hidden states in the forward layer, which entails positional biases are mostly coming from the backward layer hidden states. Putting the other observations from the direction-wise analysis together with these observations, it is clear that the positional bias is related to learning lexical semantics as it is observed in the layer that captures more lexical

semantics. The forward layer, which learns more syntax, has almost no positional bias. The forward layer also has a higher variance, which means it takes more positions into account to capture wider contexts required for encoding syntactic structure. The positional bias distributions also support our hypothesis that there is a division of responsibilities between the forward and backward layer.

5.6 Conclusion

In this chapter, we introduced an intrinsic way of investigating the information encoded in the hidden states of neural machine translation models by examining the nearest neighbors of the encoder hidden states. We conducted our study by answering the following sub-research questions:

RQ3.1 *How does the captured information in the hidden states differ from the information captured by the word embeddings?*

Although some of the information carried by the word embeddings is transferred to the corresponding hidden states, we show that the hidden state representations capture information beyond the corresponding embeddings. We show that the hidden states capture more of the WordNet relations of the corresponding word than they capture from the nearest neighbors of the embeddings.

RQ3.2 *To what extent do the hidden states of the encoder capture different types of information, such as syntactic or semantic information?*

We studied the lists of the nearest neighbors of the hidden states from various perspectives. We examined to what extent the capacity of the hidden states is devoted to capturing different types of information, including syntactic and lexical semantic information. By computing the shared portion of the nearest neighbors between the hidden states and the corresponding embeddings, we have an estimate of the capacity of the hidden states used to keep useful embedding information. Besides, our WordNet coverage score is an estimate of the extent to which the hidden states capture context-dependant lexical semantics. It is important to note that our definition of embedding coverage and WordNet coverage are not mutually exclusive. It means that there may be neighbors that are counted in computing both scores. However, WordNet coverage always trumps embedding coverage, which shows that the hidden states encode lexical information beyond what is captured by word embeddings. Additionally, the comparison of the scores from the two architectures provides additional insights into the extent to which each of the architectures encodes lexical semantics.

As mentioned before, to examine the syntactic information encoded by the hidden states, we compute the similarity of the corresponding local parse trees of the words of interest versus the parse trees of their nearest neighbors. The computed similarity scores

give an estimate of the extent to which each of the studied architectures is capable of capturing syntax.

RQ3.3 *How different is the captured information in the encoder hidden states for different NMT architectures?*

Using the proposed intrinsic methods, we compare the recurrent and transformer architectures in terms of capturing syntax and lexical semantics. For a detailed comparison we use the metrics of our intrinsic study which can be interpreted as an estimation of the amount of syntactic and lexical semantics information captured by the hidden states. We report embedding coverage, WordNet coverage, syntactic similarity, positional bias and concentration of nearest neighbors for the transformer and recurrent architectures and compare the models using these scores. We also study the changing trends of some of these measures throughout sentences to compare the behavior of the hidden states of the two architectures across time. We show that the transformer model is superior in terms of capturing lexical semantics, while the recurrent model better captures syntactic similarity.

Additionally, we investigated various types of linguistic information captured by the different directions of a bidirectional recurrent model. We show that the backward recurrent layer captures more lexical and semantic information, whereas the forward recurrent layer captures more long-distance, contextual information. We also show that the forward recurrent layer has less positional bias compared to the backward recurrent layer.

Finally, our analyses indicate that the positional bias is more prevalent as it gets closer to the borders of sentences and that it dominates lexical and semantic similarities at these regions. This leads to nearest neighbors that are irrelevant in terms of lexical and semantic similarity and only share positional similarities with the word of interest.

Considering the answers for our sub-research questions, we can now go back to our main research question.

RQ3 *What information is captured by the hidden states of the encoder of a neural machine translation model?*

By measuring the similarities of the nearest neighbors of the hidden states, we show that only part of the capacity of the hidden states is devoted to transferring the information already carried by the corresponding word embeddings. We also show that the hidden states capture WordNet-aligned lexical semantics beyond what is captured by the word embeddings. Furthermore, we demonstrate that the hidden states use some of their capacity to encode syntactic information and some positional biases, depending on the architecture of the network. Our findings contribute to the interpretability of neural machine translation by showing what information is captured by the hidden states and to what extent this differs by architecture.

6
Conclusions

In this thesis we have investigated how phrase-based and neural machine translation capture different syntactic and semantic phenomena. Our goal was to contribute to the interpretability of both phrase-based and neural machine translation by studying the role of different words in phrase reordering in phrase-based machine translation and by analyzing the behavior of the attention model and the behavior of the hidden state representations in neural machine translation.

Machine translation systems are among the most complex NLP systems. In phrase-based machine translation systems, phrase reordering is one of the most difficult tasks (Bisazza and Federico, 2016, Zhang et al., 2007) and as a result the models addressing this problem have often been the most complex components of the systems. Deep neural machine translation models showed significant improvements in handling reordering (Bentivogli et al., 2016), but at the cost of interpretability. The motivation for the work in this thesis has been to provide a deeper understanding of how phrase-based and neural machine translation systems capture different syntactic phenomena. To achieve this goal, we have focused on phrase reordering in phrase-based machine translation and the attention model plus the encoder hidden representations in neural machine translation.

In this thesis, we have addressed the following three aspects. First, although there have been many feature engineering attempts to propose stronger reordering models in phrase-based machine translation (Tillmann, 2004, Koehn et al., 2005, Galley and Manning, 2008, Cherry, 2013, Nagata et al., 2006, Durrani et al., 2014), none have looked into the phrase-internal similarities as proposed in this thesis. Second, the attention model in neural machine translation is often considered to behave like the traditional alignment model (Liu et al., 2016, Cohn et al., 2016, Chen et al., 2016, Alkhouli et al., 2016) and is trained to follow alignment models from phrase-based machine translation without fully understanding the similarities and differences between the two models. Third, there are no intrinsic studies of what syntactic and semantic information is captured by the hidden state representations in neural machine translation and how much this differs between architectures.

In Chapter 3, we have investigated the influence of the words in a phrase on the phrase reordering behavior in a phrase-based machine translation model. We have investigated different phrase generalization strategies to see which strategy sufficiently models the pattern by which the influence of the internal words on the reordering prediction of a phrase changes.

In Chapter 4, we have studied to what extent words in the attention model in neural machine translation exhibit different attention distributions in different syntactic contexts. Additionally, we have compared an attention model to a traditional alignment model to show their similarities and differences. We have also shown under which conditions to train the attention model with explicit signals from a traditional alignment model.

In Chapter 5, we have expanded our analysis to the hidden states of the encoder of neural machine translation models. We have investigated the information captured by the hidden states and have compared two prominent neural machine translation architectures based on the information they capture. Our investigation shows to what extent different syntactic and lexical semantic information is captured by the encoder hidden states. Additionally, our comparison shows which architecture is better in capturing structural syntactic information and which one is better in capturing lexical semantics.

6.1 Main Findings

In this section, we revisit the research questions and summarize our findings regarding each research question. We start with the research questions on phrase reordering behavior in a phrase-based machine translation model:

RQ1 *To what extent do the orderings of phrases in a phrase-based machine translation model depend on the full lexical forms of the phrases?*

To answer this question, we have introduced in Chapter 3 a novel method that builds on the established idea of backing off to shorter histories, commonly used in language model smoothing (Chen and Goodman, 1999). We have experimented with different backing-off strategies and shown that they can be successfully applied to smooth lexicalized and hierarchical reordering models in statistical machine translation. To investigate the influence of the internal words on reordering distributions, we experimented with different generalized forms of the phrase-pairs. We have kept the exposed heads in their original forms and have generalized the remaining words by using word classes or simply removing them. The experimental findings show that sub-phrase-pairs consisting of just the exposed heads of a phrase-pair tend to be the most important ones and most other words inside a phrase-pair have negligible influence on reordering behavior.

Our results in Section 3.4 show that the full lexical form of phrase-pairs does not play a significant role in estimating reliable reordering distributions. However, the

improvements achieved by keeping important words in their surface form show that the lexical forms of the exposed heads are important for estimating the reordering distributions more accurately.

RQ1.1 *How does the importance of the words of a phrase change in defining the ordering of the phrase?*

Contradicting earlier approaches, which assume the last and the first words of a phrase-pair to be the most influential words for defining reordering behavior (Nagata et al., 2006, Cherry, 2013), our experiments in Section 3.4.2 show that the exposed heads of phrase-pairs are stronger predictors of phrase reordering behavior. Here, we have experimented with both backing off towards the border words by assuming those as the important words, following Nagata et al. (2006) and Cherry (2013), and generalizing towards exposed heads, assuming those as the most important words, following Chelba and Jelinek (2000) and Garmash and Monz (2015). We have shown that generalized representations of phrase-pairs based on exposed heads can help decrease sparsity and result in more reliable reordering distributions.

RQ1.2 *How helpful is it to estimate the reordering distribution of phrases by backing off to shorter forms of the phrases and removing less influential words?*

The results in Section 3.4.2 show that backing off is helpful. However, our back-off models are not our best models even though they perform better than the baselines. We have found that identifying the most influential internal words on the reordering is more important than how to back off towards them to estimate better general distribution.

RQ1.3 *How accurately can the orderings of phrases be predicted by using class-based generalizations of the words in phrases when keeping the most influential words in their original form?*

Our generalized representations of phrase-pairs based on exposed heads have achieved the best results in our experiments. As mentioned above, this is mostly due to the better choice of the most influential internal words rather than the approach we took for the generalization of less important words. Interestingly, simply removing the less influential words yields the best performance improvements in our experiments, see Section 3.4.

Note that we cannot simply conclude that the exposed heads or the border words are always the important words that define the reordering behavior of phrases. However, these words arguably provide good signals on the reordering behavior of the phrase-pairs and exposed heads are more influential compared to border words on average.

Considering the analysis of the length of infrequent phrase-pairs used during translation in Section 3.4.3, we also conclude that a smoothing model that would be able to further improve the reordering distributions of single-word phrase-pairs is crucial for achieving bigger improvements.

In Chapter 4, we have continued our interpretability analysis of how machine translation systems capture different syntactic phenomena by studying attention in neural machine translation. We have compared traditional alignments, which are an essential component in phrase-based machine translation, with attention in neural machine translation.

For this part of our research, we asked the following questions:

RQ2 *What are the important words on the source side that attention-based neural models attend to while translating different syntactic phenomena?*

Attention models in neural machine translation are often considered similar to the traditional alignment model of phrase-based systems (Alkhouli et al., 2016, Cohn et al., 2016, Liu et al., 2016, Chen et al., 2016). In that sense, an attention model should pay attention to the translational equivalent of the generated target word as defined by traditional alignments. To verify this, our first step has been to compare the attention model with traditional alignments and ask the following question:

RQ2.1 *To what extent does an attention model agree with the traditional alignment model in capturing translation equivalent words?*

We have shown in Section 4.5 that attention agrees with traditional alignments to a certain extent. However, this differs substantially by attention mechanism and the type of the word being generated. When generating nouns, attention mostly focuses on the translational equivalent of the target word on the source side and behaves indeed very similar to traditional alignments. However, its behavior diverges from that of the traditional alignment model when it comes to generating words that need more contextual information to be generated correctly. This is especially the case for verbs that need to be in agreement with their subject or are modified by an auxiliary verb.

Next, we have investigated whether the divergence from traditional alignment is due to errors in the attention model or is evidence for attention capturing information beyond traditional alignments. Here, we have asked the following question:

RQ2.2 *Are the differences between attention and traditional alignments due to errors in attention models or do attention models capture additional information compared to traditional alignment models?*

In Section 4.5.3, we have examined the correlation of the translation quality with attention loss and attention concentration. The low correlations between these quantities, especially in the case of generating verbs, show that the difference is not a result of an error in attention, but a side effect of capturing additional information compared to traditional alignments.

The higher concentration of attention for nouns and adjectives shows that for these syntactic phenomena, attention behaves like traditional alignment and captures mostly translational equivalence. However, by distributing its weight to other relevant context

words, in particular while translating verbs, attention integrates other context words that influence the generation of the target word.

RQ2.3 *How does the distribution of the attention model change for different syntactic phenomena?*

We have shown in Section 4.5 that the attention behavior changes based on the POS tag of the target word. The concentrated pattern of attention and the relatively high correlations for nouns show that training attention with explicit alignment labels is useful for generating nouns. However, this is not the case for verbs, as a large portion of attention is devoted to words other than the alignment points. In these cases, training attention with alignments will force the attention model to ignore this useful information.

RQ2.4 *What types of words are attended to by the attention model while translating different syntactic phenomena?*

We have shown in Section 4.5.4 that the attention model learns to assign some attention weights to dependency relations of a word while translating it. This is especially true in the case of words that require context information in order to be translated properly. For example, verbs, adverbs, conjunctions and particles only receive half of the attention weight on average, while the rest of the attention is paid to their dependent words.

In Chapter 5, we have focused on the encoder hidden states in different neural machine translation architectures by looking at their nearest neighbors. This is a natural follow up of our study of attention on the encoder side in Chapter 4. We ask the following research questions:

RQ3 *What information is captured by the hidden states of the encoder of a neural machine translation model?*

By detecting the similarities and the differences of the nearest neighbors of the hidden states, we have shown to what extent the hidden states capture syntactic and lexical semantic information. We have shown that part of the capacity of the hidden states is devoted to transferring information already carried by the corresponding word embeddings. We have also shown that the hidden states capture some WordNet-aligned lexical semantics beyond what is captured by the word embeddings. Furthermore, we have demonstrated that the hidden states use some of their capacity to encode syntactic information and some positional biases, depending on the architecture of the network.

RQ3.1 *How does the captured information in the hidden states differ from the information captured by the word embeddings?*

Although some of the information carried by the word embeddings is transferred to the corresponding hidden states, we have shown in Section 5.5 that hidden state representations capture quite different information from what is captured by the corresponding

embeddings. We have also shown that hidden states capture more of the WordNet relations of the corresponding word than they capture from the nearest neighbors of the embeddings.

RQ3.2 *To what extent do the hidden states of the encoder capture different types of information, such as syntactic or semantic information?*

We studied the list of the nearest neighbors of the hidden states from various perspectives. We have examined to what extent the capacity of the hidden states is devoted to capturing different types of information, including syntactic and lexical semantic information. By computing the shared portion of the nearest neighbors between the hidden states and the corresponding embeddings, we have given an estimate of the capacity of the hidden states used to keep embedding information. In addition, our WordNet coverage score is an estimate of the extent to which the hidden states capture context-dependent lexical semantics. It is important to note that our definition of embedding coverage and WordNet coverage are not mutually exclusive since there may be neighbors that are counted in computing both scores. However, WordNet coverage always trumps embedding coverage, which shows that hidden states encode lexical information beyond what is captured by word embeddings.

To examine the syntactic information encoded by the hidden states, we have computed the similarity between the local parse trees of the words of interest and the parse trees of their neighbors (see Section 5.5.4). The computed similarity scores give an estimate of the extent to which each of the studied architectures is capable of capturing syntax.

Our results show that attention model and hidden state representations play a complementary role in capturing contextual information. Hidden state representations are more capable of capturing local structural information, while more global contextual information is captured by the attention model.

RQ3.3 *How different is the captured information in the encoder hidden states for different NMT architectures?*

Here, we have compared recurrent and transformer architectures in terms of capturing syntax and lexical semantics in Section 5.5. To perform a detailed comparison we have used metrics in our intrinsic study that estimate the amount of syntactic and lexical semantic information captured by the hidden states. In Section 5.5, we have reported embedding coverage, WordNet coverage, syntactic similarity, positional bias and concentration of nearest neighbors for transformer and recurrent architectures and have compared the models using these scores. We have also studied the changing trends of some of these measures throughout sentences to compare the behavior of the hidden states of the two architectures. We have shown that there are differences in the capacity that the hidden states from each architecture devote to capturing a specific type of information. We show that the transformer model is superior in terms of capturing lexical semantics, while the recurrent model better captures syntactic similarity.

Additionally, we provide a detailed analysis of the behavior of the hidden states, both direction-wise and for the concatenations. We have investigated various types of linguistic information captured by the different directions of hidden states in a bidirectional recurrent model. We have shown that the reverse recurrent layer captures more lexical information, whereas the forward recurrent layer captures more long-distance, contextual information. One can also see that the forward recurrent layer has less positional bias compared to the backward recurrent layer.

Finally, we have provided analyses on how the behavior of the hidden states change through sentences. Our analyses indicate that positional bias is more prevalent close to the border of sentences and it dominates lexical and semantic similarities at these regions. This leads to nearest neighbors that are irrelevant in terms of lexical similarity and only share positional similarities with the word of interest.

6.2 Future Work

In this thesis, we have contributed to the interpretability of both phrase-based and neural machine translation systems. We shed light on how these system capture some syntactic and semantic phenomena and how various architectures differ in capturing those phenomena. However, current neural machine translation models are so complex that more analytical studies are required to fully interpret their behavior. Below are future directions motivated by our work.

Our work motivates more complex attention models. We have shown that an attention model learns to focus its weight for translating nouns which mostly have a single translational equivalent in the source language. We have also shown that it learns to pay attention to dependent context words when translating verbs. However, improvements achieved by explicitly training an attention model using signals from traditional alignments in the e-commerce domain show that a simple attention model by itself is not sufficient (Chen et al., 2016). This encourages more complex attention models, like those of Feng et al. (2016) and Lin et al. (2018), that can differentiate more easily between different syntactic phenomena which require different attention distribution without explicit signals from traditional alignment models.

A natural follow up to this work could be an analysis of how more complex attention models or more layers of attention could capture part of the information that is captured by the hidden state representations. This may be helpful to understand how to devote hidden state capacity to similarities in the context that are not already captured by the models. For example, our comparison of the recurrent model and transformer model in Chapter 5 shows that transformers are capable of capturing syntactic structure information to a large extent, similar to recurrent LSTM models. This has been made possible by the multi-headed attention models, including self-attention, in transformers.

Finally, it is still not fully clear what information is captured in the hidden state representation of neural machine translation systems in general. Shi et al. (2016)

and Belinkov et al. (2017a,b) examined the information captured by the hidden state representations by using extrinsic classification tasks. In Chapter 5, we have taken an intrinsic approach to show what is captured and how. The fact that hidden state representations in different architectures capture different types of information needs to be analyzed more deeply.

Bibliography

T. Alkhouli, G. Bretschner, J.-T. Peter, M. Hethnawi, A. Guta, and H. Ney. Alignment-based neural machine translation. In *Proceedings of the First Conference on Machine Translation*, pages 54–65, Berlin, Germany, August 2016. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W16-2206. (Cited on pages 5, 9, 58, 60, 61, 70, 71, 77, 109, and 112.)

L. J. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. URL http://arxiv.org/abs/1607.06450. (Cited on pages 30 and 86.)

S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0130140. (Cited on page 83.)

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*, San Diego, California, USA, May 2015. URL https://arxiv.org/abs/1409.0473. (Cited on pages 1, 2, 4, 7, 23, 26, 27, 28, 57, 58, 61, 65, 76, and 84.)

Y. Belinkov. *On internal language representations in deep learning: an analysis of machine translation and speech recognition*. PhD thesis, Massachusetts Institute of Technology, Cambridge, USA, 2018. URL http://hdl.handle.net/1721.1/118079. (Cited on page 1.)

Y. Belinkov, N. Durrani, F. Dalvi, H. Sajjad, and J. Glass. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada, July 2017a. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P17-1080. (Cited on pages 2, 7, 62, 79, 80, 81, 83, and 116.)

Y. Belinkov, L. Màrquez, H. Sajjad, N. Durrani, F. Dalvi, and J. Glass. Evaluating layers of representation in neural machine translation on part-of-speech and semantic tagging tasks. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10. Asian Federation of Natural Language Processing, 2017b. URL http://aclweb.org/anthology/I17-1001. (Cited on page 116.)

Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3(Feb):1137–1155, 2003. URL https://dl.acm.org/doi/10.5555/944919.944966. (Cited on pages 23, 24, and 25.)

L. Bentivogli, A. Bisazza, M. Cettolo, and M. a. Federico. Neural versus phrase-based machine translation quality: a case study. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 257–267, Austin, Texas, Nov. 2016. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D16-1025. (Cited on pages 2, 23, 26, 57, and 109.)

A. Birch, M. Osborne, and P. Blunsom. Metrics for MT evaluation: evaluating reordering. *Machine Translation*, 24(1):15–26, 2010. URL https://dl.acm.org/doi/10.1007/s10590-009-9066-5. (Cited on page 50.)

A. Bisazza and M. Federico. A survey of word reordering in statistical machine translation: Computational models and language phenomena. *Computational Linguistics*, 42(2):163–205, June 2016. URL https://www.aclweb.org/anthology/J16-2001. (Cited on pages 26, 35, and 109.)

A. Bisazza and C. Monz. Class-based language modeling for translating into morphologically rich languages. In *Proceedings of the 25th Annual Conference on Computational Linguistics (COLING)*, pages 1918–1927, 2014. URL https://www.aclweb.org/anthology/C14-1181. (Cited on page 40.)

A. Bisazza and C. Tump. The lazy encoder: A fine-grained analysis of the role of morphology in neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2871–2876, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D18-1313. (Cited on page 79.)

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, Sept. 2015. Association for Computational Linguistics. URL https://www.

aclweb.org/anthology/W15-3001. (Cited on page 85.)

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*, pages 169–214, Copenhagen, Denmark, 2017. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W17-4717. (Cited on page 86.)

D. Britz, A. Goldie, M.-T. Luong, and Q. Le. Massive exploration of neural machine translation architectures. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1442–1451. Association for Computational Linguistics, 2017. URL http://aclweb.org/anthology/D17-1151. (Cited on page 84.)

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. URL https://www.aclweb.org/anthology/J93-2003. (Cited on page 14.)

V. Chahuneau, E. Schlinger, N. A. Smith, and C. Dyer. Translating into morphologically rich languages with synthetic phrases. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1677–1687, 2013. URL https://www.aclweb.org/anthology/D13-1174. (Cited on pages 23 and 40.)

C. Chelba and F. Jelinek. Structured language modeling. *Computer Speech & Language*, 14(4): 283–332, 2000. URL https://www.sciencedirect.com/science/article/abs/pii/S0885230800901475. (Cited on pages 4, 40, 45, 54, and 111.)

B. Chen, G. Foster, and R. Kuhn. Adaptation of reordering models for statistical machine translation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 938–946, 2013. URL https://www.aclweb.org/anthology/N13-1114. (Cited on pages 8, 36, and 50.)

D. Chen and C. D. Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 740–750, 2014. URL https://www.aclweb.org/anthology/D14-1082. (Cited on page 50.)

M. X. Chen, O. Firat, A. Bapna, M. Johnson, W. Macherey, G. Foster, L. Jones, M. Schuster, N. Shazeer, N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, Z. Chen, Y. Wu, and M. Hughes. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–86, Melbourne, Australia, July 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P18-1008. (Cited on page 57.)

S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393, 1999. URL https://www.sciencedirect.com/science/article/abs/pii/S0885230899901286. (Cited on pages 8, 19, 41, and 110.)

W. Chen, E. Matusov, S. Khadivi, and J.-T. Peter. Guided alignment training for topic-aware neural machine translation. *AMTA 2016, Vol.*, page 121, 2016. URL https://arxiv.org/abs/1607.01628. (Cited on pages 5, 9, 58, 60, 61, 65, 70, 71, 73, 77, 109, 112, and 115.)

C. Cherry. Improved reordering for phrase-based translation using sparse features. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 22–31, 2013. URL https://www.aclweb.org/anthology/N13-1003. (Cited on pages 4, 8, 23, 35, 36, 37, 38, 39, 40, 44, 49, 54, 109, and 111.)

C. Cherry and G. Foster. Batch tuning strategies for statistical machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 427–436, Montréal, Canada, June 2012. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/N12-1047. (Cited on page 1.)

C. Cherry, R. C. Moore, and C. Quirk. On hierarchical re-ordering and permutation parsing for phrase-based decoding. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 200–209, 2012. URL https://www.aclweb.org/anthology/W12-3125. (Cited on pages 22, 23, and 35.)

D. Chiang. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228, 2007. URL

https://www.aclweb.org/anthology/J07-2003. (Cited on pages 1 and 46.)

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. 2014a. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W14-4012. (Cited on pages 23, 27, and 28.)

K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, Oct. 2014b. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D14-1179. (Cited on pages 23 and 26.)

T. Cohn, C. D. V. Hoang, E. Vymolova, K. Yao, C. Dyer, and G. Haffari. Incorporating structural alignment biases into an attentional neural translation model. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 876–885, San Diego, California, June 2016. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/N16-1102. (Cited on pages 5, 58, 65, 109, and 112.)

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*, 39:1–38, 1977. URL http://web.mit.edu/6.435/www/Dempster77.pdf. (Cited on page 19.)

J. Devlin, R. Zbib, Z. Huang, T. Lamar, R. Schwartz, and J. Makhoul. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1370–1380, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P14-1129. (Cited on page 23.)

Y. Ding, Y. Liu, H. Luan, and M. Sun. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1159, Vancouver, Canada, July 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P17-1106. (Cited on page 83.)

N. Durrani, H. Schmid, and A. Fraser. A joint sequence translation model with integrated reordering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1045–1054, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P11-1105. (Cited on pages 26 and 35.)

N. Durrani, P. Koehn, H. Schmid, and A. Fraser. Investigating the usefulness of generalized word representations in SMT. In *Proceedings of the 25th Annual Conference on Computational Linguistics (COLING)*, pages 421–432, 2014. URL https://www.aclweb.org/anthology/C14-1041. (Cited on pages 37, 40, and 109.)

C. Dyer, V. Chahuneau, and N. A. Smith. A simple, fast, and effective reparameterization of IBM Model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia, June 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/N13-1073. (Cited on pages 15 and 61.)

S. Edunov, M. Ott, M. Auli, and D. Grangier. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D18-1045. (Cited on page 57.)

M. Fadaee, A. Bisazza, and C. Monz. Learning topic-sensitive word representations. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 441–447, Vancouver, Canada, July 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P17-2070. (Cited on page 26.)

M. Faruqui, Y. Tsvetkov, P. Rastogi, and C. Dyer. Problems with evaluation of word embeddings using word similarity tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations*

for NLP, pages 30–35, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W16-2506. (Cited on pages 26 and 80.)

M. Federico. Bayesian estimation methods for n-gram language model adaptation. *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP '96*, 1:240–243 vol.1, 1996. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.16.6589&rep=rep1&type=pdf. (Cited on page 19.)

C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, 1998. URL https://www.amazon.com/WordNet-Electronic-Database-Language-Communication/dp/026206197X. (Cited on pages 7 and 82.)

S. Feng, S. Liu, N. Yang, M. Li, M. Zhou, and K. Q. Zhu. Improving attention modeling with implicit distortion and fertility for machine translation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3082–3092, Osaka, Japan, Dec. 2016. The COLING 2016 Organizing Committee. URL https://www.aclweb.org/anthology/C16-1290. (Cited on pages 28 and 115.)

J. R. Firth. A synopsis of linguistic theory 1930-55. In *Studies in Linguistic Analysis (special volume of the Philological Society)*, volume 1952-59, pages 1–32, Oxford, 1957. The Philological Society. URL https://ci.nii.ac.jp/naid/10020680394/. (Cited on page 25.)

A. Fraser and D. Marcu. Squibs and discussions: Measuring word alignment quality for statistical machine translation. *Computational Linguistics*, 33(3):293–303, 2007. URL https://www.aclweb.org/anthology/J07-3002. (Cited on page 14.)

W. A. Gale and K. W. Church. What is wrong with adding one? In N. Oostdijk and P. de Haan, editors, *Corpus-based Research into Language*, pages 189–198. Rodopi, Amsterdam, 1994. URL https://pdfs.semanticscholar.org/9e09/439f009dd0d4618949e4e01964df2e1f1d2d.pdf. (Cited on page 19.)

M. Galley and C. D. Manning. A simple and effective hierarchical phrase reordering model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 848–856, 2008. URL https://www.aclweb.org/anthology/D08-1089. (Cited on pages 1, 2, 8, 10, 20, 21, 26, 35, 39, 48, and 109.)

E. Garmash and C. Monz. Bilingual structured language models for statistical machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2398–2408, 2015. URL https://www.aclweb.org/anthology/D15-1287. (Cited on pages 40, 45, 54, and 111.)

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*, ICML'17, page 1243–1252. JMLR.org, 2017. URL http://proceedings.mlr.press/v70/gehring17a.html. (Cited on page 30.)

H. Ghader and C. Monz. Which words matter in defining phrase reorderings in statistical machine translation? *Proceedings of the Twelfth Conference of the Association for Machine Translation in the Americas (AMTA 2016), Volume 1: MT Researchers' Track*, pages 149–162, 2016. URL https://staff.science.uva.nl/c.monz/html/publications/amta2016.pdf. (Cited on page 11.)

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, Nov. 2017. Asian Federation of Natural Language Processing. URL https://www.aclweb.org/anthology/I17-1004. (Cited on page 11.)

H. Ghader and C. Monz. An intrinsic nearest neighbor analysis of neural machine translation architectures. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pages 107–117, Dublin, Ireland, Aug. 2019. European Association for Machine Translation. URL https://www.aclweb.org/anthology/W19-6611. (Cited on page 12.)

M. Giulianelli, J. Harding, F. Mohnert, D. Hupkes, and W. Zuidema. Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 240–248, Brussels, Belgium, Nov. 2018. Association for Computational Linguistics. URL https:

//www.aclweb.org/anthology/W18-5426. (Cited on page 83.)

J. V. Graça, K. Ganchev, and B. Taskar. Learning tractable word alignment models with complex constraints. *Computational Linguistics*, 36(3):481–504, 2010. URL https://www.aclweb.org/anthology/J10-3007. (Cited on page 71.)

S. Green, M. Galley, and C. D. Manning. Improved models of distortion cost for statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 867–875, Los Angeles, California, June 2010. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/N10-1129. (Cited on pages 26 and 35.)

B. Hamp and H. Feldweg. Germanet - A lexical-semantic net for German. In *Proceedings of the ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, 1997. URL http://aclweb.org/anthology/W97-0802. (Cited on page 82.)

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. URL http://openaccess.thecvf.com/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf. (Cited on page 31.)

V. Henrich and E. Hinrichs. GernEdiT - the GermaNet editing tool. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may 2010. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2010/pdf/264_Paper.pdf. (Cited on page 82.)

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, Nov. 1997. ISSN 0899-7667. URL http://dx.doi.org/10.1162/neco.1997.9.8.1735. (Cited on pages 24, 26, and 84.)

M. Hopkins and J. May. Tuning as ranking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362, 2011. URL https://www.aclweb.org/anthology/D11-1125. (Cited on pages 18 and 49.)

D. Hupkes, S. Veldhoen, and W. Zuidema. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *Journal of Artificial Intelligence Research*, 61(1):907–926, Jan. 2018. ISSN 1076-9757. URL http://dl.acm.org/citation.cfm?id=3241691.3241713. (Cited on page 83.)

I. Iacobacci, M. T. Pilehvar, and R. Navigli. SensEmbed: Learning sense embeddings for word and relational similarity. 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)*, pages 95–105, Beijing, China, July 2015. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P15-1010. (Cited on page 80.)

H. Isozaki, T. Hirao, K. Duh, K. Sudoh, and H. Tsukada. Automatic evaluation of translation quality for distant language pairs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 944–952, 2010. URL https://www.aclweb.org/anthology/D10-1092. (Cited on pages 32, 48, and 50.)

S. Jean, K. Cho, R. Memisevic, and Y. Bengio. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10, Beijing, China, July 2015. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P15-1001. (Cited on pages 1, 23, 26, and 57.)

F. Jelinek and R. L. Mercer. Interpolated estimation of Markov source parameters from sparse data. In E. S. Gelsema and L. N. Kanal, editors, *Proceedings, Workshop on Pattern Recognition in Practice*, pages 381–397. North Holland, Amsterdam, 1980. URL https://www.researchgate.net/publication/321428427_Interpolated_estimation_of_Markov_source_parameters_from_sparse_data. (Cited on page 19.)

Á. Kádár, G. Chrupała, and A. Alishahi. Representation of linguistic form and function in recurrent neural networks. *Computational Linguistics*, 43(4):761–780, Dec. 2017. URL https://www.aclweb.org/anthology/J17-4003. (Cited on page 84.)

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. URL https://www.aclweb.org/anthology/D13-1176. (Cited on pages 23 and 28.)

S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 35(3):400–401, 1987. URL https://ieeexplore.ieee.org/document/1165125?denied=. (Cited on page 19.)

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. URL https://arxiv.org/pdf/1412.6980.pdf. (Cited on page 86.)

R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. In *ICASSP*, pages 181–184. IEEE Computer Society, 1995. URL https://ieeexplore.ieee.org/abstract/document/479394. (Cited on page 19.)

P. Koehn. *Statistical Machine Translation*. Cambridge University Press, USA, 1st edition, 2010. URL https://www.cambridge.org/core/books/statistical-machine-translation/94EADF9F680558E13BE759997553CDE5. (Cited on page 15.)

P. Koehn. *Neural Machine Translation*. Cambridge University Press, 2020. URL https://www.cambridge.org/core/books/neural-machine-translation/7AAA628F88ADD64124EA008C425C0197. (Cited on page 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. URL https://www.aclweb.org/anthology/W17-3204. (Cited on pages 62 and 79.)

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*, pages 48–54, 2003. URL https://www.aclweb.org/anthology/N03-1017. (Cited on pages 1, 13, 14, 16, 17, 19, 43, and 49.)

P. Koehn, A. Axelrod, A. B. Mayne, C. Callison-Burch, M. Osborne, and D. Talbot. Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation. In *Proceedings of International Workshop on Spoken Language Translation*, 2005. URL http://www.cs.jhu.edu/~ccb/publications/iwslt05-report.pdf. (Cited on pages 2, 8, 10, 13, 14, 16, 20, 35, 42, 43, and 109.)

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the Association for Computational Linguistics*, pages 177–180, 2007. URL https://www.aclweb.org/anthology/P07-2045. (Cited on pages 13 and 48.)

S. M. Lakew, M. Cettolo, and M. Federico. A comparison of transformer and recurrent neural networks on multilingual neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 641–652. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/C18-1054. (Cited on page 81.)

G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California, June 2016. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/N16-1030. (Cited on page 26.)

Q. V. Le, N. Jaitly, and G. E. Hinton. A simple way to initialize recurrent networks of rectified linear units. *CoRR*, abs/1504.00941, 2015. URL http://arxiv.org/abs/1504.00941. (Cited on page 24.)

J. Li, Z. Tu, G. Zhou, and J. van Genabith. Head-driven hierarchical phrase-based translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 33–37, 2012. URL https://www.aclweb.org/anthology/P12-2007. (Cited on pages 4, 38, 40, 45, and 46.)

P. Li, Y. Liu, M. Sun, T. Izuha, and D. Zhang. A neural reordering model for phrase-based translation. In *Proceedings of the 25th Annual Conference on Computational Linguistics (COLING)*, pages 1897–1907, 2014. URL https://www.aclweb.org/anthology/C14-1179. (Cited on pages 23 and 40.)

J. Lin, X. Sun, X. Ren, M. Li, and Q. Su. Learning when to concentrate or divert attention: Self-adaptive attention temperature for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2985–2990, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D18-1331. (Cited on page 115.)

L. Liu, M. Utiyama, A. Finch, and E. Sumita. Neural machine translation with supervised attention. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3093–3102, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee. URL https://www.aclweb.org/anthology/C16-1291. (Cited on pages 5, 9, 58, 60, 61, 65, 70, 71, 77, 109, and 112.)

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*, pages 1412–1421, Lisbon, Portugal, Sept. 2015a. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D15-1166. (Cited on pages 2, 4, 28, 57, 58, 61, 62, 65, 68, 76, and 84.)

T. Luong, I. Sutskever, Q. Le, O. Vinyals, and W. Zaremba. Addressing the rare word problem in neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 11–19, Beijing, China, July 2015b. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P15-1002. (Cited on pages 1, 7, 23, 26, and 57.)

X. Ma and E. Hovy. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P16-1101. (Cited on page 26.)

H. Mi, Z. Wang, and A. Ittycheriah. Supervised attentions for neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2283–2288, Austin, Texas, Nov. 2016. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D16-1249. (Cited on pages 61 and 65.)

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. URL https://www.fit.vutbr.cz/research/groups/speech/publi/2010/mikolov_interspeech2010_IS100722.pdf. (Cited on pages 23 and 24.)

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013a. URL http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf. (Cited on page 86.)

T. Mikolov, W.-t. Yih, and G. Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June 2013b. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/N13-1090. (Cited on page 25.)

G. A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, Nov. 1995. ISSN 0001-0782. URL http://doi.acm.org/10.1145/219717.219748. (Cited on pages 7 and 82.)

G. A. Miller and W. G. Charles. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28, 1991. URL http://eric.ed.gov/ERICWebPortal/recordDetail?accno=EJ431389. (Cited on page 25.)

P. Moradi, N. Kambhatla, and A. Sarkar. Interrogating the explanatory power of attention in neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 221–230, Hong Kong, Nov. 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D19-5624. (Cited on page 79.)

M. Nagata, K. Saito, K. Yamamoto, and K. Ohashi. A clustered global phrase reordering model for statistical

machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 713–720, 2006. URL https://www.aclweb.org/anthology/P06-1090. (Cited on pages 4, 37, 38, 39, 40, 49, 53, 54, 109, and 111.)

F. J. Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan, July 2003. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P03-1021. (Cited on pages 1 and 18.)

F. J. Och and H. Ney. Improved statistical alignment models. In *38th Annual Meeting of the Association for Computational Linguistics, Hong Kong, China, October 1-8, 2000.*, 2000. URL https://www.aclweb.org/anthology/P00-1056. (Cited on pages 14 and 15.)

F. J. Och and H. Ney. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 295–302, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P02-1038. (Cited on pages 1 and 17.)

F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003. URL https://www.aclweb.org/anthology/J03-1002. (Cited on pages 14, 15, 43, 48, 61, 68, and 69.)

F. J. Och and H. Ney. The alignment template approach to statistical machine translation. *Comput. Linguist.*, 30(4):417–449, Dec. 2004. ISSN 0891-2017. URL https://doi.org/10.1162/0891201042544884. (Cited on page 18.)

F. J. Och, C. Tillmann, and H. Ney. Improved alignment models for statistical machine translation. In *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999. URL https://www.aclweb.org/anthology/W99-0604. (Cited on pages 1 and 16.)

M. Ott, S. Edunov, D. Grangier, and M. Auli. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium, Oct. 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W18-6301. (Cited on page 57.)

N. Papernot and P. D. McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *CoRR*, abs/1803.04765, 2018. URL http://arxiv.org/abs/1803.04765. (Cited on page 83.)

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, 2002. URL https://www.aclweb.org/anthology/P02-1040.pdf. (Cited on pages 32, 48, 68, and 86.)

A. Parikh, O. Täckström, D. Das, and J. Uszkoreit. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas, Nov. 2016. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D16-1244. (Cited on page 30.)

M. Pelevina, N. Arefiev, C. Biemann, and A. Panchenko. Making sense of word embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 174–183. Association for Computational Linguistics, 2016. URL http://www.aclweb.org/anthology/W16-1620. (Cited on pages 86 and 88.)

J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL http://www.aclweb.org/anthology/D14-1162. (Cited on page 25.)

S. Petrov, D. Das, and R. Mcdonald. A universal part-of-speech tagset. In *Proceedings of the Language Resources and Evaluation Conference*, 2012. URL http://www.lrec-conf.org/proceedings/lrec2012/pdf/274_Paper.pdf. (Cited on page 70.)

Y. Qi, D. Sachan, M. Felix, S. Padmanabhan, and G. Neubig. When and why are pre-trained word embeddings useful 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 2*

(Short Papers), pages 529–535, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/N18-2084. (Cited on page 26.)

S. Riezler and J. T. Maxwell. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the Association for Computational Linguistics Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 57–64, 2005. URL https://www.aclweb.org/anthology/W05-0908. (Cited on page 49.)

H. Salehinejad, J. Baarbe, S. Sankar, J. Barfett, E. Colak, and S. Valaee. Recent advances in recurrent neural networks. *CoRR*, abs/1801.01078, 2018. URL http://arxiv.org/abs/1801.01078. (Cited on page 24.)

S. Sekine and M. J. Collins. Evalb – Bracket Scoring Program, 1997. URL http://cs.nyu.edu/cs/projects/proteus/evalb. (Cited on pages 89 and 99.)

R. Sennrich and B. Haddow. Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 83–91. Association for Computational Linguistics, 2016. URL http://aclweb.org/anthology/W16-2209. (Cited on page 89.)

R. Sennrich, M. Volk, and G. Schneider. Exploiting synergies between open resources for German Dependency parsing, pos-tagging, and morphological analysis. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, pages 601–609, Hissar, Bulgaria, September 2013. INCOMA Ltd. Shoumen, BULGARIA. URL https://www.aclweb.org/anthology/R13-1079. (Cited on page 73.)

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. Association for Computational Linguistics, 2016. URL https://www.aclweb.org/anthology/P16-1162. (Cited on pages 68 and 86.)

C. Shannon. The zero error capacity of a noisy channel. *IRE Transactions on Information Theory*, 2(3):8–19, 1956. URL https://ieeexplore.ieee.org/document/1056798. (Cited on page 15.)

X. Shi, I. Padhi, and K. Knight. Does string-based neural MT learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534, Austin, Texas, November 2016. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D16-1159. (Cited on pages 2, 7, 61, 79, 80, 81, 82, 83, 89, and 115.)

M. D. Smucker and J. Allan. An investigation of Dirichlet prior smoothing's performance advantage. Technical Report IR-391, The University of Massachusetts, The Center for Intelligent Information Retrieval, 2005. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.366.6292&rep=rep1&type=pdf. (Cited on page 36.)

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings the 7th Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 223–231, 2006. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.129.4369. (Cited on pages 32 and 48.)

F. Stahlberg, D. Saunders, and B. Byrne. An operation sequence model for explainable neural machine translation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 175–186, Brussels, Belgium, Nov. 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W18-5420. (Cited on page 1.)

M. Sundermeyer, R. Schlüter, and H. Ney. LSTM neural networks for language modeling. In *INTERSPEECH*, 2012. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.248.4448&rep=rep1&type=pdf. (Cited on page 24.)

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3104–3112, Cambridge, MA, USA, 2014. MIT Press. URL http://dl.acm.org/citation.cfm?id=2969033.2969173. (Cited on pages 1, 23, 28, and 57.)

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, 2018a. URL http:

//aclweb.org/anthology/D18-1458. (Cited on pages 2, 8, 81, 84, and 88.)

G. Tang, R. Sennrich, and J. Nivre. An analysis of attention mechanisms: The case of word sense disambiguation in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 26–35, Brussels, Belgium, Oct. 2018b. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W18-6304. (Cited on pages 62 and 79.)

C. Tillmann. A unigram orientation model for statistical machine translation. In *Proceedings of the 2004 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 101–104, 2004. URL https://www.aclweb.org/anthology/N04-4026. (Cited on pages 1, 2, 8, 10, 20, 35, and 109.)

K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics, 2003. URL https://www.aclweb.org/anthology/N03-1033/. (Cited on page 91.)

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. URL http://aclweb.org/anthology/D18-1503. (Cited on pages 2, 8, 81, 84, 88, and 99.)

Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P16-1008. (Cited on page 58.)

A. Vaswani, Y. Zhao, V. Fossum, and D. Chiang. Decoding with large-scale neural language models improves translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1387–1392, Seattle, Washington, USA, Oct. 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D13-1140. (Cited on page 23.)

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf. (Cited on pages 1, 7, 23, 26, 28, 29, 31, 57, 84, and 86.)

D. Vilar, M. Popovic, and H. Ney. AER: Do we need to "improve" our alignments? In *Proc. of the International Workshop on Spoken Language Translation*, pages 205–212, Kyoto, Japan, 2006. URL http://www2.nict.go.jp/astrec-att/workshop/IWSLT2006/proceedings/TP_7_vilar.pdf. (Cited on page 65.)

E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy, July 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P19-1580. (Cited on page 79.)

E. Wallace, S. Feng, and J. Boyd-Graber. Interpreting neural networks with nearest neighbors. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 136–144, Brussels, Belgium, Nov. 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W18-5416. (Cited on page 83.)

D. Wang, C. Gong, and Q. Liu. Improving neural language modeling via adversarial training. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6555–6565, Long Beach, California, USA, 09–15 Jun 2019a. PMLR. URL http://proceedings.mlr.press/v97/wang19f.html. (Cited on page 57.)

T. Wang and K. Cho. Larger-context language modelling with recurrent neural network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1319–1329. Association for Computational Linguistics, 2016. URL http://www.aclweb.org/anthology/P16-1125. (Cited on page 98.)

Y. Wang, Y. Xia, T. He, F. Tian, T. Qin, C. Zhai, and T.-Y. Liu. Multi-agent dual learning. In *International Conference on Learning Representations*, 2019b. URL https://openreview.net/forum?id=HyGhN2A5tm. (Cited on page 57.)

Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Łukasz Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016. URL http://arxiv.org/abs/1609.08144. (Cited on pages 1, 23, 26, and 57.)

J. Wuebker, S. Peitz, F. Rietig, and H. Ney. Improving statistical machine translation with word class models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1377–1381, 2013. URL https://www.aclweb.org/anthology/D13-1138. (Cited on pages 23 and 40.)

H. Xu, J. van Genabith, D. Xiong, and Q. Liu. Analyzing word translation of transformer layers, 2020. (Cited on page 79.)

D. Zhang, M. Li, C.-H. Li, and M. Zhou. Phrase reordering model integrating syntactic knowledge for SMT. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 533–540, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D07-1056. (Cited on page 109.)

J. Zhou, Y. Cao, X. Wang, P. Li, and W. Xu. Deep recurrent models with fast-forward connections for neural machine translation. *Transactions of the Association for Computational Linguistics*, 4:371–383, 2016. URL https://www.aclweb.org/anthology/Q16-1027. (Cited on page 27.)

M. Zhu, Y. Zhang, W. Chen, M. Zhang, and J. Zhu. Fast and accurate shift-reduce constituent parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 434–443. Association for Computational Linguistics, 2013. URL http://aclweb.org/anthology/P13-1043. (Cited on page 89.)

W. Y. Zou, R. Socher, D. Cer, and C. D. Manning. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1393–1398, Seattle, Washington, USA, Oct. 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D13-1141. (Cited on pages 23 and 25.)

Summary

Machine translation systems are large machine learning systems that learn to translate from one natural language (for example Chinese) to another (for example English). These systems scan the input text and generate part of the translation step by step. However, due to their complexity, it is difficult to interpret the decisions they make at each time step. For example, it is not possible to explain every word choice or word order that a machine translation system uses during the generation of the translation. Therefore, it is difficult to explain why a system makes certain mistakes during translation or how it is capable of regenerating complex syntactic or semantic phenomena in the target language.

Two popular types of machine translation are phrase-based and neural machine translation systems. Both of these types of machine translation systems are composed of multiple complex models or layers. Each of these models and layers learns different linguistic aspects of the source language. However, for some of these models and layers, it is not clear which linguistic phenomena are learned or how this information is learned. For phrase-based machine translation systems, it is often clear what information is learned by each model, and the question is rather how this information is learned, especially for its phrase reordering model. For neural machine translation systems, the situation is even more complex, since for many cases it is not exactly clear what information is learned and how it is learned.

To shed light on what linguistic phenomena are captured by machine translation systems, we analyze the behavior of important models in both phrase-based and neural machine translation systems. We consider phrase reordering models from phrase-based machine translation systems to investigate which words from inside of a phrase have the biggest impact on defining the phrase reordering behavior. Additionally, to contribute to the interpretability of neural machine translation systems we study the behavior of the attention model, which is a key component in neural machine translation systems and the closest model in functionality to phrase reordering models in phrase-based systems. The attention model together with the encoder hidden state representations form the main components to encode source side linguistic information in neural machine translation. To this end, we also analyze the information captured in the encoder hidden state representations of a neural machine translation system. We investigate the extent to which syntactic and lexical-semantic information from the source side is captured by hidden state representations of different neural machine translation architectures.

Samenvatting

Computervertaalsystemen zijn complexe lerende algoritmes die vertalingen van een natuurlijke taal (zoals Chinees) naar een andere natuurlijke taal (zoals Engels) kunnen maken. Deze systemen werken door een ingevoerde tekst stapsgewijs te doorlopen en daarbij telkens een deel van de vertaling op te bouwen. Door de complexiteit in deze algoritmes is het moeilijk om te interpreteren welke beslissingen in elke stap van dit proces door het algoritme worden gemaakt. Het is bijvoorbeeld onmogelijk om elke woordkeuze of woordvolgorde in de opbouw van de vertaling te verklaren. Het is daardoor ook moeilijk om te begrijpen waarom bepaalde fouten zich in de vertaling voordoen, of in te schatten hoe goed een vertaalsysteem complexe syntactische of semantische structuren in de vertaling kan doorvoeren.

Twee populaire vormen van computervertaalsystemen zijn phrase-gebaseerde (Engels: *phrase-based*) en neurale vertaalsystemen. Beide soorten vertaalsystemen zijn opgebouwd uit meerdere complexe modellen of lagen. Elk van deze modellen of lagen leert andere linguïstische aspecten van de taal van waaruit vertaald moet worden. Voor sommige modellen en lagen is het echter onduidelijk welke linguïstische aspecten worden geleerd en hoe deze informatie wordt geëxtraheerd. Voor phrase-gebaseerde vertaalsystemen is het weliswaar duidelijk welke informatie door elk model wordt geleerd, maar niet hoe deze informatie wordt geleerd, vooral voor herordeningsmodellen (Engels: *reordering models*). Voor neurale vertalingssystemen is de situatie nog complexer, omdat voor veel gevallen zowel onduidelijk is welke informatie wordt geleerd als hoe deze informatie wordt geleerd.

Om inzicht te verschaffen in welke linguïstische aspecten worden opgevangen in computervertaalsystemen, analyseren we het gedrag van belangrijke modellen in zowel phrase-gebaseerde als neurale vertalingssystemen. We nemen herordeningsmodellen van phrase-gebaseerde modellen om te onderzoeken welke woorden van een zin de grootste impact op het gedrag van het herordeningsmodel hebben. Ook dragen we bij aan de interpreteerbaarheid van neurale vertaalmodellen, door de aandachtsmodellen die een sleutelrol in deze systemen vormen én het meest vergelijkbaar zijn met herordeningsmodellen in phrase-gebaseerde modellen, te bestuderen. Het aandachtsmodel vervult samen met de verborgen coderingslaag (Engels: *encoder hidden state*) de sleutelrol voor het opnemen van de linguïstische informatie in de invoertaal. Om dit te begrijpen analyseren we de informatie die wordt vastgelegd in de verborgen coderingslaag van het neurale vertaalsysteem. We kijken hierbij naar de mate waarin syntactische en lexicaal-semantische informatie van de invoer-tekst wordt vastgelegd in de verborgen coderingslaag van verschillende neurale vertalings-architecturen.