KARLSRUHER INSTITUT FÜR TECHNOLOGIE

Multilingual Neural Translation

zur Erlangung des akademischen Grades eines
Doktors der Ingenieurwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte Dissertation
von
Thanh-Le HA

Hauptreferent: Prof. Dr. Alexander WAIBEL
Korreferent: Dott. Marcello FEDERICO
Tag der mündlichen Prüfung: 09.01.2019

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe, sowie dass ich die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht habe und die Satzung des KIT, ehem. Universität Karlsruhe (TH), zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet habe.

Karlsruhe, den 14.11.2018

Thanh-Le HA

Abstract

Machine translation (MT) refers to the technology that can automatically translate contents in one language into other languages. Being an important research area in the field of natural language processing, machine translation has typically been considered one of most challenging yet exciting problems. Thanks to research progress in the data-driven **statistical machine translation** (SMT), MT is recently capable of providing adequate translation services in many language directions and it has been widely deployed in various practical applications and scenarios.

Nevertheless, there exist several drawbacks in the SMT framework. The major drawbacks of SMT lie in its dependency in separate components, its simple modeling approach, and the ignorance of global context in the translation process. Those inherent drawbacks prevent the over-tuned SMT models to gain any noticeable improvements over its horizon. Furthermore, SMT is unable to formulate a multilingual approach in which more than two languages are involved. The typical workaround is to develop multiple pair-wise SMT systems and connect them in a complex bundle to perform multilingual translation. Those limitations have called out for innovative approaches to address them effectively.

On the other hand, it is noticeable how research on artificial neural networks has progressed rapidly since the beginning of the last decade, thanks to the improvement in computation, i.e faster hardware. Among other machine learning approaches, neural networks are known to be able to capture complex dependencies and learn latent representations. Naturally, it is tempting to apply neural networks in machine translation. First attempts revolve around replacing SMT sub-components by the neural counterparts. Later attempts are more revolutionary by fundamentally changing the whole core of SMT with neural networks, which is now popularly known as **neural machine translation** (NMT). NMT is an end-to-end system which directly estimate the translation model between the source and target sentences. Furthermore, it is later discovered to capture the inherent hierarchical structure of natural

language. This is the key property of NMT that enables a new training paradigm and a less complex approach for multilingual machine translation using neural models.

This thesis plays an important role in the evolutional course of machine translation by contributing to the transition of using neural components in SMT to the completely end-to-end NMT and most importantly being the first of the pioneers in building a neural multilingual translation system.

First, we proposed an advanced neural-based component: the **neural network discriminative word lexicon**, which provides a global coverage for the source sentence during the translation process. We aim to alleviate the problems of phrase-based SMT models that are caused by the way how phrase-pair likelihoods are estimated. Such models are unable to gather information from beyond the phrase boundaries. In contrast, our discriminative word lexicon facilitates both the local and global contexts of the source sentences and models the translation using deep neural architectures. Our model has improved the translation quality greatly when being applied in different translation tasks. Moreover, our proposed model has motivated the development of end-to-end NMT architectures later, where both of the source and target sentences are represented with deep neural networks.

The second and also the most significant contribution of this thesis is the idea of extending an NMT system to a **multilingual neural translation** framework without modifying its architecture. Based on the ability of deep neural networks to modeling complex relationships and structures, we utilize NMT to learn and share the cross-lingual information to benefit all translation directions. In order to achieve that purpose, we present two steps: first in incorporating language information into training corpora so that the NMT learns a common semantic space across languages and then force the NMT to translate into the desired target languages. The compelling aspect of the approach compared to other multilingual methods, however, lies in the fact that our multilingual extension is conducted in the preprocessing phase, thus, no change needs to be done inside the NMT architecture. Our proposed method, a **universal approach** for multilingual MT, enables a seamless coupling with any NMT architecture, thus makes the multilingual expansion to the NMT systems effortlessly. Our experiments and the studies from others have successfully employed our approach with numerous different NMT architectures and show the universality of the approach.

Our multilingual neural machine translation accommodates cross-lingual information in a learned common semantic space to improve altogether every translation direction. It is then effectively applied and evaluated in various scenarios. We develop a multilingual translation system that relies on both source and target data to boost up the quality of a single translation direction. Another system could be deployed as a multilingual translation system that only requires being trained once using a multilingual corpus but is able to translate between many languages simultaneously and the delivered quality is more favorable than many translation systems trained separately. Such a system able to learn from large corpora of well-resourced languages, such as English→German or English→French, has proved to enhance other translation direction of low-resourced language pairs like English→Lithuania or German→Romanian. Even more, we show that kind of approach can be applied to the extreme case of zero-resourced translation where no parallel data is available for training without the need of pivot techniques.

The research topics of this thesis are not limited to broadening application scopes of our multilingual approach but we also focus on improving its efficiency in practice. Our multilingual models have been further improved to adequately address the multilingual systems whose number of languages is large. The proposed strategies demonstrate that they are effective at achieving better performance in **multi-way translation scenarios** with greatly reduced training time. Beyond academic evaluations, we could deploy the multilingual ideas in the **lecture-themed spontaneous speech translation service** (Lecture Translator) at KIT. Interestingly, a derivative product of our systems, the **multilingual word embedding corpus** available in a dozen of languages, can serve as a useful resource for cross-lingual applications such as cross-lingual document classification, information retrieval, textual entailment or question answering. Detailed analysis shows excellent performance with regard to semantic similarity metrics when using the embeddings on standard cross-lingual classification tasks.

Zusammenfassung

Maschinelle Übersetzung bezeichnet eine Technologie die automatisch Text von einer Sprache in eine andere Sprache übersetzt. Sie ist ein wichtiges Forschungsfeld im Bereich der natürlichen Sprachverarbeitung und wird häufig als eines der schwersten, aber auch spannendsten Probleme in diesem Bereich angesehen. Dank Fortschritte in der Forschung an datengetriebener statistischer Maschinellen Übersetzung (SMT) existieren nun praxisreife Übersetzungsdienste in viele Sprachrichtungen, die schon in vielen Anwendungen und praktischen Szenarien eingesetzt werden.

Dennoch hat dieses SMT Rahmenwerk auch mehrere Nachteile. Einige Nachteile sind die notwendige Unterteilung in mehrere getrennte Komponenten, der simplistische Modellierungsansatz, und das Ignorieren von globalem Kontext im Übersetzungsprozess. Diese inherenten Nachteile lassen Forscher keine spürbaren weiteren Fortschritte in der bereits jetzt über-optimierten SMT mehr erwarten. Außerdem ist es nicht möglich, SMT-basierte Modelle zur multilingualen Übersetzung zu formulieren, also Modelle die mehr als zwei Sprachen berücksichtigen. Die typische Praxislösung ist es, mehrere bilinguale SMT Systeme zu entwickeln, und diese in Reihe hintereinander zu schalten um multilinguale Übersetzung zu ermöglichen. Diese limitierenden Faktoren müssen durch innovative neue Ansätze angegangen werden.

Auf der anderen Seite gab es seit Anfang der letzten Dekade bemerkenswerten Fortschritt im Bereich der neuronalen Netze, dank verbesserter Rechenleistung moderner Hardware. Neuronale Netze sind als parametrisierte Funktionsapproximatoren bekannt, welche komplexe Abhängigkeiten zwischen Variablen modellieren und latente Darstellungen erlernen. Es ist daher verlockend, neuronale Netze für Aufgaben der maschinellen Übersetzung einzusetzen. Die ersten Ansätze hierzu ersetzten Subkomponenten des SMT Rahmenwerks durch neuronale Gegenstücke. Spätere Ansätze sind revolutionärer und ersetzen den Kern der SMT durch neuronale Netze, heute allgemein als neuronale maschinelle Übersetzung

(NMT) bekannt. NMT ist ein Ende-zu-Ende System welches die Übersetzung zwischen Quell- und Zielsätzen direkt modelliert. Später wird sogar entdeckt, dass NMT in der Lage ist die hierarchische Struktur natürlicher Sprache abzubilden. Diese Schlüsseleigenschaft der NMT ermöglicht nun neue Trainingsparadigmen und einen weniger komplexen Ansatz zur multilingualen Übersetzung mit neuronalen Modellen.

Diese Dissertation spielt eine wichtige Rolle im evolutionären Kurs der maschinellen Übersetzung, indem sie den Übergang von durch neuronale Netze angereichterter SMT hin zu Ende-zu-Ende neuronaler maschinellen Übersetzung (NMT) bezeugt und aktiv zu diesem beiträgt, und insbesondere durch wichtige Pionierarbeit in der Entwicklung eines neuronalen multilingualen Übersetzungssystems.

Als erstes stellen wir eine erweiterte neuronale Übersetzungskomponente vor: ein neuronales diskriminatives Wortlexikon das den Quellsatz während des Übersetzungsprozesses global abdecken kann. Wir zielen damit auf ein Problem in der phrasenbasierten SMT ab, das im Zusammenhang mit der Schätzung von Likelihoods für Phrasenpaare entsteht. Solche Modelle sind nicht in der Lage, Informationen von außerhalb der jeweiligen Phrasengrenzen zu berücksichtigen. Unser diskriminatives Wortlexikon ermöglicht es dagegen durch Nutzung von tiefen neuronalen Architekturen sowohl lokalen als auch globalen Kontext der Quellsätze zu berücksichtigen. Wir zeigen dass dieses Modell die Übersetzungsqualität in verschiedenen Übersetzungsszenarien stark verbessert. Unser neues Modell hat zusammen mit anderen neuronalen Komponenten die anschließende Entwicklung von Ende-zu-Ende NMT motiviert, in welcher sowohl Quellsätze als auch Zielsätze durch tiefe neuronale Netze repräsentiert werden.

Als zweiten und wichtigsten Beitrag leistet diese Dissertation Pionierarbeit in der Erweiterung eines NMT Systems zu einem multilingualen neuronalen Systems, ohne dabei seine Architektur zu modifizieren. Wir nutzen dabei die Fähigkeit tiefer neuronaler Netze, komplexe Relationen und Strukturen zu modellieren, und wenden NMT zum Lernen und Teilen sprachübergreifender Informationen an, durch die alle Sprachrichtungen gleichermaßen profitieren. Um dieses Ziel zu erreichen, stellen wir zwei Schritte vor: zunächst wird Sprachinformation in den Trainingsdaten enkodiert, sodass das NMT Modell einen gemeinsamen semantischen Raum über alle Sprachen erlernen kann, und erzwingen dann die Übersetzung in die erwünschte Zielsprache. Die Attraktivität unserer Methode im Vergleich zu anderen Ansätzen ergibt

sich jedoch dadurch, dass unsere multilingualen Erweiterungen während der Datenvorverarbeitung geschieht, und somit keine Änderung der NMT Architektur erforderlich ist. Dies führt zu großen Vorteilen in vielerlei Hinsicht. NMT stand zuletzt im Fokus vieler Forschungsarbeiten, in denen Forschungsgruppen aktiv neue, verbesserte Architekturen entwickelten. Unser Ansatz, den wir unversellen Ansatz zur multilingualen MT nennen, ermöglichst hierbei die nahtlose Kupplung mit beliebigen vorteilhaften NMT Architekturen. Er arbeitet außerdem mit beliebigen NMT und Deep Learning Systemen und macht deren multilinguale Erweiterung ohne besonderen Aufwand möglich. Experimente von uns und auch von anderen Forschern haben unseren Ansatz erfolgreich mit unterschiedlichen NMT Architekturen und zahlreichen NMT Systemen benutzt und somit die Universalität unseres Ansatzes gezeigt.

Unser multilinguales maschinelles Übersetzungssystem besitzt sprachübergreifende Informationen, die im geteilten semantischen Raum erlernt werden und welche die Übersetzung in jeder Richtung im allgemeinen Sinne verbessern. Dies nutzen und evaluieren wir in verschiedenen Szenarien. Wir entwickeln zunächst multilinguale Übersetzungssysteme, welche Daten sowohl der Quellsprache als auch der Zielsprache nutzen um die Übersetzungsqualität einer einzelnen Übersetzungsrichtung zu verbessern. Ein anderes System konnte als multilinguales Übersetzungssystem in Produktion geschaltet werden das nur einmal auf einem multilingualen Korpus trainiert werden musste, aber in der Lage ist, zwischen vielen Sprachen gleichzeitig zu übersetzen und bessere Qualität liefert als wenn man viele Übersetzungssysteme separat trainiert hätte. Ein ähnliches System, das auf Sprachen mit großen Mengen an verfügbaren Daten, wie etwa Englisch→Deutsch oder Englisch→Französisch, trainiert wurde, konnte erfolgreich Übersetzung von Übersetzungsrichtungen mit wenig verfügbaren Daten, etwa Englisch→Litauisch oder Deutsch→Rumänisch, verbessern. Darüber hinaus zeigen wir dass dieser Ansatz auch im Extremfall, wenn keine Daten für ein bestimmtes Sprachpaar existieren, angewendet werden kann, und er somit den Umweg über eine Pivotsprache vermeiden kann.

Diese Dissertation erforscht nicht nur die neuen Anwendungsfälle des multilingualen Ansatzes, sondern betrachtet auch Effizienzgewinne die durch diesen Ansatz in der Praxis entstehen. Unsere multilingualen Modelle wurden weiter verbessert, sodass multilinguale Systeme mit einer großen Zahl an abgebildeten Sprachen praktikabel erstellt werden

können. Dank verschiedener effektiver neuer Strategien erreicht unser System bessere Performance in Mehrweg-Übersetzungsszenarien mit stark reduzierter Trainingszeit. Dadurch sind wir in der Lage, die multilingualen Ideen in unserem Vorlesungsübersetzungssystem für spontane gesproche Sprache (Lecture Translator) am KIT im Produktionseinsatz zu nutzen. Interessanterweise ist auch eine von unseren Systemen abgeleitete Resource, der in einem Dutzend Sprachen verfügbare multilinguale Word Embedding Corpus, für viele sprachübergreifende Probleme von Nutzen, so etwa für sprachübergreifende Dokumentklassifikation, Information Retrieval, Textual Entailment, oder Question Answering. Eine detaillierte Analyse zeigt dabei die hervorragende Leistung im Hinblick auf semantische Ähnlichkeitsmaße, wenn diese Embeddings für gebräuchliche, sprachübergreifende Klassifikationsaufgaben angewendet werden.

Acknowledgements

I would like to thank my advisor Prof. Dr. Alexander Waibel for giving me the opportunity to perform the research leading to this thesis and for the constructive discussions. Furthermore, I would like to thank Prof. Dr. Marcello Federico for co-advising this thesis. He showed great interest in the thesis and gave me valuable input.

I would like to thank all members of the MT team at KIT who have been helping me during all phases of my research and only by the joint work in this team, it was possible to achieve the results presented in this thesis. My thanks go to: Eunah Cho, Stefan Constantin, Teresa Herrmann, Mohammed Mediani, Jan Niehues, Elizabeth Salesky, Felix Schneider, Isabel Slawik, Matthias Sperber, Ngoc-Quan Pham and Yuqi Zhang. The special thank goes to Jan Niehues for helping me learn a lot about doing research in machine translation.

Also all the other colleagues of the Interactive System Labs have been very helpful with any problem I had and the exchange in the team led to very interesting ideas. Therefore, I would like to thank all of them: Silke Dannenmaier, Sarah Fünfer, Jonas Gehring, Michael Heck, Klaus Joas, Kevin Kilgour, Narine Kokhlikyan, Florian Kraft, Bastian Krüger, Patricia Lichtblau, Christian Mohr, Markus Müller, Van-Huy Nguyen, Quoc-Bao Nguyen, Thai-Son Nguyen, Kay Rottmann, Florian Dessloch, Margit Rödder, Virginia Roth, Christian Saam, Maria Schmidt, Mirjam Simantzik, Sebastian Stüker, Franzisca Vogel, Joshua Winebarger, Thomas Zenkel and Liang-Guo Zhang.

I would give the special thanks to Matthias Sperber, Elizabeth Salesky, Thai-Son Nguyen and Ngoc-Quan Pham, whom I shared my coffee gossips during the stressful time of writing this thesis and they also helped me a lot in proofreading.

I would like to thank my family: my parents and my sister from Vietnam, my mother-in-law and my sibling-in-law in Germany who have been always

supporting me. Last but not the least, I would like to give my thanks to my beloved: my wife, Ngoc-Linh and my daughter, Ngoc-Nhi for sharing every moments in my life during these years.

Contents

List of Figures

List of Tables

Dedicated to my family

Chapter 1

Introduction

Machine Translation (MT), the idea of having computers automatically translate between languages, emerged around 60 years ago from the initial works of (Weaver, 1955). After several decades of ups and downs, the field has grown to be recognized as one of the most exciting yet challenging problems in Natural Language Processing (NLP) and Artificial Intelligence (AI). Data-driven MT, as the name suggest, concerns with building translation agents based on data has achieved remarkable successes, leading to the current situation that MT systems are being developed both in academia and industry, and employed towards end-users as commercial products.

Statistical machine translation (SMT) was the dominant approach in research as well as application in the last decade. The abilities of automatic learning the translation rules and easy adapting domain knowledge from parallel data make SMT becomes the favorite choice over rule-based approaches when it is less expensive and faster to build a new translation system. The emergence of MT and the success of SMT in various scenarios and language pairs even led many people to seriously consider that MT may soon take over and substitute human translation. But the dominant SMT systems, while offer high-quality solutions for a limited amount of specific situations, especially in computer-aided translation community, are far from being able to provide fully-automatic, reliable translation services in a wider range of sectors due to several issues.

Firstly, translation quality of an SMT system is strongly influenced by the quantity and the quality of the training data (Bertoldi and Federico, 2009; Haddow and Koehn, 2012; Niehues, 2014). Thus, besides requiring domain adaptation techniques, it is not trivial to build MT systems that can efficiently deal with special situations with scarce data availability, i.e. low-resourced language pairs, or even in more extreme cases, where there is no direct parallel

data at all, i.e. zero-resourced situations. In these cases, we needs some ways to conduct data augmentation for the low-resourced and share the common knowledge from the language pairs that have abundant amount of data. The only way SMT systems can do in zero-resourced situations, however, is to rely on pivot techniques, which allow SMT to combine two indirect data or models in order to produce the translation.

Secondly, SMT framework itself has an inherent drawback. It basically consists of many sub-components, called features, where each of them often employs statistically linear model. Typically, those individual features are trained separately, often not take into account the information that other features have learned, and eventually combined in a log-linear framework. The parameters of this log-linear can be tuned on an independent set in order to determine the (weighted) contributions of the features in producing final translations. Poor quality from individual components easily leads to the overall system's degradation. Such combination is also a hindrance for industrial level deployment, e.g. a spontaneous speech translation system, where the errors are hard to detect and the integration is not trivial.

Thirdly, since SMT systems treat the translation units as discrete, symbolic items, it is impossible to afford a multilingual machine translation system using SMT approach. At the moment, the only workaround is to develop many SMT systems, each of them is in charge of one translation direction and trained from the parallel data of that direction, and bundle them. This approach is inconvenient, costly and maybe hard to maintain the complex system. On the other hand, there is classic interlingual machine translation which shows its potential applications to build a more economical multilingual system. Interlingual machine translation conducts the translation via an intermediate language, called interlingua. We need to build the systems to translate from and to the languages of interest and the interlingua. This interlingua theoretically covers all the linguistic phenomena and entities of the source and target languages as well as the rules to transform between it and those languages. Practically, it needs to cover most vocabulary in the domain of interest, the large part of grammar and the important transformation rules related to the source and target languages. This is the obvious disadvantage of interlingual machine translation, where it is difficult, expensive and time-consuming and even, impossible to build such an interlingua for general domains. This reason limits the application scope of interlingual machine translation systems.

Analogous to recent development of neural networks, research community working on natural language processing fields and machine translation in particular have observed rapid progress in applying neural models in their classic problems. First attempts in MT concern with replacing SMT sub-components by their neural-based counterparts. Then a revolutionary shifting in MT paradigms has comes under a new approach which fundamentally changing the core of machine translation with neural networks. Instead of relying on a log-linear combination of different features, a neural architecture is used to directly approximate the mapping function between the source and target sentence in a continuous space. The new approach, called neural machine translation (NMT), addresses the aforementioned weaknesses of SMT directly.

Despite its newly-born status, NMT has marked an evolutionary move in the field not only as a potential framework alleviating the limitations of SMT but also as a new approach which performs superior to the once-dominant SMT in many language directions. Furthermore, the ability to learn a hidden semantic representation and latent structures gives NMT the potentials to build multilingual translation systems.

This thesis focuses on the topics of neural-based translation models, especially the possibilities to construct a realistic multilingual neural translation system capable of being deployed in a wide range of domains, scenarios and languages.

1.1 Contributions of this Work

First, we attempted to answer the question: "*Is there any advantageous way to incorporate global context into the framework of SMT?*". In order to answer this question, we analyzed the inherent problems of SMT regarding to this matter and investigated the abilities of neural models. Afterwards, we proposed an advanced component: the **neural network discriminative word lexicon** which provides a preferable way to model and consolidate the global context from the source side into the translation system.

The second research problem we addressed in this thesis is "*Can we employ the interlingual approach over a wide range of domains and scenarios in order to build an economical multilingual translation system?*". The entailment of the problem is then "*Can we build a multilingual translation system with one*

single MT architecture?". Later on in this thesis we provided a solution for both of these questions . By utilizing the ability of an NMT architecture in inducing hidden representation from the inputs and translating this representation to the desired language, we are able to extend a standard NMT to a genuine **multilingual neural translation** framework. Our system following this approach is much more convenient and economical than the traditional, complex multilingual translation bundle of many single SMT systems. Furthermore, compared to other contemporaneous neural-based approaches, our proposed approach alleviates the need of complicated architecture re-designing when accommodating the standard NMT or any other advanced NMT architectures in the multilingual settings. And it can be seamlessly adapted to work in any domains like a standard NMT. Thus, we call it the **universal approach**. We then applied our propsosed framework in various multilingual translation tasks under two important and practical scenarios: under-resourced translation and zero-resourced translation. These scenarios show that our proposed method is impactful for many languages in the world, especially languages with little resource, not limited to common languages.

Despite the simplicity and elegance of our universal approach, it remains a few limitations that prevent the approach from being applied in more complicated and realistic multilingual scenarios. The basic problem lies in the fact that our universal approach, when accommodating many languages into one single architecture, creates a bottleneck that affects to the performance and computational cost of the whole architecture. By studying and analyzing those limitations, we then proposed some advanced techniques to release the bottle neck and address the problem. We are therefore able to improved our multilingual translation model in zero-resourced situation and deploy a large translation system capable of translating English and German texts to twenty four other European languages.

Finally, we have extracted and published `KIT-Multi`, a multilingual word embedding corpus, as a derivative product of training our multilingual translation systems. Preliminary evaluations show that the corpus is helpful in improving cross-lingual tasks. `KIT-Multi` is expandable and continues growing as we are adding more languages into our multilingual translation systems.

1.2 Overview and Structure

This section gives an overview of the contents of the individual chapters of this thesis:

Chapter 1 gives an introduction to the topics of this thesis.

Chapter 2 describes the backgrounds of this work. In particular, we briefly introduces the core concepts in machine translation and discuss several MT approaches. Statistical machine translation, which is among the study objects of this thesis, is described. Other components related to our experiments such as how to evaluate MT systems, the data used in the experiments and our typical scenario, lecture translator, are discussed as well. The second part in this chapter is reserved for the fundamental theory in the field of neural networks, including how to train a neural network, several advanced neural network architectures and a general neural framework to solve a family of popular natural language processing problems.

Chapter 3 starts with the discussion of SMT problems that motivates us to replace SMT sub-components by neural models. Then we present our proposed neural model to incorporate global context into the translation process and the experiments being conducted to prove its efficiency. Other related work about neural translation models is also briefly described. Our work as well as others relates well to the introduction of neural machine translation. Then, several NMT architectures are shortly reviewed.

Chapter 4 describes our main idea, the universal approach, to build a multilingual neural translation system from a standard NMT framework without changing its internal architecture. We then present the experiments and their results of our universal approach on under-resourced and zero-resourced scenarios.

Chapter 5 shows how we alleviate the large mix-language vocabulary problem in order to deploy our universal approach in more complicated and realistic multilingual translation applications. The experimental results in such applications are reported and discussed.

Chapter 6 summarizes the contribution of this thesis, draws the conclusion and discusses the future directions.

Chapter 2

Background

This chapter briefly describes the task of machine translation (MT) and also the required theoretical foundations in the field of artificial neural networks (ANN) as a machine learning technique applied to MT.

The traditional approach to MT, statistical machine translation (SMT), is reviewed to give background and context to newly-proposed neural machine translation (NMT) models and to this thesis. Related work about the applications of neural networks to other natural language processing tasks is discussed due to their growing popularity and the insights used in this thesis.

2.1 Machine Translation

Machine translation is the task of learning to automatically translate texts from one human language, called the 'source language', to another, called the 'target language'. This section provides a short overview of various approaches used for MT, with the focus on statistical machine translation. Currently, new ways of conducting machine translation using neural networks are gaining popularity. We will discuss them in the next chapter. Another important aspect of MT, how to evaluate the quality of MT systems and progress efficiently in the development cycle of building MT systems, is also reviewed in this section. After that, we describe the data used in this thesis to conduct research and build our MT systems. Finally, we pay particular attention to a realistic and challenging application of our MT research: Lecture Translation.

2.1.1 Rule-based Machine Translation

Since natural languages are generally considered to have hierarchical structures, there are different MT approaches which attempt to solve the problem at different levels of processing. These approaches often require corresponding rules composed by language experts in order to "transform" the source text into the target text. Thus, they are named **rule-based machine translation** approaches. Figure 2.1 depicts the famous Vauquois MT pyramid, which explains the levels of transformation in those approaches.

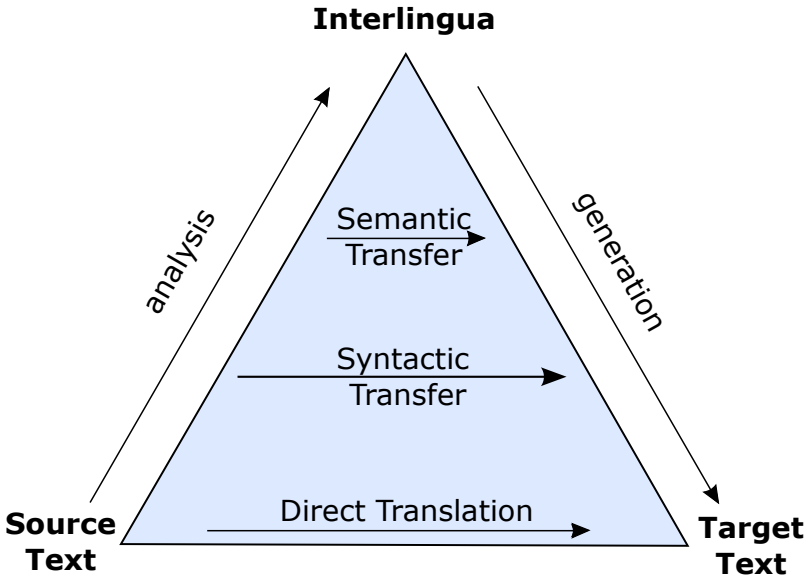


FIGURE 2.1: The famous Vauquois Triangle.

The most simple approaches, **direct translation**, conduct the transformation at the surface word level, using dictionary lookups to do word-by-word translation. **Transfer-based** approaches perform translation on a higher linguistic degree, employing either syntactic or semantic information in order to transfer abstract entities from the source language to the target language. On the highest level stand **interlingual** approaches, which require a new intermediary language, called the *interlingua*, which theoretically represents all the linguistic phenomena and entities of the source and target languages as well as the rules to transform from and to those languages. As you go higher on the pyramid, more abstraction is performed, which also requires more complicated analysis and generations.

The main advantage of the **rule-based** approaches is that they perform translation similar to the way human interpreters and translators do. Because the rules are directly built and compiled by the linguistic experts, the

translations produced by rule-based often achieve both adequacy and fluency. The disadvantages, on the other hand, are the requirement for both linguistic knowledge and expensive human efforts to create rules. Thus, **rule-based** approaches are hard to be extended and adapted to other language pairs or even other domains.

2.1.2 Statistical Machine Translation

In contrast to rule-based approaches, corpus-based approaches aim to learn translation rules from parallel texts automatically, making use of sentence-aligned parallel corpora where each source sentence $\boldsymbol{s}$ is aligned to a target sentence $\boldsymbol{t}$. There exist several corpus-based MT approaches, including example-based, decipherment, and statistical machine translation (SMT). Among them, SMT remains the most popular approach and has achieved state-of-the-art translation performance on many language pairs (Freitag et al., 2013; Cettolo et al., 2014; Bojar et al., 2015; Ha et al., 2015a).

Word-based models. SMT approaches the MT problem by breaking down the translation process to smaller tasks and modeling them using statistical models. SMT was first proposed by Brown et al. (1990), which employed Bayesian inference to separately model the translation and fluency problems over word-based units:

$$\boldsymbol{t}^* = \underset{\boldsymbol{t}}{\operatorname{argmax}}\, p(\boldsymbol{t}|\boldsymbol{s}) = \underset{\boldsymbol{t}}{\operatorname{argmax}}\, p(\boldsymbol{t})p(\boldsymbol{s}|\boldsymbol{t})$$

Here translation is constructed by the translation model $p(\boldsymbol{s}|\boldsymbol{t})$ and fluency is handled by the language model $p(\boldsymbol{t})$. While the language model has become regarded as a relatively well-solved problem in the language and speech research community, the word-based translation model became the main focus of the group. In a later study, Brown et al. (1993) describes several refined methods for efficiently estimating the translation model parameters, known as the IBM models. In those methods, the translation model is in turn broken into smaller models computing aspects such as lexical translation, alignment and fertility, where the latter computes how many target words each source word translates to.

For short sentences or phrases with simple structure between two typologically similar languages, word-based models can work efficiently. But, it becomes significantly harder in cases where the number of target

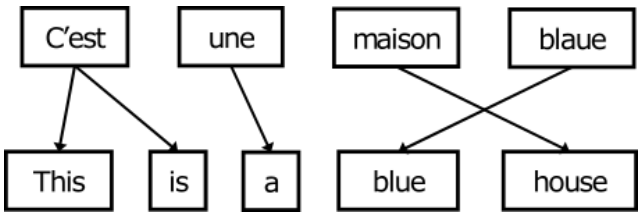


FIGURE 2.2: An example of word-based alignment.

words is very different to the number of source words due to compounds, morphology and idioms. Furthermore, word-based SMT often suffers from complex reordering rules and word order differences, and many ambiguous problems arise when translating between largely dissimilar languages.

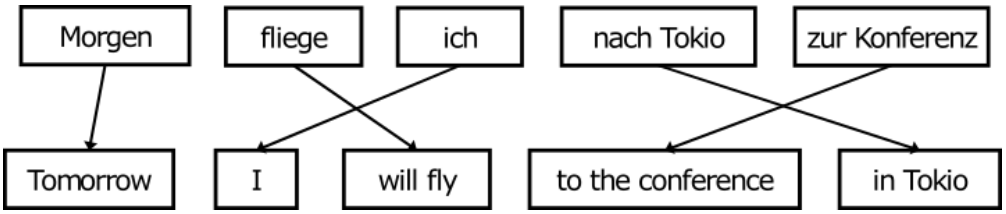


FIGURE 2.3: An example of phrase-based alignment.

Phrase-based models. Phrase-based SMT (Marcu and Wong, 2002; Koehn et al., 2003; Venugopal et al., 2003) aims to overcome the aforementioned limitations of word-based models by considering the translation unit as a consecutive sequence of words, called *phrases*[1], instead of a single word. By translating between phrases, phrase-based SMT (PBMT) can naturally model the phenomenon of many-to-many alignments which is inherent in the translation process. Moreover, it is not necessary to directly and model the fertility rate between source and target sentences. The problem of word ambiguity is also partially resolved by the additional information provided by phrases to disambiguate between contexts.

Another fundamental change from word-based models to phrase-based is the introduction of *discriminative frameworks* (Och and Ney, 2004) for machine translation. The word-based models before follow the generative approach where we decompose translation into smaller steps and model and solve them independently. In phrase-based SMT, those generative models, e.g. fertility models, local reordering or lexical translation, are built without referring to each others where they optimize their parameters to maximize the likelihood of observations inside each models. Discriminative models, on the other hand,

[1]The term *phrase* here represents consecutive segment of words, and not necessarily a linguistic constituent such as a noun phrase or an adjective phrase.

maximize the posterior probability given the data. Each model $h_m(\boldsymbol{t}, \boldsymbol{s})$, now called a feature, is combined in the framework with its corresponding weight λ_m expressing the contribution of that feature to the overall translation model:

$$\boldsymbol{t}^* = \underset{\boldsymbol{t}}{\operatorname{argmax}} \exp\left(\sum_m \lambda_m h_m(\boldsymbol{t}, \boldsymbol{s})\right)$$

This discriminative framework used in PBMT is realized as the linear combination of the features, under a log scale. Thus, it is often called *log-linear PBMT framework*. As we will see shortly, this log-linear framework is extremely useful since it allows PBMT to integrate a large number of features. In generative models, we are searching through the space of possible translations to find $\boldsymbol{t}^*$. When a new feature comes and we add it to our models, we are going to add a new dimension to our search space, making search complexity to increase exponentially with the number of features. With this discriminative framework, adding new features does not require searching the whole space of possible translations. Instead, we only need a list of translation candidates and try to discriminate good candidates from the bad ones. Many studies have set their focus on designing new features covering different aspects of translation. By learning how to scale the feature reasonably, the discriminative framework could propose a number of features which are highly related and hope for the discriminative framework figuring out the correlations and their contributions via a suitable set of weights $\{\lambda_m^* : m = 1, \ldots, M\}$.

Training. Given a training parallel corpus with N sentence pairs $\{(\boldsymbol{s}^{(n)}, \boldsymbol{t}^{(n)}) : n = 1, \ldots, N\}$, we train a log-linear PBMT system to estimate the parameters of individual features and find the weight set $\{\lambda_m^*\}$ which maximizes the posterior probability:

$$\{\lambda_m^*\} = \underset{\{\lambda_m\}}{\operatorname{argmax}} \sum_{n=1}^{N} \log p_{\{\lambda_m\}}(\boldsymbol{t}^{(n)}, \boldsymbol{s}^{(n)})$$

This optimization corresponds to directly maximizing the likelihood of the translation model. It has very nice properties: there exists a global maximum, and there are algorithms that are guaranteed to converge to the global maximum. So it is possible to train a perfect model to fit the training data. However, our goal is producing high-quality translations of unseen data, not fitting the training data. So the optimal set of weights for our SMT system is often another $\{\lambda_m^*\}$ than the result of a training procedure.

Tuning. Finding the set of weights that helps the SMT system performs well on unseen data is referred to as *tuning*. Tuning is usually conducted on a small parallel corpus, which is independent of the training data, called the *tuning set* or *development set*. There have been several methods proposed for tuning. Here we briefly discuss one of the most popular methods, called minimum error rate training (MERT) (Och, 2003; Bertoldi et al., 2009). Following MERT, first, the whole tuning set is translated using some initial set of weight $\{\lambda_m^0\}$ to produce an n-best list. Then the weights are optimized following the criterion:

$$\{\lambda_m^*\} = \underset{\{\lambda_m\}}{\operatorname{argmin}} \operatorname{Err}(\boldsymbol{t}^*_{\{\lambda_m\}}, ref)$$

Here ref is the reference from the tuning set and Err is some function to calculate the total errors between the n-best candidates and the references, based on some automatic translation evaluation metric[2]. The optimization process is then repeated to find better weights and this iterative procedure continues until some stop condition is reached.

We now shortly describe the popular features used in many PBMT systems.

Phrase translation features. Those features score the probability $p(\bar{\boldsymbol{t}}, \bar{\boldsymbol{s}})$ of a phrase pair $(\bar{\boldsymbol{t}}, \bar{\boldsymbol{s}})$ and are stored in a *phrase table* for faster look-ups. Those scores are often derived from word-based IBM models with some heuristics considering the counts and the alignments among them (Koehn et al., 2003). There are other phrase translation features developed from other criteria and using other methods than statistical models; Some of them will be reviewed in Chapter 3.

Distortion and reordering features. The distortion models learn how to conduct local movements of words and phrases when translating between two languages having different word orderings. For example, the distance based distortion generally penalizes long-distance movements of words. Another model widely used in PBMT is the lexicalized reordering model (Koehn et al., 2005; Galley and Manning, 2008) which conditions on specific words to predict the orientation of a given phrase consisting of those words. Other complicated reordering features try to incorporate morphological or syntactic information from the source or target sentence to better model the reordering phenomenon.

Language models. Language models estimate the fluency of a sequence of symbols $\boldsymbol{S} = s_1, \ldots, s_N$ by giving a larger probability $p(\boldsymbol{S})$ to sequences that

[2]The evaluation metrics will be introduced later in Section 2.1.3.

are more fluent. In other words, they indicates how probable a sequence is to appear in that language. For example, in the word-based SMT, $p(\boldsymbol{t})$ is the language model of the target sentence $\boldsymbol{t}$. The language model helps the SMT system in selecting appropriate words for the local context as well as the word order. In phrase-based SMT, the target language model $p(\boldsymbol{t})$ still keeps an important role, but we can handily incorporate different types of language models thanks to the log-linear framework and discriminative training. We can train our language model by estimating the probability of a word conditioned on all preceding words:

$$p(\boldsymbol{S}) = \prod_{i=1}^{N} p(s_i|s_1, \ldots, s_{i-1})$$

Normally, to deal with the data sparsity problem when estimating $p(s_i|s_1, \ldots, s_{i-1})$, the context of the language model is usually restricted to a window of n few words, called the n-gram. Then we can achieve a reliable estimate for those probabilities using the maximum likelihood principle:

$$p(s_i|s_1, \ldots, s_{i-1}) \approx p(s_i|s_{i-n+1}, \ldots, s_{i-1}) = \frac{\text{count}(s_{i-n+1}, \ldots, s_{i-1}, s_i)}{\text{count}(s_{i-n+1}, \ldots, s_{i-1})}$$

However, the coverage of the corpus for language modeling cannot be large enough to avoid zero counts in the estimation. Thus, back-off and smoothing techniques are often applied in order to assign probability mass to unseen sequences instead of unreasonably zeros (Chen and Goodman, 1999).

These types of language models are called (statistical) n-gram language models. Another direction is the approaches which use neural networks to estimate $p(\boldsymbol{S})$. Those approaches are then called neural language models and will be discussed in detail in Section 2.2.5.

2.1.3 Translation Evaluation.

In order to improve our machine translation system, we need a way to automatically evaluate the quality of our system outputs. In this section, we review a number of popular approaches for translation evaluation.

Human Evaluation. Obviously, the best way to evaluate an MT system is to ask translators and interpreters, or at least bilinguals, to conduct the evaluation. Human evaluators can rate the translated outputs of our MT

systems according to different aspects of quality. They can also identify and pay particular attention to specific types of errors that our MT system produced. The latter often refers to conducting *error analysis* of the translation outputs. Then we can narrow down the problems and set research efforts to focus on some concrete directions.

One of the most widely-used error analysis frameworks was proposed by Vilar et al. (2006). Following this framework, we consider the following error categories: incorrect words, missing words, word order, and punctuation. Incorrect word errors refer to the errors in which a word is wrongly translated, either because of semantic ambiguity, incorrect inflection or is a fully incorrect word selection. Missing word errors refer to the absence of some word in the translation. Missing word errors are considered to be more serious if the missing word is a content word as opposed to a function word. The word order category accounts for wrong word order errors. The final error category is related to wrong places for punctuation or wrong types of punctuation in the translation.

Automatic Evaluation. Human evaluation is expensive and time-consuming. It is impractical for human evaluators to evaluate a large sample of translated sentences. Thus, there has been significant interest in designing an automatic metric that can evaluate the quality of a translation system. An automatic metric is often used in two scenarios. First, it is employed as a fully automatic and fast way of tuning an MT system (c.f. Section 2.1.2). Second, as an alternative way to evaluate an MT system. Because of the latter scenario, it is necessary for an automatic evaluation metric to correlate well with human evaluation.

The first metric we mentioned is the *translation error rate* or *translation edit rate* (TER). TER is determined by the number of post-edits needed to correct a system output into one of the references (Snover et al., 2006):

$$\text{TER}(out, ref) = \frac{\text{number of needed edits}}{\text{average number of words in } ref}$$

Another metric is the *precision* of the translation, calculated as:

$$\text{precision}(out, ref) = \frac{\text{number of correct words}}{\text{number of words in } out}$$

The precision metric leads us to the most popular metric for machine

translation: the score of bilingual evaluation understudy (BLEU). BLEU (Papineni et al., 2002) is essentially a modified n-gram precision, based on the number of n-gram matches between the translation and the reference:

$$\text{BLEU}_n(out, ref) = BP. \sum_{i=1}^{n} \log precision_i$$

Here BP signifies a 'brevity penalty'. This implicitly accounts for recall by penalizing for missing words in the translation if the output is too short. Often, the BLEU score where $n = 4$ is used:

$$\text{BLEU}_4(out, ref) = \min(1, \frac{\text{number of words in } out}{\text{number of words in } ref}) \sum_{i=1}^{4} \log precision_i$$

BLEU score is usually computed for an entire corpus, rather than for individual sentences, in order to avoid zero matching counts in any case of n. For example, a translated sentence consisting of three words will give an invalid BLEU score since $precision_4 = 0$. However, the problem of using BLEU on the sentence level is still there, which makes it unreliable for training or tuning SMT systems as an objective function. Furthermore, optimizing a log-linear PBMT using BLEU on the level of the sentence might not lead to the global optimization of BLEU on the whole corpus. In the case of neural systems discussed later, the aforementioned BLEU is also unusable as an objective function, since it is not differentiable for gradient-based optimization.

Lin and Och (2004) suggest a sentence-level approximation of corpus-level BLEU, called BLEU+1. BLEU+1 has applied the simple add-one smoothing in order to avoid zero-precision problem. Nakov et al. (2012) improve BLEU+1 by suggesting two advancements. The first applies add-one smoothing not only to the precision calculations but also to the brevity penalty to make the score more balanced. The second grounds the precision by subtracting some value when there are no matches. "Grounded" BLEU+1 slightly favors longer sentences (Nakov et al., 2012).

There are many other MT metrics but BLEU remains the most popular due to its lightweight calculation, overall robustness and its correlation to human evaluation. In recent years, the Workshop on Statistical Machine Translation (WMT) has organized a shared task in order to encourage researchers to propose new and useful MT metrics. Several metrics, especially

character-based ones, are allegedly better than BLEU for some scenarios and language pairs.

2.1.4 Data for Machine Translation

Parallel corpora, i.e. large collections of texts in one language and their translations into one or more other languages, are the central data for building machine translation systems. In the most simple case, a parallel corpus contains only texts in two languages, therefore, it is called a bilingual corpus. A multilingual parallel corpus, which contains sentence-aligned texts in more than two languages, can be viewed as multiple bilingual corpora. Another type of data used for MT is monolingual corpora, which containing text in only one language. The traditional usage of monolingual corpora is for language modeling. Another usage of monolingual data which has gained prominence recently is to produce synthetic data for neural machine translation training.

In available parallel and monolingual corpora, domains vary greatly from laws, political discussions or academic lectures to daily and travel dialog, news, movies subtitles or even bible. For any MT system, there will be in-domain corpora, which cover the domain of interests of such MT systems. The out-of-domain corpora, normally available in a larger amount, can also be helpful in building an MT system. However, to use these corpora, we need to apply data selection and domain adaptation techniques on those corpora prior to or during MT training.

The majority of parallel data for building MT systems are well-formed, written texts extracted from parallel documents with high-quality translation. Some other corpora consist of spoken-style data, which might be scripted such as speeches and talks, or partly spontaneous like presentations and lectures, or even entirely spontaneous like meeting logs, discussions and daily conversations. In order to leverage these data sources, especially in a spoken language translation system, formalization and style adaptation phases need to be conducted.

Most of the conventional MT systems are trained on the sentence level. For monolingual data, this means that the training texts are required to have adequate sentence boundaries. For parallel data, the corpora need to be sentence-aligned, i.e. any sentence in the source language corresponds to only

one sentence in the target language[3]. To achieve this property, bilingual or multilingual bodies of texts are first fed through the extraction and alignment phases using sentence alignment tools Gale and Church (1993); Moore (2002).

As in other machine learning problem, data in MT are always divided into *training*, *development* and *test sets*. They are non-overlapped datasets, use for training, tuning and testing the systems, respectively. In popular MT evaluation campaigns such as the Workshop on Statistical Machine Translation (WMT)[4] or the International Workshop on Spoken Language Translation (IWSLT)[5], *official test sets* are hidden from participants during the evaluation period and only revealed during the official evaluation of the participants' MT systems.

In this thesis, we use two different types of MT data, depending on our application scenarios. Written texts and highly-scripted texts are used for training our conventional machine translation systems. They can be in-domain or out-of-domain data with respect to the domain of the test set. Many written-style corpora for European languages are introduced in the news translation shared task of WMT every year. On the other hand, the second type of data includes the spoken-style corpora used to train our spontaneous speech translation system. This kind of data is mostly available in the shared task of IWSLT related to spoken language translation. In this section, we briefly introduce those different sources and styles of MT corpora used in the thesis. We will describe this data in more detail when conducting specific MT experiments.

European Parliament parallel corpus. One large, high-quality and wide-coverage parallel corpus suitable for building a large-scale SMT system is the European parliament proceedings parallel corpus (EPPS or Europarl) (Koehn, 2003), which contains the proceedings from the European parliament. It is available for 20 language pairs with English as the source language and 20 target languages of the European Union's country members. This corpus belongs to the written-style category. The statistics of the newest version of Europarl are included in the Appendix.

[3]However, the source sentence can have several distinct translations, or several different source sentences can be translated into the same target sentence. In these cases, the same source or target sentence will be duplicated correspondingly to reserve the sentence-aligned property of the corpora.

[4]E.g. http://www.statmt.org/wmt15/. It has now become the Conference on Machine Translation, e.g. http://www.statmt.org/wmt18/.

[5]https://iwslt.org/.

News Commentary parallel corpus. The parallel texts extracted from news commentary editorials (NC) on the project Syndicate website[6] are available in five European languages. Due to its high quality and its coverage of news domain, NC is often used as the in-domain data for the news translation shared task of WMT.

CommonCrawl parallel corpus. Starting in 2013, WMT have introduced parallel corpora collected from web sources in several languages. While the size of the corpus is large and has become larger over the years, it contains some noise and should be used with care.

Gigaword monolingual corpus. Another corpus released from WMT is the Gigaword corpus, which contains monolingual texts in many languages. In the thesis, we use the English, French and German parts of Gigaword corpus.

TED parallel corpus. This corpus, published under Cettolo et al.'s Web Inventory of Transcribed and Translated Talks (WIT3), is a spoken-style multilingual corpus extracted from TED talks. TED talks are short presentations from inspiring people talking their stories in various topics. They are published on the website of TED conferences [7]. The speeches are recorded, transcribed and translated into many languages, and the transcriber and translators need to conform to strict guidelines in order to produce fluent, high-quality and easy-to-follow subtitles. Although TED subtitles are spoken-style, the talks are often scripted by the presenter and transcribers are recommended to produce correct sentences as well as remove speech disfluencies from the original audio. Thus, compared to other spoken-style spontaneous data such as meeting or lectures, TED data contains a lesser degree of spontaneousness, such as stammers, correctness or other speech disfluencies. Due to its broad domain coverage and its availability in many languages, TED parallel corpus is used thoroughly in most of our experiments as the primary in-domain data. Furthermore, we often employ the TED corpus for certain translation directions as the sole data, simulating low-resourced scenarios.

University lecture corpus. The university lecture data consists of lectures on computer science and other areas given to the students at the Karlsruhe Institute of Technology (Stüker et al., 2012). Recordings of selected lectures are transcribed by research assistants according to detailed guidelines. These guidelines differ from the TED transcription guidelines in the way that the

[6] https://www.project-syndicate.org/
[7] https://www.project-syndicate.org/

lecture transcriptions are intended as training and test data in automatic speech recognition and machine translation systems. Therefore, they have to meet particular requirements, necessary for research in those fields. As a consequence, the transcriptions need to be very close to the actual spoken words. Speech artifacts such as hesitations, stuttering, mumbled words, aborted and restarted words are annotated as they are spoken. This may result in ungrammatical text, which poses a difficulty for statistical translation models which are typically trained primarily on well-formed written text. In the experiments presented in this thesis, we use a test set of seven lectures given by five different speakers covering the subjects of computer science. The length of each lecture varies between 35 and 91 minutes.

2.1.5 Lecture Translation.

A great motivation for the research of this thesis is to contribute scientific and practical values to the KIT lecture translation system (KIT LT). The KIT LT is an open domain speech translation system which offers an automatic lecture translation service to KIT students and has been deployed in several lecture halls at KIT. Starting from the initial version of lecture translation Fügen et al. (2006), our system has been continuously developing Kolss et al. (2008); Cho et al. (2013); Niehues et al. (2018); Dessloch et al. (2018). The KIT LT plays an important role in motivating and acknowledging our research in speech translation area over the recent years.

The KIT LT system is a client-server architecture, consisting of three main components: automatic speech recognition (ASR), segmentation, and machine translation. An overview of our LT system is shown in Figure 2.4. The KIT LT works in a pipeline. Lecture speech is simultaneously recorded by a recording client then sent to the LT server, where it first goes through the ASR to get a long transcribed texts. The segmentation puts sentence boundaries and adds punctuation in order to help the following machine translation deals with texts in written form. The machine translation produced the translation in desired languages and the translated texts are sent to the display client. While the main use case is online translation, where the user can follow the lecture concurrently on his smart phone or laptop, we also offer a web-based archive for viewing previously recorded lectures.

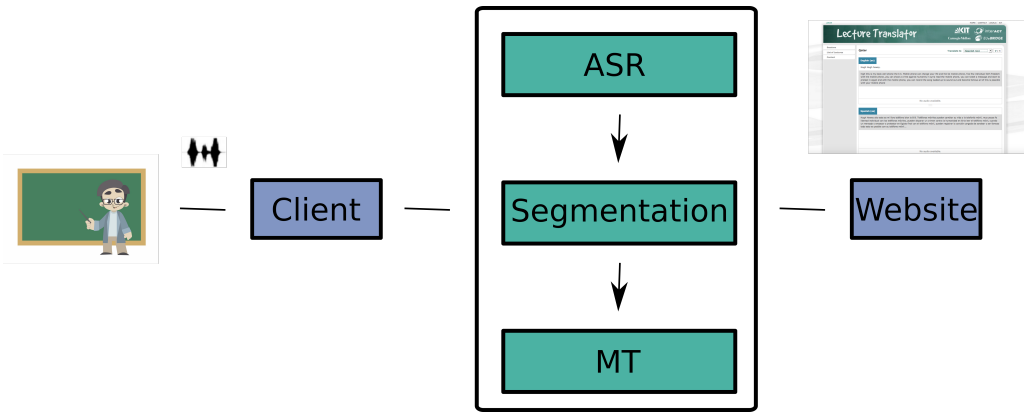


FIGURE 2.4: KIT LT System Architecture (Dessloch et al., 2018)

An efficient design of the Lecture translation system has to deal with certain challenges include low latency, domain-specific knowledge in academic domains, and multilingualism:

- Low-latency: Smooth user experience is only achieved with transcrition and translation being in sync with the streaming talk. It is crucial to provide spontaneous speech translation models with minimum latency.
- Domain Adaptation: The speech translation system needs to be flexibly adapted to different domains to achieve the best performance for each talk providing with a possibly different area of contents.
- Multilingualism: An international education system demands multilingual support from the lecture translator. The translation models, therefore, have to be equipped with the ability to rapidly support and extend with new languages.

2.2 Neural Networks

Neural networks are a family of machine learning techniques which powerfully model non-linear relationships between inputs and outputs of a predictive system. Neural networks are currently the backbone of the popular deep learning trend fueling the new Artificial Intelligence era. Compared to other machine learning methods, they have prominent strengths such as universal approximation ability. On the other hand, models employing neural networks are questionably black boxes since there are lacks of research in interpreting and explaining what and how neural networks learned from data. Nonetheless, in this section, we will limit the discussion about neural

networks from the perspective of machine learning methods and how they have been applied to various Natural Language Processing tasks under a unified framework.

2.2.1 Linear Models

Linear models are the machine learning models which predict a linear relationship between inputs of a system and its outputs. Perceptrons and linear regression are typical examples of single-output linear models that can be graphically represented as a simple network (Figure 2.5). This network has only one processing unit, taking N inputs and produce an output which is a transformation f over the linear combination of the inputs. One weight corresponds to one input and is illustrated by an arrow connecting from that input and the output. The processing unit in those models is called *a neuron* (or *an artificial neuron*) and this network is the minimal type of neural network which has only one neuron.

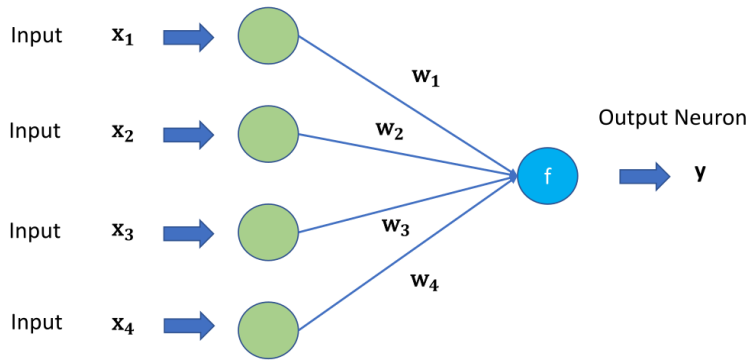


FIGURE 2.5: Linear models with a single output neuron.

$$y = f(z) = f(\mathbf{w} \cdot \mathbf{x}) = f(\sum_{i=1}^{N} w_i x_i)$$

In case of perceptrons (Rosenblatt, 1958), a weighted sum of the inputs $\mathbf{w} \cdot \mathbf{x}$ is first calculated, following by a threshold function $f(z) = \mathcal{H}(z)$ to produce a binary response. Thus, a perceptron is normally considered a linear classifier. Linear regression, on the other hand, uses the weighted sum directly to estimate the scalar output value ($f(z) = z$).

There exists an important linear model, logistic regression, which the function f is a non linearity: $f(z) = \sigma(z)$. Here σ is the *sigmoid* function, defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Despite the name, logistic regression performs a binary classification decision, *True* or *False*, 1 or 0 . The output value of the sigmoid function represents the probability of a specific example $\mathbf{x}$ to be classified as *True*: $p(y = 1|\mathbf{x})$. Naturally, that value then can be thresholded by e.g. 0.5 in order to obtain a hard decision.

Although sigmoid function σ is a non-linear function on both inputs x_i and weights w_i, standard logistic regression is in general considered as a linear model since its outcome always depends on the linear combination of its inputs (weighted sum). In other words, the decision boundary of a logistic regression is a linear one. More specific, the decision boundary of a logistic regression separates the set of x such that:

$$\begin{aligned} \frac{1}{1 + e^{-\mathbf{w}\cdot\mathbf{x}}} &= 0.5 \\ \Rightarrow e^{-\mathbf{w}\cdot\mathbf{x}} &= 1 \\ \Rightarrow \mathbf{w}\cdot\mathbf{x} &= 0 \end{aligned}$$

Thus, the decision is linear to the inputs $\mathbf{x}$ (and also the weights $\mathbf{w}$).

Using perceptron or standard logistic regression, it would be straightforward to adapt those linear models in order to produce more than one output. We can simply use multiple perceptrons or logistic regressions sharing the inputs, each of them corresponds to one output and has a distinct weight set to be learned. This forms a network with one input layer and one output layer, as shown in Figure 2.6.

The main limitation of a linear model lies in its inability of modeling complex relationships among inputs. The inputs of a linear model are considered to be under an independence assumption: there is little or no correlation between them. Features for machine translation and textual features in general, however, relate deeply to each other. A sentence has many features forming its linguistic hierarchy. Words in a sentence share syntactic and semantic dependencies. Reordering models, language models and even phrase-based translation models in SMT all reflect the local ordering of words at different

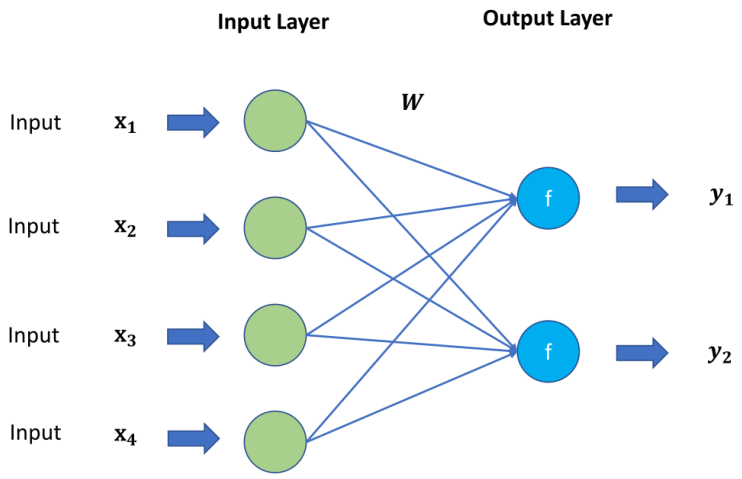


FIGURE 2.6: Linear models as a simple neural network.

levels. A linear model is unable to effectively discover and represent those complicated connections.

2.2.2 Loss Function and Gradient Descent Principle

The purpose of training linear models is to find the weights corresponding to the inputs so that the produced output is close to the expected output of the system. One way to do that is to look for the weight set minimizing the difference between the produced outputs and the expected outputs. This difference is measured by a scalar value, which depends on the values of produced outputs and expected outputs. We model this as a function of the inputs and their expected outputs as well as the weights of the model. This function has several names: *loss function*, *cost function*, or *error function*. The *loss function* is central to discriminative machine learning models where one would define a suitable loss function for a specific model to attempt to find the weights that minimize error in a convenient and effective way.

Gradient descent is an iterative method to find weights that minimize the loss function by moving weights towards the direction of the steepest descent. In term of weight updating for neural models, it is implemented as:

$$w_i = w_i - \eta \frac{\partial \mathcal{L}}{\partial w_i}$$

, where $\frac{\partial \mathcal{L}}{\partial w_i}$ is the partial derivative of the loss function $\mathcal{L}$ with respect to the weight w_i and η is the learning rate, measuring the update speed.

In the multi-dimensional space of model weights, besides one global minimum of the loss, there exist many local minima. Since in many linear models such as logistic regression, the loss function is convex. Thus, it has only one minimum, gradient descent always converges to that minimum. In theory, however, for other more complicated machine learning models and loss functions, gradient descent does not assure to find the global minimum but it often has a higher chance to stop at one local minimum if the model is trained in a proper and suitable way. In practice, the local minimum gradient descent found is often good enough for a model trained in that way.

In this section, we are discussing loss functions and gradient descent principle for neural linear models. However, gradient descent is applied in the case of more complex neural architectures and actually, it is among the most popular methods to train deep neural networks. The critical point is how to calculate $\frac{\partial \mathcal{L}}{\partial w_i}$ effectively and efficiently in those complicate neural nets.

2.2.3 Multi-layer Perceptrons and Feedforward Neural Nets

As discussed in the previous section, each neuron using a non-linear function represents a mapping between its inputs and its output, and when used together, multiple neurons are able to model distinct relationships from the inputs to the outputs. If we continue combining the outputs of those neurons this way, i.e. introducing an intermediate layer of neurons between input and output layers, we can exempt the limitation of (multiple) linear models. The neurons in the intermediate layer now encode different relations of the original inputs, and then the combinations of those neurons establish complicated, non-linear connections among the inputs and the outputs of the whole network. This network is called multi-layer perceptron (MLP). We can add more intermediate layers to represent hierarchical structures of the connections. Since we can only observe the inputs and outputs of the network but not the intermediate processes that connects them, these layers are called *hidden* layers. Figure 2.7 shows such a network.

Hornik (1991) and Csáji (2001) show that an MLP with one hidden layer and a sufficient number of neurons in the hidden layer is a universal function approximator. It can approximate any real function mapping from any multi-dimensional discrete space to another. In general, any MLP or other neural network architecture can be considered as a parameterized

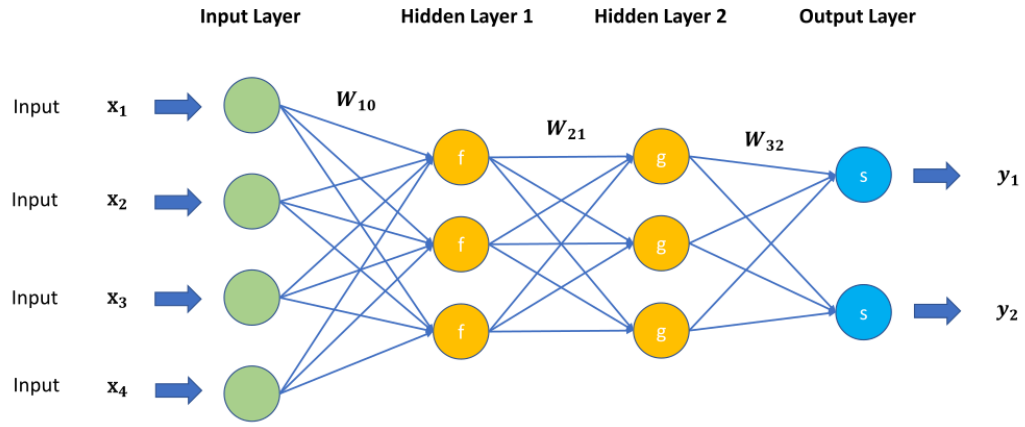


FIGURE 2.7: A multi-layer perceptron with three layers.

mathematical expression over $\{\mathbb{R}^m \rightarrow \mathbb{R}^n : m, n \in \mathbb{N}^+\}$ made from basic mathematical functions.

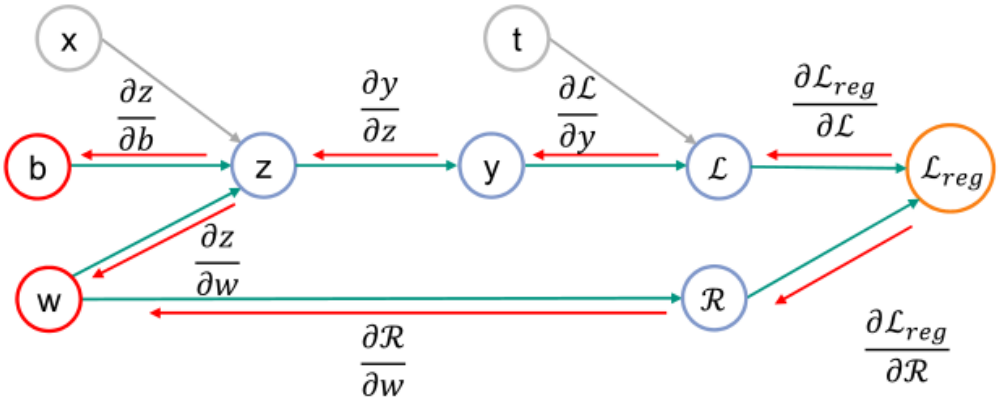
MLP is an instance of a neural architecture family, called Feedforward Neural Nets (FNN). As the name suggests, FNNs are a neural architecture in which the connections between neurons do not form any loop. By understanding FNN in this way, we can easily represent any FNN as a computational graph and apply a general procedure over this computational graph to calculate the partial derivative of the loss function with respect to any weight or any intermediate value of the network.

2.2.4 Computational Graph and Backpropagation.

Backpropagation is an algorithm used to efficiently and effectively train an FNN. It allows us to compute the gradients of any parameter of that FNN by iteratively applying the chain rule backward. The idea is similar to the way we train linear models yet through more layers. First, we do a forward pass in order to calculate the output of the network and the loss value from the input; then we propagate the error backward to the inputs and update the weights using gradient descent principle. While we can compute the partial derivative of the loss function with respect to any parameter $\frac{\partial \mathcal{L}}{\partial w_i}$ by hand, this work is error-prone, especially with a complicated architecture. Thus, it is preferable to use automatic tools for gradient computation. Computational graph and its allies procedure, auto differentiation, serve this purpose.

A computational graph is a representation of an arbitrary mathematical expression in graph. It is a directed acyclic graph in which each node

represents a variable and each edge represents the direction of information flow from a node to another. The variable can be a bounded value or a mathematical operation.



Computational Graph representing a Regularized Logistic Regression.

As we have seen, any FNN can be represented as a computational graph. In the meanwhile, the directed structure of a computational graph defines the order of the computations from one node to another node in that graph. If we can establish a reasonable ordering of the computation graph, we can follow that ordering in order to compute the loss of the corresponding FNN through the edges which begin with the input nodes. There exist at least one of such orderings which we call a *topological ordering*.

Topological orderings of a directed acyclic graph rean ordering in which for every edge $u \rightarrow v$, where u always comes before v. Let us define $\text{Pa}(v)$ to be the set of parent nodes of the node v ($\forall u \in \text{Pa}(v) : \exists(u \rightarrow v)$) and $\text{Ch}(v)$ to be the set of children nodes of the node v ($\forall u \in \text{Ch}(v) : \exists(v \rightarrow u)$)

With these definitions in mind, now we can approach an automatic way to train an arbitrary FNN using auto differentiation. Auto differentiation is a set of general procedures to compute the value for a mathematical expression and the derivative of that value. Applied in neural network training, auto differentiation becomes our familiar backpropagation algorithm. Algorithm 1 describes the backpropagation algorithm under this view.

Algorithm 1 Backpropagation over Computational Graph.

Input: CG representation of the NN
Output: Derivative of the loss w.r.t all nodes v_i: $d(v_i)$

Create topological orderings of the CG:
$\Rightarrow (v_1, v_2, \cdots, v_N)$

# Forward pass:
for $i \leftarrow 1 \cdots N$ **do**
    Compute $v_i = f_{v_i}\text{Pa}(v_i)$
end for

Backward pass:
$d(v_N) = 1$
for $i \leftarrow N-1 \cdots 1$ **do**
    $d(v_i) = \sum_{u \in \text{Ch}(v_i)} d(u) \frac{\partial u}{\partial u}$
end for

2.2.5 Neural Networks on Natural Language Processing

Natural Language Processing (NLP) is among the most important corners of Artificial Intelligence (AI), where we teach machines to understand, process and generate human languages. At the moment, NLP problems have been realized by machine learning approaches that automatically learn from data. Those approaches have to cope with typical challenges of NLP such as working with very high-dimensional yet sparse feature vectors and modeling hierarchical structures of discrete symbols. For a long time, linear non-neural models such as support vector machines or maximum entropy models have been used extensively to tackle problems in NLP.

Recently, there has been success in replacing those linear models by neural networks, since the latter are able to better cope with high dimensional inputs and effectively learn latent structures of NLP entities. Here we will review a general neural framework (Bengio et al., 2003) to solve a classic NLP problem: *sequence modeling*. This problem is closely connected to machine translation using neural architectures later.

Sequence modeling refers to the problem of modeling a variable length sequence of symbols $\boldsymbol{S} = s_1, \ldots, s_N$ and producing discrete outputs from a finite set of choices in order to reflect some aspect of that sequence. For example, given a German sentence, our machines need to point out the part of speech (POS) of every word in that sentence, which one is a noun, which

one is a verb, and so on. Here the outputs are another sequence of POS tags corresponding to the input sentence and the problem is called *part of speech tagging*. Another similar example is the problem of *named entity recognition* (NER) where given a sentence, computers need to tell which word refers to a name of a person or a location, which word is just a normal word. In those two examples, the set of outputs are discrete and finite (the set of predefined POS tags or NER tags).

The simplest case of sequence modeling results when the model only needs to produce one output for a given sequence. For example, in the sentiment analysis, given a sentence, machines need to predict the sentiment behind that sentence. This output can even be a binary output. In the sentiment analysis example above, the sentiment prediction can be "Good" or "Not Good". This simple case can be viewed as a multi-class classification task in the field of machine learning where we train some machine learning model with its parameters $\boldsymbol{\theta}$ to predict the most probable output class c among a finite set of discrete outputs:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p_{\boldsymbol{\theta}}(c|\boldsymbol{S})$$

Other sequence modeling problems which need to predict many outputs can be solved by applying our multi-class classifier over the time steps where in each time step i we use the information of $\overline{\boldsymbol{S}}$ contextually conditioned to i:

$$\boldsymbol{\theta}_i^* = \underset{\boldsymbol{\theta}_i}{\operatorname{argmax}}\, p_{\boldsymbol{\theta}_i}(s_i \rightarrow c|\overline{\boldsymbol{S}_i})$$

Normally, machine learning models try to avoid the data sparsity problem by limiting the context $\overline{\boldsymbol{S}_i}$ to a fixed-size input window moving along the time step i. If we look closely, the language model described in Section 2.1.2 is exactly an instance of our sequence modeling problem and the context $overline\boldsymbol{S}_i$ here is the n-gram window consisting of the previous $n-1$ words: $s_{i-n+1}, \ldots, s_{i-1}$. Unlike part of speech tagging or named entity recognition, the output set of the language model is the vocabulary of the language and it is often a very large to intractable set.

Figure 2.9 depicts a general neural framework to solve the language model problem as the multi-class classification over a natural language sentence. First, each input word will be encoded as an one-hot high dimensional vector based on the vocabulary of the input, where the value of that vector at the index of the word in the vocabulary is 1 and all others are 0. The one-hot

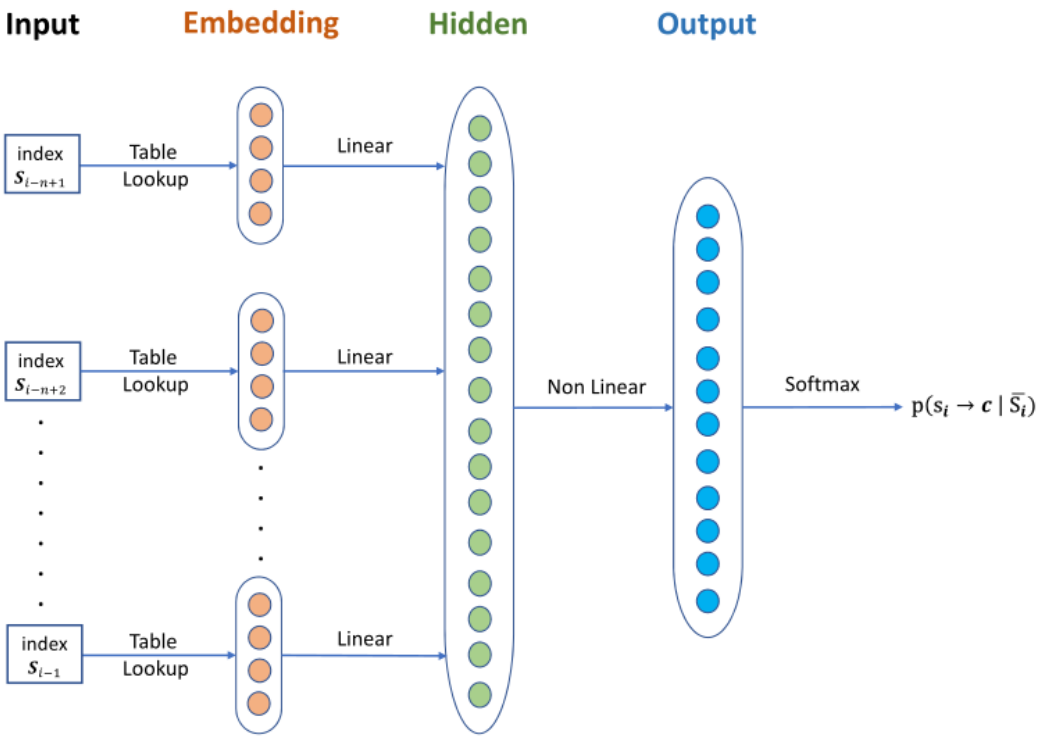


FIGURE 2.8: A neural language model architecture.

vectors of $n-1$ words in the context windows $\overline{S_i}$ are projected into a shared *embedding* space[8]. Then the concatenation of the embedding vectors is fed into another linear hidden layer prior to the output layer. The output layer is a non-linear layer with its own parameters followed by a softmax to estimate the probabilistic distribution over outputs o_j for the word s_i as the class c_j:

$$\text{softmax}(o_j) = \frac{e^{o_j}}{\sum_k e^{o_k}} = p(s_i \rightarrow c_j | \overline{S_i})$$

The vector of each word in embedding is called a *word embedding* or a *word representation*. The embedding is an important concept where discrete inputs, i.e. input words here, are embedded in a shared space that words are similar in some aspect would behave similarly in the prediction. The traditional n-gram language model cannot afford to have this property where each input word is considered independently to the final prediction. Furthermore, the hidden and non-linear output layers help the architecture implicitly learn hidden relationships from the embeddings in order to perform better prediction. For

[8] Actually, this projection - or linear transformation - over one-hot vectors can be efficiently implemented as a simple lookup operation on the index of the words instead of a vector-matrix inner product operation.

example, there may be a neuron in those layers determining whether the last word in the context window could be a noun, or the whole windows could be a noun phrase, and another neuron determining the plurality of the windows. All of the information which has been learned implicitly are then combined automatically to predict the class of the next word s_i.

2.2.6 Advanced Neural Network Architectures

In this section, we describe several advanced neural architectures which have been proposed to solve typical machine learning problems. The common property of those neural networks is that they are, or can be treated as feedforward architectures, therefore, we can build computational graphs representing them and train them using backpropagation.

Time-Delay Neural Networks and Convolutional Neural Networks. *Time-Delay Neural Networks* (TDNNs), introduced by Waibel (1997), and its successor *Convolutional Neural Networks* (LeCun et al., 1995) are a special kind of feedforward neural networks which can effectively deal with variable-length (time-shifted) sequences and model long distance dependencies within them. TDNNs therefore have been applied in many machine learning problems, mostly in speech recognition (Waibel, 1989; Waibel et al., 1990; Lang et al., 1990; Dellaert et al., 1996; El-Bakry and Mastorakis, 2009), but also widely popular in other areas such as computer vision (Chen et al., 2004; Jaeger et al., 2001; Schenkel et al., 1995; Kim et al., 2000), robotics (Lin et al., 1995; Wöhler and Anlauf, 1999; Kaiser, 1994), time series prediction (Kim, 1998; Jiang et al., 2009) and natural language processing (Collobert et al., 2011; Kalchbrenner et al., 2014).

Instead of using a fully-connected layer, TDNNs run a window from the delayed input D_i to D_{i+N} with its parameters $\boldsymbol{W}$ and computes the weighted sum of its inputs and feed it through a nonlinear function f:

$$\boldsymbol{h}_i = \mathrm{f}(\boldsymbol{W}^T \cdot \boldsymbol{S}_{i:i+N})$$

This non-linear transformation in the TDNN architecture are tied across the delays (time steps), thus TDNN layers are forced to learn features shifted within the patterns. Those features are called translation-invariant features, where they do not change over time. E.g. the noun phrase "*a blue house*" will still be a noun phrase and has specific meaning (semantics) no matter where it

is located in the sentence. And its syntactic function and semantics are learned automatically. Applied in speech recognition, TDDNs are able to detect of time-independent sub-patterns, therefore, one does not need to perform an additional step for time alignment.

After that, some sampling operation would be conducted. In TDNN architecture, max pooling is often used. By this way, the most active features can be selected as the inputs for the next TDNN layer.

In CNN architecture, the sliding window is called kernels, and in principle, CNNs are TDNNs but they have been widely applied in computer vision and natural language processing.

TDNNs or CNNs are feedforward neural nets, hence, the backpropagation described in Section 1 can be used straight-forward with the concerns to limited computation of the sliding window at a specific time step.

Recurrent Neural Networks. Recurrent neural networks (RNNs - Elman, 1990) are a variety of neural network that makes it possible to model these long-distance dependencies. The idea is to take some "*memory*" of the past into account by adding a connection to the previous hidden state when calculating current hidden state $\boldsymbol{h}_t$ at time t:

$$\boldsymbol{h}_t = RNN(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}) = f(\boldsymbol{W}_{xh}\boldsymbol{x}_t + \boldsymbol{W}_{hh}\boldsymbol{h}_{t-1} + \boldsymbol{b}_h)$$

According to the structure, the hidden state $\boldsymbol{h}_t$ is determined by, in addition to the input, also the last hidden state $\boldsymbol{h}_{t-1}$ as well. This enables the model to learn long-distanced dependency with a variable length of context. The RNNs can be views as a feedforward neural net why we unroll the hidden state over time. Then we can use our back propagation over that unrolled network represented in a computational graph to train it. That version of backpropagation is called backpropagation through time (Rumelhart and McClelland, 1985; Robinson and Fallside, 1987; Werbos, 1988; Mozer, 1995).

There is a serious problem with the RNNs when training them on a long sequence. On the backward pass, the error derivative w.r.t. to the inputs of simple recurrent hidden units can be extraordinarily large or very small close to zero over the time steps. More specifically, the gradient signal will always be multiplied by the same matrix W_{hh} when it is going back through one time step. So it is basically proportional to W_{hh}^N with N is the number of time steps. If all elements in W_{hh} larger than 1, the gradient becomes very large then the

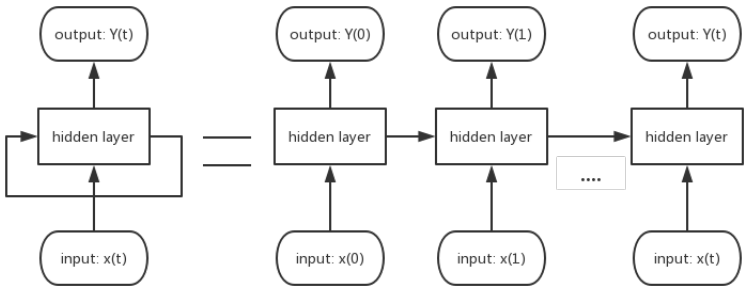


FIGURE 2.9: An unrolled RNN.

exploding gradient occurs. On the other hand if all elements in W_{hh} smaller than 1, the gradient becomes smaller over time then the *vanishing gradient* occurs.

Hochreiter and Schmidhuber (1997) proposed an extended version of RNNs named Long Short-Term Memory (LSTM). LSTM is able to alleviate the vanishing and exploding gradient problems through gating functions:

$$
\begin{aligned}
\boldsymbol{u}_t &= \tanh(\boldsymbol{W}_{xu}\boldsymbol{x}_t + \boldsymbol{W}_{hu}\boldsymbol{h}_{t-1} + \boldsymbol{b}_u) \\
\boldsymbol{i}_t &= \text{sigmoid}(\boldsymbol{W}_{xi}\boldsymbol{x}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \\
\boldsymbol{f}_t &= \text{sigmoid}(\boldsymbol{W}_{xf}\boldsymbol{x}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \\
\boldsymbol{o}_t &= \text{sigmoid}(\boldsymbol{W}_{xo}\boldsymbol{x}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \\
\boldsymbol{c}_t &= \boldsymbol{i}_t * \boldsymbol{u}_t + \boldsymbol{f}_t * \boldsymbol{c}_{t-1} \\
\boldsymbol{h}_t &= \boldsymbol{o}_t * \tanh(\boldsymbol{c}_t)
\end{aligned}
$$

Here $\boldsymbol{c}_t$ is the memory cell at time t and $\boldsymbol{i}_t$, $\boldsymbol{o}_t$, $\boldsymbol{f}_t$ are the input, output and forget gates, respectively. Imagine the extreme example with the above equation where the input gate is closed ($\boldsymbol{i}_t \approx 0$) and the forget gate is fully opened ($\boldsymbol{f}_t \approx 1$), this means the memory cell from the last time step $\boldsymbol{c}_{t-1}$ would be directly copied to the hidden output and therefore maintain the long dependencies without losing the magnitude of the gradient while back propagating the gradient signal, thus prevents vanishing gradient.

Chapter 3

Neural Translation Models

Inspired by the breakthroughs of deep neural networks in computer vision and speech recognition, natural language processing researchers started seeking ways to apply neural-based approaches to their problems. In the area of machine translation, there have been attempts from the research community to replace statistical machine translation (SMT) components by neural models.

In this chapter, we identify several problems with traditional SMT that motivate research on using neural models. Then we propose a neural architecture which proved to be effective on dealing with the issues. Several well-known neural techniques driven by the same motivations are going to be mentioned in the subsequent literature review. Our model and other neural translation models, despite being bounded in the traditional SMT framework, have been the first milestones and playing a crucial role for the eventually popular neural machine translation (NMT) models which follow a different end-to-end paradigm.

3.1 Motivations for Neural Translation Models

While statistical machine translation achieves many successes in the field and has been widely deployed and used in both research and commercial sections, it still holds inherent drawbacks which have prevented its advancement. Improvements become stagnant and require a massive overhaul of the approach to be seen. Those drawbacks have called out for innovative neural translation models able to address them effectively.

The Problem of Local Context. In standard phrase-based statistical machine translation (PBMT), translation adequacy is ensured by the translation model, which is essentially a phrase table estimated by counting phrase pairs from

bilingual corpora. Since the translation units are phrasal segments instead of words, PBMT can exploit some local source and target contexts within those segments. However, the translation model cannot leverage the context information outside the phrases, thus they do not model long-distance dependencies beyond the phrase boundaries, let alone the global context across sentences. The proposed solutions for the problem can be characterized into two categories: reordering approaches and lexical translation models. Pre-reordering approaches (Niehues and Kolss, 2009; Rottmann and Vogel, 2007) learn to change the word order of the source sentences prior to the translation so that the long dependencies have less adverse effect on the induced phrase table. Other reordering models (Herrmann et al., 2013; Durrani et al., 2011) are trained as separated components in the SMT frameworks hoping to provide more context beyond phrases. The disadvantages of those approaches are that they often rely on explicit linguistic knowledge, and that the reordering information they learn cannot fully make up for the lack of global contexts in standard SMT. On the other hand, lexical translation models aim to employ two or more source words, which do not necessarily belong to a phrase, in order to derive translation probability of the target sentence (Hasan et al., 2008; Mauser et al., 2009; Niehues and Waibel, 2013). Nevertheless, the global contexts and long-distance dependencies are fairly complicated to be captured by such simple models.

The Problem of Symbolic Translation. Like other statistical methods for Natural Language Processing, an SMT system is trained to estimate statistical models over discrete random variables such as words, n-grams or phrases. The discrete representations of those symbolic units thus omit useful relationships that exist among them, lead to sparsity issues and decrease the generalization power of the models when applied to other domains. Those relationships includes interconnected information helpful for the translation between the languages, such as morphological, syntactic or semantic of the units in the sentences. For example, the two phrases "*a blue car*" and "*an azure automobile*" should be treated more similarly in translation than the two phrases "*a blue car*" and "*a horse car*". But in SMT, the latter might be estimated wrongly by counting the discrete symbols and discarding the morphological and semantic relationships they might imply. There exists number of methods proposed to fix the problem, such as smoothing techniques or separated models enriching linguistic information. They have, however, their own drawbacks and also do not solve the problem entirely since they still rely on discrete representations.

As we show in the following sections, the neural-based approaches are relevant and powerful enough to find comprehensive solution for the aforementioned problems. Neural architectures have the ability to model complex relationships between inputs and outputs. They are also capable of transforming the discrete space of input features into a continuous space that learns latent connections and shares the global context among different inputs or outputs.

3.2 Proposed Neural Discriminative Word Lexicon

In order to solve the problem of local context and integrate wider contexts into PBMT, we propose a neural translation model which is inspired by earlier works on discriminative word lexicon (Mauser et al., 2009; Niehues and Waibel, 2013). Discriminative word lexicon (DWL) is a specific type of lexical translation models which exploits the occurrence of source words to perform lexical choices of the target words. The original DWL employs a discriminative machine learning framework, called *maximum entropy* or *MaxEnt* (Jaynes, 1957). While DWL has shown that it could improve the translation quality in different conditions, MaxEnt is basically a linear classifier, thus it has limited power in modeling complex dependencies (Berger et al., 1996). On the contrary, hierarchical non-linear classifiers such as neural networks can model dependencies between different source words better since they perform some abstraction over the input. Furthermore, since many pairs of source and target words co-occur quite rarely, a way of sharing global context between different classifiers, which can be offered by a neural architecture, could improve the modeling as well.

3.2.1 Discriminative Word Lexicon

For one sentence pair, DWL deploys many discriminative models in parallel. Each model, given the source words, determines the probability of the event that a specific word t_j from the target language vocabulary V_t is present in the translated sentence: $p(t_j \in \mathbf{t}|\mathbf{s})$. All of the discriminative models $p(t_j \in \mathbf{t}|\mathbf{s})$, $\forall t_j \in V_t$ thus share the features derived from the source sentences.

In DWL models, the source sentence are represented as a bag of indicator features. More formally, a given source sentence $\mathbf{s}$ is represented by the feature

bag $\mathbf{F}(\mathbf{s}) = \{f(s_i) : \forall s_i \in V_s\}$, where V_s is the source vocabulary:

$$f(s_i) = \begin{cases} 1 & \text{if} \quad s_i \in \mathbf{s} \\ 0 & \text{if} \quad s_i \notin \mathbf{s} \end{cases} \tag{3.1}$$

Notice that the source sentence represented in this way loses the word order information. We can include parts of word order information by utilizing n-gram indicators instead of word indicators (Niehues and Waibel, 2013). In this way, local word order can be taken into account, hence, providing more information for the global context. Utilizing n-gram indicators can be done conveniently by extending our n-gram indication features over the **union** of the word (or unigram) vocabulary V_s^1 and higher-n-gram vocabularies V_s^n, $n > 1$. The source vocabulary now becomes $V_s = \uplus_n V_s^n$. In our work, we perform this extension to bigrams and trigrams: $\mathbf{F}(\mathbf{s}) = \{f(s_i) : n = 1..3, \forall s_i \in \uplus_n V_s^n\}$:

$$f(s_i) = \begin{cases} 1 & \text{if} \quad s_i \in \mathbf{s} \\ 0 & \text{if} \quad s_i \notin \mathbf{s} \end{cases} \tag{3.2}$$

The models are trained on examples generated by the parallel training data. The labels for training the classifier of target word t_j are defined as follows:

$$label(t_j|\mathbf{s}, \mathbf{t}) = \begin{cases} 1 & \text{if} \quad t_j \in \mathbf{t} \\ 0 & \text{if} \quad t_j \notin \mathbf{t} \end{cases} \tag{3.3}$$

Considering the advantages of non-linear models mentioned before, in our work, we use a deep neural network to approximate the presence probability $p(t_j \in \mathbf{t}|\mathbf{s})$ of every target word $t_j \in V_t$.

After inducing the presence probability for every word t_j given the source sentence s, with the independence assumption among target words, the probability of the whole target sentence $p(\mathbf{t}|\mathbf{s})$ are calculated simply as follows:

$$p(\mathbf{t}|\mathbf{s}) \approx \prod_{t_j \in V_t} p(t_j|s) \tag{3.4}$$

This $p(\mathbf{t}|\mathbf{s})$ is our proposed lexical translation model, and it is integrated into the PBMT framework like other components.

In Equation 3.4, we need to update the lexical translation score only if a new word appears in the hypothesis. That means we do not take into account the frequency of words but multiply the probability of one word only once even if the word occurs several times in the sentence. Other models in our translation system, however, would hopefully restrict overusing a particular word. Furthermore, to keep track of which words whose probabilities have been calculated already, additional book keeping would be required. To avoid those difficulties, we come up with the following approximation, where J is the length of the target sentence $\boldsymbol{t}$:

$$p(\mathbf{t}|\mathbf{s}) \approx \prod_{j=1}^{J} p(t_j|\mathbf{s}) \tag{3.5}$$

In practice, the DWL score being integrated into PBMT framework is in fact the following **log-prob**:

$$score_{\mathbf{DWL}} = \log p(\mathbf{t}|\mathbf{s}) \approx \sum_{j=1}^{J} \log p(t_j|\mathbf{s}) \tag{3.6}$$

3.2.2 Neural Architecture of Discriminative Word Lexicon

In this section, we describe the neural network architecture that models a DWL for calculating the probabilities $p(t_j|\mathbf{s})$, and we call it **NNDWL**.

The input and output of our neural-based DWL are the source and target sentences from which we would like to learn the lexical translation relationship. We represent each source sentence s as a binary column vector $\hat{\mathbf{s}} \in \{0|1\}^{|V_s|}$ with V_s being the considered vocabulary of the source corpus. If a source n-gram s_i appears in that sentence s, the value of the corresponding index i in $\hat{\mathbf{s}}$ is 1, and 0 otherwise. Hence, the source sentence representation is a sparse vector, depending on the considered vocabulary V_s. The same representation scheme is applied to the target sentence t to get a sparse binary column vector $\hat{\mathbf{t}}$ with the considered target vocabulary V_t.

As the Figure 3.1 depicts, our main neural network-based DWL architecture for learning lexical translation is a feed-forward neural network with three hidden layers. The matrix $\mathbf{W}^{(1)} \in \mathbb{R}^{V_s \times |H_1|}$ connects the input layer to the first hidden layer. Two matrices $\mathbf{W}^{(2)} \in \mathbb{R}^{|H_1| \times |H_2|}$ and $\mathbf{W}^{(3)} \in \mathbb{R}^{|H_2| \times |H_3|}$ encodes the learned translation mapping between two compact global feature

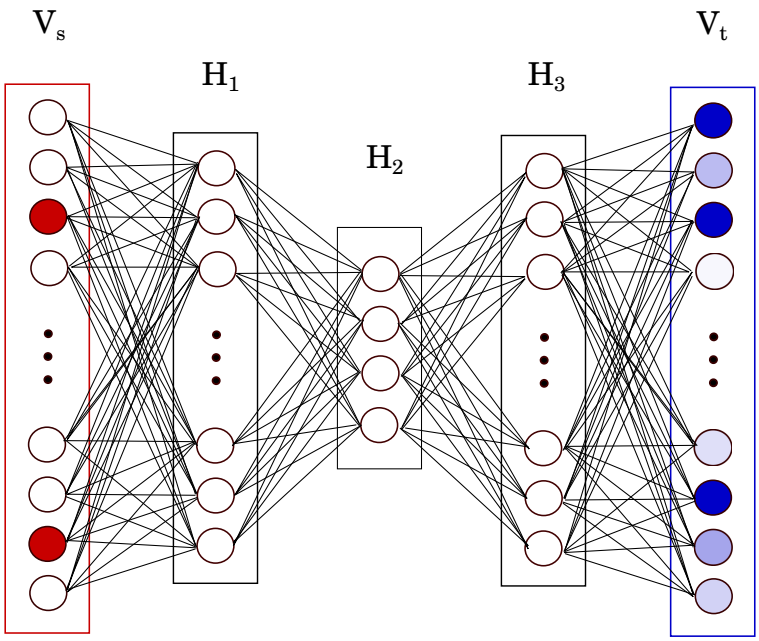


FIGURE 3.1: Neural Architecture for Learning Lexical Translation

spaces of the source and target contexts. And the matrix $\mathbf{W}^{(4)} \in \mathbb{R}^{|H_3| \times |V_t|}$ computes the lexical translation output. $|H_1|$, $|H_2|$, and $|H_3|$ are the number of units in the first, second and third hidden layers, respectively. The lexical translation distribution of the words in the target sentence $p(t_i|\mathbf{s})$ for a given source sentence s is computed by a forward pass:

$$p(t_i|\mathbf{s}) = \sigma({\mathbf{W}^{(4)}}^T \mathbf{O}^{(3)})$$

where:

$$\mathbf{O}^{(k)} = \left[\sigma({\mathbf{W}^{(k)}}^T \mathbf{O}^{(k-1)})\right] \quad k \in \{1,2,3\}$$
$$\mathbf{O}^{(0)} = \hat{\mathbf{s}} \quad \text{and} \quad \mathbf{O}^{(4)} = \mathbf{p}(\hat{\mathbf{t}}|\hat{\mathbf{s}})$$

and σ is the *sigmoid* function:

$$\sigma(x) = \frac{1}{1 + \mathbf{e}^{-x}}$$

So the parameters of the network are:

$$\theta = (\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{W}^{(3)}, \mathbf{W}^{(4)})$$

3.2.3 Network training

In neural network training, for each training sentence pair $(\boldsymbol{s}, \boldsymbol{t})$, we maximize the similarity between the conditional probability $p_i = p_\theta(t_i|s)$ to either 1 or 0 depending on the appearance of the corresponding word t_i in the target sentence $\boldsymbol{t}$. The neural network operates as a multivariate classifier which gives the probabilistic score for a binary decision of independent variables, i.e the appearances of target words. Here we minimize the cross entropy error function between the binary target sentence vector $\hat{\mathbf{t}}$ and the output of the network $\mathbf{p} = \left[p_i\right]$:

$$E = -\frac{1}{V_t}\sum_{i=1}^{V_t}(\hat{\mathbf{t}}_i \ln p_i + (1-\hat{\mathbf{t}}_i)\ln(1-p_i))$$

We train the network by back-propagating the error based on the gradient descent principle. The error gradient for the weights between the last layer and the output is calculated as:

$$\frac{\partial E}{\partial w_{ij}^{(4)}} = (\mathbf{O}_j^{(4)} - \hat{\mathbf{t}}_j)\mathbf{O}_i^{(3)}$$

The error gradient for the weights between the other layers is calculated based on the error gradients for activation values from the previous layers:

$$\frac{\partial E}{\partial w_{ij}^{(k)}} = \frac{\partial E}{\partial \mathbf{O}_j^{(k)}}\mathbf{O}_i^{(k-1)}$$

Then the weight matrices are batch-updated after each epoch:

$$\mathbf{W}^{(k)}[T+1] = \mathbf{W}^{(k)}[T] - \eta\sum_{i=1}^{N}\frac{\partial E}{\partial \mathbf{W}^{(k)}}$$

$$\frac{\partial E}{\partial \mathbf{W}_{(k)}} = \sum_{i=1}^{n}(\mathbf{y}_{(k)}^i - \hat{\mathbf{t}}_i)\mathbf{y}_{(k)}^i(1-\mathbf{y}_{(k)}^i)\mathbf{x}_{(k)}^i$$

$$\frac{\partial E}{\partial \mathbf{W}_{(k)}} = \sum_{i=1}^{n}(\mathbf{y}_{(k)}^i - \hat{\mathbf{t}}_i)\mathbf{y}_{(k)}^i(1-\mathbf{y}_{(k)}^i)\mathbf{y}_{(k-1)}^i,$$

where:

- N is the number of training instances.

- η is the learning rate of the network.
- $\mathbf{W}^{(k)}[T+1]$ is the weight matrix of the layer k after $T+1$ epochs of training.

3.2.4 Experiments

In this section, we describe the configurations for our **NNDWL** and experiments we conducted on three different language pairs: English⇒French, German⇒English and English⇒Chinese.

Data. The training, validation and test data for **NNDWL** in each language pair of interest is derived from TED corpus (Cettolo et al.). The statistics of those data are shown in Table 4.2.

		En-Fr	**En-Zh**	**De-En**
Training	Sent.	149991	140006	130654
	Tok. (avg.)	3.1m	3.3m	2.5m
Validation	Sent.	6153	8962	7430
	Tok. (avg.)	125k	211k	142k

TABLE 3.1: Statistics of the corpora used to train NNDWL.

Baselines. For each language pair, we compare our **NNDWL** with two baseline systems. The first one is a strong PBMT baseline without any DWL model. This was trained using large, out-of-domain corpora (EPPS, NC and CommonCrawl) and adapted to TED[1]. The phrase translation probabilities are extracted using the scripts from Moses (Koehn et al., 2007). Moses is also used for extracting distortion and lexicalized reordering scores. Several language models, including a bilingual language model as described in Niehues et al. (2011) as well as a cluster language model based on word classes generated by the MKCLS algorithm (Och, 1999), are integrated. Short-range pre-reordering (Rottmann and Vogel, 2007) is performed in the systems of English→French and German→English. Those baseline systems achieved the best performances in the IWSLT'12 campaign (Mediani et al., 2012; Federico et al., 2012). The second baseline is the same PBMT system with the DWL model trained using MaxEnt.

NNDWL architecture. Our **NNDWL** models are integrated into the strong PBMT baseline as a feature. One system features word-based (or unigram)

[1]Excepts English→Chinese system which was trained only on TED.

NNDWL, i.e. $V_s = V_s^1$, the other one ulilizes bigrams and trigrams as well. Since it is very expensive to calculate DWLs for all the words, we limit the source and target vocabularies to the most frequent ones. All words outside the lists are treated as unknown words. More specifically, V_s^1 and V_t contains the 2000 most-frequent words from the source and target languages, respectively. The bigram and trigrams of the source vocabulary, V_s^2 and V_s^3, are 500 and 200 most-frequent bigrams and trigrams, respectively. The size of our three hidden layers: $|H_1| = 1000$, $|H_2| = 500$, $|H_3| = 1000$. For training our proposed architecture, minibatch gradient descent with a batch size of 15 and a learning rate of 0.02 is used. Gradients are calculated by averaging across a minibatch of training instances and the process is performed for 35 epochs. After each epoch, the current neural network model is evaluated on a separate validation set, and the model with the best performance on this set is utilized for calculating lexical translation scores afterwards. We regularize the models with the L_2 regularizer. The training is done on GPUs using the Theano Toolkit (Bergstra et al., 2010).

Results on English→French. Here we report the results using different NNDWL configurations mainly for an English→French translation system (Table 3.2).

System (En→Fr)	BLEU	ΔBLEU
Baseline	31.94	–
MaxEnt DWL	32.17	+0.23
NNDWL 500	32.06	+0.12
NNDWL 2000	32.38	+0.44
NNDWL 2000 SC-500-200	**32.44**	**+0.50**

TABLE 3.2: Results of the English→French NNDWL.

Varying the vocabulary sizes for both source and target sentences not only helps to dramatically reduce neural network training time but also affects the translation quality. In our experiments, the network with 2000-most-frequent-word vocabularies show a large improvements with around 0.32 BLEU points better than the network with 500-most-frequent-word vocabularies. 500 might not cover enough helpful information for NNDWL. The network with 2000-most-frequent-word vocabularies also perform better than the DWL using the maximum entropy approach, showing a better context sharing mechanism and the ability to model more complicate relationships between source and target words. Furthermore, providing local contexts with 500 most-frequent bigrams and

200 most-frequent trigrams, we achieve the best improvements of 0.5 BLEU points over the baseline.

When we look at some translated samples from the models, we notice that our model suggests better word choices compared to the baseline and the maximum entropy DWL (Table 3.3). For example, our model can take the context from the phrase "*where do we run*" to translate "*run away*" into "*échapper*" ("*escape*") instead of "*laissé tomber*" ("*gave up*") like other models.

Source	So we run away from it , and where do we run ?
Reference	Alors on s'en écarte . Pour aller où ?
Baseline	Nous avons donc laissé tomber , et où ?
Engl. gloss	We gave up, and where ?
MaxEnt DWL	Nous avons donc laissé tomber , et où nous ne courons ?
Engl. gloss	We then gave up , and where don't we run ?
Our NNDWL	Nous y *échapper* et *pour aller où* ?
Engl. gloss	We *escaped* from it and *where do we run* ?

Source	and this resulted in the first soft - surface character , CG animation that was ever in a movie .
Reference	Et cela a donné le premier personnage électronique en image de synthèse qu'on ait jamais vu dans un film .
Baseline	A la première surface douce de caractère , CG animation qui était dans un film .
Engl. gloss	For the first time a soft surface of character , CG animation that was in a movie .
MaxEnt DWL	A la première surface douce-personnage , CG animation qui était dans un film .
Engl. gloss	For the first surface soft character , CG animation that was in a movie . ?
Our NNDWL	*Et cela a entraîné le premier personnage de surface douce* , animation CG qui était dans un film .
Engl. gloss	*and this has resulted in the first character of soft surface* , CG animation that was in a movie .

TABLE 3.3: Examples show that NNDWL produced better word choices.

We also trained our NNDWL models on a bigger corpus concatinating different-domain corpora and TED with the size 15 times bigger than TED corpus. The results in Table 3.4 show that using a bigger corpus does not improve the translation quality. The DWL models trained on in-domain data only, i.e. TED, perform similar or better than the models trained on more

data but broader domains. This observation also holds true for the maximum entropy models reported in Ha et al. (2013).

System (En→Fr)	BLEU	ΔBLEU
Baseline	31.94	–
NNDWL on TED	**32.44**	**+0.50**
NNDWL on EPPS+NC+TED	32.33	+0.39

TABLE 3.4: Results of the NNDWL trained on different-size corpora.

Results on other language pairs. We conducted the experiments with NNDWL models mainly on our English→French translation system in order to investigate the impact of our method on a strong baseline. However, we would also like to inspect the effect of the DWL on language pairs with long-range dependencies or differences in word order. For that purpose, we built similar NNDWL models and integrate them to our translation systems for other language pairs.

Tables 3.5 and 3.6 show the results of our NNDWL in German→English and English→Chinese translation directions, respectively.

System (De→En)	BLEU	ΔBLEU
Baseline	29.70	–
MaxEnt DWL	29.95	+0.25
NNDWL 500	29.82	+0.12
NNDWL 2000 SC-500-200	**30.04**	**+0.34**

TABLE 3.5: Results of the German→English NNDWL.

System (En→Zh)	BLEU	ΔBLEU
Baseline	17.18	–
MaxEnt DWL	16.78	-0.40
NNDWL 500	17.09	-0.09
NNDWL 1000	17.58	+0.40
NNDWL 1000 SC-200-100	**17.63**	**+0.45**

TABLE 3.6: Results of the English→Chinese NNDWL

We can see the similar improvements confirming the effects of our NNDWL. In case of the German→English direction, the NNDWL also helps to gain 0.34 BLEU points over the baseline with the best model. However, the improvements is not notably different compared to the original MaxEnt

DWL. In case of the English→Chinese direction, the NNDWL significantly improves the translation quality, with an increment of 0.45 BLEU points over the baseline. Interestingly, for this direction, we need smaller number of most frequent words and n-grams in our vocabularies to achieve the best performance. We speculate that because a small number of basic Chinese words could cover broader meaning than in other European languages.

3.2.5 Discussion of NNDWL

In this section, we described our proposed neural translation model which employ a deep neural network to model the global context of source sentence in order to predict the target words. We show that using a feedforward achitecture we can directly model the translation relationship between source and target sentences. Due to the lack of explicit position and word order information, our model cannot work alone but needs to be integrated into a standard PBMT system so that other models can cover the lacking information. However, our model shows a potential direction that a simple neural architecture with suitable extensions could be used directly as an end-to-end machine translation framework.

3.3 Related Work on Neural Translation Models

In this section, we will go over shortly other popular neural translation models which are able to cover word order information by extending the neural language model idea to translation models. But they can barely stand alone without the PBMT frameworks since they remain limiting themselves in translation local contexts and data sparsity issues bound by n-gram approaches.

First, let us recall the feedforward architecture used in neural n-gram language models. The language probability of the word s_i, $p(s_i|s_{1:i-1})$, is estimated by the probability of that word given $n-1$ previous words: $p(s_i|s_{1:i-1}) \approx p(s_i|s_{i-n+1:i-1})$. This n-gram language probability is modeled by a feedforward architecture, whose inputs are concatenated by the one-hot vectors of $n-1$ previous words. It often starts with a shared projecting layer, following by one or several non-linear hidden layers, and finally a softmax layer to model the distribution of the word s_i.

Obviously, this feedforward architecture also works when modeling more general probabilistic distributions like $p(x_i|h_{1:i-1})$ or $p(x_i|h_{i-n+1:i-1})$, in which x_i is the output of time step i and $h_{i-n+1:i-1}$ is some corresponding *history* up to the current time step i. For convenience, let us call this feedforward architecture a *continuous space model* (CSM). CSM suffers from the problem of large softmax, where the last softmax output layer is costly due to a large number of words in the vocabulary. As we have seen in Section 2.2.5, several solutions for this problem have been proposed, including hierarchical softmax or noise contrastive estimation. A less severe problem of CSM also comes from the large vocabulary size: the first layer of CSM is often a big matrix and therefore memory-intensive. Those problems are known as the inherent problems of CSM, thus, they are the common problems that any CSM-based translation model needs to face.

3.3.1 Continuous Space Translation Model

Schwenk (2012) proposed a continuous space translation model (CSTM) to calculate the probability of a phrase pair $(\bar{\boldsymbol{t}}, \bar{\boldsymbol{s}})$. In order to estimate $p(\bar{\boldsymbol{t}}, \bar{\boldsymbol{s}})$, they assume the target words t_j in $\bar{\boldsymbol{t}}$ are independent to each others when conditional to the source phrase $\bar{\boldsymbol{s}}$:

$$p_(\bar{\boldsymbol{t}}|\bar{\boldsymbol{s}}) = \prod_j p(t_j|\bar{\boldsymbol{s}})$$

Then they use the CSM to estimate n target word probabilities $p(t_j|\boldsymbol{s})$ at the same time:

$$p(t_j|\bar{\boldsymbol{s}}) \approx p(t_j|h_{\bar{\boldsymbol{s}}})$$

where $h_{\bar{\boldsymbol{s}}}$ is a sequence of n source words in the phrase $\bar{\boldsymbol{s}}$. In their CSM network, n is set to 7. The considering phrase pairs belong to the phrase table extracted by their SMT framework from parallel corpora. Incomplete phrase pairs whose either source phrase or target phrase is shorter than n words are zero-padded in order to make the CSM works.

While there exist several differences, we observe that CSTM shares some similarities with our NNDWL: First, it models a fixed number of source and target words (their n is 7 and ours is $|V_s| = |V_t| = 2000$). Second, it relies on the same independent assumption of the target words in the sequence. Although they argue that the common hidden layer forces the target words to learn a distributed representation, thus, re-introducing some dependencies among

them, this reasoning has not been supported by any concrete experiment and result.

As other translation models, CSTM are integrated into PBMT systems under two schemes: 1) as a feature score in the framework, and 2) through n-best list rescoring.

3.3.2 Continuous Space Bilingual Language Model

Le et al. (2012) used the same n-gram approach to calculate $p(\bar{t}, \bar{s})$:

$$p(\bar{t}, \bar{s}) \approx \prod_j p(t_j | \bar{h}_{i-n+1:i-1})$$

Unlike Schwenk (2012), they introduced an elegant way for the CSM to avoid the naive independence assumption: instead of some history over $n-1$ source words, $\bar{h}_{i-n+1:i-1}$ denotes the history over $n-1$ bilingual tuples (s_k, t_k), $k = i-n+1..i-1$. Now they can use the standard CSM to model the language model of those *bilingual units* without any change in the architecture. Indeed, the tricks in their studies are mostly to deal with the inherent problems of a neural language model implemented by CSM architectures.

One problem of the Continuous Space Bilingual Language Model (CSBiLM) is that the data sparsity becomes quadratically more severe since now it happens to bilingual tuples. To reduce the adverse effect of the problem, bilingual units (s_k, t_k) are extracted from the phrase table in a synchronized way, meaning that the number of source and target words in an n-gram windows are identical to each other and to n.

By this way, CSBiLM explicitly models $p(\bar{t}, \bar{s})$ as well as the dependencies among the target words in the phrase $\bar{t}$. They conducted the experiments where CSBiLM is utilized as a feature in their PBMT system and achieve significant improvements. On the other research, Ha et al. (2015b) used CSBiLM in n-best list rescoring and it also helps to produce better translations.

3.3.3 Neural Network Joint Translation Model

Neural Network Joint Translation Model (NNJM), proposed by Devlin et al. (2014), is another CSM-based translation model that keeps itself free from the independence assumption of target words when directly calculating the

translation model $p(\boldsymbol{t}, \boldsymbol{s})$ of a target sentence $\boldsymbol{t}$ given a source sentence $\boldsymbol{s}$:

$$p(\boldsymbol{t}, \boldsymbol{s}) \approx \prod_{j=1}^{|T|} p(t_j | t_{j-1}, t_{j-2}, .., t_{j-n+1}, \bar{\boldsymbol{s}}_j)$$

Here $\bar{\boldsymbol{s}}_j$ is an m-size window of source sentence affiliated with the current target word t_j. Devlin et al. (2014) define the affiliation as a sequence centering around the source word aligned to t_j. Note that $m \neq n$, so NNJM eases the synchronized constraint of bilingual tuples in case of CSBiLM. Furthermore, this CSM architecture takes $n-1$ previous target words and m source words as its direct inputs, thus, NNJM learns the dependencies among target words and the translation between source and target sentence jointly and more explicitly than CSBiLM.

Similar to Le et al. (2012), many practical techniques in Devlin et al. (2014) are presented in order to cope with the inherent issues of CSM architectures and make the training significantly faster. Due to its advantages in both conceptual models and speeding-up techniques, NNJM famously achieves considerable improvements on Arabic→English and Chinese→English. Additionally, since NNJM considers both previous target words and relevant source information in order to predict the next target word at a time, it allegedly inspires the studies of the neural architectures able to perform machine translation end-to-end.

3.4 Neural Machine Translation

When neural machine translation (NMT) models first appeared, the research community considered them as a neural translation model since it also estimates the translation probabilities using neural methods and then integrated into an SMT system either directly as a sub-component or through n-best rescoring (Kalchbrenner and Blunsom, 2013; Cho et al., 2014). However, NMT models differ from other neural translation models in several ways.

The models from Schwenk (2012), Le et al. (2012) and Devlin et al. (2014) are limited to fixed-size source and target phrases, while NNDWL, even though does not suffer from this limitation, lacks of word order information. On the other hand, NMT models first try to encode the whole variable-length source sentence into a fixed-length vector, then use that encoded source vector to generate target words, once at a time (similar to Devlin et al. (2014)

strategy). This kind of encoder-decoder approach has shown promising foundations to address the limitations of other neural translation models and more importantly, to tackle the inherent drawbacks of PBMT.

First, NMT utilizes the global context of the whole source sentence instead of local contexts in order to produce the translation. Second, as other neural models, it operates on continuous representations of words and phrases, thus, overcoming sparsity issues and being able to share syntax and semantic information hidden from the discrete inputs. Third, it leverages the power of neural networks in modeling complex relationships, including long-distance dependencies, among source words and between the source and target sentences. Furthermore, the tranlation process is carried out from the source sentence to the target sentence in an *end-to-end* manner where NMT learns the translation models, language models and reordering models jointly.

Because of its importance not only to this thesis but also to the machine translation area, we devote this section to describe neural machine translation. We approach it as an encoder-decoder framework to solve machine translation problem. However, this approach can be applied to solve a broader class of machine learning problems: *sequence-to-sequence* modeling. This class refers to any model that maps one sequence to another, in which machine translation is one of the primary instances.

3.4.1 Encoder-Decoder Framework

In NMT, encoder-decoder framework consists of an encoder which models the characteristics of a source sentence and a decoder responsible to produce most probable words in some order.

The encoder is a neural architecture which reads every words x_i of a source sentence $\mathbf{x} = \{x_1, ..., x_n\}$ and encodes a representation of the sentence into a fixed-length vector $\mathbf{c}$, called the *context vector*. In the encoder, a discrete representation of input words is usually transformed into a continuous space of the source language, called embedding space, prior to the next layers. The only requirement for encoder architecture is the capability to deal with variable-length inputs.

The decoder, which is another neural architecture, generates one target word every time step to form a translated target sentence $\mathbf{y} = \{y_1, ..., y_m\}$ in the end. It works the similar way as the language model previously mentioned.

However, in addition to the information from previous generated sequence $\mathbf{y}' = \{y_1, ..., y_{j-1}\}$, the decoder is also conditioned on the context vector $\mathbf{c}$, which encodes the source sentence, to produce the next target word y_j at the time step j.

The general encoder-decoder framework for machine translation is depicted in Figure 3.2. In the original NMT framework, Sutskever et al. (2014) and Cho et al. (2014) utilize recurrent neural networks to model source and target sentences, each word corresponds to a hidden state in the encoder ($\boldsymbol{h}_i$ with $i = 1..n$) or decoder ($\boldsymbol{z}_i$ with $j = 1..m$). Typical architectures for the encoder and decoder are described more detailed in Section 3.4.3.

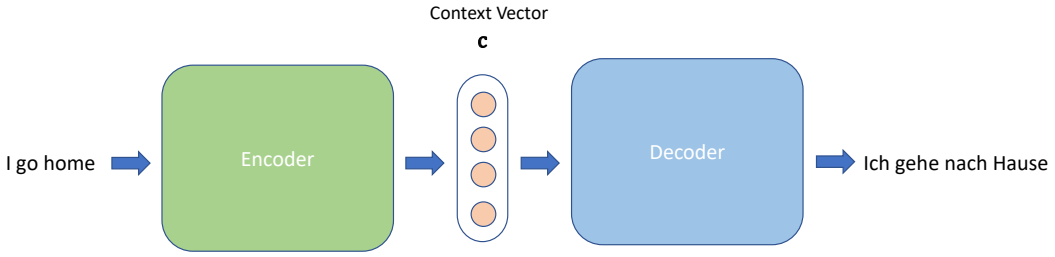


FIGURE 3.2: General encoder-decoder architecture for NMT.

3.4.2 Attention mechanism

One problem of the encoder-decoder framework lies in the way the context vector $\mathbf{c}$ is derived. After the encoder reads the whole source sentence, it comes up with the context vector $\mathbf{c}$, and then the decoder starts using $\mathbf{c}$ to generate target words. It is obviously inefficient to encode the source sentence into a fix-length vector regardless to how long the sentence is. Although the source context is considered when the decoder generates the translation at some time step, it would also be "disconnected" to a specific state of the source sentence at that moment.

Bahdanau et al. (2014) proposed to include an *attention mechanism* into the encoder-decoder framework, which takes the source states specific to different time steps into account when calculating the source context vector and generating an output. In other words, the decoder would look at different states $\boldsymbol{h}_i$, each state corresponding to a source word, and decide to put its "attention" on certain words in order to help decide to generate the target word at this step. Thus, the context vector is not constant across time steps, now denoted $\boldsymbol{c}_j$, but it is calculated on-the-fly by weightedly combining

different states from the source sentence. Intuitively, the weights would depend upon some "relevance" $rel(\boldsymbol{z}_{j-1}, \boldsymbol{h}_i)$ between the source words $\boldsymbol{h}_i$ and the previously-generated target word $\boldsymbol{z}_j$. Bahdanau et al. (2014) used *an addictive attention* which is implemented by an MLP. On the other hand, the formular 3.4.2 below shows a simpler alternative called *dot-product attention* (Luong et al., 2015b). This learnable attention mechanism is employed as another neural architecture in the encoder-decoder framework and it is trained end-to-end along with the whole framework:

$$
\begin{aligned}
rel(\boldsymbol{z}_{j-1}, \boldsymbol{h}_i) &= \boldsymbol{z}_{j-1}^T \cdot \boldsymbol{h}_i \\
\boldsymbol{\alpha}_{ij} &= \text{softmax}(rel(\boldsymbol{z}_{j-1}, \boldsymbol{h}_i)) \\
\boldsymbol{c}_j &= \sum_i \boldsymbol{\alpha}_{ij} \boldsymbol{h}_i
\end{aligned}
$$

Cheng et al. (2016) and Lin et al. (2017) used a similar type of attention in sentence modeling, called *self attention*. Self attention refers to the mechanism that finds the relevance inside a sentence in the process of modeling the sentence itself. Vaswani et al. (2017) extended both types and formalized them to as a unique attention mechanism. They then used them in the NMT's encoder-decoder framework to model the source and target sentences as well as to conduct the original attention suggested by Bahdanau et al. (2014).

Attention according to Vaswani et al. (2017) formalization is a way to ask a model to focus on relevant information between queries q and a set of key-value pairs (k, v). In the original attention, each query is a decoder state at some time step, and set of key-value pairs (k, v) are all the encoder' states ($k \equiv v$), and the relevance scores between source and target states define the weighted scheme when combining the source states into the context vector. In the self attention used to model a sentence, each query q corresponds to a current considering state of that sentences, the set of key-value pairs (k, v) are all the other words, and the relevance scores define some hidden relationship that we would like to learn between that word and other words. Vaswani et al. (2017) used a simple, feed-forward neural architecture to realize their attention. It is called Scaled Dot-Product Attention (Figure 3.3):

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

Excepts for the scaling factor $\sqrt{d_k}$, where d_k is the dimension of the keys k,

this attention is identical to the original dot product attention (Luong et al., 2015b).

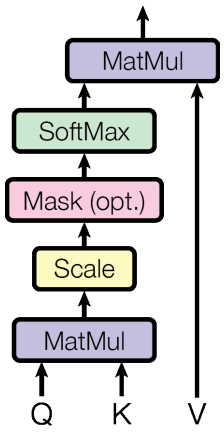


FIGURE 3.3: General Scaled Dot-Product Attention (Vaswani et al., 2017).

3.4.3 NMT architectures

Recent NMT systems usually implement the aforementioned encoder-decoder framework with attention. Basically they differ only with regards to the specific architectures employed in the encoder and decoder. It is necessary for those architectures to have the ability of modeling variable-length inputs and long-distance dependencies between them. We will describe such architectures in this section.

Recurrent-based architecture. Recurrent neural networks are able to naturally deal with sequences having different lengths. Furthermore, advanced recurrent units such as Long Short-Term Memory (Hochreiter and Schmidhuber, 1997) or Gated Recurrent Unit (Cho et al., 2014) have better memorizing mechanism to capture the relationships between words positioned in distance to each others. In the most popular NMT to date, Bahdanau et al. (2014) employed recurrent architecture for both the encoder and decoder.

In the encoder, the hidden state $\boldsymbol{h}_i$ is computed by taking into account the current word's embeddings $\boldsymbol{s}_i$ and the hidden state of the previous word $\boldsymbol{h}_{i-1}$. $\boldsymbol{h}_i$ encodes the source sentence up to the time i from both forward and

backward directions.

$$\begin{aligned}
\boldsymbol{h}_i &= [\overrightarrow{\boldsymbol{h}}_i, \overleftarrow{\boldsymbol{h}}_i] \\
\overrightarrow{\boldsymbol{h}}_i &= \text{Recurrent}(\overrightarrow{\boldsymbol{h}}_{i-1}, \boldsymbol{s}_i) \\
\overleftarrow{\boldsymbol{h}}_i &= \text{Recurrent}(\overleftarrow{\boldsymbol{h}}_{i+1}, \boldsymbol{s}_i)
\end{aligned}$$

The decoder calculates the target hidden state $\boldsymbol{z}_j$ based on the previous hidden state of the decoder $\boldsymbol{z}_{j-1}$, the embeddings of the previous target word $\boldsymbol{t}_{j-1}$ and the time-specific context vector $\boldsymbol{c}_j$ as inputs to calculate the current hidden state $\boldsymbol{z}_j$. At the very end of the decoder, like other architectures, a softmax is usually applied to yield a distribution over the target word y_j

$$\begin{aligned}
\boldsymbol{z}_j &= \text{Recurrent}(\boldsymbol{z}_{j-1}, \boldsymbol{t}_{j-1}, \boldsymbol{c}_j) \\
p(y_j|\boldsymbol{x}) &= \text{Softmax}(\boldsymbol{z}_j)
\end{aligned}$$

In recent NMT architectures, the encoder and decoder are constructed by stacking several recurrent layers, and residual connections (He et al., 2016) are added between them. Figure 3.4 shows the recurrent NMT architecture with original attention mechanism.

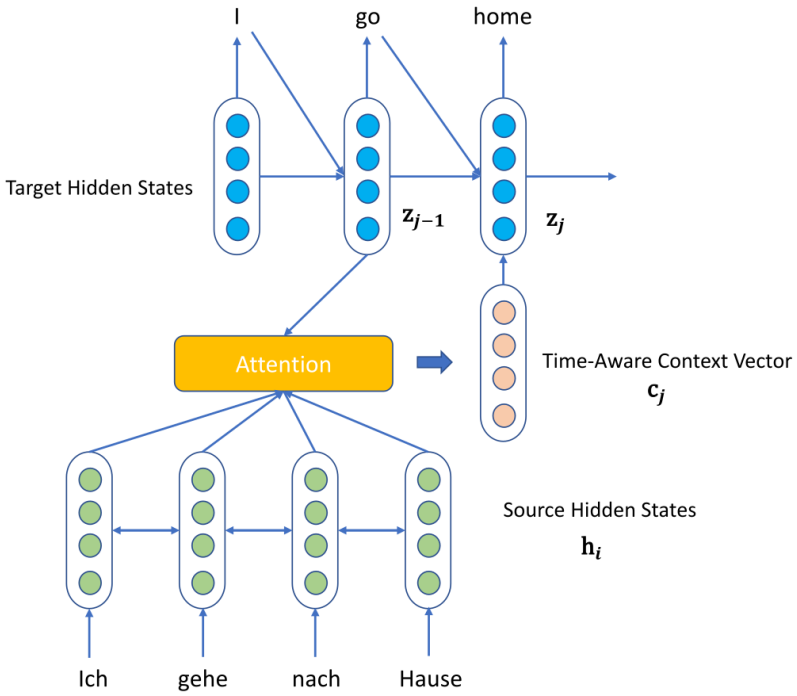


FIGURE 3.4: Recurrent NMT architecture with attention.

Convolutional architecture. Convolutional Neural Networks (CNNs) or their preceding, Time-Delay Neural Networks (TDNNs), are a special kind of feedforward neural networks which can effectively deal with variable-length sequences. Instead of using a fully-connected layer, CNNs or TDNNs run a smaller layer with its weights $\boldsymbol{W}$ via a sliding windows of size k over the input sequence $\boldsymbol{S}$:

$$\boldsymbol{h}_i = \mathrm{f}(\boldsymbol{W}^T \cdot \boldsymbol{S}_{i-k+1:i})$$

Here f is a non-linear activation function. The convolution operation is applied l times to get $\boldsymbol{h}_i^l$, where $^{\boldsymbol{S}}$ in the Equation 3.4.3 is the concatenation of the outputs $\boldsymbol{h}_i^{l-1}$ from previous convolutional step. With suitable paddings, the number of outputs after the last convolutional step is equal to the length of input sequence. Those final outputs, in case of the convolutional encoder, are actually source hidden states $\boldsymbol{h}_i$, similar to the ones from recurrent architectures.

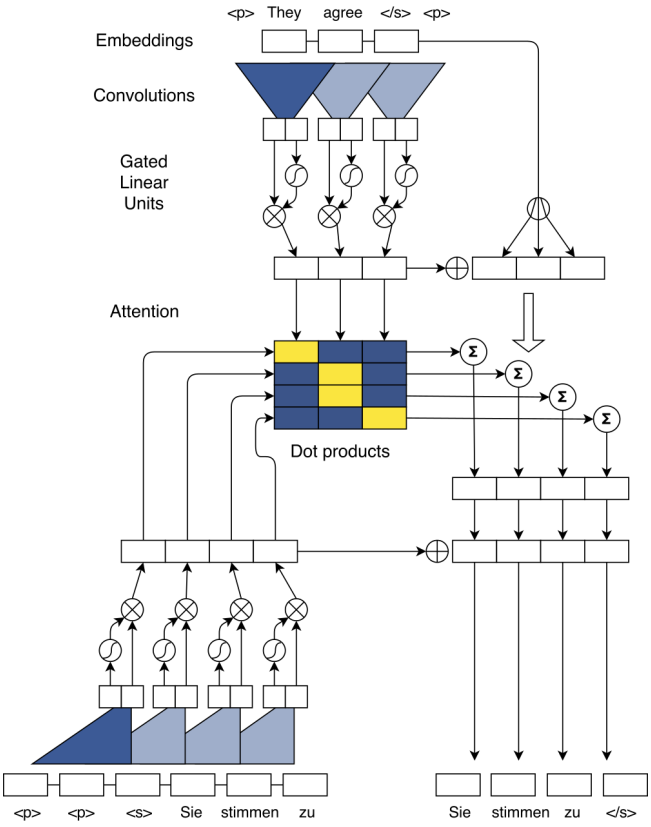


FIGURE 3.5: Convolutional NMT architecture (Gehring et al., 2017).

Kalchbrenner and Blunsom (2013) and Gehring et al. (2016) proposed a hybrid architecture where they employed CNNs for the encoder, and stayed with

a recurrent decoder, which is in principle identical to the one mentioned in Section 3.4.3. The reason to use a recurrent decoder is because in the inference step, we cannot utilize any information from the future when calculating hidden state z_j and predicting the next word y_j, and we want to do the same in training. In order to overcome that problem and use a convolutional decoder, a practical trick is presented: We mask the inputs of any convolutional step by zero vectors and remove the corresponding future outputs.

One issue of convolutional architecture is that the hidden states h_i, after convolutions, lose the position information of input words. The solution is to integrate positional embeddings (Gehring et al., 2017) into those hidden states. By that way, the model has a sense about which part of the sequence it is currently considering.

An important advantage of convolutional architectures over recurrent architectures lies in the fact that CNNs are feedforward architectures which can be highly parallelized. Therefore, it takes much less time to train a convolution NMT system compared to a recurrent one having similar capacity. Figure 3.5 shows the NMT architecture which fully employed CNNs in both encoder and decoder (Gehring et al., 2017).

Transformer architecture. Transformer is an NMT architecture whose encoder and decoder are realized by the scaled dot-product attention described in Section 3.4.2 (Vaswani et al., 2017). The encoder of Transformer architecture consists of a few blocks which features self-attention layers as the main component. The decoder of this architecture stacks several blocks which are also attention-based. Similar to the convolutional NMT, the decoder of Transformer is masked to prevent using future information in the current time step of prediction. Obviously, Dot-Product Attention also takes parts of the attention mechanism lies between the encoder and decoder.

The major difference of Transformer architecture from other NMT architectures is *multi-head attention*. Each attention component in the architecture incorporates several Dot-Product Attention channels, where each channel attends to different representations of the inputs. In other words, multi-head attention allows the component to learn a variety of relationships at the different positions of the inputs. At the end, those channels are concatenated and projected to produce the final outputs of a multi-head attention component (Figure 3.6).

Beside the multi-head attention layer, each transformer block also contains

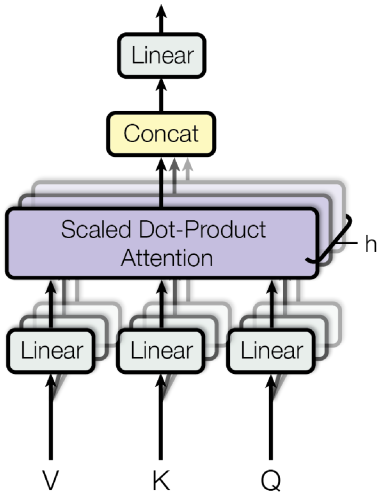


FIGURE 3.6: Multi-Head Attention (Vaswani et al., 2017).

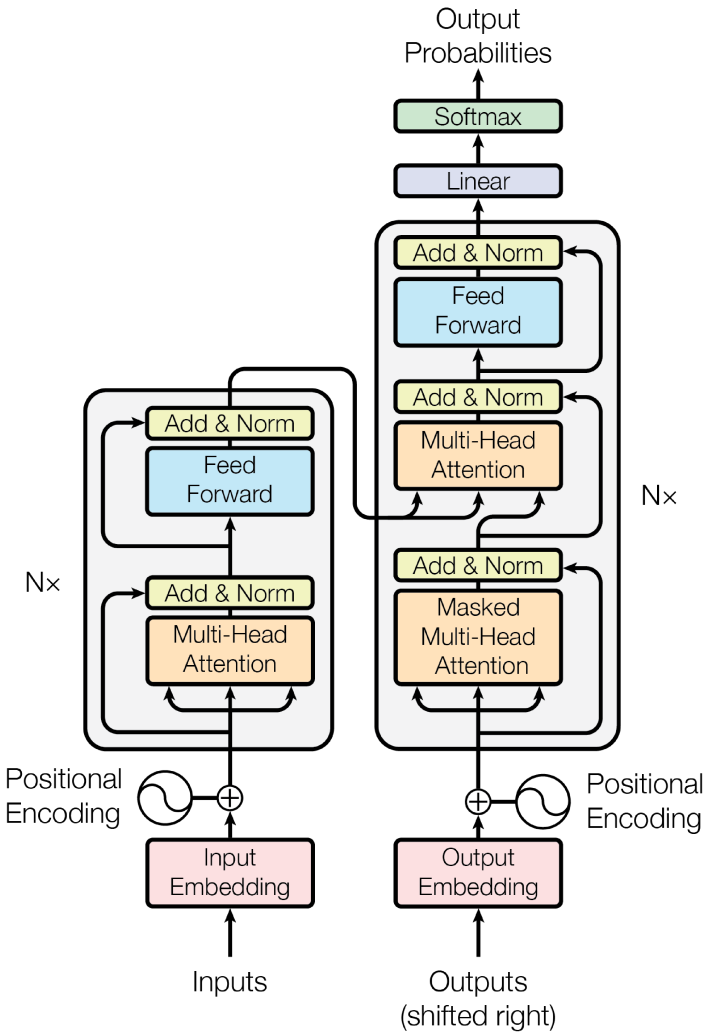


FIGURE 3.7: Transformer Architecture for NMT (Vaswani et al., 2017).

other layers, which features layer normalization (Ba et al., 2016) and residual connections (He et al., 2016). The input of the encoder and the decoder are the embeddings of the source and target sentences, respectively. The embeddings includes word embeddings and positional encoding, similar to the convolutional architecture. Because all components of Transformer architecture are feedforward, the architecture enjoys the advantage of being highly parallelized and fast training time. The whole transformer NMT is shown in Figure 3.7.

As we have seen, from recurrent architecture to transformer, NMT frameworks become more complicated and have higher performance. The recurrent NMT is still the most popular architecture used in research and applications, however, transformer architecture, inspired by and similar in many ways to the convolutional one, is catching fast due to its state-of-the-art performances in many systems (Vaswani et al., 2017; Junczys-Dowmunt et al., 2018; Popel, 2018; Pham et al., 2018).

Chapter 4

Multilingual Neural Translation

As can be seen from the previous chapters, the main philosophy of SMT is revolving around using learning and combining individual components in a log-linear framework. Initially these sub components are statistically based models over discrete units, such approach struggles to represent the relationship between those units, for example words with the same lemma and different morphological forms. Neural network-based components improved by learning distributed representation of the discrete units (also called as continuous space representation). As a result, connections between words (or phrases) can be established mathematically and deep neural architectures can learn complex structured relationships that outmatches conventional count-based methods used in statistical models.

Since SMT systems treat the translation units as discrete, symbolic items, it is practically hard to afford a multilingual machine translation system using SMT approach. At the moment, the only workaround is to develop many SMT systems, each of them is in charge of one translation direction and trained from the parallel data of that direction, and bundle them. If we do not have the parallel data of a specific language pair, we need to use pivot techniques, i.e. translating the source text into an intermediate language and then from that output, doing another translation into the target language.

On the other hand, the more classic interlingual machine translation approach (InterMT) employs an intermediate language in the translation process. By a quick glance, InterMT shares some similarity with the pivot techniques, however the main difference is that the intermediate language in InterMT is typically artificial. This so called interlingua language is supposed to covers all the linguistic phenomena and entities of the source and target languages as well as the rules to transform between the natural languages. The obvious disadvantage of InterMT is that heavy hand-crafted grammar and translation

rules have to be made by professional linguists, leading to high expense. These systems are also difficult to built upon general domains due to large vocabulary and complex rules required. InterMT approach, therefore, is not suitable for building robust multilingual machine translation systems that are feasible to construct, and fast to adapt to any domain when required.

However, the InterMT approach hints that it is theoretically possible to construct an intermediate representation that is not language specific. Such idea is in line with the key property of neural translation models: distributed representation of translation units (words, phrases, sentences, etc). To clarify, neural translation models transform the discrete source and target sentences to continuous spaces and learn complex mapping functions to transform between those spaces. If we can force neural models to learn and derive an intermediate representation from the involved languages in an automatic way, we can realize the interlingual machine translation to any domain that we provide them via multilingual corpora. In this chapter, we propose such an simple yet elegant way to do that by extending an ordinary neural machine translation system. We call it the *universal approach* for multilingual machine translation.

The chapter starts with some reviews on the attempts to build a multilingual system by bundling up several conventional MT systems and employing pivot techniques. Then we discuss the related work on building multilingual MT systems based on the recurrent NMT architecture as well as their pros and cons in Section 4.1.2. The details of our proposed universal approach is explained in Section 4.2. After that, in Section 4.3 we describe and report the experiments and evaluations of our approach applied in different translation scenarios where the amount of available training data matters. Section 4.4 concludes the chapter with the description of a derivative product from our universal approach, a multilingual word embedding corpus, which shows its usefulness when significantly improving the well-known cross-lingual document classification task.

4.1 Multilingual Machine Translation

Since the sub-components of an SMT system are basically statistical models over discrete units, such as words, n-grams of words or bilingual tuples, it is impossible to afford a genuine multilingual machine translation system

using SMT approach. Instead, we need to train separating translation systems from the parallel corpora of many language pairs. If the parallel data of a specific language pair is not largely available, i.e. a low-resourced language pair for MT, pivot techniques are the only choice to deploy a multilingual SMT system.

There are normally two types of topology for such a multilingual SMT system: *Fully Connected* and *Star*. The *Fully Connected* takes advantage of data availability. If there are n languages, we need $\frac{n(n-1)}{2}$ parallel corpora and train $n(n-1)$ systems in order to have a full multilingual SMT system containing all n languages. On the other hand, in the *Star* topology, we only need $n-1$ parallel corpora and train $2(n-1)$ SMT systems to cover the same translation directions by using pivot approaches. However, pivot-based systems often suffer from the error propagation problem where the erroneous degree of a system is accumulated by the poor performances of bridging systems it pivots. In both topologies, the quality of a specific system depends highly on the amount of parallel data it is trained from.

4.1.1 Pivot Approaches

Due to the greedy demand of parallel data, the *Fully Connected* is rarely deployed in practice. The *Star* systems utilizing pivot approaches are more popular. Depending on the level of integration when pivoting, there are several approaches:

Direct Pivot. It is also called *bridge in translation time* Bertoldi et al. (2008). As the name suggests, this simple technique directly uses two MT systems in the translation phase to acquire the ultimate output. The first system translates the source text in F into the pivot language G and then the second system translates that output in G into the target language E. This method is straight-forward and allows the usages of any MT architecture, where they are treated as black boxes Wu and Wang (2007); Bertoldi et al. (2008). The major disadvantage, however, is that it often suffers from error propagation, where the errors made by the first system tend to amplify the errors produced in the translation process of the second system. One can alleviate this issue by instead of sending an unique output from the first system, outputting an n-best list to the second one, then choosing the best final result Utiyama and Isahara (2007); Bertoldi et al. (2008). However, this solution results in an

increase proportional to n at translation time for the second system, which may not be desirable in practice.

Model Pivot. Also referred to as *bridge in training time* or *triangulation*: This technique needs a deeper integration in the participating systems, where it combines the trained models of the two systems in some specific ways (Bertoldi et al., 2008; Cohn and Lapata, 2007). In Cohn and Lapata (2007), the first step is to train the two systems and get the phrase tables $p(g|f)$ and $p(e|g)$, respectively. Then $p(e|f)$ is estimated from those $p(g|f)$ and $p(e|g)$ by summing all $p(e|g)$ that g could be found in the first phrase table. In Bertoldi et al. (2008), the criterion of maximizing the marginal distribution $\sum_e p(f|e)p(e|g)$ over the combined alignments is used to train the parameters of the pivot system. In general, model pivot approach achieves better performances than the direct approach. But it can be applied only to the participating systems which are phrase-based SMT.

Data Pivot. Finally, this is another simple pivot method which also utilizes available MT systems and considers them as black boxes, i.e. independent to the underlying architectures. The idea is that we can create synthetic direct parallel corpus $E - F$ by using the trained $G \rightarrow F$ MT system to translate the source side of the $G - E$ corpus into F[1]. This synthetic corpus can then be used to train the direct $F \rightarrow E$ translation system, although its quality relies on the quality of the trained MT system used to generate the synthetic. Often, it would not be as good as in the case of direct data due to the fact that it is generated by machine and thus affected by MT noise. However, as we will see, if the quality of the trained MT system is sufficiently good and the direct MT system is neural-based, there would be positive effects when MT noise is introduced into the training of an NMT system.

4.1.2 Related Work in Multilingual NMT

The pivot techniques from the previous section have been applied to the scenarios in which no or little direct parallel corpus of an specific translation direction exists. They often employ SMT, therefore, most modern multilingual translation systems are the results of research focusing on SMT and pivot techniques.

[1]Or in an alternative way: use the trained $G \rightarrow E$ MT system to translate the target side of the $F - G$ corpus into E. Choosing which way depends on the quality of the trained MT system.

On the other hand, neural machine translation provides a promising tool for learning intermediate representation, not only among the latent relationships in one language, but also across languages. Recent work has started investigating potential means to conduct the translation involved in multiple languages using a single NMT system. The possible reason explaining these efforts lies in the unique architecture of NMT whose encoder maps a sentence in any language to a representation in an embedding space which is believed to share common semantics among the involved languages. From that shared space, the decoder, with some implicit or explicit relevant constraints, could transform the representation into a concrete sentence in any desired language. In this section, we review some related work on this matter.

Multi-task Learning. Multi-task learning refers to the machine learning problem that use a single model to solve several tasks at the same time (Collobert et al., 2011; Niehues and Cho, 2017). Those tasks often shares the representation of input features. Research following this direction considers multilingual machine translation as a multi-task learning where the translation to one target language is viewed as a separate task. By extending the solution of sequence-to-sequence modeling using encoder-decoder architectures to multi-task learning, Luong et al. (2016) managed to achieve better performance on some *many-to-many* tasks such as translation, parsing and image captioning compared to individual tasks. Specifically in translation, the work utilizes multiple encoders to translate from multiple languages, and multiple decoders to translate to multiple languages. In this view of multilingual translation, each language in source or target side is modeled by one encoder or decoder, depending on the side of the translation. Due to the natural diversity between two tasks in that multi-task learning scenario, e.g. translation and parsing, it could not feature the attention mechanism although it has proven its effectiveness in NMT. Furthermore, by modeling the source languages with different encoders, it is indeed problematic to find the good way incorporating different attentions which are language-specific into one NMT framework.

One-to-Many Encoder-Decoder. In contrast to the multi-task approach from Luong et al. (2016), there exist others directions which proposed for multilingual translation scenarios where they can employ attention mechanism. The first one is illustrated in the work from Dong et al. (2015), where it introduces an *one-to-many* multilingual NMT system to translates from one source language into multiple target languages. Having one source

language, the attention mechanism is then handed over to the corresponding decoder. The objective function is changed to adapt to multilingual settings. In testing time, the parameters specific to a desired language pair are used to perform the translation.

Many-to-One Encoder-Decoder. In a separate effort to achieve multilingual NMT, the work of Zoph and Knight (2016) leverages available parallel data from other language pairs to help reducing possible ambiguities in the translation process into a single target language[2]. They employed the multi-source attention-based NMT in a way that only one attention mechanism is required despite having multiple encoders. To achieve this, the outputs of the encoders were combined before feeding to the attention layer. They implemented two types of encoder combination; One is adding a non-linear layer on the concatenation of the encoders' hidden states. The other is using a variant of LSTM taking the respective gate values from the individual LSTM units of the encoders. As a result, the combined hidden states contain information from both encoders, thus encode the common semantic of the two source languages.

Many-to-Many Encoder-Decoder. Firat et al. (2016) proposed another approach which genuinely delivers attention-based NMT to multilingual translation. As in Luong et al. (2016), their approach utilizes one encoder per source language and one decoder per target language for *many-to-many* translation tasks. Instead of a quadratic number of independent attention layers, however, one single attention mechanism is integrated into their NMT, performing an affine transformation between the hidden layer of m source languages and that one of n target languages. It is required to change their architecture to accommodate such a complicated shared attention mechanism.

4.2 Universal Approach for Multilingual NMT

Our motivation comes from the fact that, multi-source NMT as additional parallel data in several languages are expected to improve the performance for a single language pair, we have been investigating a simple but elegant approach toward multilingual NMT that we called the "*universal*" approach. Being one of the first attempts to design multilingual neural machine

[2] An example taken from the paper is when we want to translate the English word *bank* into French, it might be easier if we have an additional German sentence containing the word *Flussufer* (*river bank*).

translation systems, we aim at realizing the great advantages of using neural methods for multilingual translation, especially without the need of architecture modification which normally requires heavy changes in hyper-parameter tuning and training speed.

The intuition behind our universal approach is that we deploy multilingual translation processing using a single encoder-decoder NMT architecture. In order to realize this important target, we managed to discover a convenient and efficient way to ensure that a single NMT model can acquire an intermediate representation across the languages involved in the source side and then use such representation to project into the target languages of interest.

Interestingly, this approach is feasible by modifying the preprocessing phase while the NMT architecture is remained unchanged. Therefore, one of the most important advantages is that our approach can utilise any improvement in modeling for the single-language-pair translation task. For example, architectural improvement for NMT such the convolutional neural networks or the transformer can be applied to a multilingual setup according to our method easily. Such universality also enable this approach to be applied in various translation scenarios, such as for the translation of low-resourced language pairs or even in the extreme case, where we do not have any direct parallel data. Furthermore, unlike interlingual translation, we can train and deploy our multilingual system in any domain in which there exists parallel corpora. In addition, it is proven that the number of free parameters to learn in our network does not go beyond that magnitude of a many-to-many encoder-decoder NMT system, bringing a great computational advantage both in training and translation time.

Given the idea, our practical implementation consists of two steps conducted in the preprocessing phase:

1. **Language-specific Coding**: Coding the words in different languages as different words in the language-mixed vocabularies in order to force our NMT learn the shared representation across languages

2. **Target Forcing**: Forcing our NMT to translate that representation of the source sentence into the sentences in a desired target language.

4.2.1 Language-specific Coding

When the encoder of a NMT system considers words across languages as different words, with a well-chosen architecture, it is expected to be able to learn a good representation of the source words in an embedding space in which words carrying similar meaning would have a closer distance to each others than those are semantically different. This should hold true when the words have the same or similar surface form, such as (*Obama**@de***, *Obama**@en***) or (*Projektion**@de***, *projection**@en***)[3]. This should also hold true when the words have the same or similar meaning across languages, such as (*car**@en***, *automobile**@en***) or (*Flussufer**@de***, *bank**@en***). In this way, the same words in different languages are treated as synonyms in one language or as the words having the same meaning across languages. Our encoder then acts similarly to the one of multi-source approach (Zoph and Knight, 2016), collecting additional information from other sources for better translations, but with a much simpler embedding function. Unlike them, we need only one encoder, so we could reduce the number of parameters to learn. Furthermore, we neither need to change the network architecture nor depend on which recurrent units (GRU, LSTM or simple RNN) are currently been using in the encoder.

We could apply the same trick to the target sentences and thus enable *many-to-many* translation capability of our NMT system. Similar to the multi-target translation (Dong et al., 2015), we exploit further the correlation in semantics of those target sentences across different languages. The main difference between our approach and the work of Dong et al. (2015) is that we need only one decoder for all target languages. Given one encoder for multiple source languages and one decoder for multiple target languages, it is trivial to incorporate the attention mechanism as in the case of a regular NMT for single language translation. In training, the attention layers were directed to learn relevant alignments between words in specific language pair and forward the produced context vector to the decoder. Now we rely totally on the network to learn good alignments between source and target sides. In fact, giving more information, our system are able to form nice alignments.

In comparison to other research that could perform complete multi-task learning, e.g. the work from Luong et al. (2016) or the approach proposed

[3] *a_word**@lang_code*** is a simple way that transforms the word *a_word* into a different surface form associated with its language ***lang_code***. For example, *Projektion**@de*** is referred to the word *Projektion* appearing in a German (***de***) sentence.

by Firat et al. (2016), our method is able to accommodate the attention layers seamlessly and easily. It also draws a clear distinction from those works in term of the complexity of the whole network: considerably less parameters to learn, thus reduces overfitting, with a conventional attention mechanism and a standard training procedure.

4.2.2 Target Forcing

While language-specific coding allows us to implement a multilingual NMT with attention, there are two issues we have to consider before training the network. The first issue lies on the fact that the number of rare words would increase in proportion with the number of languages involved. This might be solved by applying a rare word treatment method with appropriate awareness of the vocabularies' size. The second one is more problematic: Ambiguity level in the translation process definitely increases due to the additional introduction of words having the same or similar meaning across languages at both source and target sides.

We deal with the problem by explicitly forcing the attention and translation to the direction that we prefer, expecting the information would limit the ambiguity to the scope of one language instead of all target languages. We realize this idea by adding at the beginning and at the end of every source sentence a special symbol indicating the language they would be translated into. Due to the nature of sequence labeling implemented in the encoder and decoder, in training, those starting symbols[4] encourage the network learning the translation of following target words in a particular language pair. At translation time, information of the target language we provided help to limit the translated candidates and guide the decoder, which is basically a language model, to follow those starting symbols, hence forming the translation in the desired language.

Figure 4.1 illustrates the essence of our approach. With two steps in the preprocessing phase, namely language-specific coding and target forcing, we are able to employ multilingual attention-based NMT without any special treatment in training such a standard architecture. Our encoder and attention-enable decoder can be seen as a shared encoder and decoder across languages, or an *universal* encoder and decoder. The flexibility of our approach

[4]For a bidirectional encoder, they are actually the starting symbols of a source sentence from two directions.

allow us to integrate any language into source or target side. As we will see, it has proven to be extremely helpful not only in low-resourced scenarios but also in translation of well-resourced language pairs as it provides a novel way to make use of large monolingual corpora in NMT.

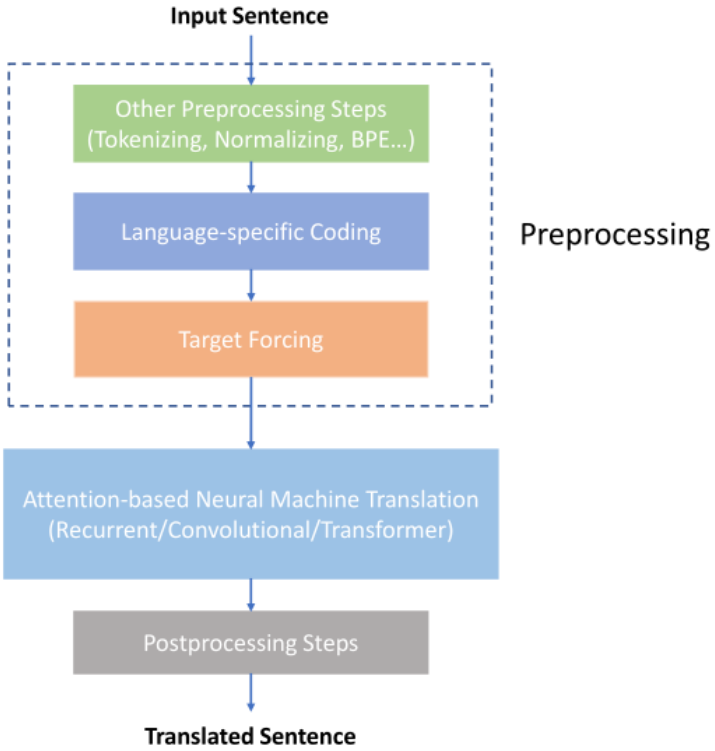


FIGURE 4.1: Universal Approach for Multilingual NMT.

4.3 Evaluation

In this section, we describe the evaluation of our proposed approach in comparisons with the strong baselines using NMT in two scenarios: the translation of an under-resourced language pair and the translation of a language pair that does not exist any parallel data at all, which we called it *zero-resourced*.

4.3.1 Experimental Settings

Data. We choose TED parallel corpus as the basic training data of our experiments since it might be the only high-quality parallel data of many low-resourced language pairs.

Original sentence pair		
Source	De	versetzen Sie sich mal in meine Lage
Target	En	put yourselves in my position
Preprocessed sentence pair		
Source	De	**<en>** @*de*@versetzen @*de*@Sie @*de*@sich @*de*@mal @*de*@in @*de*@meine @*de*@Lage **<en>**
Target	En	@*en*@put @*en*@yourselves @*en*@in @*en*@my @*en*@position
Original sentence pair		
Source	En	I flew on Air Force Two for eight years .
Target	Nl	ik heb acht jaar lang met de Air Force Two gevlogen .
Preprocessed sentence pair		
Source	En	**<nl>** @*en*@I @*en*@flew @*en*@on @*en*@Air @*en*@Force @*en*@Two @*en*@for @*en*@eight @*en*@years @*en*@. **<nl>**
Target	Nl	@*nl*@ik @*nl*@heb @*nl*@acht @*nl*@jaar @*nl*@lang @*nl*@met @*nl*@de @*nl*@Air @*nl*@Force @*nl*@Two @*nl*@gevlogen @*nl*@.

TABLE 4.1: Two examples of training sentence pairs in German→English and English→Dutch are preprocessed following our universal approach. German words with language code `De`, English words with language code `En` and Dutch words with language code `Nl`. Language-specific coding tags are denoted in *italic* and target forcing tags are denoted in **bold**.

In addition, we use a much larger corpus provided freely by WMT organizers[5] when we evaluate the impact of our approach in a real machine translation campaign. It includes the paralled corpus extracted from the digital corpus of European Parliament (EPPS), the News Commentary (NC) and the web-crawled parallel data (CommonCrawl). While the number of sentences in popular TED bilingual corpus varies from 13 thousands to 17 thousands, the total number of sentences in those larger corpus is approximately 3 million sentences.

		En-De	**En-Fr**	**Fr-De**
Training	Number of sentences	196k	165k	165k
	Number of tokens (average)	3.7m	3.3m	3.4m
Development	Number of sentences	993	887	887
	Number of tokens (average)	20k	19k	21k
Testing	Number of sentences	993	887	887
	Number of tokens (average)	20k	19k	21k

TABLE 4.2: The statistics of TED corpora used in our experiments.

[5] http://www.statmt.org/wmt15/

NMT Setup. All experiments in this section have been conducted using the NMT frameworks `Nematus`[6] and `OpenNMT`[7]. Following the work of Sennrich et al. (2016b), subword segmentation is handled in the prepocessing phase using Byte-Pair Encoding (BPE). Excepts stated clearly in some experiments, we set the number of BPE merging operations at 40,000 on the joint of source and target data. When training all NMT systems, we take out the sentence pairs exceeding 50-word length and shuffle them inside every minibatch. Our short-list vocabularies contain 40,000 most frequent words while the others are considered as rare words and applied the subword translation. We use an 1024-cell GRU layer and 1000-dimensional embeddings with dropout at every layer with the probability of 0.2 in the embedding and hidden layers and 0.1 in the input and ourput layers. We trained our systems using gradient descent optimization with Adadelta (Zeiler, 2012) on minibatches of size 80 and the gradient is rescaled whenever its norm exceed 1.0. All the trainings last approximately seven days if the early-stopping condition could not be reached. At a certain time, an external evaluation script on BLEU (Papineni et al., 2002) is conducted on a development set to decide the early-stopping condition. This evaluation script has also being used to choose the model archiving the best BLEU on the development set instead of the maximal loglikelihood between the translations and target sentences while training. In translation, the framework produces n-best candidates and we then use a beam search with the beam size of 12 to get the best translation.

4.3.2 Low-resourced Translation

First, we consider the translation for an under-resourced pair of languages. Here a small portion of the large parallel corpus for English-German is used as a simulation for the scenario where we do not have much parallel data. We perform language-specific coding in both source and target sides. By accommodating the German monolingual data as an additional input (German→German), which we called the *mix-source* approach, we could enrich the training data in a simple, natural way. Given this under-resourced situation, it could help our NMT obtain a better representation of the source side without any additional data, hence, able to learn the translation relationship better. Including monolingual data in this way might also improve the translation of some rare word types such as named entities.

[6] https://github.com/rsennrich/nematus
[7] https://github.com/OpenNMT/OpenNMT-py

As approaching to the ultimate goal of our work, we would like to investigate the advantages of the real multilinguality in our NMT systems from additional data in other languages. We incorporate a similar portion of French-German parallel corpus into the English-German one. As discussed in Section 4.2, it is expected to help reducing the ambiguity in translation between one language pair since it utilizes the semantic context provided by the other source language. We name this *multi-source*.

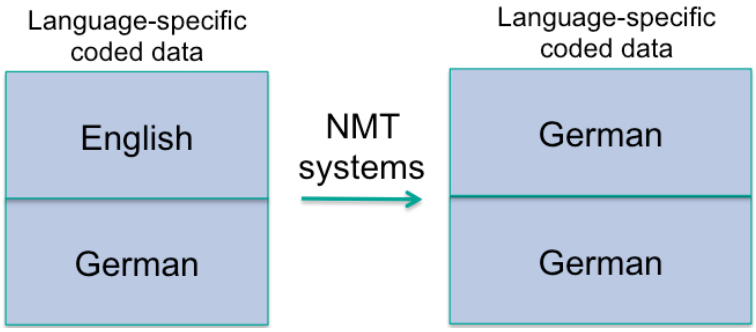


FIGURE 4.2: Description of data prepared for mix-source NMT

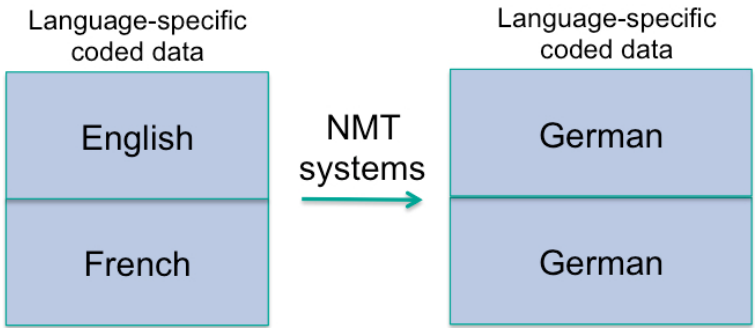


FIGURE 4.3: Description of data prepared for multi-source NMT

Table 4.3 summarizes the performance of our systems measured in BLEU[8] on two test sets, *tst2013* and *tst2014*. Compared to the baseline NMT system which is solely trained on TED English-German data, our *mix-source* system achieves a considerable improvement of 2.6 BLEU points on *tst2013* and 2.1 BLEU points on and *tst2014* . Adding French data to the source side and their corresponding German data to the target side in our *multi-source* system also help to gain 2.2 and 1.6 BLEU points more on *tst2013* *tst2014*, respectively. We observe a better improvement from our *mix-source* system compared to our *multi-source* system. We speculate the reason that the *mix-source* encoder utilize the same information shared in two languages while the *multi-source* receives

[8]We used the script `mteval-v13a.pl` of the Moses framework (`http://statmt.org/moses/`) as the official way to calculate BLEU scores in main machine translation campaigns.

and processes similar information in the other language but not necessarily the same.

System	tst2013		tst2014	
	BLEU	ΔBLEU	BLEU	ΔBLEU
Baseline (En→De)	24.35	–	20.62	–
Mix-source (En,De→De,De)	26.99	+2.64	22.71	+2.09
Multi-source (En,Fr→De,De)	26.64	+2.21	22.21	+1.59

TABLE 4.3: English→German systems in under-resourced scenario.

4.3.3 Zero-resourced Translation

Among low-resourced scenarios, zero-resourced translation task stands in an extreme level. A zero-resourced translation task is one of the most difficult situation when there is no parallel data between the translating language pair. In this section, we extend our strategies using the proposed multilingual NMT approach as first attempts to this extreme situation.

We employ language-specific coding and target forcing in a strategy called `bridge`. Unlike the strategies used in under-resourced translation task which employs multiple corpora in the source side only, `bridge` is an entire *many-to-many* multilingual NMT. Simulating a zero-resourced German→French translation task given the available German-English and English-French parallel corpora, after applying language-specific coding and target forcing for each corpus, we mix those data with an English-English data as a "bridge" creating some connection between German and French. We also propose a variant of this strategy that we incorporate French-French data. And we call it `universal`.

We evaluate `bridge` and `universal` systems on two German→French test sets. They are compared to a `direct` system, which is an NMT trained on German→French data, and to a `pivot` system, which essentially consists of two separate NMTs trained to translate from German to English and English to French. The `direct` system should not exist in a real zero-resourced situation. We refer it as the perfect system for comparison purpose only. In case of the `pivot` system, to generate a translated text in French from a German sentence, we first translate it to English, then the output sentence is fed to the English→German NMT system to obtain the French translation. Since

there are more than two languages involved in those systems, we increase the number of BPE merging operations proportionally in order to reduce the number of rare words in such systems. We do not expect our proposed systems to perform well with this primitive way of building direct translating connections since this is essentially a difficult task. We report the performance of those systems in Table 4.4.

System	BLEU	ΔBLEU
Direct (De→Fr)	16.65	+3.24
Pivot (De→En→Fr)	13.41	–
Bridge (De,En,En→En,Fr,En)	9.70	-3.71
Universal (De,En,En,Fr→En,Fr,En,Fr)	10.77	-2.64

TABLE 4.4: German→French systems in zero-resourced scenario.

Unsurprisingly, both `bridge` and `universal` systems perform worse than the `pivot` one. We consider two possible reasons:

Our target forcing mechanism is moderately primitive. Since the process is applied after language-specific coding, the target forcing symbol is the same for all source sentences in every languages. Thus, the forcing might not be strong enough to guide the decision of the next words. Once the very first word is translated into a word in wrong language, the following words tend to be translated into that wrong language again. Table 4.5 shows some statistics of the translated words and sentences in wrong language.

System	Percentage of wrong words	Percentage of wrong sentences
`bridge`	21.27%	9.70%
`universal`	17.57%	9.47%

TABLE 4.5: Percentages of language identification mistakes.

Balancing level of the training corpus. By conducting several experiments with our multilingual translation system which has a limited number of sentences of a specific language compared to significant larger corpora for other languages, we have observed that unbalanced data can affect the quality of the system. Unbalanced data might introduce bias in the training as well as ambiguities in the translation. The difference of 1.07 BLEU points between *bridge* and *universal* might explain this assumption. Compared to the *bridge*

system, in the *universal* system we added more target data (French), therefore reduced the unbalance problem and brought some improvements.

4.4 Multilingual Word Embeddings

Cross-lingual word embeddings are the representations of words across languages in a shared continuous vector space. Cross-lingual word embeddings have been shown to be helpful in the development of cross-lingual natural language processing tools. In case of more than two languages involved, we call them multilingual word embeddings. In this section, we introduce a multilingual word embedding corpus as the derivative product of our multilingual NMT following the universal approach. The corpus also stands to prove our points on multilingual NMT that our universal approach successfully forces NMT to learn an intermediate and multilingual representation across languages.

Unlike other cross-lingual embedding corpora, the embeddings can be learned from significantly smaller portions of data and for multiple languages at once. And then by conducting an intrinsic evaluation on monolingual tasks, we show that our method is fairly competitive to the prevalent methods which focus on learning monolingual representations from monolingual corpora. On the other hand, applying our corpus on the cross-lingual document classification task obtains the best figures compared to other popular cross-lingual corpora.

There have been various methods of cross-lingual embedding induction being proposed, but most of them are essentially bilingual in the perspective that they learn to induce bilingual embeddings from bilingual data. Basically these methods optimize some cross-lingual constraints so that the semantic similarity between words corresponds to the closeness of these representations in a common vector space. Consequently, if they need cross-lingual embeddings for a new language pair, they must apply their inducing method on that new bilingual data. Furthermore, there would be some domain mismatch between the new acquired embeddings and the others if the new bilingual data are from different domain. The aforementioned limitations of those cross-lingual corpora motivates us to design a method for multilingual embedding induction from a single corpus which is available in as many languages as possible.

In this section, we describe our proposed approach utilizing a multilingual NMT system to constrain the embeddings from n source languages while translating into the same target language (as we previously called it *multi-source* NMT). The source embeddings employed in this model are implicitly forced to learn the common semantic regularities in order to maximize the translation quality of every language pair in the system. Once the multi-source NMT model is trained to a good state, the source word embeddings can be simply extracted from the model and used as a multilingual word embeddings. Our method results in `KIT-Multi`, consisting multilingual word embeddings of {English, German, French, Dutch, Italian, Romanian}[9]. Then we conducted some preliminary evaluations on `KIT-Multi` and compares to other cross-lingual embedding corpora. It has been shown that our multilingual corpus achieves competitive performances in standard evaluations as well as it has better coverage while using much less data for the training process.

4.4.1 `KIT-Multi` Corpus

In our multi-source NMT system where the sentences from *several sources languages* are translated to *one target language*, the source embeddings are tied to a common semantic space across languages. So the source embeddings has its inherent cross-lingual characteristics, which could be extremely helpful for the cross-lingual applications employing the embeddings. So in order to induce the multilingual embeddings, we simply train our multi-source NMT system following the universal approach to a good state and then we extract the source embeddings from this multi-source system. The figure 4.4 describes the process.

The corpora we used to induce the multilingual embeding originates from the {French, German, Dutch, Italian, Romanian}-English parts of TED corpus, and then we took the English target side of the French-English part and mixed it to the source side following the *mix-source* strategy (Section 4.3.2). This resulted in a multi-source corpus ({French, German, Dutch, Italian, Romanian, English}$\rightarrow$ English) prior training. The multi-source NMT is trained using the NMT framework `OpenNMT`[10] (Klein et al., 2016) to translate from aforementioned languages to the only target language English. The

[9] But we can of course expand it seamlessly by training our *multi-source* NMT on the corpora in more languages.

[10] `http://opennmt.net`

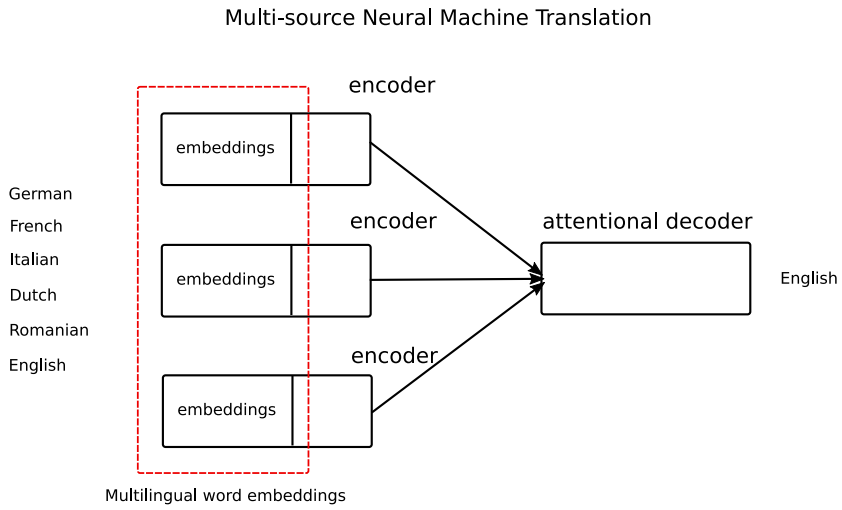


FIGURE 4.4: Multi-source NMT used to produce KIT-Multi

statistics of the TED bilingual corpora and our multilingual embedding corpus are shown in Table 4.6 and Table 4.7, respectively.

Language pairs	Number of sentences
German-English	196794
French-English	195025
Dutch-English	230866
Italian-English	220812
Romanian-English	210402

TABLE 4.6: Statistics of pair-wise TED bilingual corpora

Languages	Number of entries
English	21001
French	25685
German	24182
Dutch	24167
Italian	23422
Romanian	25505

TABLE 4.7: The size of the KIT-Multi embedding corpus

Table 4.8 shows the closest words in the semantic space based on Cosine similarity with respect to some examples. We also include the language codes to clarify the origin of each word. From the table, we can see that the most close words are actually the words having the same meaning but in other languages.

Figure 4.5 illustrates the visualization of multilingual word embeddings extracted from the multi-source NMT system and projected to the 2D space

@en@research	
Word	Cosine Similarity
@de@Forschung	0.727675
@fr@recherches	0.697122
@de@Forschungs	0.671166
@fr@recherche	0.643990
@de@geforscht	0.637604
@en@humanity	
Word	Cosine Similarity
@de@Menschlichkeit	0.691524
@fr@humanité	0.684639
@de@Menschheit	0.645123
@de@Menscheit	0.634902
@en@mankind	0.621472

TABLE 4.8: Top 5 closest words by Cosine similarity.

using *t-SNE* (van der Maaten and Hinton, 2008). It shows how different words in different languages, i.e. English-German-French, can be close in the shared semantic space after being trained to translate into a common language.

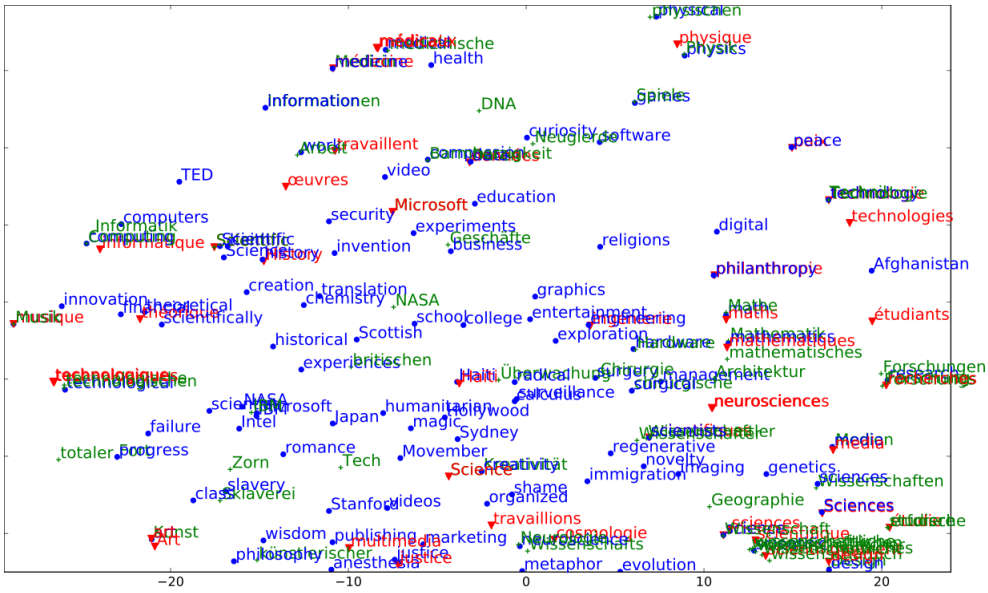


FIGURE 4.5: The multilingual word embeddings derived from our Multi-source NMT. To illustrate more clearly, only the word vectors of the words related to "*science*" are visualized. The blue words are the English words, green for German and the red ones are the French words.

4.4.2 Evaluation of `KIT-Multi`

In this section, we describe some initial evaluation of our multilingual embedding corpus over some standard intrinsic and extrinsic evaluations, in comparisons with some other popular approaches for cross-lingual word embedding induction.

We follow the experimental layout and settings of Upadhyay et al. (2016), conducting intrinsic and extrinsic evaluations on three European languages: English, French and German. The intrinsic evaluation is the *monolingual word similarity* task. The extrinsic evaluation focuses on the *cross-lingual document classification*. In this task, a document classifier is trained on a training set composed by a language L_1 and then predict the test set which is in the different language L_2. The process is then reversed for the language pair, and the classification accuracy is used to judge the quality of the cross-lingual embeddings. The corpora chosen to be compared are the corpora induced by Skip - Bilingual Skip-gram (Luong et al., 2015a), CVM - Bilingual Compositional Model (Hermann and Blunsom, 2014) and VCD - Bilingual Vectors from Comparable Data (Vulic and Moens, 2015), which are all trained on much bigger Europarl v7 parallel corpora. To show the impact of the corpus size, we also train the Bilingual Skip-gram embeddings with the same corpora used to train our model, and name it Skip-TED. For the details of those methods, please refer to Upadhyay et al. (2016).

In the intrinsic monolingual evaluation, we consider the word embeddings in one language at a time, i.e. the monolingual word embeddings, in order to conduct the *word similarity*. The Spearman's rank correlation coefficient (Myers et al., 1995) between system similarity and human is the measure to judge the quality of the induced word embeddings. The English evaluation datasets are SimLex999 (*En-999*) and WordSim353 (*En-353*), in which the former (Hill et al., 2016) is claimed to better capture the similarity rather than both similarity and relatedness like in the latter (Finkelstein et al., 2002). The German (*De*) and French (*Fr*) datasets are the WordSim353 counterparts Camacho-Collados et al. (2015); Leviant and Reichart (2015).

The scores in Table 4.9 show that our word embeddings are competent in term of monolingual aspect even though they are not trained to be adapted to monolingual quality. Moreover, our word embeddings perform better than the Skip embeddings trained on the same data by a large margin.

Language	Skip	Skip-TED	CVM	VCD	*KIT-Multi*
En-999	0.34	0.22	**0.37**	0.32	***0.37***
En-353	0.53	0.39	0.43	**0.59**	*0.45*
De	0.52	0.40	0.40	**0.54**	*0.51*
Fr	**0.50**	0.09	0.38	0.43	*0.48*

TABLE 4.9: Monolingual evaluation tasks.

As shown in Table 4.10, the classifiers trained on our embeddings achieve highest accuracy on both directions of English⇔German, considerably better than other approaches. It is notable that, our model is trained on a substantially smaller corpus.

L_1	L_2	Skip	Skip-TED	CVM	VCD	*KIT-Multi*
En	De	85.2	84.3	85.0	79.9	***86.6***
De	En	74.9	73.5	71.1	74.1	***79.7***

TABLE 4.10: The accuracy of cross-lingual document classification task using the word embeddings.

4.4.3 Discussion of `KIT-Multi`

In Upadhyay et al. (2016), the most popular and advantageous techniques for multilingual word embedding induction have been thoroughly evaluated. Corpora induced by Skip and VCD are the methods having the capability of monolingual adaptation by adjusting a hyper-parameter (in Skip models) or the portion of texts in each language (in VCD models). Furthermore, since they are designed based on the *skip-gram* models (Mikolov et al., 2013), it is unsurprising that they perform well on monolingual tasks.

Corpora induced by CVM and our `KIT-Multi`, in contrast, are designed with cross-lingual orientation so that they focus more on similarity instead of relatedness. Aforementioned, our `KIT-Multi` corpus has shown its potential by achieving high accuracies on the task despite being induced from a significally smaller corpus. Compared to the corpora acquired by their method, our embedding inherently induced in multilingual settings, with an arbitrary number of source and target languages, instead of being limited to bilingual. Those advantages allow us to extend our corpus seamlessly to many languages using small multilingual corpus, ideally from TED talks.

Chapter 5

Advanced Methods in Multilingual Neural Translation

In the previous chapter, we have presented our proposed universal approach for multilingual neural machine translation. Although it shows great potential in several multilingual translation scenarios thanks to its simplicity and elegance, there exist a few limitations with the current setting that prevent our universal approach from being applied in more complicated and realistic applications. In this chapter, we propose a handful of solutions to address those practical limitations. We are therefore able to improve our zero-resourced translation systems significantly and efficiently. Furthermore, we can deploy the refined multilingual approach in our Lecture Translator for KIT students and build a large speech translation system translating speeches from English and German to twenty four European languages using a single neural translation model.

5.1 Limitations of Universal Approach

The original universal approach leverages an ordinary neural machine translation (NMT) system to perform multilingual translation by forcing it to learn the common representation space across languages and translate sentences in that space to the desired target language. In order to do that, in the preprocessing phase prior to training, we apply these two steps:

1. **Language Coding**: Add the language codes to every word in source and target sentences.
2. **Target Forcing**: Add a special token in the beginning/end of every source sentence indicating the desired target language.

As we discussed in the previous chapter, due to the diversity of mixed-language vocabulary and the potential bias of training data, the target forcing mechanism might be not strong enough to guide the NMT translate to the desired target language. This is consider as *language bias* problem in our multilingual system: If the very first word is mis-translated into a wrong language, the imminent words are likely to fall into the same language again . The problem is more severe when the degree of unbalance is high in some scenarios (whereas the zero-resourced is a typical example). We reported a figure of 9.7% of the sentences incorrectly translated in our basic zero-resourced German→French system. A simple temporary fix is that we can add the target language token several times both to the beginning and to the end of every source sentence to make the forcing effect stronger. Furthermore, every target sentence starts with a pseudo word playing the role of a start token in a specific target language. This pseudo word can later be removed along with sub-word tags in post-processing steps. Table 5.1 illustrates this simple fix, as we call it *reinforced guiding*, on the same example from the previous chapter.

Original sentence pairs		
Source	De	versetzen Sie sich mal in meine Lage
Target	En	put yourselves in my position
Original universal approach		
Source	De	**<en>** @*de*@versetzen @*de*@Sie @*de*@sich @*de*@mal @*de*@in @*de*@meine @*de*@Lage **<en>**
Target	En	@*en*@put @*en*@yourselves @*en*@in @*en*@my @*en*@position
Universal approach with reinforced guiding		
Source	De	**<en> <en> <en>** @*de*@versetzen @*de*@Sie @*de*@sich @*de*@mal @*de*@in @*de*@meine @*de*@Lage **<en> <en> <en>**
Target	En	@*en*@___ @*en*@put @*en*@yourselves @*en*@in @*en*@my @*en*@position @*en*@___

TABLE 5.1: Example preprocessed following our universal approach with reinforced guiding

The same example of German→English from the previous chapter is preprocessed following our universal approach with reinforced guiding. The target language token is placed three times instead of once. The pseudo word in the target language is "@*en*@___", meaning the start (or the end) of an English target sentence.

Our experiments show that with the reinforced guiding, the universal approach performs better when evaluated with both the accuracy of target language identification and the quality of the translation (Pham et al., 2017; Ha et al., 2017). However, the problem of diversity in mixed-language vocabulary

is still there. The beam search still needs to find the correct word in the desired language from a big pool of words; Among them, there are many other candidates that should not be considered in the beam search.

The underlying problem is basically the ineffective way to represent the mixed-language vocabularies. If we have n languages in the source or target sides, and we consider keeping m most frequent words for each language, the mixed-language vocabulary's size is around $n \times m$. This obviously introduces n-fold ambiguities, thus, potentially reduces the accuracy of the beam search. Furthermore, large vocabularies means that there is an immerse number of parameters in the input embedding in both the encoder and decoder sides. And a more serious problem is the size of the decoder's output softmax layer, which takes the majority of computation in the network, due to the fact that the size of the vocabulary is much larger than the hidden layer size. Those problems make the training as well as the inference (translation process) become slower and inefficient, and require more hardware memory to store the model.

In the next section, we propose two techniques specifically address the aforementioned problem in the zero-resourced tasks organized by the International Workshop on Spoken Language Translation 2017 (IWSLT'17). But first, let us review the tasks and describe the baselines for comparison purpose.

5.2 Multilingual Translation Tasks at IWSLT'17

In the end of the year 2017, the International Workshop on Spoken Language Translation for the first time and being the first MT workshop to organize multilingual translation tasks. The task would focus on multilingual translation for different scales, including large data size, small data size and a zero-shot scenario in which the participants cannot use any available direct parallel data to train.

5.2.1 Small and Large Multilingual Translation Tasks

The participants can train any or all of twenty systems from the provided multilingual corpora of five languages {English, German, Dutch, Italian, Romanian}.

For the *small translation tasks*, the only corpus to used is the multilingual corpus of TED in those five languages. It is considered to be in-domain data since the development and test sets are also extracted from this corpus.

For the *large translation tasks*, in addition to the in-domain TED corpus, other large and out-of-domain corpora listed as permissible in the website of IWSLT'17[1] can also be used to build the MT systems. There are several permissible multilingual corpora which are much larger than TED, but we only use the multilingual Europarl corpus for the five languages as the out-of-domain data when training our multilingual systems.

5.2.2 Zero-Resourced Translation Tasks

IWSLT'17 organize the *zero-resourced translation tasks* (which they call *zero-shot translation tasks*) in four directions: German↔Dutch and Italian↔Romanian (Cettolo et al., 2017). In any of the four translation directions, we can use all of the bilingual corpora from {English, German, Dutch, Italian, Romanian} to {English, German, Dutch, Italian, Romanian} excepts the direct corpus of that direction. The four directions have been chosen as the IWSLT'17 organizers reasoned that the similar between languages (like German and Dutch or Italian and Romanian) might help to perform better zero-resourced translation between those languages. However, we would like to figure out how independent the multilinguality relies on the similarity of the involved languages. Thus, we conducted two following zero-resourced tasks: German→Dutch and German→Romanian. Besides not using the direct corpus between two languages in ech direction, we also do not use all of the corpora but we only use the involved corpora with English as the pivot language. For instance, to translate German→Dutch, we use German-English and English-Dutch, and to translate German→Romanian, we use German-English and English-Romanian.

The data are extracted from TED corpus. The validation and test sets are `dev2010` and `tst2010` which are provided by the IWSLT17 organizers.

[1] https://sites.google.com/site/iwsltevaluation2017/data-provided

5.2.3 Zero-Resourced NMT System Setups

We use `OpenNMT`[2] (Klein et al., 2016) framework to conduct all experiments mentioned below. Subword segmentation is performed using Byte-Pair Encoding (Sennrich et al., 2016b) with 40000 merging operations. All sentence pairs in training and validation data which exceeds 50-word length are removed and the rest are shuffled inside each of every minibatch. We use 1024-cell LSTM layers (Hochreiter and Schmidhuber, 1997) and 1024-dimensional embeddings with dropout of 0.3 at every recurrent layers. The systems are trained using Adam(**?**). In decoding process, we use a beam search with the size of 15.

5.2.4 Zero-Resourced Baseline Systems

We use the following baseline systems:

- `Direct`: A system which does not exist in the real world is trained using the parallel corpus. It is only for comparison purpose.
- `Pivot`: A system which uses English as the pivot language. This is the *direct pivot* technique mentioned in Section 4.1.1: The output of the first Source→Pivot translation system was pipelined into the second system trained to translate from Pivot to Target.
- *Zero 2L*: To build this system, we followed the idea of Johnson et al. (2016). We added a target token to every source sentences in the parallel corpus of Source→Pivot, added another target token to every pivot sentences in the parallel corpus of Pivot→Target, merged those two parallel corpora into a big corpus and used a standard NMT architecture to train and decode. The only differences are the actual data and a simpler NMT architecture we used to train the system.
- *Zero 4L*: Same as *Zero 2L* but in addition applying to two other directions Pivot→Source and Target→Pivot. The result is a parallel corpus two times larger than the corpus in *Zero 2L*.
- *Zero 6L*: This is our universal multilingual approach with reinforced guiding. There are two main differences compared *Zero 2L* and *Zero 4L*: we conducted both Language Coding and Target Forcing preprocessing steps, the data used to trained are actually six parallel

[2] `http://opennmt.net/`

corpora: Source↔Pivot, Pivot→Pivot, Pivot↔Target, Target→Target. Finally we merged them at the end to form a big parallel corpus.

- *Back-Trans*: This is not a real zero-resourced system where we back-translated (Sennrich et al., 2016a) the English part of the Pivot-Target parallel corpus using a Target-Pivot NMT system. At the end we have a Source-Target parallel corpus with back-translation quality. This is actually the *data pivot* technique (Section 4.1.1). After we obtained that direct corpus, we apply the same steps as in the *Zero 6L* setting to all corpora we have (in total it consists of eight bilingual corpora).

The results of these baselines is reported in Table 5.2. We can see that translating from German→Romanian is more difficult than German→Dutch, which is reasonable when German and Dutch are considered to be similar. The `direct` approach and the `pivot` approach have similar performance in term of BLEU score with a little bit degraded by the `pivot`. Interestingly, the `Back-Trans` performed better that the direct approach on German→Romanian. We spectaculate that back translation might pose some translation noise which makes the translation from German→Romanian more robust.

	System	**Zero-shot?**	**German→Dutch**		**German→Romanian**	
			dev2010	tst2010	dev2010	tst2010
(1)	Direct	No	17.83	20.49	12.41	15.14
(2)	Pivot	Yes	16.11	19.12	12.88	15.04
(3)	Zero 2L	Yes	4.79	5.75	1.55	2.05
(4)	Zero 4L	Yes	6.31	7.93	3.15	3.73
(5)	Zero 6L	Yes	11.58	14.95	8.61	10.83
(6)	Back-Trans	No	17.33	20.36	12.92	15.62

TABLE 5.2: Comparisons of multilingual NMT baselines.

Compared to the *Zero 6L* model (5), two models *Zero 2L* (3) and *Zero 4L* (4) from Johnson et al. (2016) achieved quite low scores. This explains the language-bias problem when these models used less and unbalanced corpora than the *Zero 6L* system. However, the real zero-resourced systems (2, 3, 4, 5), excepts the pivot one (2), performed worse than those using direct parallel corpora (1) and (7), since the zero-resourced systems have not been shown the direct data, hence, having little or no guide to learn the translation. Among

those real zero-resourced non-pivot systems, the *Zero 6L* system got the best performance due to the amount and the balance of the data used in training. Thus, from hereinafter we consider the *Zero 6L* as the baseline to analyze the effectiveness of our proposed strategies.

5.3 Target Dictionary Filtering

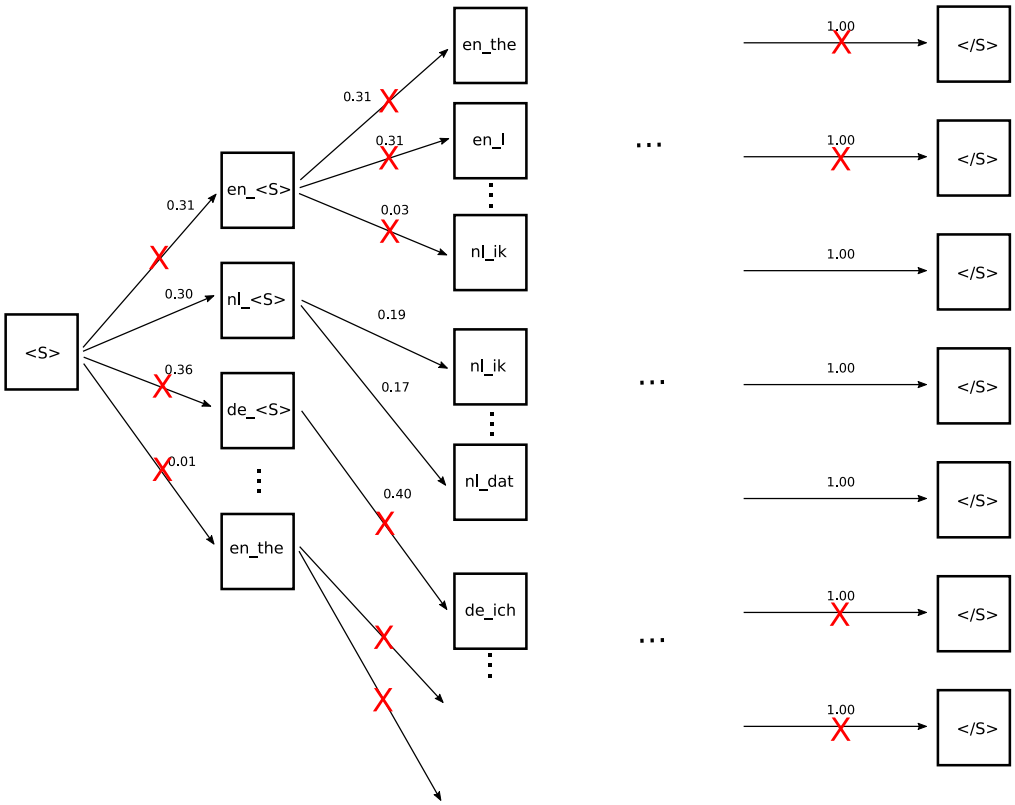


FIGURE 5.1: Search paths are removed by filtering dictionaries.

The first solution we describe here aims to reduce time and complexity in the translation phrase by filtering out unwanted candidates. In the translation phase, our trained NMT system already knows in which language it should translate the source sentence into. Hence, we can remove the words not belonging to the desired target language for this translation. It is a straight-forward method since our entries in the target vocabulary contains their corresponding language code. This *target dictionary filtering* would significantly reduce the translation time due to the fact that many search paths containing the unwanted candidates are removed. More importantly, it assures the translated words and sentences are in the correct language.

	System	German→Dutch		German→Romanian	
		dev2010	tst2010	dev2010	tst2010
(1)	Zero 2L	4.79	5.75	1.55	2.05
(2)	Zero 4L	6.31	7.93	3.15	3.73
(3)	Zero 6L	11.58	14.95	8.61	10.83
(4)	Zero 6L **Filtered Dict**	**12.50**	**16.02**	**9.10**	**11.00**
(5)	Back-Trans	17.33	20.36	12.92	15.62
(6)	Back-Trans **Filtered Dict**	**17.13**	**20.22**	**13.10**	**15.67**

TABLE 5.3: Effects of target dictionary filtering method.

Figure 5.1 shows how this strategy removes many unwanted search paths decoding process.

When we applied target dictionary filtering method, it is interesting to see its effects on different types of systems (Table 5.3). Since *Zero 2L* and *Zero 4L* do not have the language identity for words, we cannot directly apply our strategies on those systems. In contrast, it is straight-forward to use target dictionary filtering on *Zero 6L* and *Back-Trans*. On `tst2010`, *target dictionary filtering* (4) brought an improvement of 1.07 on German→Dutch. On German→Romanian zero-shot task, the improvements were not as great as on German→Dutch, but it still help, especially on `dev2010`.

Table 5.4 shows two examples where *Target Dictionary Filtering* clearly improves the quality and readability of the translation over the *Zero 6L*.

5.4 Language as a Word Feature

In order to truly alleviate the problem of having large mixed-language vocabularies, we need some way to represent the language mix better than the original universal approach.

There exist works on integrating linguistic information into NMT systems in order to help predict the output words (Sennrich and Haddow, 2016; Hoang et al., 2016; Niehues et al., 2016). In those works, the information of a word (e.g. its lemma or its part-of-speech tag) are integrated as a word features. It is conducted simply by learning the feature embeddings instead of the word embeddings. In other words, their system considers a word as a special feature together with other features of itself.

German→Dutch example	
Zero 6L	Een collega van mij had **Zugang** tot **investment** van de autoriteiten van Fox guard
Engl. gloss	A colleague of mine had **Zugang** to **investment** from the authorities of Fox guard
Filtered Dict	Een collega van mij had **toegang** tot **investeringsgegevens** van Fox guard
Engl. gloss	A colleague of mine had **access** to **investment data** from Fox guard
Reference	Een collega van me kreeg **toegang** tot **investeringsgegevens** van Vanguard
Engl. gloss	A colleague of mine received **access** to **investment data** from Vanguard
German→Romanian example	
Zero 6L	Pentru că s-ar aștepta să apelăm la medic în **nächsten** dimineață
Engl. gloss	Because he would expect to call a doctor in **nächsten** morning
Filtered Dict	Pentru că s-ar aștepta să-l chemăm pe doctori în **următorul** dimineață
Engl. gloss	Because he would expect us to call the doctors the **next** morning
Reference	Răspunsul e că cei care fac asta se așteaptă ca noi să ne sunăm doctorii în dimineața **următoare**
Engl. gloss	The answer is that people who do this expect us to call our doctors the **following** morning

TABLE 5.4: Examples of the sentences with the words in wrong languages produced by *Zero 6L* system and the corrected version produced by the same system having the target dictionary filtered in decoding phase. Target dictionary filtered is not only helpful in producing readable and fluent outputs but also clearly affects to the choices of the next words.

More specially, the embedding matrices of the encoder and decoder are the concatenation of all features' embeddings:

$$\mathbf{E} \cdot \mathbf{x} = \Big[,\Big]_{f \in F} (\mathbf{E}^f \cdot \mathbf{x}^f)$$

where $\left[,\right]$ is the vector concatenation operation, concatenating the embeddings of individual feature f in a finite, arbitrary set F of word features. The target features of each target word would be jointly predicted along the word. Figure 5.2 denotes this modified architecture.

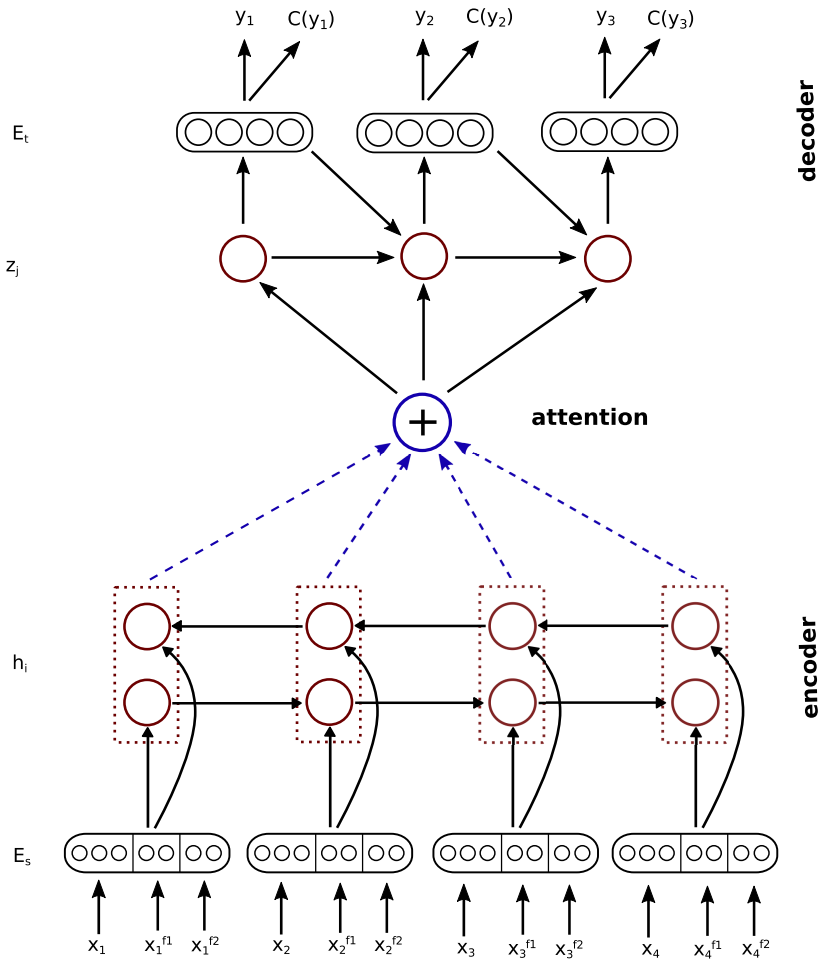


FIGURE 5.2: The NMT architecture which allows the integration of linguistic information as word features.

Inspired by their work, we attempt to encode the language information directly in the architecture instead of performing language token attachment in the preprocessing step. Being applied in our model, instead of the linguistic information at the word level, our source word features are the language of the considering word and the correct language the target sentence. The only target feature is the language of the produced word by the system. For example, when we would like to translate from the sentence "*put yourselves in my position*" into German, the features of each source word would be the word itself, e.g. "*yourselves*", and two additional features "*en*" and "*de*". Similarly, the features of the target words are the word and "*de*".

This scheme of using language information looks alike the universal approach's *language-specific coding* and *target forcing*, but the difference is the way the language information are integrated into the NMT framework. In the

	System	German→Dutch		German→Romanian	
		dev2010	tst2010	dev2010	tst2010
(1)	Zero 2L	4.79	5.75	1.55	2.05
(2)	Zero 4L	6.31	7.93	3.15	3.73
(3)	Zero 6L	11.58	14.95	8.61	10.83
(4)	Zero 6L **Lang Feature**	**13.95**	**17.15**	**9.88**	**11.37**
(5)	Back-Trans	17.33	20.36	12.92	15.62
(6)	Back-Trans **Lang Feature**	**17.48**	**20.24**	**13.43**	**15.70**

TABLE 5.5: Effects of the method used language codes as the word features in the factored architecture.

universal approach, those information are implicitly injected into the system. In this work, they are explicitly provided along with the corresponding words. Furthermore, when being used together in the embedding layers, they can share useful information and constraints which would be more helpful in choosing both correct words and language to be translated to.

During decoding, the beam search is only conducted on the target words space and not on the target features. When the search is complete, the corresponding features are selected along the search path. In our case, we do not need the output of the target language features excepts for the evaluation of language identification purpose.

When we applied *Lang Feature* on top of *Back-Trans* system, it seems that the data it used to train is sufficient to avoid the language bias problem. Thus, *Lang Feature* did not have a significant effect of performance on this system (5 vs. 6). But on the real zero-resourced system *Zero 6L*, *Lang Feature* helps to improve by notable margins. On `tst2010` of the German→Dutch direction, *Lang Feature* has achieved the gains of 2.20 BLEU scores compared to *Zero 6L* while on German→Romanian, the improvement is only 0.54 BLEU points (4 vs. 3). We can see that *Lang Feature* helps more for German→Dutch zero-resourced task than in German→Romanian task.

Considering the effectiveness of our strategy *Lang Feature* on computation perspective, which is shown in Table 5.6, we observed very positive results. We compared the *Zero 6L* configuration and our *Lang Feature* system in term of training times, size of source&target vocabularies[3] and the total number of model parameters on both zero-shot translation tasks. The models were usually trained on the same GPU (Nvidia Titan Xp) for 8 epochs so they

[3]In all cases, these sizes are similar numbers.

are fairly compared (seeing the same dataset the same number of times). Each type of models has the same configuration between two zero-shot tasks, excepts the parts related to vocabularies[4].

By encoding the language information into word features, the number of vocabulary entries reduces to almost half of the original method. Thus, it leads to the similar reduction in term of the number of parameters. This reduction allows us to use bigger minibatches as well as perform faster updates, resulting in substantially decreased training time (from 7.3 hours to 1.5 hours for each epoch in case of German→Dutch and from 6.0 hours to 1.3 hours for each epoch in case of German→Romanian). The strategy requires minimum modifications in the standard NMT framework, yet it still achieved better performance with much less training time.

German→Dutch			
System	#parameters (millions)	Vocab Size (thousands)	Training Time (hours/epoch)
Zero 6L	243	68	7.3
Lang Feature	130	28	1.5
German→Romanian			
System	#parameters (millions)	Vocab Size (thousands)	Training Time (hours/epoch)
Zero 6L	247	69	6.0
Lang Feature	122	31	1.3

TABLE 5.6: Effects on model size and training time.

We have proposed two techniques in order to release the burden of having large mixed-language vocabularies. Then we apply them to the zero-resourced tasks to reduce the severeness of language bias problem. The two techniques substantially improved the multilingual systems in terms of both performance and training resources.

5.5 Shared Components in Multilingual NMT

Motivated from the zero-resouced results, we investigate a general way to avoid large mixed-language vocabularies as well as large-sized components in our multilingual NMT architecture while keeping the ability to learn

[4]While the total number of parameters on German→Romanian is bigger than that of German→Dutch, the training time of German→Romanian systems is less due to the fact that its training corpus is smaller.

shared intermediate representation space. In the next section, we describe our investigation on the effectiveness of sharing different parts of the architecture across languages when applying NMT in multilingual settings. This work is jointly conducted by me and my colleagues, and the content of the next section in this chapter is mainly from Pham et al. (2017).

Based on the advanced computation techniques of modern Deep Learning frameworks in building complex neural architectures as computational graphs, we have implemented our own NMT framework which is very flexible when mentioned about architectural choices. In the multilingual setting, we investigate the effectiveness of sharing different parts of the model. The break-down of the neural machine translation models is illustrated as in Figure 5.3.

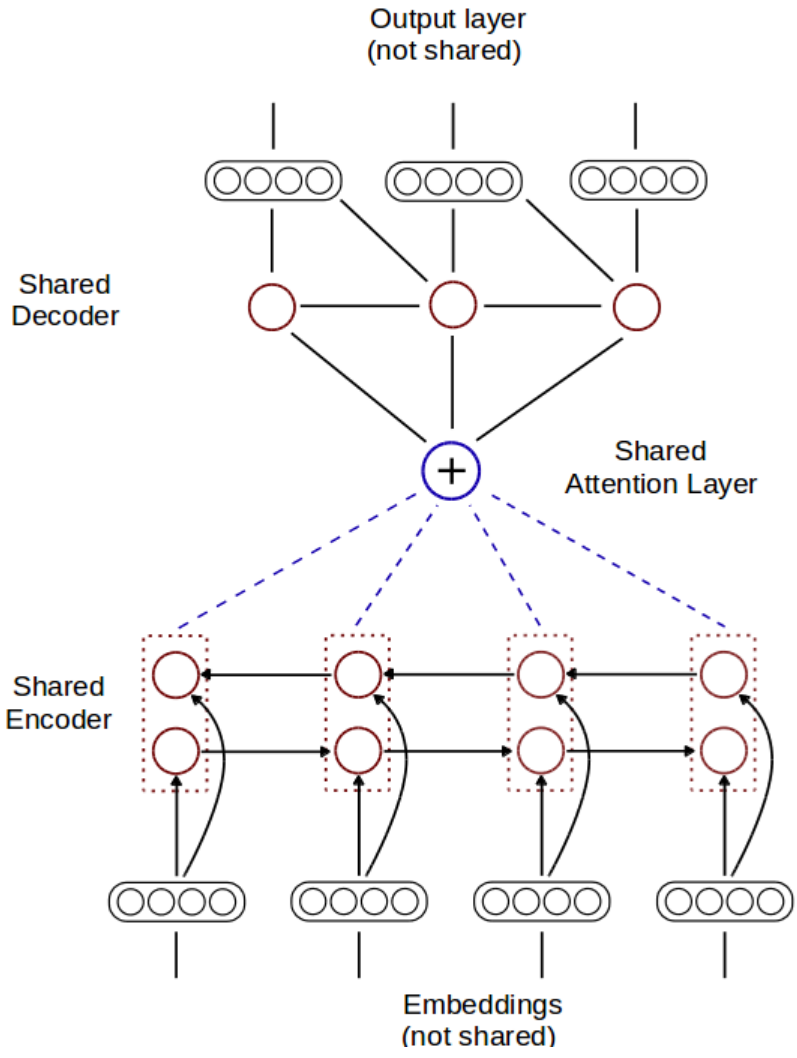


FIGURE 5.3: NMT architecture with shared components.

Our NMT encoder has been adapted so it can either share one common encoder to encode sentences (regardless of their language), or make use of one encoder per source language. We can do similar choices for the decoder layer and for the attention layer. If the attention layer is shared, the whole network is used across all language pairs, and if attention is not shared, each language pair is assigned to one attention layer.

In addition, we also considered sharing embedding layers. We chose to share one when we want to ensure that the model has the same view of the embeddings on the source and target sides, e.g. a German word on the source data has the same embedding values as the same German word on the target sentences. For the output layer which computes the probabilities of the words, there are two different scenarios. First, if we use distinct vocabularies for each language, we then end up constructing n different output layers but we can dynamically load and use only one of them in the same mini-batch training. Because of this architectural choice, each mini-batch contains only sentences from one single language pair. In the second scenario, the probability distribution of all words and all languages is computed, then the output layer is not separated (as opposed to the first scenario). This would have the same problem of large mixed-language vocabularies. The two output layer scenarios are almost equivalent, but the former is computationally much faster than the latter, because the softmax layer required for each mini-batch is considerably n-fold smaller.

5.5.1 Evaluation on IWLST'17 Multilingual Tasks

To evaluate our architectural sharing choices, we conducted our experiments on large data and small data multilingual translation tasks described in Section 5.2.1. Here is the system setup of our multilingual system.

System Description. We built a neural machine translation framework which is customized with multiple encoders-decoders-attention for this multilingual task using PyTorch [5]. For the small data task, we use a small network configuration with word embedding and hidden layer size of 512 for all experimented architectures. For the large data task, all of the models are trained with a larger config, with layer size of 1024. We applied dropout between the vertical connection of the recurrent networks with probability 0.5. We sampled minibatches containing sentences from only one language-pair

[5] http://pytorch.org/

System	tokenized BLEU	case-sensitive BLEU
Separate-All	24.7	22.6
+ Lang-adapted	25.8	24
Share-RNN	26.0	24.2
+ Lang-adapted	26.3	**24.5**
Share-All	25.2	23.5
+ Lang-adapted	26.2	24.2
Universal Multilingual	25.6	23.8
Share-All + Lang-adapted + Average	25.7	23.8
Ensemble	27.4	**25.6**

TABLE 5.7: Average BLEU scores on the test set for small task.

so that the model can observe all sentences once every epoch. The parameters are updated using Adam optimizer (**?**) with the gradients clipped at 5. We observed the training progress with the average *perplexity* on the validation sets, and used the models with the lowest perplexity to translate the test sets.

Results. For the small task reported in Table 5.7, BLEU scores on the test data show the effectiveness of RNN encoders and decoders sharing in our multilingual setups. Both the architectures with shared RNNs outperformed their Separate-All counterpart, by 0.9 and 1.6 BLEU points. For the attention mechanism, we found out that sharing the attention reduces translation performance by 0.7 BLEU. It might explain our assumption about language-dependent attention, and motivates us to not share the attention layer.

In the meanwhile, the multilingual neural model following our universal approach performed best. Unsurprisingly, the scores from that system are similar to Share-All's. Due to its expensive training and different preprocessing pipeline, however, we did not attempt to employ adaptation and ensemble on that architecture.

Moving over the large data set, we observe the same phenomenon as the small one, in which the Separate architecture fell behind the other two (Table 5.8). Interestingly, Shared-All and Shared-RNN produce the same translation performance. One reason why may be that the shared-attention mechanism requires more data to become robust to language-specific mappings.

We conducted almost all combinations of sharing choices, and the experiments show that the most effective architecture for multilingual NMT in low-resourced settings is to share encoders and decoders as well as embedding

System	tokenized BLEU	case-sensitive BLEU
Share-All	26.9	25.1
Share-RNN	26.9	25.1
Separate-All	25.1	23.4
Ensemble	27.8	**26.0**

TABLE 5.8: Average BLEU scores on the test set for large task.

layers but not the attention layer. For more details, please refer to Pham et al. (2017).

5.5.2 Evaluation on 2×24 Speech Translation System

Since the Share-All architecture alleviates the problem of large mixed-language vocabularies compared to the universal approach, we would like to deploy it to a realistic and tempting scenario: Build a multilingual system that is able to handle the translations among a large number of languages. More specifically, this system is a multy-way speech translation system which can be deployed in the lecture translator to simultaneously translate the professor's lecture into many other languages. Or we can use it as an alternative interpreting service in a multi-way meeting, e.g. European Commission meetings, to translate one's speech to many other languages at the same time.

In order to build a single neural translation model able to translate into more than twenty European languages, we choose the Share-All architecture. In principle, any multilingual approach could work. In our case, however, there are many considerations taken into account to make this feasible given the scope of our system and resource limitations in practice. Our goal is to keep the neural architecture as compact as possible while still maintaining parity with the translation quality of systems trained on individual language pairs on the same data.

Fundamentally, our system employs different softmax output layers and word embedding layers for different target languages based on their vocabularies. In this way, the system does not need to calculate over all the words from all target languages, which in this case is a huge number, but instead only considers the words in the appropriate target language. Thus, translation time is significantly reduced. While at training time we have to load all

vocabularies, at run-time we only load the language-specific ones, making memory consumption no higher than that of a bilingual translation system.

Our multilingual translation system shares its main components across languages: the encoder, the decoder and the attention layer. Sharing encoders and decoders does not only make the model considerably more economical in space, but also leverages transfer learning across language. In a similar setting, a single shared attention layer benefits from multilingual information and performs on par with the Share-All-but-not-Attention architecture mentioned before. Furthermore, when shared, a single attention layer is used across many language pairs, compared to separate $n \times m$ attention layers where m, n are the number of source and target languages, respectively[6].

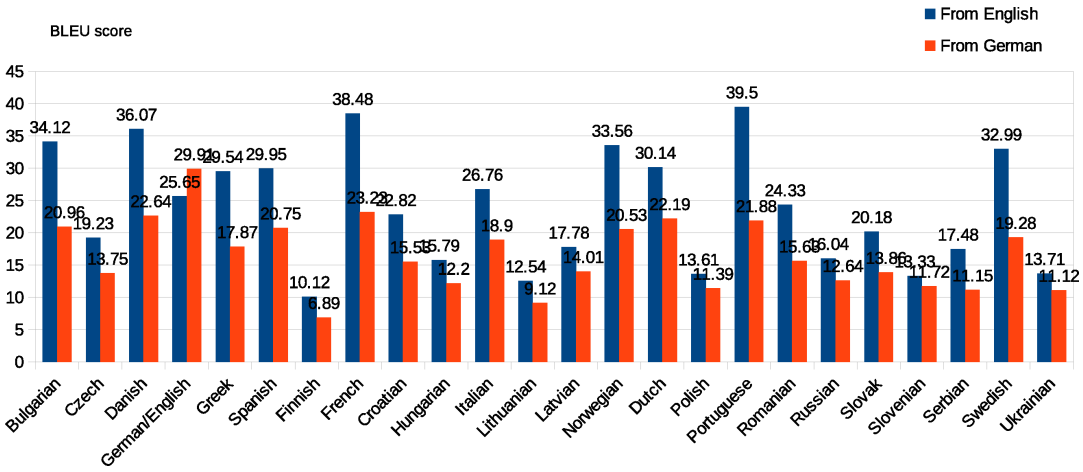


FIGURE 5.4: Performance of the system evaluated on individual language pairs when translating from English or German to 24 other European languages with a single multilingual neural translation architecture.

Figure 5.4 shows the results of the multilingual system, translating from English and German to 24 European languages using a single model trained on the multilingual data. Compared to a standard bilingual system trained on the same data, it achieves better performance: for English↔German, we see 25.65 BLEU as compared to 24.92 translating into German, and 29.91 BLEU as compared to 28.74 translating into English. The results confirm our assumption that multilingual information helps to improve low-resourced translation system trained individually. Furthermore, instead of building and training 48 separate systems, our shared architecture allows us to build a

[6] In our system, $m = 2$ and $n = 24$.

single system able to fit on a moderately-sized GPU, and translate from two languages to 24 languages with reasonable quality.

In term of BLEU, this system performs best translating from English to Portuguese and German to English, which is reasonable since there are adequate amount of corpora in those directions and related languages assist by providing more context. At the other end, the system obtains its worst results when translating into Finnish because data size and language difficulty.

Chapter 6

Conclusion and Future Work

Along with the rapid movements of neural networks and deep learning as well as their fast growing application scope, we have observed the important change in machine translation (MT), from the once-dominant Statistical Machine Translation (SMT) to the new-born-yet-already-strong neural machine translation (NMT). Furthermore, NMT offers the opportunities to push multilingual translation research to a new horizon and allows deploying those research lines in real-world applications and scenarios with efficiency.

In this thesis we have analyzed the limitations of Statistical Machine Translation (SMT) approach and investigated the influence and potential of neural-based translation models. SMT relies on simple machine learning models to conduct the translation over symbolic, discrete units staying inside the phrase boundaries. Hence, it is unable to learn complex translation relationships, discards the global context beyond phrases and cannot model long-distance dependencies well. We first suggest an advanced neural model to exploit the global contexts from the source side in order to better model the translation. Then we discussed other models which together with ours, inspire the emergence of neural machine translation.

We have observed nice properties that NMT architectures possess:

- NMT learns directly the translation process from parallel corpora without any explicit phrase or word segmentation and alignment.
- NMT is able to share the common information from the discrete units, by projecting them into a shared continuous representation space.
- NMT is able to learn hidden representations and latent hierarchical structures, so the feature learning is conducted automatically and jointly, as well as how to combine them in a complicate and effective way.

From the observations of what NMT architectures are capable of, we derived an elegant multilingual neural translation approach by forcing NMT to learn a common representation across languages and translate into the desired target language without any architectural modification. Due to its universal, our approach has been employed and evaluated thoroughly in several scenarios and various language pairs and domains. The approach shows its advantages especially on low-resourced situations.

We then proposed better techniques to improve the multilingual approach in order to deploy it in more realistic scenarios where it might involve in a large number of languages. By factoring the language information as a feature, limiting the unwanted candidates in beam search and dynamically share the NMT components, we were able to reduce the adversity of the large mix-language vocabulary problem previously existing in the original approach. The refined approach has proven its positive effects on a number of realistic multilingual translation tasks, e.g the IWSLT 20-language-direction multilingual translation IWSLT tasks or the 2×24 speech translation systems.

In addition, we have built and share a multilingual word embedding corpus which is achieved from our mult-source systems. It would be helpful in other cross-lingual natural language processing tasks.

For the future work, we would like to analyze more deeply the abilities of our multilingual neural translation and the neural translation models in general. We also aim to more effectively tackle low-resourced scenarios, where we can leverage the cross-lingual information in data augmentation settings in order to provide more data to the gready neural translation models. The multilingual word embedding corpus would be constantly expanded to have larger coverage and in more languages.

Bibliography

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Berger, A. L., Pietra, V. J. D., and Pietra, S. A. D. (1996). A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., and Bengio, Y. (2010). Theano: a cpu and gpu math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*.

Bertoldi, N., Barbaiani, M., Federico, M., and Cattoni, R. (2008). Phrase-based statistical machine translation with pivot languages. In *International Workshop on Spoken Language Translation (IWSLT) 2008*.

Bertoldi, N. and Federico, M. (2009). Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the fourth workshop on statistical machine translation*, pages 182–189. Association for Computational Linguistics.

Bertoldi, N., Haddow, B., and Fouet, J.-B. (2009). Improved minimum error rate training in moses. *The Prague Bulletin of Mathematical Linguistics*, 91:7–16.

Bojar, O., Buck, C., Federmann, C., Haddow, B., Koehn, P., Leveling, J., Monz, C., Pecina, P., Post, M., Saint-Amand, H., et al. (2015). Findings of the 2015 workshop on statistical machine translation. In *Proceedings of the tenth workshop on statistical machine translation*, pages 12–58.

Brown, P. F., Cocke, J., Pietra, S. A. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. (1990). A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85.

Brown, P. F., Pietra, S. A. D., Pietra, V. J. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Camacho-Collados, J., Pilehvar, M. T., and Navigli, R. (2015). A framework for the construction of monolingual and cross-lingual word similarity datasets. In *ACL (2)*, pages 1–7.

Cettolo, M., Federico, M., Bentivogli, L., Niehues, J., Stüker, S., Sudoh, K., Yoshino, K., and Federmann, C. (2017). Overview of the IWSLT 2017 Evaluation Campaign. In *Proceedings of the 14th International Workshop on Spoken Language Translation (IWSLT)*, Tokyo, Japan.

Cettolo, M., Girardi, C., and Federico, M. Wit3: Web inventory of transcribed and translated talks. In *Proceedings of the 16ĥ Conference of the European Association for Machine Translation (EAMT)*, pages 261–268, Trento, Italy.

Cettolo, M., Niehues, J., Stüker, S., Bentivogli, L., and Federico, M. (2014). Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*.

Chen, S. F. and Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–394.

Chen, X., Yang, J., Zhang, J., and Waibel, A. (2004). Automatic detection and recognition of signs from natural scenes. *IEEE Transactions on image processing*, 13(1):87–99.

Cheng, J., Dong, L., and Lapata, M. (2016). Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*.

Cho, E., Fügen, C., Herrmann, T., Kilgour, K., Mediani, M., Mohr, C., Niehues, J., Rottmann, K., Saam, C. h., Stüker, S., et al. (2013). A real-world system for simultaneous translation of german lectures. In *INTERSPEECH*, pages 3473–3477.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations

using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Cohn, T. and Lapata, M. (2007). Machine translation by triangulation: Making effective use of multi-parallel corpora. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 728–735.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Csáji, B. C. (2001). Approximation with artificial neural networks. *Faculty of Sciences, Etvs Lornd University, Hungary*, 24:48.

Dellaert, F., Polzin, T., and Waibel, A. (1996). Recognizing emotion in speech. In *Fourth International Conference on Spoken Language Processing*.

Dessloch, F., Ha, T.-L., Müller, M., Niehues, J., Nguyen, T. S., Pham, N.-Q., Salesky, E., Sperber, M., Stüker, S., Zenkel, T., et al. (2018). Kit lecture translator: Multilingual speech translation with one-shot learning. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 89–93.

Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R., and Makhoul, J. (2014). Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the Association for Computational Linguistics (ACL), Baltimore*.

Dong, D., Wu, H., He, W., Yu, D., and Wang, H. (2015). Multi-Task Learning for Multiple Language Translation. In *Proceedings of ACL-IJNLP 2015*, pages 1723–1732, Beijing, China. Association for Computational Linguistics.

Durrani, N., Schmid, H., and Fraser, A. (2011). A joint sequence translation model with integrated reordering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1045–1054. Association for Computational Linguistics.

El-Bakry, H. M. and Mastorakis, N. (2009). Fast word detection in a speech using new high speed time delay neural networks. *WSEAS Transactions on Information Science and Applications*, (7):261–270.

Elman, J. L. (1990). Finding structure in time. *Cognitive science*, 14(2):179–211.

Federico, M., Cettolo, M., Bentivogli, L., Michael, P., and Sebastian, S. (2012). Overview of the iwslt 2012 evaluation campaign. In *IWSLT-International Workshop on Spoken Language Translation*, pages 12–33.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., and Ruppin, E. (2002). Placing search in context: the concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.

Firat, O., Cho, K., and Bengio, Y. (2016). Multi-Way, Multilingual Neural Machine Translation with a Shared Attention Mechanism. *CoRR*, abs/1601.01073.

Freitag, M., Peitz, S., Wuebker, J., Ney, H., Durrani, N., Huck, M., Koehn, P., TL, H., Niehues, J., Mediani, M., et al. (2013). Eu-bridge mt: Text translation of talks in the eu-bridge project. In *IWSLT*, pages 128–135.

Fügen, C., Kolss, M., Bernreuther, D., Paulik, M., Stüker, S., Vogel, S., and Waibel, A. (2006). Open domain speech recognition & translation: Lectures and speeches. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing, ICASSP 2006, Toulouse, France, May 14-19, 2006*, pages 569–572.

Gale, W. A. and Church, K. W. (1993). A program for aligning sentences in bilingual corpora. *Computational linguistics*, 19(1):75–102.

Galley, M. and Manning, C. D. (2008). A simple and effective hierarchical phrase reordering model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 848–856. Association for Computational Linguistics.

Gehring, J., Auli, M., Grangier, D., and Dauphin, Y. N. (2016). A convolutional encoder model for neural machine translation. *arXiv preprint arXiv:1611.02344*.

Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N. (2017). Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*.

Ha, T.-L., Herrmann, T., Niehues, J., Mediani, M., Cho, E., Zhang, Y., Slawik, I., and Waibel, A. (2013). The kit translation systems for iwslt 2013. In *Proceedings of the 2013 International Workshop on Spoken Language Translation (IWSLT)*.

Ha, T.-L., Niehues, J., Cho, E., Mediani, M., and Waibel, A. (2015a). The kit translation systems for iwslt 2015. In *International Workshop on Spoken Language Translation (IWSLT) 2012*.

Ha, T.-L., Niehues, J., and Waibel, A. (2017). Effective strategies in zero-shot neural machine translation. *arXiv preprint arXiv:1711.07893*.

Ha, T.-L., Quoc-Khanh, D., Cho, E., Niehues, J., Allauzen, A., Yvon, F., and Waibel, A. (2015b). The kit-limsi translation system for wmt 2015. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 120–125.

Haddow, B. and Koehn, P. (2012). Analysing the effect of out-of-domain data on smt systems. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 422–432. Association for Computational Linguistics.

Hasan, S., Ganitkevitch, J., Ney, H., and Andrés-Ferrer, J. (2008). Triplet lexicon models for statistical machine translation. In *Proc. of Conference on Empirical Methods in NLP*, Honolulu, USA.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hermann, K. M. and Blunsom, P. (2014). Multilingual models for compositional distributed semantics. *Acl*, pages 58–68.

Herrmann, T., Niehues, J., and Waibel, A. (2013). Combining word reordering methods on different linguistic abstraction levels for statistical machine translation. In *Proceedings of the Seventh Workshop on Syntax, Semantics and Structure in Statistical Translation*, Altanta, Georgia, USA. Association for Computational Linguistics.

Hill, F., Reichart, R., and Korhonen, A. (2016). Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*.

Hoang, C. D. V., Haffari, R., and Cohn, T. (2016). Improving neural translation models with linguistic factors. In *Proceedings of the Australasian Language Technology Association Workshop 2016*, pages 7–14.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257.

Jaeger, S., Manke, S., Reichert, J., and Waibel, A. (2001). Online handwriting recognition: the npen++ recognizer. *International Journal on Document Analysis and Recognition*, 3(3):169–180.

Jaynes, E. T. (1957). Information theory and statistical mechanics. *Physical review*, 106(4):620.

Jiang, W., Xu, Y., Cao, D., and Luo, F. (2009). A novel nonlinear time series forecasting of time-delay neural network. In *Granular Computing, 2009, GRC'09. IEEE International Conference on*, pages 278–283. IEEE.

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F. B., Wattenberg, M., Corrado, G., Hughes, M., and Dean, J. (2016). Google's multilingual neural machine translation system: Enabling zero-shot translation. *CoRR*, abs/1611.04558.

Junczys-Dowmunt, M., Grundkiewicz, R., Dwojak, T., Hoang, H., Heafield, K., Neckermann, T., Seide, F., Germann, U., Aji, A. F., Bogoychev, N., Martins, A. F. T., and Birch, A. (2018). Marian: Fast neural machine translation in c++. *arXiv preprint arXiv:1804.00344*.

Kaiser, M. (1994). Time-delay neural networks for control. In *Proceedings of the symposium on robot control*, volume 94.

Kalchbrenner, N. and Blunsom, P. (2013). Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709.

Kalchbrenner, N., Grefenstette, E., and Blunsom, P. (2014). A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.

Kim, K. K., Kim, K., Kim, J., and Kim, H. J. (2000). Learning-based approach for license plate recognition. In *Neural Networks for Signal Processing X, 2000. Proceedings of the 2000 IEEE Signal Processing Society Workshop*, volume 2, pages 614–623. IEEE.

Kim, S.-S. (1998). Time-delay recurrent neural network for temporal correlations and prediction. *Neurocomputing*, 20(1-3):253–263.

Klein, G., Kim, Y., Deng, Y., Senellart, J., and Rush, A. M. (2016). Opennmt: Open-source toolkit for neural machine translation. *ArXiv e-prints*.

Koehn, P. (2003). Europarl: A multilingual corpus for evaluation of machine translation.

Koehn, P., Axelrod, A., Mayne, A. B., Callison-Burch, C., Osborne, M., and Talbot, D. (2005). Edinburgh system description for the 2005 iwslt speech translation evaluation. In *International Workshop on Spoken Language Translation (IWSLT) 2005*.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *ACL 2007, Demonstration Session*, Prague, Czech Republic.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *HLT/NAACL 2003*.

Kolss, M., Wölfel, M., Kraft, F., Niehues, J., Paulik, M., and Waibel, A. (2008). Simultaneous german-english lecture translation. In *2008 International Workshop on Spoken Language Translation, IWSLT 2008, Honolulu, Hawaii, USA, October 20-21, 2008*, pages 174–181.

Lang, K. J., Waibel, A. H., and Hinton, G. E. (1990). A time-delay neural network architecture for isolated word recognition. *Neural networks*, 3(1):23–43.

Le, H.-S., Allauzen, A., and Yvon, F. (2012). Continuous space translation models with neural networks. In *Proceedings of the 2012 Conference of the NAACL-HLT*, Montréal, Canada.

LeCun, Y., Bengio, Y., et al. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995.

Leviant, I. and Reichart, R. (2015). Separated by an un-common language: Towards judgment language informed vector space modeling. *arXiv preprint arXiv:1508.00106*.

Lin, C.-Y. and Och, F. J. (2004). Orange: a method for evaluating automatic evaluation metrics for machine translation. In *Proceedings of the 20th international conference on Computational Linguistics*, page 501. Association for Computational Linguistics.

Lin, D.-T., Dayhoff, J. E., and Ligomenides, P. A. (1995). Trajectory production with the adaptive time-delay neural network. *Neural Networks*, 8(3):447–461.

Lin, Z., Feng, M., Santos, C. N. d., Yu, M., Xiang, B., Zhou, B., and Bengio, Y. (2017). A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*.

Luong, M.-T., Le, Q. V., Sutskever, I., Vinyals, O., and Kaiser, L. (2016). Multi-task sequence to sequence learning. In *International Conference on Learning Representations (ICLR)*, San Juan, Puerto Rico.

Luong, M.-T., Pham, H., and Manning, C. D. (2015a). Bilingual word representations with monolingual quality in mind. *Workshop on Vector Modeling for NLP*, pages 151–159.

Luong, T., Pham, H., and Manning, C. D. (2015b). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Marcu, D. and Wong, W. (2002). A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 133–139. Association for Computational Linguistics.

Mauser, A., Hasan, S., and Ney, H. (2009). Extending statistical machine translation with discriminative and trigger-based lexicon models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, Singapore.

Mediani, M., Zhang, Y., Ha, T.-L., Niehues, J., Cho, E., Herrmann, T., Kärgel, R., and Waibel, A. (2012). The kit translation systems for iwslt 2012. In *International Workshop on Spoken Language Translation (IWSLT) 2012*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Moore, R. C. (2002). Fast and accurate sentence alignment of bilingual corpora. In *Conference of the Association for Machine Translation in the Americas*, pages 135–144. Springer.

Mozer, M. C. (1995). A focused backpropagation algorithm for temporal. *Backpropagation: Theory, architectures, and applications*, page 137.

Myers, J. L., Well, A., and Lorch, R. F. (1995). *Research design and statistical analysis*. Routledge.

Nakov, P., Guzman, F., and Vogel, S. (2012). Optimizing for sentence-level bleu+ 1 yields short translations. *Proceedings of COLING 2012*, pages 1979–1994.

Niehues, J. (2014). Adaptation in machine translation. In *PhD Thesis*. KIT.

Niehues, J. and Cho, E. (2017). Exploiting linguistic resources for neural machine translation using multi-task learning. *arXiv preprint arXiv:1708.00993*.

Niehues, J., Ha, T.-L., Cho, E., and Waibel, A. (2016). Using factored word representation in neural network language models. In *Proceedings of the First Conference on Machine Translation (WMT16)*, Berlin, Germany. Association for Computational Linguistics.

Niehues, J., Herrmann, T., Vogel, S., and Waibel, A. (2011). Wider context by using bilingual language models in machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, Edinburgh, Scotland.

Niehues, J. and Kolss, M. (2009). A pos-based model for long-range reorderings in smt. In *Fourth Workshop on Statistical Machine Translation (WMT 2009)*, Athens, Greece.

Niehues, J., Pham, N.-Q., Ha, T.-L., Sperber, M., and Waibel, A. (2018). Low-latency neural speech translation. *arXiv preprint arXiv:1808.00491*.

Niehues, J. and Waibel, A. (2013). An mt error-driven discriminative word lexicon using sentence structure features. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 512–520.

Och, F. J. (1999). An efficient method for determining bilingual word classes. In *EACL'99*.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *41st Annual Meeting of the Association for Computational Linguistics (ACL)*, Sapporo, Japan.

Och, F. J. and Ney, H. (2004). The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. Technical Report RC22176 (W0109-022), IBM Research Division, T. J. Watson Research Center.

Pham, N.-Q., Niehues, J., and Waibel, A. (2018). The karlsruhe institute of technology systems for the news translation task in wmt 2018. In *Proceedings of the Third Conference on Machine Translation (WMT 2018)*.

Pham, N.-Q., Sperber, M., Salesky, E., Ha, T.-L., Niehues, J., and Waibel, A. (2017). Kit's multilingual neural machine translation systems for iwslt 2017. In *Proceedings of the 14th International Workshop on Spoken Language Translation (IWSLT 2017)*.

Popel, M. (2018). Cuni transformer neural mt system for wmt18.

Robinson, A. and Fallside, F. (1987). *The utility driven dynamic error propagation network*. University of Cambridge Department of Engineering.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Rottmann, K. and Vogel, S. (2007). Word reordering in statistical machine translation with a pos-based distortion model. In *TMI*, Skövde, Sweden.

Rumelhart, D. E. and McClelland, J. L. (1985). On learning the past tenses of english verbs. Technical report, CALIFORNIA UNIV SAN DIEGO LA JOLLA INST FOR COGNITIVE SCIENCE.

Schenkel, M., Guyon, I., and Henderson, D. (1995). On-line cursive script recognition using time-delay neural networks and hidden markov models. *Machine Vision and Applications*, 8(4):215–223.

Schwenk, H. (2012). Continuous space translation models for phrase-based statistical machine translation. In *COLING (Posters)*, pages 1071–1080.

Sennrich, R. and Haddow, B. (2016). Linguistic input features improve neural machine translation. *CoRR*, abs/1606.02892.

Sennrich, R., Haddow, B., and Birch, A. (2016a). Improving Neural Machine Translation Models with Monolingual Data. In *Association for Computational Linguistics (ACL 2016)*, Berlin, Germany.

Sennrich, R., Haddow, B., and Birch, A. (2016b). Neural Machine Translation of Rare Words with Subword Units. In *Association for Computational Linguistics (ACL 2016)*, Berlin, Germany.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200.

Stüker, S., Kraft, F., Mohr, C., Herrmann, T., Cho, E., and Waibel, A. (2012). The kit lecture corpus for speech translation. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Upadhyay, S., Faruqui, M., Dyer, C., and Roth, D. (2016). Cross-lingual models of word embeddings: An empirical comparison. *Acl 2016*, pages 1661–1670.

Utiyama, M. and Isahara, H. (2007). A comparison of pivot methods for phrase-based statistical machine translation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 484–491.

van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Venugopal, A., Vogel, S., and Waibel, A. (2003). Effective phrase translation extraction from alignment models. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 319–326. Association for Computational Linguistics.

Vilar, D., Xu, J., Luis Fernando, D., and Ney, H. (2006). Error analysis of statistical machine translation output. In *LREC*, pages 697–702.

Vulic, I. and Moens, M.-F. (2015). Bilingual word embeddings from non-parallel document-aligned data applied to bilingual lexicon induction. *Acl-2015*, pages 719–725.

Waibel, A. (1989). Modular construction of time-delay neural networks for speech recognition. *Neural computation*, 1(1):39–46.

Waibel, A. (1997). Phoneme recognition using time-delay neural networks. In *Meeting of the Institute of Electrical, Information and Communication Engineers (IEICE), Tokyo, Japan*. Elsevier.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. J. (1990). Phoneme recognition using time-delay neural networks. In *Readings in speech recognition*, pages 393–404. Elsevier.

Weaver, W. (1955). Machine translation of languages. 14:15–23.

Werbos, P. J. (1988). Generalization of backpropagation with application to a recurrent gas market model. *Neural networks*, 1(4):339–356.

Wöhler, C. and Anlauf, J. K. (1999). A time delay neural network algorithm for estimating image-pattern shape and motion. *Image and Vision Computing*, 17(3-4):281–294.

Wu, H. and Wang, H. (2007). Pivot language approach for phrase-based statistical machine translation. *Machine Translation*, 21(3):165–181.

Zeiler, M. D. (2012). Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Zoph, B. and Knight, K. (2016). Multi-Source Neural Translation. In *The North American Chapter of the Association for Computational Linguistics (NAACL 2016)*, San Diego, CA, USA.