Exploring deep learning techniques in the task of English-Arabic Machine Translation

Mohamed Seghir Hadj Ameur

$N°$ d'ordre : 07/2020-C/INF

République Algérienne Démocratique et Populaire
Ministère de L'Enseignement Supérieur et de la Recherche Scientifique
Université des Sciences et de la Technologie HOUARI BOUMEDIENE
Faculté d'électronique et d'informatique

THÈSE

Présentée pour l'obtention du **diplôme** de **DOCTORAT**

En : INFORMATIQUE

Spécialité : Intelligence Artificielle

Par : HADJ AMEUR Mohamed Seghir

Sujet

Exploration des techniques de l'apprentissage profond au service de la traduction automatique Anglais-Arabe

Soutenue publiquement, le 29/06/2020, devant le jury composé de :

M.	AZZOUNE Hamid	Professeur	à l'USTHB	Président
M.	GUESSOUM Ahmed	Professeur	à l'USTHB	Directeur de thèse
M.	MEZIANE Farid	Professeur	à l'Univ de Salford	Co-directeur de thèse
Mme.	OUAMMOUR Sihem	Professeur	à l'USTHB	Examinatrice
Mme.	CHEROUN Hadda	Professeur	à l'Univ de Laghouat	Examinatrice
M.	ABBAS Mourad	DR	à CRSTDLA	Examinateur

$N°$ of order : 07/2020-C/INF

People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University of Science and Technology HOUARI BOUMEDIENE
Faculty of Electronics and Informatics

THESIS

Presented to obtain a **DOCTORATE DEGREE**

In : COMPUTER SCIENCE

Specialty : Artificial Intelligence

By : HADJ AMEUR Mohamed Seghir

Subject

Exploring deep learning techniques in the task of English-Arabic Machine Translation

Publicly defended on 29/06/2020, before the jury composed of:

Mr.	AZZOUNE Hamid	Professor	at the USTHB	President
Mr.	GUESSOUM Ahmed	Professor	at the USTHB	Thesis supervisor
Mr.	MEZIANE Farid	Professor	at Salford Univ	Thesis co-supervisor
Mrs.	OUAMMOUR Sihem	Professor	at the USTHB	Examiner
Mrs.	CHEROUN Hadda	Professor	at Laghouat Univ	Examiner
Mr.	ABBAS Mourad	RD	at CRSTDLA	Examiner

I would like to dedicate this thesis to my loving parents ...

Acknowledgements

First and foremost, I would like to thank the Almighty, the Most Gracious God, for His help, His protection, and His guidance throughout this journey. Without His help, this work would have never been concluded. I also use this opportunity to express my deepest gratitude to the people who have participated directly or indirectly in the completion of this thesis. I would like to express my sincere gratitude to my advisors Prof. Ahmed Guessoum and Prof. Farid Meziane for their help and support, for their patience, motivation, and vast knowledge. Their guidance helped me finish this thesis in a satisfactory manner. Besides my advisors, I would like to thank the members of my thesis committee: Prof. Hamid Azzoune, Prof. Hadda Cherroun, Prof. Sihem Ouammour, and Dr. Mourad Abbas for the suggestions they made to improve the quality of this thesis. I also want to take the time to thank Prof. Nacera Bensaou who stood by my side and helped me a lot from the moment I entered the university until the moment I defended my thesis, I also don't forget Prof. Baya Kadri and Prof. Ahmed Chawki Benchikh who also did a lot for me. Furthermore, I want to thank my dearest friends Mohamed Houcine Hadoud and Moussa Nadjib Djouda who have always inspired, motivated, and encouraged me. Finally, I would like to express my deepest affection to my dear parents, brother, and sisters, and all my loved ones for their encouragement, their patience, and their great support during all these years.

Declaration

I certify that this dissertation is the result of my own personal work. I state that every part of my thesis (sentence, paragraph, example, etc.) is done by me, except where indicated by a clear reference. I further state that no part of my dissertation has already been submitted, or is concurrently being submitted, for a degree, diploma or any other qualification at "University of Science and Technology Houari Boumediene" or elsewhere.

Hadj Ameur Mohamed Seghir
August 2020

Abstract

Given that Arabic is one of the most widely used languages in the world, the task of Arabic Machine Translation has recently received a great deal of attention from the research community. Indeed, the amount of focus that has been devoted to this task has led to some important achievements and improvements. However, the current state of Arabic Machine Translation systems has not reached the quality achieved for some other languages such as English and French. In this thesis, we are interested in the task of English-to-Arabic Machine Translation for which we propose several contributions: First, we propose a method that handles both long- and short-distance word reorderings in the context of English-to-Arabic Statistical Machine Translation (SMT). Secondly, we propose a method for named entity transliteration that can accurately transliterate English named entities into Arabic. Finally, our main contribution concerns re-ranking the n-best list in the context of English-to-Arabic Neural Machine Translation (NMT). Our solution uses a set of sophisticated features that cover lexical, syntactic and semantic aspects of the n-best list candidates. All our contributions are evaluated carefully and the results obtained for the tests we have carried out show the effectiveness of our proposals.

Keywords: Natural language Processing, Machine Translation, Deep Learning, Arabic Language

الملخص

نظرا لأن اللغة العربية هي واحدة من أكثر اللغات إنتشارا في العالم، فإن عملية الترجمة الآلية من وإلى اللغة العربية قد حظيت في الآونة الأخيرة باهتمام كبير في مجال البحث العلمي. هذا الإهتمام ساعدعلى تحسين نتائجها بشكل ملحوظ، ومع ذلك لحد الان جودة هذه الترجمة لم تصل الى نفس النوعية المحققة بالنسبة للغات اخرى مثل الإنجليزية والفرنسية. في هذه الأطروحة، نحن مهتمون بمهمة الترجمة الآلية من الإنجليزية إلى العربية ولتي في خضمها قمنا بإقتراح عدة مساهمات: أولا، اقترحنا طريقة للتعامل مع مشكل الإختلاف النحوي الموجود في تركيب جمل كل من اللغتين العربية والإنجليزية، ثانياً، اقترحنا طريقة تعتمد على أسلوب التعليم العميق لترجمة أسماء الأعلام بين اللغتين العربية والإنجليزية. أخيرا، مساهمتنا الرئيسية تتمثل في إعادة ترتيب الترجمات الآلية المتحصل عليها من نضام ترجمي عميق بلإعتماد على مجموعة من الخواص التركيبية والنحوية. فيما يتعلق بنتائج أبحاثنا العلمية فقد قمنا بتجربة وتقييم جميع مساهماتنا بعناية وبدقة وأثبتت نتائج هذا التقييم فعالية ونجاعة مقترحاتنا.

الكلمات الدالة: معالجة اللغات الطبيعية ، الترجمة الآلية ، التعلم العميق ، اللغة العربية

Résumé

Etant donné que l'Arabe est l'une des langues les plus utilisées au monde, la traduction automatique de la langue Arabe a reçu récemment beaucoup d'attention de la part des chercheurs. En effet, la quantité d'attention consacrée à cette tâche a conduit à d'importantes contributions et améliorations. Toutefois, l'état actuel des systèmes de traduction automatique de la langue Arabe n'atteint pas celles d'autres langues telles que l'Anglais et le Français. Dans cette thèse, nous nous sommes intéressés à la tâche de la traduction automatique de l'Anglais vers l'Arabe, et pour laquelle nous avons proposé plusieurs contributions. Premièrement, nous avons proposé une méthode qui gère les différences syntaxiques qui existent entre l'Anglais et l'Arabe. Deuxièmement, nous avons proposé une méthode de translittération d'entités nommées capable de translittérer avec précision les entités nommées de l'Anglais vers l'Arabe. Enfin, notre principale contribution concerne le réordonnancement de la liste des n-meilleures traductions dans le contexte d'un système de traduction automatique basé sur les réseaux de neurones profonds en utilisant un ensemble de caractéristiques lexicales, syntaxiques et sémantiques. Toutes nos contributions sont évaluées avec soin et les résultats obtenus pour les tests que nous avons effectués montrent l'efficacité de nos propositions.

Mots-clés : Traitement Automatique des Langues, Traduction automatique, Apprentissage Profond, Langue Arabe

Table of contents

List of figures

List of tables

Chapter 1

Introduction and Motivation

1.1 Brief History of MT

Machine Translation (MT) is a sub-field of Natural Language Processing (NLP) devoted to the development and enhancement of computer-based machine translation systems. The goal of an MT system is to automatically translate a given textual content from one language to another in a way that best preserves its meaning and style while ensuring that the produced translation is as linguistically fluent as possible.

Machine Translation has a very long history of creativity, research, and ambition. The idea of automatic translation was first proposed in the late 1940s by Warren Weaver the director of the Natural Sciences Division at the Rockefeller Foundation. He suggested the use of cryptography techniques and statistical methods to perform automatic translation between languages in a memorandum entitled "Translation", that he wrote in July 1949 (Hutchins, 1995). In 1952, and after a few years of research at several US centers, new proposals to deal with different MT problems such as morphology and syntax were made. Furthermore, the researchers recognized that human assistance for post-editing MT outputs is needed until a human-level translation is achieved. A demonstration of an MT system was held in 1954 by a group of researchers belonging to both Georgetown University and IBM research center. Their system was used to translate from Russian to English using very limited source and target vocabularies of around 250 words each, and a few grammar rules. Though this first translation system was very basic it allowed demonstrating the task of MT in a practical fashion (Hutchins, 1995).

Since then, the evolution of MT benefited from several factors such as the increase of computing power, the availability of large parallel corpora, and also the rapid

progress in the field of computer science and artificial intelligence (Hutchins, 1995)[1]. These factors have led to the emergence of Statistical Machine Translation (SMT) (Koehn, 2009), the approach that has dominated the field of Machine Translation for the last two decades. Currently, another paradigm called Neural Machine Translation (NMT) (Bahdanau et al., 2014) has also emerged and managed to push the boundaries of machine translation quality even further. These technologies have been used to build large-scale translation systems such as the well known Google [2] and Microsoft [3] translators which have provided the Internet users with high-quality instant translation services across several language pairs. A glimpse of the current translation quality that has been obtained on the task of MT between English and several Indo-European and Asian languages is provided in Figure 1.1[4].

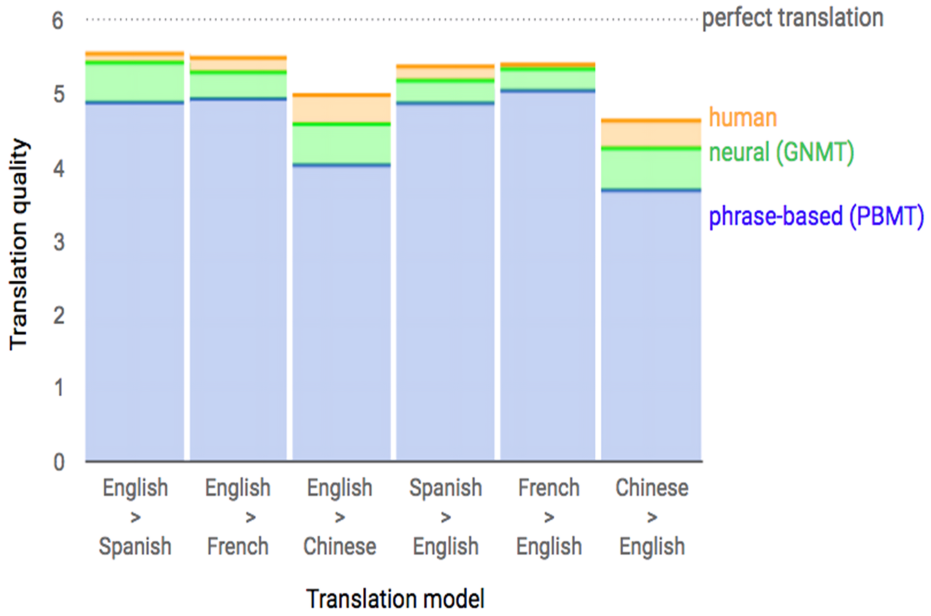


Fig. 1.1: A comparison between Google's Neural Machine Translation (GNMT) and the Phrase-based Machine Translation (PSMT) systems on the task of MT between English and several Indo-European and Asian languages (Wu et al., 2016)

Figure 1.1 shows the considerable gain that has been achieved when using neural instead of phrase-based statistical machine translation. It also shows that Google's NMT results attained near the human-level quality in the task of French-to-English

[1]For a detailed history of machine translation, we point the reader to Hutchins (1986, 1995); Hutchins and Somers (1992).

[2]https://translate.google.com

[3]https://www.bing.com/translator/

[4]Figure 1.1 is taken from the Google AI Blog describing the paper of Wu et al. (2016) https://ai.googleblog.com/2016/09/a-neural-network-for-machine.html

translation. Though near human translation quality is achieved in the task of MT that involves close languages such as English and French or English and Spanish, the translation between distant languages such as English-Chinese is still not as good (Fig. 1.1).

1.2 Arabic MT

Arabic is one of the five most spoken languages in the world with more than 300 million native speakers [5] and one of the six official languages of the United Nation (UN) [6]. It is a Semitic language that is well known for its rich and complex morphology which is substantially different from that of Indo-European languages (such as English and French). The morphology added to other linguistic aspects of Arabic has made the automatic translation from and to Arabic a lot more challenging. Indeed, even though a great deal of improvement has been achieved due to the recent advances in data-driven translation paradigms (e.g. statistical and neural methods), these Arabic linguistic aspects are still causing many difficulties (Alkhatib and Shaalan, 2018; Habash and Sadat, 2006).

The below example shows the quality of current translation systems (ex. Google Translate) when dealing with a fairly simple Arabic passage taken from a free online language portal[7].

Arabic Sentence:

تشجع لجنة بناء السلام التابعة للأمم المتحدة على دعم الجهود الرامية إلى تحقيق
الاستقرار السياسي والتعمير في البلدان بعد انتهاء الصراع

Reference Translation:

The United Nations Peacebuilding Commission encourages support efforts for the political stabilization and reconstruction of post-conflict countries.

Google Translation:

The UN Peacebuilding Commission encourages support for efforts to achieve political stability and reconstruction in post-conflict countries.

[5]https://www.conservapedia.com/List_of_languages_by_number_of_speakers
[6]http://www.un.org/en/sections/about-un/official-languages/
[7]https://en.bab.la/company/

Though the current Arabic MT systems can give fairly good translation results especially when translating from and to the English language (as shown in the above example), much work is still needed to achieve near human-level translation quality. To illustrate the difficulty of Arabic MT, the below example shows the results of translating a verse taken from an Arabic poem that was written by "Imru' al-Qais" (امْرُؤُ الْقَيْسِ) in the Pre-Islamic era [8].

Arabic Sentence:

"قِفَا نَبْكِ مِنْ ذِكْرَى حَبِيبٍ ومَنْزِلِ :: بِسِقْطِ اللِوى بَيْنَ الدَخولِ فَحَوْمَلِ"

Reference Translation:

"Stop, oh my friends, let us pause to weep over the remembrance of my beloved"
::
"Here was her abode in Lawa between Dakhool and Howmal"

Google Translation:

"Nuba from the memory of Habib and the house :: the fall of Alloi between the entry Fhmall"

The above example shows the limits of current translation tools when dealing with fairly difficult Arabic texts such as poetry. It proves that though the task of Arabic machine translation has recently received a great deal of attention from the research community, the current state of Arabic machine translation systems has not reached the quality achieved for some other languages, which means that much research work is still needed to improve it. For more details about the Arabic language characteristics and the difficulties involved in its translation, we refer the reader to Chapter 4.

1.3 Motivation

The approaches that have been used in the field of natural language processing have encountered three major eras: the rule-based, the empirical, and the deep learning ones (Deng and Liu, 2018). The first era has focused on using experts' knowledge (ex. handcrafted rules) to solve problems. The second era focused on using statistical

[8]The English reference translation is taken from https://blogs.harvard.edu/sulaymanibnqiddees/2013/03/04/hafez-and-imrul-qays/

probabilistic-based methods to build statistical models from training data. The third era, appeared with the rise of deep learning and just like the previous one the models are built from training data, however, the capability of these models to benefit from the huge amount of available corpora and its superior feature extraction aspects have helped it to surpass the empirical-based methods on several NLP applications (Deng and Liu, 2018). Motivated by the recent advances of deep learning in the field of NLP in general and in MT specifically, and also by the fact that Arabic MT still needs much more attention, we have decided to explore this new paradigm of deep learning to improve the current Arabic MT results. The specific goal of this research is to push the current Arabic-to-English and English-to-Arabic results further and to help the research community working on Arabic MT to advance at a faster pace toward achieving human-level translation quality.

1.4 Aim and Objectives

The overall aim of this thesis is to propose new contributions that can advance and improve the quality of current English-to-Arabic (and also Arabic-to-English) machine translation systems. We focus on the two currently used translation paradigms namely the statistical and the neural-based translation approaches. To this end, we have defined three main objectives:

1. Find a way to deal with the syntactic difference that exists between the English and Arabic languages which pose a serious challenge to the current statistical MT systems.

2. Find a way to handle the problem of Arabic named entity transliteration which is extremely important in the context of MT.

3. Find a way to improve the current translation quality of Neural based MT systems.

1.5 Research Methodology

The research methodology that we have planned and followed during the lifetime of this thesis is summarized in the below-mentioned steps:

1. Covering the background notions in the fields of Deep Learning, and Machine Translation.

2. In-depth reading about Arabic language processing and the challenges involved in it.

3. In-depth converging of the state-of-the-art methods regarding the field of machine translation and specify Arabic MT.

4. Identify the specific subjects (sub-problems) of research on which we will try to find new ideas and propose contributions.

5. Propose several contributions in regards to both statistical and neural MT in the selected subjects and evaluate their proposed contributions rigorously.

6. Publish the contributions in domain-specialized and reputable conferences and journals.

7. Finish the writing of the Ph.D. thesis with the necessary proofing and corrections and defend my Ph.D.

1.6 Problem Statement and Contributions

The goal of this thesis is to improve the quality of current English-to-Arabic MT which still needs a lot of attention to achieve the same quality that has been obtained for other language pairs such as English-to-French. This thesis focuses mainly on the exploration of deep learning techniques. However, it also presents some contributions regarding statistical methods. Our contributions are as follows:

- Our first contribution tackles the problem of word order that exists between distant languages such as English and Arabic and which leads to a degradation in the result of statistical MT. In this regard, we propose a word reordering method that efficiently handles both long- and short-distance word reordering phenomena in the context of an English-to-Arabic statistical MT. The proposed method can automatically learn reordering rules and incorporate them to change the syntactic word order of the source sentences, making them as close as possible to the target ones. We show that this preordering process does lead to a noticeable improvement in the overall translation results.

- Our second contribution deals with the translation of named entities which is also an important problem in MT. To this end, we propose a transliteration system that can accurately transliterate English named entities into Arabic.

- The third and most important contribution concerns the problem of re-ranking MT system outputs. We present a method that re-ranks the n-best list candidates via a set of sophisticated features. The features we propose cover the lexical, syntactic and semantic aspects of the n-best list candidates. The weights of these features are automatically optimized via a swarm-based optimization algorithm.

1.7 Thesis Overview

The remainder of this thesis is structured as follows:

- Chapter 2 presents deep learning methods and their applications in the field of Natural Language Processing (NLP).
- Chapter 3 presents all the paradigms that have been proposed to build machine translation systems. It also presents the MT evaluation task, the different ways to perform it, and the methods and metrics involved in it.
- Chapter 4 introduces the Arabic language, its characteristics, and the difficulties involved in its translation. It also gives as a comprehensive review as possible of the research works that have tackled Arabic MT along with a discussion about the limitations that still exist in this area.
- Chapter 5 talks about the first contribution of this thesis namely the syntactic reordering system. The approach is proposed to address the problem of word ordering differences that exist between the Arabic and English languages. The method is tested on an English-to-Arabic statistical translation system.
- Chapter 6 presents the second contribution of this thesis regarding named entity transliteration in which we propose a deep learning model to accurately transliterate English named entities into Arabic.
- Chapter 7 presents our final and main contribution namely a system for n-best list re-scoring in the context of English-to-Arabic NMT. In this contribution, we propose a set of features that cover the lexical, syntactic and semantic aspects of the translation candidates (the n-best list candidates). We also use a Quantum-behaved Particle Swarm Optimization (QPSO) algorithm to optimize the weights of these features.
- Chapter 8 is the conclusion of this thesis in which we state our main findings and shed light on some possible research directions.

1.8 Published Work

The material presented in our Ph.D. thesis has been published in various papers as follows:

- Chapter 5 which presents an approach for the syntactic word reordering in the context of a statistical English-to-Arabic translation system is published in the International Conference on Arabic Language Processing (ICALP 2017) with the title "A POS-based preordering approach for English-to-Arabic statistical machine translation" (Hadj Ameur et al., 2017a).
- Chapter 6 which presents an approach for the transliteration of English named entities into Arabic is published in the Third International Conference on Arabic Computational Linguistics (ACLing 2017) with the title "Arabic Machine Transliteration using an Attention-based encoder-decoder Model" (Hadj Ameur et al., 2017b).
- Chapter 7 which presents our main contribution regarding n-best list re-scoring is published in the Springer Machine Translation journal under the title "Improving Arabic neural machine translation via n-best list re-ranking" (Hadj Ameur et al., 2019).

I have also worked on related NLP problems that have given me a better understanding of Arabic language processing namely:

- Arabic Text Diacritization: This is the process of assigning Arabic diacritics to a given text. I have published a conference paper in which I proposed a vocalization system for the Arabic language using a multi-level statistical HMM model. This work is published in the International Conference on Computer Science and its Applications (IFIP 2015) with the title "Restoration of Arabic diacritics using a multilevel statistical model" (Hadj Ameur et al., 2015).
- Arabic WordNet Enrichment: which interests in enriching the current Arabic WordNet by inserting new synsets, lemmas, vocalizations, etc. In this regard, I proposed an automatic approach and used it to enrich the existing Arabic WordNet. This work is published in the International Conference on Arabic Language Processing (ICALP 2017) with the title "An Automatic Approach for WordNet Enrichment Applied to Arabic WordNet" (Hadj Ameur et al., 2017a).

Chapter 2

Deep Learning: Background and Usage in NLP

2.1 Introduction

In the last few years, deep learning models have made noticeable improvements in various fields such as computer vision, natural language processing, and bioinformatics. The advantage of these models is their high capacity to learn meaningful features from data in a fully automated way without needing any task-specific features engineering. In this chapter, we present some background notions that are necessary for a better understanding of deep learning and its usage in the NLP field. We start by introducing the field of machine learning in a very brief manner, then we introduce the three most important and heavily used neural network architectures, namely: feed-forward, convolutional, and recurrent neural networks. Then we present the concept of deep learning and show its impact on the field of NLP.

2.2 Machine learning

Machine learning (ML) is a subfield of artificial intelligence devoted to developing algorithms and techniques which help to understand, extract, learn, and discover important information from data in an automated way without the need for explicit programing (Kelleher et al., 2015). In the remainder of this subsection we will present

two important machine learning approaches that are relevant to our study, namely: supervised and unsupervised learning [1].

2.2.1 Supervised Learning

When writing a classic computer-based algorithm, the steps (instructions) that are needed for solving it need to be mentioned explicitly, thus it requires a good knowledge about the problem. Supervised learning approaches, on the other hand, do not require any explicit knowledge about the problem; instead, they use the available input-to-output labeled data to automatically train (infer) a model that can map any given data point from the input space to the output space.

Formally, given a labeled set of samples $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ of size n, where x_i is a data point from the input space X and y_i is its corresponding output (label) from the output space Y. Supervised learning attempt to find a function $f : X \rightarrow Y$ (a model) that predicts/approximates the output of any given input $x \in X$ such that $f(x) \approx y$ for any given $x \in X$.

In order to find the best approximating function $\hat{f}$ from the space of all possible functions F, a loss function Q (Eq. 2.1) is used to evaluate the output of any given function $f \in F$.

$$Q(f(x_i), y_i) = e_i, \quad e_i \in \mathbb{R} \tag{2.1}$$

The function Q estimates the difference (or the error e_i) between the predicted output $f(x_i)$ and the reference output y_i of the function f for a given input data x_i.

The best function $\hat{f}$ is defined as the function that has the lowest empirical risk R which is simply the result of averaging the loss function Q on all the training data set (Eq. 2.2).

$$R(f) = \frac{1}{n} \sum_{i=1}^{n} Q(f(x_i), y_i) \tag{2.2}$$

This optimization problem that attempt to find $\hat{f}$ by minimizing the empirical risk is known as Empirical Risk Minimization (ERM) (Vapnik, 1992) (Eq. 2.3.

$$\hat{f} = argmin_{f \in F}(R(f)) \tag{2.3}$$

[1] We note that this section covers only a few aspects that are relevant to our study, thus it merely scratches the surface on this field.

2.2.2 Unsupervised Learning

Unsupervised learning algorithms aim at learning hidden structures (e.g. features or patterns) from unlabeled data. There are two main categories of unsupervised learning namely clustering and association rule learning.

- Clustering: aims at grouping similar data points (objects) together in groups called clusters. The data points that belong to the same cluster should be more similar to those found in other clusters (Han et al., 2011). Clustering can be performed via a number of different algorithms that differ in the way they organize and construct their clusters. Some of the most commonly used clustering algorithms are K-means (Han et al., 2011), Mean-Shift (Cheng, 1995), Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (Ester et al., 1996), Expectation-Maximization (EM) (Dempster et al., 1977), and Agglomerative Hierarchical Clustering (Day and Edelsbrunner, 1984).
- Association rule learning: aims at discovering the frequent associations and correlations that exist between data (objects) (Han et al., 2011). An example of that is to discover that the clients who bought product X tend to also buy products Y and Z in a given store. Some of the most used association rule learning algorithms are Apriori (Agrawal et al., 1994) and FP-Growth (Han et al., 2011).

2.3 Artificial Neural Networks

Artificial Neural Networks are computer-based learning algorithms that attempt to mimic the biological human brain by using a large number of interconnected processing units (called artificial neurons) that work together to solve problems (Goodfellow et al., 2016). Neural Networks can learn the necessary features to solve a specific task directly and automatically from data without needing to be explicitly programmed to do so.

In the remainder of this subsection, we will present the three most important types of neural networks namely: feed-forward, convolutional, and recurrent neural networks.

2.3.1 Feed-forward Neural Networks

Feedforward Neural Network (FFNN) (Goodfellow et al., 2016) also known as Multilayer Perceptrons (MLP) is an artificial neural network composed of an input layer, an output layer, and at least one hidden layer in which the information flows only in the forward

direction from the input layer toward the output layer. This kind of neural network has a specific property in which each neuron of a given layer is connected to all the neurons of its subsequent layer. A non-linear activation function (also known as transfer function) such as Hyperbolic tangent (Tanh) is applied to the output of all the neurons belonging to each layer. Figure 2.1 shows a simple example of a FFNN which contains an input layer, a hidden layer, and an output layer.

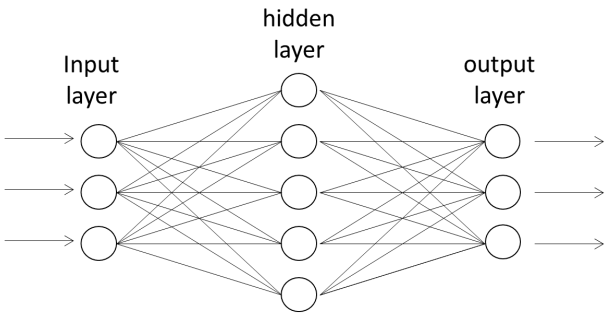


Fig. 2.1: Architecture of a Feed-forward Neural Network

FFNN can be modeled by using algebraic operations that involve matrix and vector multiplications and the application of non-linear functions. In a formal way, we can define a FFNN having k layers using Equation 2.4:

$$f_\theta(x) = \Phi_k(\Phi_{k-1}(...\Phi_2(\Phi_1(x)))) \tag{2.4}$$

where f is the function that represents our neural network, x is the input of the neural network, θ is the set of all the parameters (weights) of the network, and Φ_i is a function that estimates the output of the i^{th} layer using equation 2.5:

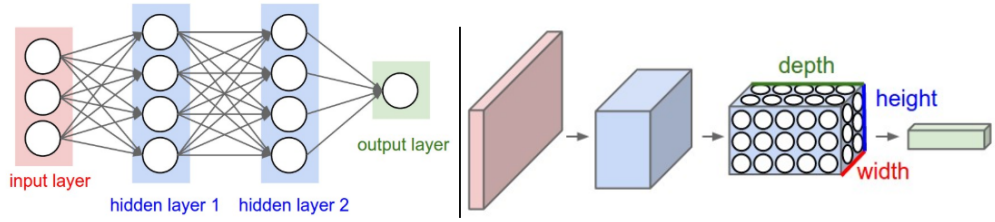
$$\Phi_i(x) = h(W_i * x + b_i) \tag{2.5}$$

W_i is the matrix of weights (connections) that exist between the $(i-1)^{th}$ and the i^{th} layers, b_i is the bias of the i^{th} layer, and h is a point-wise activation function (e.g. the sigmoid function).

2.3.2 Convolutional Neural Networks

In an FFNN architecture (as we have shown in the previous section) a neuron that is found in a given layer is connected to all the neurons of the subsequent layer. Convolutional Neural Networks (CNNs) (Goodfellow et al., 2016) use a different architecture in which a neuron of a given layer is connected only to a small number

of neurons (a region) of its previous layer. This idea of connecting neurons to only a limited region is inspired by the functioning mechanism of the mammals visual cortex where only a small group of cells is sensitive to some specific region of the visual field (Hubel and Wiesel, 1968). The CNN architecture is designed specifically to deal with imagery data; thus, unlike a regular FFNN, the layers of a CNN arrange its neurons in 3 dimensions: width, height, and depth which are the 3 specific dimensions of imagery data (Skansi, 2018). The difference between the architecture of a standard FFNN and a CNN is emphasized in Figure 2.2 (Karpathy, 2019).



Left: A regular 3-layer Neural Network. Right: A ConvNet arranges its neurons in three dimensions (width, height, depth), as visualized in one of the layers. Every layer of a ConvNet transforms the 3D input volume to a 3D output volume of neuron activations. In this example, the red input layer holds the image, so its width and height would be the dimensions of the image, and the depth would be 3 (Red, Green, Blue channels).

Fig. 2.2: Architecture of a Convolutional Neural Network (Karpathy, 2019)

A CNN architecture involves three main building layers: a convolutional layer, a pooling layer, and a fully connected layer.

- Convolution layer: this layer applies convolution operations to the input volume. Each neuron is connected only to a local area of the input volume, the size of this area is called the "receptive field" of the neuron. The goal of using convolutions is to learn a set of useful filters that can detect various types of visual imagery features (e.g. edges).

- Pooling layer: this layer is generally used after the convolutional layer to perform a down-sampling operation which reduces the size of the representation that has been produced from the convolutional operation.

- Fully connected layer: this layer is the same as FFNN in which each neuron on the layer is connected to all the neurons of the subsequent layer. The fully connected layer is used as a classifier which uses the features that have been extracted in the previous layers (convolutional and pooling layers) to find the most probable class for a given input data (e.g. an image). This layer is generally

used along with a Softmax activation function (Nasrabadi, 2007) to compute the class scores for all the available categories.

The most common way of building a CNN architecture involves two parts:

1. The Feature Learning Part: this part uses multiple Convolutional-Pooling layers to produce a small spatial representation of the most important features.

2. The Classification Part: this part is used just after the feature learning part; it consists of several fully-connected layers which work together as a classifier to predict the most suitable category for the input.

2.3.3 Recurrent Neural Networks

Recurrent Neural Network (RNN) (Bengio et al., 2015; Williams and Zipser, 1989) is an artificial neural network that is designed specifically to handle sequential data in which each piece of information depends on its preceding ones.

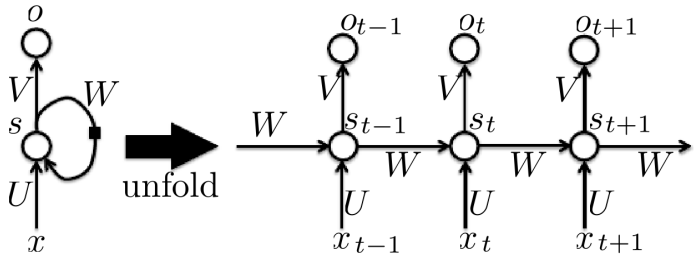


Fig. 2.3: Architecture of a Recurrent Neural Network (RNN) (Bengio et al., 2015)

Figure 2.3 provides the architecture of a Recurrent Neural Network (RNN). The left hand side of the figure shows the vanilla recurrent network, and on the right hand side, the same RNN is shown in its time-unfolded flow graph.

Formally, when given a time-series input sequence $X = \{x_1, x_2, ..., x_d\}$ of length d, where each symbol depends on its preceding ones (the order of the symbols is important), the RNN scans X from left to right. At each time step t, the RNN summarizes the whole sequence of symbols $\{x_1, x_2, ..., x_t\}$ up till x_t in what is called a "state" s_t. Thus, at the end of the sequence (at time step d), the RNN will have a summary of the whole input sequence X in its last hidden state s_d. At each time step t the RNN receives an input x_t and estimates its current state s_t on the basis of its previous state s_{t-1} (Eq. 2.6) and produces an output o_t (Eq. 2.7).

$$s_t = f(\mathbf{W}s_{t-1} + \mathbf{U}x_t) \tag{2.6}$$

$$o_t = softmax(\mathbf{V}s_t) \tag{2.7}$$

where U, V, and W are the RNN's weight matrices, and f is a nonlinear activation function such as hyperbolic tangent (Tanh). The output o_t is produced using an activation function such as $softmax$.

Recurrent Neural Network (RNN) architectures suffer from some serious limitations, especially when dealing with long-term dependencies. A common solution is to use either the Long Short-term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) or the Gated Recurrent Unit (GRU) (Cho et al., 2014a) neural networks which will be presented in the next subsections.

2.3.3.1 Gated Recurrent Unit (GRU)

Unlike a standard RNN, a GRU unit (Cho et al., 2014a) does not use an activation function directly in each recurrent component; instead, two gates are used to control the flow of information throughout the network. A graphical illustration of the Gated Recurrent Unit is presented in Figure 2.4.

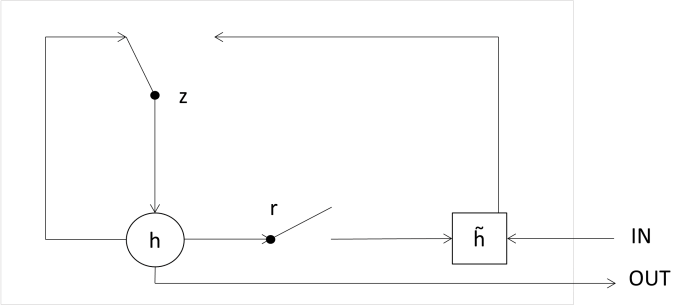


Fig. 2.4: Graphical illustration of the Gated Recurrent Unit (Cho et al., 2014a)

As illustrated in Figure 2.4, the first gate of the GRU unit is called a reset gate r, which is responsible for combining the previous memory with the new input, allowing the cell to remember or forget information as needed. The second gate is an update gate z, which decides how much information needs to be kept from the previous memory. These two gates can take values between 0 and 1, where a value of zero indicates that the gate is off, and a value of 1 indicates that the gate is on.

Given an input sequence $X = \{x_1, x_2, ..., x_d\}$, the activation h_t at time-step t is calculated by combining the previous activation h_{t-1} with a candidate activation $\tilde{h_t}$ (Eq. 2.8).

$$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t \tag{2.8}$$

The candidate activation $\tilde{h_t}$ is estimated using Eq. 2.9.

$$\tilde{h_t} = \sigma(W x_t + U(r_t \circ h_{t-1})) \tag{2.9}$$

The update gate z_t and the reset gate r_t are calculated using Eq. 2.10 and Eq. 2.11, respectively.

$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \tag{2.10}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{2.11}$$

where W_z, U_z and W_r, U_r are the weight matrices corresponding to the update and reset gates, $\circ$ is an element-wise multiplication, and σ is a logistic sigmoid activation function.

2.3.3.2 Long Short-term Memory

Long Short-term Memory (LSTM) RNN architecture (Hochreiter and Schmidhuber, 1997; Patterson and Gibson, 2017) is also proposed to address limitations of the classical RNNs. The LSTM network is known to have a memory that allows it to remember long term past events. An LSTM unit uses three gating units: an input gate, an output gate and a forget gate. Figure 2.5 shows a graphical illustration of the Long Short-term Memory Unit (Britz, 2016).

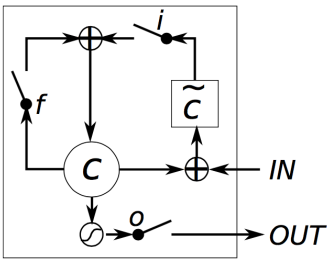


Fig. 2.5: A graphical illustration of the Long Short-term Memory Unit (Britz, 2016)

Given an input sequence $X = \{x_1, x_2, ..., x_d\}$, the LSTM unit uses the current input symbol/word x_t at time-step t and the previous hidden state h_{t-1} to generate a memory $\tilde{c_t}$:

$$\tilde{c_t} = tanh(W_c\ x_t + U_c\ h_{t-1}) \tag{2.12}$$

The new memory c_t is then calculated by applying the result of the forget and the input gates to the past memory c_{t-1} and the current memory $\tilde{c_t}$, respectively (Eq.

2.13). Then,the memory c_t is used to find the new hidden state h_t (Eq. 2.14).

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c_t} \tag{2.13}$$

$$h_t = o_t \circ tanh(c_t) \tag{2.14}$$

The three gates of the LSTM unit are calculated as follows:

- The input gate takes the current input symbol and the previous hidden state and decides to what extent it should preserve the input or not:

$$i_t = \sigma(W_i \; x_t + U_i \; h_{t-1}) \tag{2.15}$$

- The forget gate takes the current input symbol and the previous hidden state and decides whether to use the previous memory cell to compute the current memory cell or not:

$$f_t = \sigma(W_f \; x_t + U_f \; h_{t-1}) \tag{2.16}$$

- The output gate decides the parts of the memory cell c_t that needs to be exposed/present when computing the new hidden state h_t:

$$o_t = \sigma(W_o \; x_t + U_o \; h_{t-1}) \tag{2.17}$$

where $W_i, W_f, W_o, U_i, U_f, U_o$ are the weight matrices corresponding to the three gatings units, $\circ$ is an element-wise multiplication, and σ is the logistic sigmoid activation function.

2.3.3.3 Bidirectional Recurrent Neural Networks

Bidirectional Recurrent Neural Networks (BiRNNs) (Schuster and Paliwal, 1997) are an extension of the standard RNNs consisting of forward and backward RNN cells that scan the input sequence in the direct and reversed directions. The general architecture of a Bidirectional Recurrent Neural Network is shown in Fig. 2.6.

Given an input sequence $X = \{x_1, x_2, ..., x_d\}$, the Bidirectional Recurrent Neural Network calculates both the forward hidden state $\overrightarrow{h}$ and the backward hidden state $\overleftarrow{h}$ resulting from traversing the input sequence X from its direct and reversed direction respectively. Then the two hidden states $\overrightarrow{h}$ and $\overleftarrow{h}$ are combined to generate an output y_t at time step t.

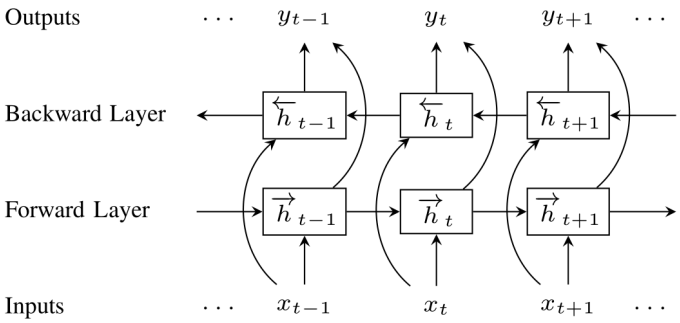


Fig. 2.6: Architecture of a Bidirectional Recurrent Neural Network (Wang et al., 2015b)

The most interesting thing about this BiRNN architecture is that at the level of each time step t, a summary of the whole input sequence surrounding that step (its left and right contexts) is obtained.

2.4 Deep Learning and its Applications in NLP

The approaches that have been used in the field of natural language processing have gone through three major eras: the rule-based, the empirical, and the deep learning ones (Deng and Liu, 2018). The first era has focused on using human knowledge to solve problems. Indeed, many NLP systems have been developed using handcrafted rules designed by linguistic experts. The downside of these approaches was the amount of human effort needed to design, maintain and update these rule-based systems. In the second era, the focus has gone toward using empirical probability-based methods that can solve problems by using large corpora without requiring any human intervention. In the third era, another paradigm called deep learning has appeared and managed to surpass the empirical methods on several NLP applications (Deng and Liu, 2018).

In the remainder of this section, we will start by briefly presenting the concept of deep learning, then we will talk about its applications in the NLP field.

2.4.1 Deep Learning

Classical neural networks have been historically used with only a few hidden layers (also known as shallow neural networks). This restriction in the number of layers was mainly due to the difficulties that have been encountered when attempting to train deeper neural networks such as the lack of computing power and the unavailability of large annotated corpora. These difficulties have forced the researchers to use neural

networks under a feature engineering scenario. First, the researchers had to carefully design and extract some useful features from data, then they had to feed those learned features to a shallow neural network to obtain the final predictions. During the last few years, two main aspects have changed: first, the amount of annotated data has increased noticeably; and second, the available computing power has reach such a level that it is now possible to train very large neural networks in a reasonable amount of time. These changes gave rise to deep neural networks which can be trained directly from raw data, thus skipping the features engineering step. This new way of using deeper neural networks to learn abstract features directly in an end-to-end fashion from raw data, without needing any intermediate steps, is currently known as "deep learning". An example of a deep feed-forward neural network with three hidden layers is provided in Figure 2.7.

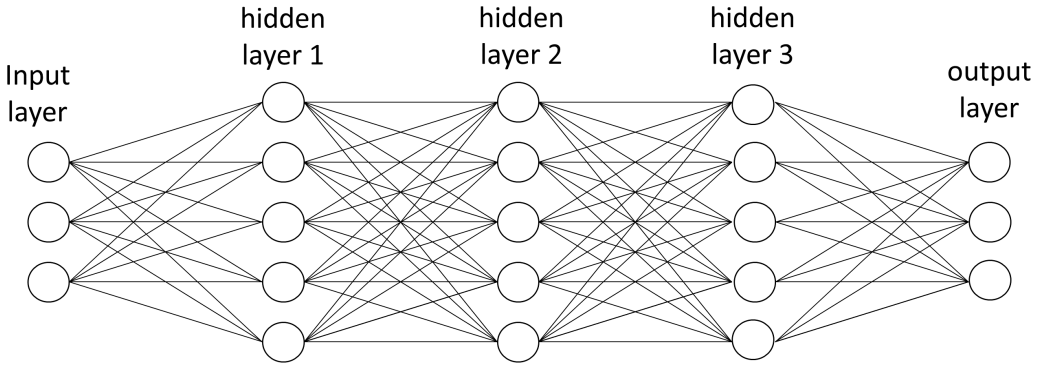


Fig. 2.7: An example of a deep feed-forward neural network

In the last few years, the term "deep learning" has been used very heavily in the research field; yet, as stated by Schmidhuber (2015), we still do not have a universal threshold of depth upon which we can clearly separate a shallow from a deep neural network. However, most researchers tend to consider as "deep" a neural network that has more than two hidden layers. Convolutional and Recurrent Neural Networks are considered as Deep Learning.

2.4.2 Deep Learning in NLP

In the past few decades, most of the research studies that have been done in NLP have used feature engineering techniques. These approaches generally require a good knowledge about the problem in order to propose and develop adequate features for solving it. However, more recently, the rise of deep learning (especially convolutional and

recurrent neural networks) and the huge success of word embeddings (low dimensional word representations) have achieved better results on several NLP applications.

In the sequel, we will introduce word embeddings and highlight some of the applications of deep learning in the NLP field.

2.4.2.1 Word Embeddings

Word embedding refers to a set of techniques that aim at learning a dense vector representation of words. These representations encode each word (or phrase) into a fixed-length vector of real numbers that maintain its linguistics aspects. The intuition of these methods is to represent the words that have similar meanings using vector representations that are also close to each other in terms of special distance.

One of the most used methods to create word embeddings from monolingual data is word2vec (Mikolov et al., 2013). Word2vec can be trained from textual data (a set of texts) using two different shallow neural networks namely: Skip Gram (skip-gram) and Common Bag Of Words (CBOW).

1. CBOW Model: it takes the context of a given word as input (its surrounding right and left words) and tries to predict the word (the center word) that is the most suited for that context.

2. Skip-gram Model: its functioning is the exact opposite of CBOW: it takes a given word as input (the center word) and tries to predict its context (its surrounding left and right words).

Figure 2.8 illustrates the functioning of the two models CBOW and Skip-gram when dealing with the center word “man”, the left context “the old”, and the right context “was sick”.

These kinds of representations have been proven to be very helpful for various NLP applications such as part-of-speech tagging (Ling et al., 2015a; Santos and Zadrozny, 2014), text/document classification (Dai et al., 2015; Kusner et al., 2015), machine translation (Bahdanau et al., 2014; Cho et al., 2014b), question answering (Aouichat et al., 2018; Zhou et al., 2015), etc.

2.4.2.2 Applications in NLP

Deep learning models have been used to address various NLP applications that go from the basic tagging tasks to the more complex text summarization and translation problems. In this subsection, we will try to highlight (in a very brief manner) some of

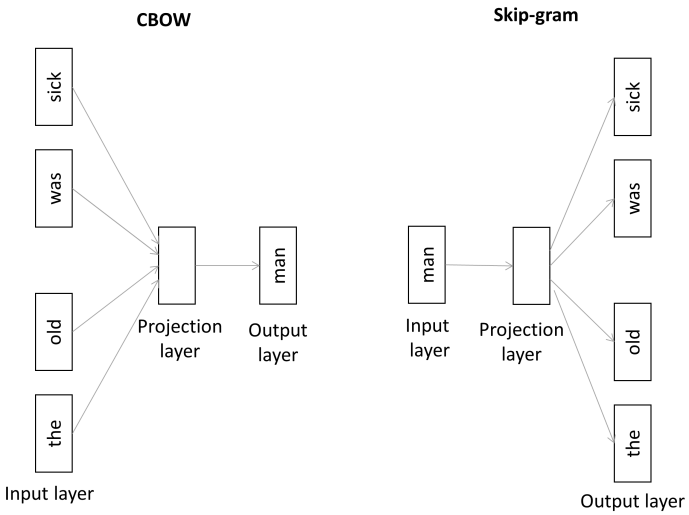


Fig. 2.8: An example illustrating the functioning mechanism of the two embedding models CBOW and skip-gram

the most important applications of deep learning in NLP. For a more detailed overview about the applications of deep learning in the field of NLP we point the reader to (Deng and Liu, 2018).

- Text-Based Dialog Systems: also known as interactive conversational agents (or chatbots) are computer-based systems designed to converse with humans using a coherent textual language (Deng and Liu, 2018). In the last few years, several end-to-end deep learning systems have been proposed to address this problem; we cite (Serban et al., 2016, 2017). These systems have been used in a wide range of applications such as technical support services, language learning tools, entertainment, etc (Deng and Liu, 2018).

- Syntactic Parsing: its goal is to analyze and recognize the syntactic structure (e.g constituents) of a given sentence, generally producing a parse tree that shows its different constituents along with the relation between them. Several end-to-end deep learning models have been proposed to infer the syntactic structure of a given sentence in a fully automated manner. We can cite (Collobert and Weston, 2008; Socher et al., 2011).

- Dependency Parsing: its goal is to grammatically analyze the structure of a sentence by identifying the different grammatical relationships that exist between its words. Deep learning has been used as an end-to-end model to predict the different relations between words (Andor et al., 2016; Chen and Manning, 2014).

- Machine Translation: it is concerned with the automatic translation of texts from one language to another. End-to-end deep learning systems have been proposed for MT, eg. by Cho et al. (2014b) and Bahdanau et al. (2014) [2].

- Question Answering: these are computer-based systems that get designed to answer the questions posed by humans using natural language (Deng and Liu, 2018). Two of the most important end-to-end deep learning systems that have been proposed to address this task are (Bordes et al., 2015; Iyyer et al., 2014).

- Text Summarization: the aim here is to create a shorter, fluent, and accurate text summary of a larger text or document. Two of the most important end-to-end deep learning systems that have been proposed to address this task are (Nallapati et al., 2016; Rush et al., 2015).

- Text Classification: its goal is to assign a document/text to one of a predefined set of classes or categories. Two of the most important end-to-end deep learning systems that have been proposed to address this task are (Kim, 2014; Zhang et al., 2015).

- Part-of-Speech Tagging: it assigns a part of speech tag (e.g. noun, verb, or adjective) to each token in a given sentence. Various deep learning systems have been proposed to address this task, from which we can cite (Plank et al., 2016; Wang et al., 2015a).

- Named Entity Recognition: it is the task of finding named entities (e.g. person names, organizations, or locations) in a given text. Two of the end-to-end deep learning models that have been proposed to address this task are (Chiu and Nichols, 2016; Lample et al., 2016).

2.5 Conclusion

In this chapter, we have introduced some background notions concerning deep learning and machine learning in general. We have also highlighted some of the most important applications of deep learning in the field of natural language processing. In the next chapter, we will focus on the machine translation task. We will try to cover all the paradigms that have been proposed for solving it along with the different ways to evaluate a given machine translation system.

[2] This neural-based MT method will be detailed in the next section.

Chapter 3

Machine Translation: Background and Approaches

3.1 Introduction

Machine Translation (MT) is a sub-field of Natural Language Processing (NLP) devoted to the development and enhancement of computer-based machine translation systems. The goal of an MT system is to automatically translate a given textual content from one language to another in a way that best preserves its meaning and style while still ensuring that the produced translation output is as linguistically fluent as possible. A huge amount of research has been accomplished in regards to MT and as a consequence, many translation paradigms have been developed in the field.

In this chapter, we will cover the different paradigms that have been proposed to solve the machine translation task along with the different approaches that have been used to evaluate the performance of a given machine translation system.

3.2 Machine Translation Paradigms

In this thesis, we classify the existing machine translation approaches based on the sources of information required for their construction as suggested by Costa-Jussa and Fonollosa (2015). These sources can be either rules (i.e. rule-based methods) or data (i.e. data-driven methods). Combining these two sources will lead to a third category of methods known as hybrid MT approaches. Figure 3.1 presents the main translation paradigms that can be found under each of these categories.

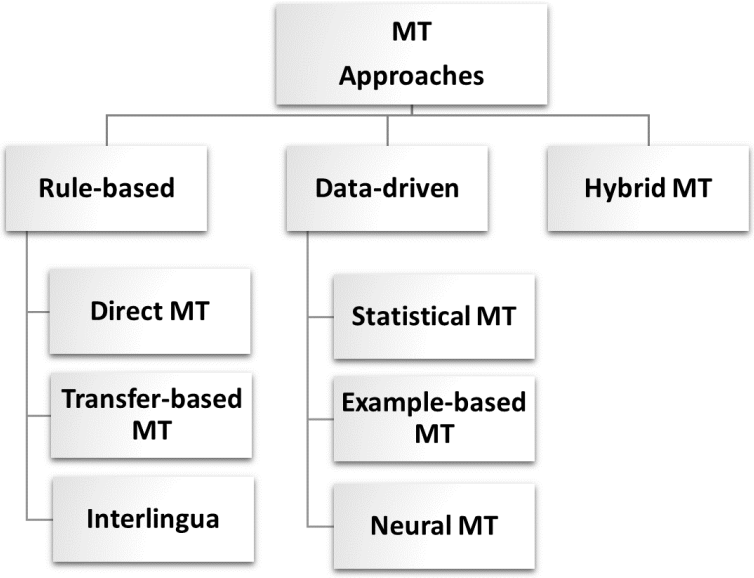


Fig. 3.1: Hierarchical categorization of the main machine translation approaches (Costa-Jussa and Fonollosa, 2015)

3.2.1 Rule-based Machine Translation

Rule-based Machine Translation (RBMT) is the first class of methods that have been used in the field of machine translation (Hutchins and Somers, 1992). These kinds of methods incorporate linguistic knowledge (i.e. handcrafted rules) about the source and target languages that vary from low-level morphological information to high-level syntactic and semantic knowledge. Indeed, rule-based MT methods can be divided into three classes: (1) the direct MT methods which attempt to model the relation between the source and target languages by relying only on morphology level knowledge (because most often captured by rules), (2) transfer methods which involve some syntax level knowledge, and (3) the interlingua methods which attempt to model the semantic level. The linguistic level of analysis involved in each class of methods increases gradually from the lowest direct MT level to the highest interlingua semantic-based translation; these methods are generally represented via the "Vauquois triangle" (Martin and Jurafsky, 2009) shown in Figure 3.2.

As shown in Figure 3.2, the direct MT methods are placed at the bottom of the triangle given that they involve almost no linguistic analysis. As we go higher in the triangle the level of analysis which is required increases until we reach the Interlingua MT which involves the highest level of linguistic analysis. In the following, we will briefly introduce the direct, transfer, and interlingual rule-based translation methods.

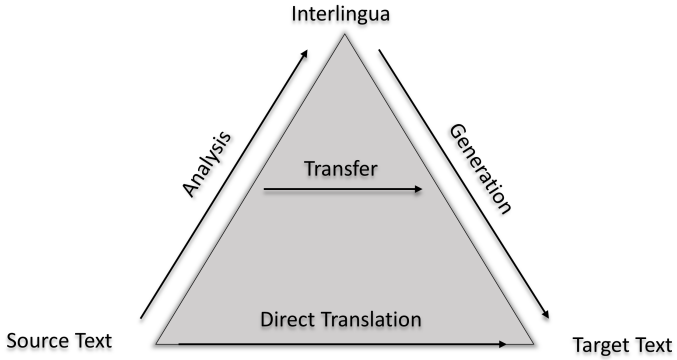


Fig. 3.2: The Vauquois triangle

3.2.1.1 Direct Machine Translation

The direct MT approach directly translates a source text to a target text without involving any linguistic analysis that goes beyond the morphological level (Hutchins and Somers, 1992). Figure 3.3 shows the overall translation phases involved in the direct MT approach.

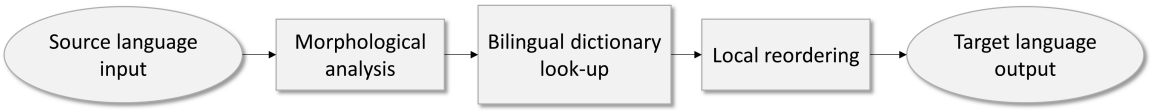


Fig. 3.3: Direct machine translation phases (Hutchins and Somers, 1992).

As illustrated in Figure 3.3, the source text is first morphologically analyzed, then, the translation process is carried out word by word by using a large bilingual dictionary that maps each source word to a target word. Each entry (word) in the bilingual dictionary is generally associated with some simple rules that specify its different translation scenarios. An example of such a procedure is given in Figure 3.4.

```
function DIRECT_TRANSLATE_MUCH/MANY(word) returns Russian translation

if preceding word is how return skol'ko
else if preceding word is as return stol'ko zhe
else if word is much
   if preceding word is very return nil
   else if following word is a noun return mnogo
else /* word is many */
   if preceding word is a preposition and following word is a noun return mnogii
   else return mnogo
```

Fig. 3.4: A simple procedure that guides the translation process of "much" and "many" from English to Russian under the direct MT method (Martin and Jurafsky, 2009)

As shown in Figure 3.4, a set of handcrafted rules have been written by experts to cover the different translation scenarios of the two words "much" and "many" from English to Russian (Martin and Jurafsky, 2009). After each word is translated, some simple reordering rules are applied (e.g. rules for moving adjectives after nouns) to arrange the target words in correct syntactic order.

3.2.1.2 Transfer-based Machine Translation

Instead of considering the translation as a direct mapping process, transfer-based MT methods (Hutchins and Somers, 1992; Martin and Jurafsky, 2009) use an intermediate structural representation to capture the different linguistic aspects of the input sentence. Then, the translation output is generated from that representation using a set of sophisticated transfer rules.

The functioning mechanism of this approach (Figure 3.5) can be summarized in three consecutive steps: analysis, transfer, and generation.

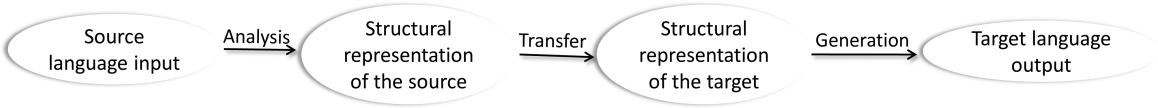


Fig. 3.5: The steps involved in a transfer-based MT system

As shown in Figure 3.5, an analysis step is first performed in which the source sentence is analyzed both morphologically and syntactically to create an internal representation that captures its syntactic aspects (e.g. a parse tree of the source sentence). Then, a set of linguistic rules are used to transform the structural representation of the input sentence into the target one. The morphological and syntactical differences that exist between the source and target languages are also handled at this stage via these transfer rules. Finally, the output translation is generated from the resulting target representation by using large bilingual dictionaries. Figure 3.6 illustrates the steps needed to translate a simple English sentence into French using the transfer-based translation approach.

As illustrated in Figure 3.6, the English sentence structural representation (parse tree) is transformed into a French parse tree by applying a simple Noun-Adjective syntactic reordering rule and using a bilingual English-to-French dictionary.

The downside of this approach is the high cost of building and maintaining a consistent set of transfer rules that transform the structural representation of the source language into the target one.

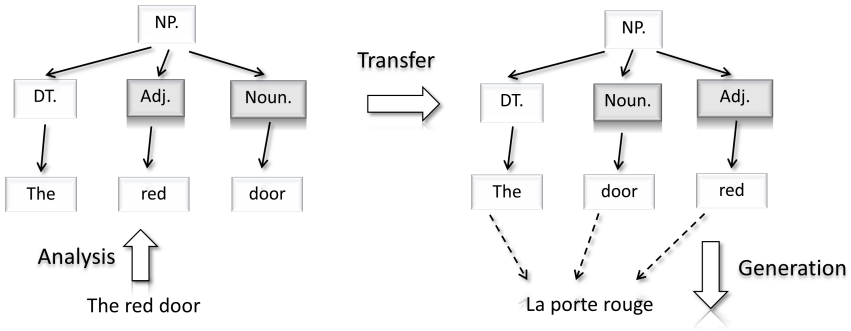


Fig. 3.6: Example of a Transfer-based approach translating between English and French.

3.2.1.3 Interlingua-based Machine Translation

One main problem about transfer-based MT is that it requires the development of a separate set of transfer rules to handle the translation between each pair of languages. Instead of building a specific language-related representation of the source sentence and then transforming into the target one, the interlingua-based MT approach uses a pivot representation called "Interlingua", that is independent of any natural language (Hutchins and Somers, 1992). This representation holds the meaning of a given sentence in a purely abstractive way regardless of its source language. This means that the transfer component of the transfer-based MT (shown in Figure 3.5) is no longer needed leaving the interlingua-based MT with only two steps: analysis and generation as shown in Figure 3.7.

Fig. 3.7: Interlingua-based machine translation phases

As illustrated in Figure 3.7, the analysis step involves all the necessary morphological, syntactical, and semantic linguistic analysis needed to transform the source language text into the interlingua, an abstractive representation of meaning. Then, the generation step performs the reverse processing in which the target output is generated from the interlingua representation.

This approach has the advantage of being extremely suited for multilingual translation systems due to the fact that all the language pairs will share the same abstract representation (no transfer component is needed). However, it suffers from the fact

that, in practice, building such a universal language-independent representation is extremely hard. Consequently, such methods are only used in very specific domains and still cost a huge amount of human effort (Martin and Jurafsky, 2009).

3.2.2 Data-driven Machine Translation

Unlike the rule-based methods which rely on human knowledge (e.g. handcrafted rules), data-driven methods use sophisticated algorithms and mathematical models to automatically learn the translation process from data. This class of approaches relies heavily on large bilingual aligned corpora which are currently available across multiple languages. The availability of such corpora and the fact that this class of methods does not require any human knowledge have allowed it to dominate the field of machine translation for the past few decades. Data-driven MT includes Example-Based MT (EBMT), statistical MT (SMT), and neural machine translation (NMT) approaches.

3.2.2.1 Example-based Machine Translation

Translating by example was first proposed by Nagao (1984) and the idea behind it is to translate by analogy. Indeed, it follows the intuition that people do not translate using deep linguistic analysis; instead, they split the text into smaller segments and translate them separately, then they combine those smaller segments together to produce the final translation. Thus, to translate a new input text, this approach attempts to adapt the previously translated sentences to produce its translation instead of trying to translate it from scratch. This approach uses a large bilingual corpus that contains a set of parallel texts (segments) as its knowledge base. The phases involved in this method are shown in Figure 3.8.

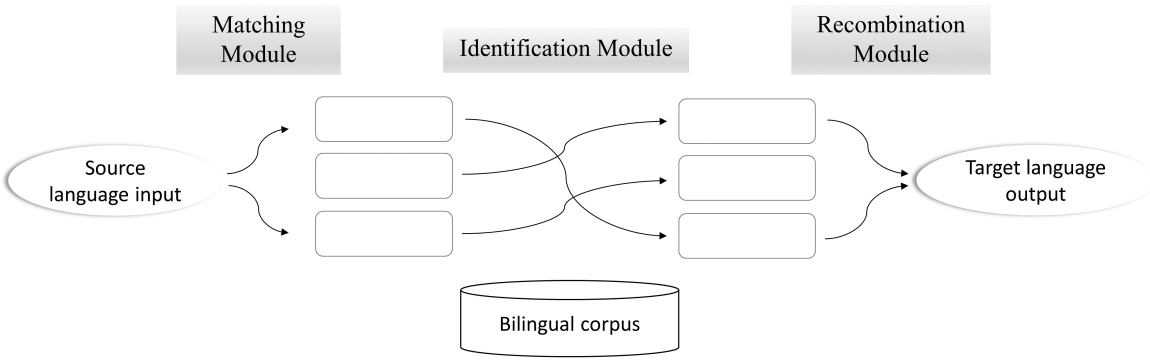


Fig. 3.8: The steps involved in Example-based MT

As illustrated in Figure 3.8, this approach involve three modules:

1. The matching module attempts to decompose the input sentence into smaller fragments.
2. The identification module (alignment module) tries to align each segment into its best matching translation using the example database.
3. The recombination module tries to combine the translated fragments (phrases) together to form the translation output.

Even though this approach does not require any handcrafted linguistic rules, its performance is heavily influenced by the quality of the example database.

3.2.2.2 Statistical Machine Translation

The statistical machine translation (SMT) paradigm was proposed in the early 90s by a group of researchers working at the IBM Watson Research Center (Brown et al., 1988a, 1990b, 1993) and has evolved to become one of the leading approaches in machine translation. Unlike those based on the rule-based methods, statistical MT systems are built automatically using monolingual and parallel corpora.

To formulate the problem, we assume the task of translation to be from a foreign language (e.g. French) to English which is a convention used in all the documentations regarding SMT (Collins, 2011; Koehn, 2009) [1]. Let us consider a parallel corpus, which is a collection of sentence pairs (f, e) where f=$\{f_1, f_2, ..., f_{l_f}\}$ is the source sentence composed of l_f foreign words, and e=$\{e_1, e_2, ..., e_{l_e}\}$ is the target sentence composed of l_e English words.

Given a source sentence f along with its translation e, the problem of statistical machine translation can be formulated as follows:

$$\hat{e} = argmax_e p(e|f) \tag{3.1}$$

The goal is to find the best translation $\hat{e}$ that maximizes $p(e|f)$, the probability of e being the translation of f (Brown et al., 1988a, 1990b). The noisy channel model (Brown et al., 1990b; Koehn, 2009) is used to decompose $p(e|f)$ into a translation model $p(f|e)$ and a language model $p(e)$.

$$\hat{e} = argmax_e p(f|e) * p(e) \tag{3.2}$$

[1] Following this convention, throughout this chapter, we assume the task of statistical translation to be form French to English.

The components of a statistical machine translation system are provided in Figure 3.9.

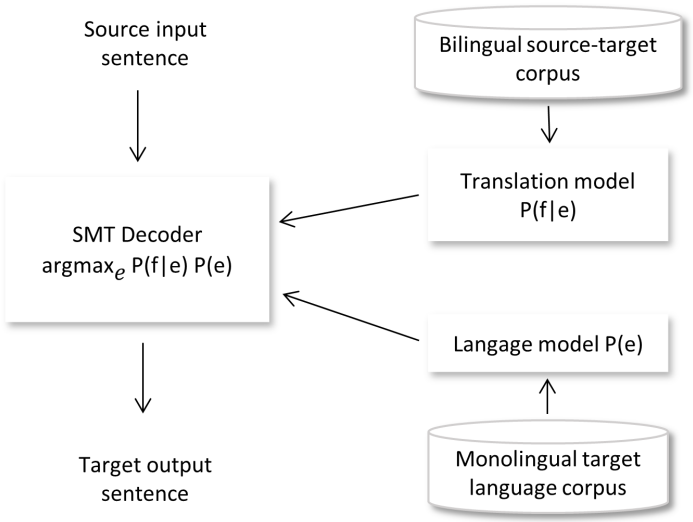


Fig. 3.9: The components of a statistical machine translation system

The SMT components resulting from the noisy channel formulation are as follows:

- A language model for computing $p(e)$ which ensures the fluency of the generated target output. The language model is built using a monolingual target corpus.
- A translation model for computing $p(f|e)$ which ensures the accuracy of the translation between the source and the target languages. The translation model is built using a parallel source-target corpus.
- A decoder, which is used to find the most probable translation output ê for the input sentence f from the space of all possible translations of f.

More details will be added about word alignment and language models in the following section.

3.2.2.3 Neural Machine Translation

Neural Machine Translation (NMT) (Cho et al., 2014b) is a recently proposed approach for tackling the machine translation task. Unlike the traditional statistical machine translation approach which involves several components that are tuned separately, neural machine translation uses a single large neural network which is tuned at once to increase the translation quality (Bahdanau et al., 2014). In the following, we

present the basic Attention-based NMT model (Bahdanau et al., 2014). Given a source sentence $X = (x_1, x_2, ..., x_d)$ and a target sentence $Y = (y_1, y_2, ..., y_{d'})$, where each x_t and y_t represent the source and target words at time-step t, d and d' represent the maximum source and target sentence lengths respectively [2]; the attention-based Neural Machine Translation estimates the conditional probability of generating the target sentence Y given the source sentence X as $P(Y = (y_1, y_2, ..., y_{d'})|X = (x_1, x_2, ..., x_d))$.

The NMT architecture (Figure 3.10) involves two components: an encoder and a decoder.

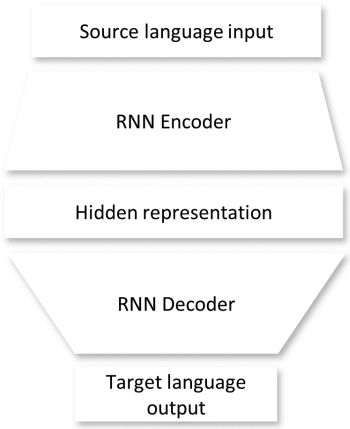


Fig. 3.10: The components of a Neural Machine Translation (NMT) system (Cho et al., 2014b)

The encoder is usually a bidirectional Recurrent Neural Network (bRNN) (Schuster and Paliwal, 1997) that reads the input sentence word by word from left-to-right (direct direction Eq. 3.3) and from right-to-left (reversed direction Eq. 3.4) :

$$\overrightarrow{h_t} = \Phi_{enc}(\overrightarrow{h_{t-1}}, x_t) \tag{3.3}$$

$$\overleftarrow{h_t} = \Phi_{enc}(\overleftarrow{h_{t-1}}, x_t) \tag{3.4}$$

where $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ are the hidden states at time-step t generated by the direct and reversed recurrent neural networks respectively. This is done by taking into consideration the previous hidden state h_{t-1} at time-step $t-1$ and the current input word x_t at time-step t. Φ_{enc} is the recurrent activation function responsible for combining the previous hidden state with the current input word. In practice the function Φ_{enc} is usually implemented as a Long Short-term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) or a Gated Recurrent Unit (GRU) (Cho et al., 2014a). Applying this process to the whole input

[2]Generally a padding process is used to pad all the sentences into the same length.

sentence produces a direct encoder hidden representation $\overrightarrow{H} = (\overrightarrow{h_1}, \overrightarrow{h_2}, ..., \overrightarrow{h_d})$ and a reverse encoder hidden representation $\overleftarrow{H} = (\overleftarrow{h_1}, \overleftarrow{h_2}, ..., \overleftarrow{h_d})$ where $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ represent the direct and reversed encoder hidden states at time-step t respectively. The direct and reversed hidden states will be concatenated at each time-step t to form what is called the "annotation vector" $H = (h_1, h_2, ..., h_d)$, where each single annotation h_t at time-step t (Eq. 3.5) is a tuple:

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \tag{3.5}$$

Each annotation h_t conserves the information about the word at position t along with all the words surrounding it (its left and right contexts).

The decoder is generally a unidirectional recurrent neural network that uses an attention mechanism to select a target word at each time-step t based on its annotation vector. This is done by attributing a relevance weight α_{tj} to each annotation h_j via a feed-forward neural network that takes the annotation h_j, the previous output y_{t-1} and the previous decoder hidden state s_{t-1} to produce an output e_{tj} as shown in Eq. 3.6.

$$e_{tj} = f(s_{t-1}, h_j, y_{t-1}) \tag{3.6}$$

The function f is a dense neural network with one hidden layer. The output e_{tj} will then be normalized via Eq. 3.7.

$$\alpha_{tj} = \frac{exp(e_{tj})}{\sum_{k=1}^{d}(exp(e_{tk})} \tag{3.7}$$

The normalized scores are then used to compute the weighted sum of the annotation vectors (Eq. 3.8)

$$c_t = \sum_{j=1}^{d} \alpha_{tj} h_j \tag{3.8}$$

Then the decoder neural network can update its own hidden state using the encoder weighted sum of the annotation vectors c_t, the decoder previous hidden state s_{t-1} and the decoder previous output word y_{t-1}:

$$s_t = \Phi_{dec}(s_{t-1}, y_{t-1}, c_t) \tag{3.9}$$

where Φ_{dec} can be implemented as an LSTM or a GRU unit in a similar way to what we have seen for the encoder. The NMT model is trained to maximize the probability of generating the target sentence when given the source sentence in an end-to-end learning

manner on the basis of a substantially large parallel corpus via the backpropagation algorithm (Goller and Kuchler, 1996).

3.2.3 Hybrid Machine Translation

Hybrid machine translation is a class of methods that attempt to combine the aspects of several translation approaches into a single translation system. The goal of doing so is to benefit from the advantages of each individual approach and to produce a better translation. There are two main hybridization methods (Costa-Jussa and Fonollosa, 2015), hybrid rule-based MT and hybrid data-driven MT.

3.2.3.1 Hybrid rule-based MT

These hybridization methods attempt to use data information (extracted via data-based methods) into a preexisting rule-based translation system. For example, word-based alignment methods (e.g. IBM 1-5 or HMM alignment methods (Och and Ney, 2000)) have been used by Habash et al. (2009a) to enhance the bilingual dictionaries of a rule-based system by adding new entries (e.g. new words or phrases) extracted automatically from a bilingual corpus.

3.2.3.2 Hybrid data-driven MT

These approaches attempt to either include rules into an existing data-based MT system (e.g. statistical or example-based systems) or to combine several data-driven methods into a single MT system (Costa-Jussa and Fonollosa, 2015).

- Enhancing a system using rules: the idea here is to use rules to enhance a preexisting data-driven system via a preprocessing or a post-processing task. For example, handcrafted rules can be used to change the syntactic order of the source language to match the target one (Xia and McCord, 2004).

- Enhancing a system using a combination: the idea here is to combine several data-driven approaches into a single MT system to boost their performance. For instance, Groves and Way (2005) proposed a hybrid system that combines subsentential alignments from both a phrase-based and example-based MT systems and managed to outperform both of these individual systems.

3.3 Language Models and Word Alignments

In this section, we cover two important components of Statistical Machine Translation (SMT): language models and word alignment models.

3.3.1 Language Models

Language modeling is one of the most important tasks in the field of natural language processing. It can be used in a wide range of applications, such as speech recognition, machine translation, question answering, automatic summarization, etc.

A Language model assigns a probability to a given sequence of words on the basis of its syntactic correctness (how linguistically correct its word order is). Language models are generally trained on the task of predicting the next word in a given sequence of words using monolingual language corpora. For example, given the sentence "I am reading a", the model will try to predict the most probable upcoming word (e.g. book, paper, journal, etc).

There are currently two main approaches to language modeling namely: statistical and recurrent language models.

3.3.1.1 Statistical Language Models

A statistical language model (Martin and Jurafsky, 2009) is a probabilistic model that estimates the probability $P(W)$ of a given sequence of words $W = w_1, w_2, ..., w_n$ using the chain rule (Eq. 3.10).

$$P(W) = P(w_1, w_2, ..., w_n) = P(w_1)P(w_2|w_1)...P(w_n|w_1^{n-1}) \tag{3.10}$$

The chain rule predicts the probability of each word in the sequence on the basis of all the words that precede it (Eq. 3.11).

$$P(W) = \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \tag{3.11}$$

Relying on the full history of a given word (all the words that precedes it) becomes very troublesome when dealing with long sequences. Indeed, the statistics that we can derive from the corpus regarding long sequences are inaccurate because their frequency of appearance is very small (Martin and Jurafsky, 2009). As a solution to this issue the independence assumption (also known as the Markov assumption) is used to limit the history of a given word to only he last k words that precede it. The probability

will thus be estimated using equation 3.12.

$$P(w_n|w_1^{n-1}) = P(w_n|w_{k-(n-1)}^{n-1}) \tag{3.12}$$

When using the Markov assumption in the case of a bigram language model (when the threshold $k = 1$), we will end up with equation 3.13. It says that a word is predicted only on the basis of one word that precedes it.

$$P(w_n|w_1^{n-1}) = P(w_n|w_{n-1}) \tag{3.13}$$

$P(w_n|w_{n-1})$ is computed using the Maximum Likelihood Estimation (MLE) from the monolingual corpus as shown in equation 3.14.

$$P(w_i|w_{i-1}) = \frac{C(w_{i-1}, w_i)}{C(w_{i-1})} \tag{3.14}$$

where $C(w_{i-1}, w_i)$ and $C(w_{i-1})$ are the counts of the bigram $w_{i-1}w_i$ and the unigram w_{i-1} respectively.

3.3.1.2 Recurrent Language Models

As we have seen previous section, the statistical n-gram language models use the Markov assumption to limit the history of each word to only a small number of words. Recurrent neural networks, however, do not need to make such an assumption given that they are theoretically capable of modeling long-term dependencies (Mikolov et al., 2010). Recurrent neural network language models are trained in a supervised fashion using a corpus of monolingual sentences. They are trained on the task of predicting the upcoming word for a given sequence of words by providing all its previous words (its history) as shown in Fig. 3.11.

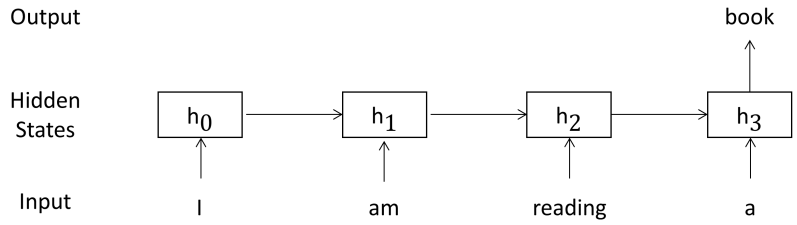


Fig. 3.11: An example illustrating the functioning of an RNN-based language model

Even though a basic recurrent neural network can work very well for language modeling tasks (Mikolov et al., 2010), in practice more advanced types of RNNs can also be used such as GRUs and LSTMs (Jozefowicz et al., 2015), already presented in Section 2.3.3.

3.3.2 Word Alignment Models

Word alignment is a very important task in the field of machine translation; it aims at finding word (or group of word) correspondences (also called alignments) between source and target sentences belonging to two different languages.

As we stated in the previous section, statistical machine translation (Koehn, 2009) splits the translation process into two models: $p(e)$ which is known as the language model which gives the probability of finding the sentence e in the English language, and $p(f|e)$ known as the translation model which gives the probability of translating a sentence e into a sentence f.

The word alignment process which is the object of this section attempts to model the translation probability $p(f|e)$. For an input sentence pair (f, e) the goal is to map/associate each word from the source sentence f to its corresponding word (or group of words) from the target sentence e. An alignment variable a=$\{a_1,a_2,...,a_{l_f}\}$ is used to indicate for each foreign word at position i; the English word at position j that is aligned to it. An example of word-to-word alignment between French and English sentences is presented in Figure 3.12.

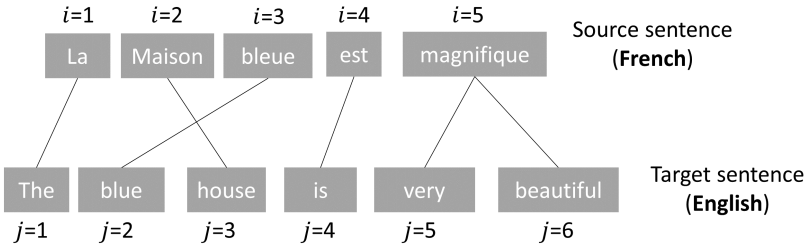


Fig. 3.12: An example of word-to-word alignment between French and English

Each source word can be aligned to a single target word (one-to-one alignment), multiple target words (one-to-many alignment) or no target words (one-to-zero alignment, also known as "dropping" the source word). In our example the length of the French sentence is 5; hence our alignment variable will contain 5 values a=$\{a_1,a_2,...,a_5\}$, where $a_i = [j]$ indicates that the French word found at position i is aligned to the

English word found at position j. Thus, The alignment showed in Fig. 3.12 can be represented by a=$\{a_1 = [1], a_2 = [3], a_3 = [2], a_4 = [4], a_5 = [5,6]\}$.

There are two main approaches for preforming word-to-word alignment, namely: the generative and the discriminative methods.

3.3.2.1 Generative Word Alignments

Generative word alignments are probabilistic models that rely solely on large bilingual corpora without involving any external linguistic knowledge. The most popular generative models are the IBM word alignment model: 1, 2, 3, 4, and 5, and also the Hidden Markov Model (HMM) alignment model.

IBM Models IBM models (Brown et al., 1990a, 1988b, 1993) refer to the 5 most popular and highly used word-to-word alignment models namely: IBM model 1, 2, 3, 4, and 5, that have been introduced by researchers from the IBM Research Center between the late 1980s and early 1990s. The complexity of these models increases gradually from the first to the fifth model as each one of them handles the inconveniences of its predecessor.

1. IBM Model 1 (Lexical Translation): It is the first and simplest model to be introduced among the five IBM models and it is based solely on the lexical translation probability. It assumes the independence between all the alignment points. We recall that our goal is to estimate $p(f|e)$ which is defined as the sum of all possible alignments:

$$p(f|e) = \sum_{a} p(f,a|e) \tag{3.15}$$

IBM model 1 estimates $p(f,a|e)$ as follows (Eq. 3.16):

$$p(f,a|e) = p(f|e,a)p(a|e) \tag{3.16}$$

The first IBM model uses two probabilities $p(a|e)$ and $p(f|e,a)$. The first probability $p(a|e)$, is the probability of aligning a source word to one of the target words. Since this model makes the assumption that all alignments are equally likely, and since each source word must come from one of the target words or from NULL then we will end up having $(l_e+1)^{l_f}$ possible alignments (3.17).

$$p(a|e) = \frac{\epsilon}{(l_e+1)^{l_f}} \tag{3.17}$$

where ϵ is a small normalization constant that is used to ensure that the result is a valid probability distribution.

The second probability, $p(f|e,a)$, is the conditional probability of aligning a source word f to a target word e according to the alignment a (Eq. 3.18).

$$p(f|e,a) = \prod_{j=1}^{l_f} p(f_j|e_a(j)) \tag{3.18}$$

where $e_a(j)$ is the target word that is aligned to the source word f_j, and $p(f_j|e_i)$ is the conditional probability for choosing f_j as the translation of e_i.

2. IBM2 is an extension of the first IBM model which introduces a distortion model that takes into account the absolute position of the source word. Thus, the probability $p(a|e)$ that has been considered to be uniform in IBM1 is modified as follows:

$$p(a|e) = \prod_{j=1}^{l_f} p(a_i|j,l_e,l_f) \tag{3.19}$$

This equation can be interpreted as the probability of alignment variable a_i taking the value j, conditioned on the lengths of both the source and target sentences. The full IBM2 equation becomes:

$$p(f,a|e) = \prod_{j=1}^{l_f} p(a_j|j,l_e,l_f) \prod_{j=1}^{l_f} p(f_j|e_a(j)) \tag{3.20}$$

3. IBM model 3 introduces a fertility model $n(\phi|e)$ which models the number of target words ϕ that can be generated from each source word e. In other words, it models the number of words that are usually produced for each source word. For example, a unique word in English such as "shallow" translates to two French words "peu profond", thus, $n(\phi|\text{ɪ}shallow\text{ɪ}) \simeq 2$.

4. The IBM4 model brings several new improvements to its predecessor, namely:

   - A better distortion model that uses relative instead of absolute positions. This specifically aims to improve the unreliable statistics that are obtained from the IBM 2 and 3 models when dealing with large source/target sentence pairs.

- A dependency on word classes, thus each word becomes dependent not only on the previously aligned word but also on the word classes of the surrounding words.
- A dependency between alignment points is introduced in order to capture the phrases (multiple words) that tend to move together.

5. IBM Model 5 improves its predecessor by introducing new training parameters to address the problem where multiple output words are placed in the same position. Indeed, in the IBM 3 and 4 models, there are no means of prohibiting the placement of output words in a position that is already taken. Thus, the added parameters in IBM 5 ensures the placement of output words in free positions only, which is performed by allowing placements in only the remaining free positions.

HMM Model As we stated earlier, when talking about IBM models, the goal is to model the translation probability $p(f|e)$. The HMM alignment model (Vogel et al., 1996) is similar to IBM model 2, with one main difference that is making the alignment of each word dependent on its previous word rather than its absolute position. The model uses a first-order Hidden Markov model (HMM) that follows the equation below:

$$p(f|e) = \sum_{a} \prod_{j=1}^{l_f} p(a_j|a_{j-1}, l_e) * p(f_j|e_a(j)) \tag{3.21}$$

Two probabilities are involved: the transition probability $p(a_j|a_{j-1}, l_e)$ that models the distance between the current aligned word and the previous one, and the emission probability $p(f_j|e_a(j))$ which is exactly the same as presented in IBM model 1 and 2.

3.3.2.2 Discriminative Word Alignment

The generative models consider only the features that can be learned from raw data (lexical word forms) without introducing any additional linguistic information that may be helpful for the alignment task. This branch of discriminative methods tries to incorporate rich linguistic features (lemma, morphosyntactic tags, word senses, lexical classes, semantic information, etc.) to enhance the alignment quality. Research studies that have been done on discriminative word alignment have used various supervised learning methods to build their models. Among these algorithms we cite the Perceptron

algorithm (Moore, 2005), maximum entropy approach (Ayan and Dorr, 2006), neural networks (Ayan et al., 2005), max-margin methods (Cherry and Lin, 2006), Support Vector Machines (Moore et al., 2006), Conditional Random Fields (Blunsom and Cohn, 2006). The downside of these discriminating models is their need for word-for-word annotated parallel training data which are not available for all languages. Given the difficulty of gathering a large amount of annotated word-to-word alignment data, these models are generally trained on a relatively small amount of data, thus their quality will depend mainly on the considered linguistic features [3].

3.4 Evaluation of Machine Translation Systems

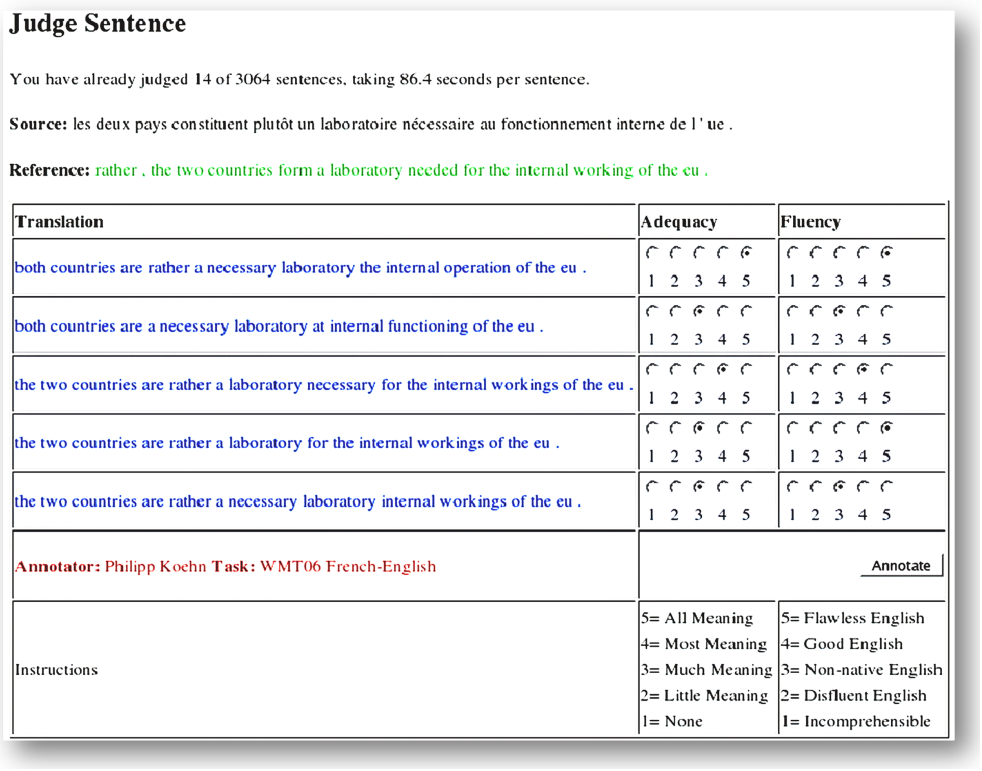
Once the automatic translation system is used to translate a given textual data, MT evaluation can be used to measure the quality of that translation. This evaluation can be done by relying on either manual or automatic methods.

3.4.1 Manual Evaluation

This approach requires the intervention of linguistic experts to assess the quality of MT systems outputs. First, the MT system is used to translate a set of sentences taken from a given test set. Then, the input sentences along with the translations produced for them by the MT systems will be given to the translation experts to measure their quality. The quality is generally assessed on the basis of two main criteria: "fluency" and "fidelity". Fluency measures the understandability of the translation output by taking into account the grammatical soundness of its constructions and the correctness of its word order. Fidelity, on the other hand, is used to make sure that the meaning of the original sentences is preserved. Figure 3.13 shows English translations of a given French sentence being evaluated by an expert using the two criteria "fluency" and "fidelity" (adequacy).

The main disadvantage of manual evaluation is its time and effort consumption. Also since the linguistic experts do not evaluate the automatic translations in the same way, their evaluation results are not reproducible.

[3] We have covered the discriminative alignment models in a very brief manner given that they are not very relevant to this thesis; for more information about them we refer the reader to (Koehn, 2009; Xiong and Zhang, 2016).



Judge Sentence

You have already judged 14 of 3064 sentences, taking 86.4 seconds per sentence.

Source: les deux pays constituent plutôt un laboratoire nécessaire au fonctionnement interne de l ' ue .

Reference: rather , the two countries form a laboratory needed for the internal working of the eu .

Translation	Adequacy	Fluency
both countries are rather a necessary laboratory the internal operation of the eu .	1 2 3 4 5	1 2 3 4 5
both countries are a necessary laboratory at internal functioning of the eu .	1 2 3 4 5	1 2 3 4 5
the two countries are rather a laboratory necessary for the internal workings of the eu .	1 2 3 4 5	1 2 3 4 5
the two countries are rather a laboratory for the internal workings of the eu .	1 2 3 4 5	1 2 3 4 5
the two countries are rather a necessary laboratory internal workings of the eu .	1 2 3 4 5	1 2 3 4 5
Annotator: Philipp Koehn **Task:** WMT06 French-English		Annotate
Instructions	5= All Meaning 4= Most Meaning 3= Much Meaning 2= Little Meaning 1= None	5= Flawless English 4= Good English 3= Non-native English 2= Disfluent English 1= Incomprehensible

Fig. 3.13: An example showing the process of manual MT evaluation (Koehn and Monz, 2006)

3.4.2 Automatic Evaluation

Many metrics have been proposed to automatically evaluate MT translation outputs. These metrics rely on preexisting test sets (benchmarks) in which for each source sentence, one or multiple target (reference) translations are prepared. The automatic evaluation metrics calculate a score that is mainly based on the distance or the degree of similarity between the MT system outputs and the preexisting test set references. In the following we cite the most important automatic MT evaluation metrics.

3.4.2.1 Word Error Rate

Word Error Rate (WER) (Su et al., 1992) is one of the first automatic evaluation metrics that have been used to evaluate machine translation outputs. It uses the Levenshtein distance, that is the minimal number of editing operations that are needed to transform the MT system output to one of its references. The considered operations are insertion (adding a word), deletion (removing a word), and substitution (replacing

a word). One main weakness of the WER metric is word order since it is not taken care of appropriately.

3.4.2.2 Translation Error Rate

Translation Error Rate (TER) (Snover et al., 2006) is a machine translation evaluation metric that is designed as an improvement to the classical WER metric. It addresses the problems of word order that exists in WER by adding a novel editing step called "shift" which allows word/phrase movements from one part of the output to another. By adding this new operation, the possible edit operations become four: insertion, deletion, substitution, and shift.

3.4.2.3 BiLingual Evaluation Understudy (BLEU)

BiLingual Evaluation Understudy (BLEU) introduced by Papineni et al. (2002) is the most commonly used evaluation metric in the field of machine translation and the first automatic metric to report a high correlation with human judgments (Coughlin, 2003). The method works by matching the n-grams of the hypothesis (the system output) against the test set translation references corresponding to that input. Formally, we define a test set as a collection of pairs $(f, r(f))$ where f is the source input sentence containing l_f words and $r(f)$ is a list of its target reference translations. We refer to the MT system output for the input sentence f as $o(f)$ of length l_o. The BLEU score equation is as follows:

$$Bleu = BP * exp(\frac{1}{4}\sum_{n=1}^{4} log(p_n)) \tag{3.22}$$

where p_n is the modified precision of all the n-grams belonging to the system output $o(f)$:

$$p_n = \frac{\sum_{n\text{–}gram \in o(f)} modified_{count}(n\text{–}gram)}{\sum_{n\text{–}gram \in o(f)} count(n\text{–}gram)} \tag{3.23}$$

BP is the brevity penalty which is used to penalize short translations (Eq. 3.24).

$$BP = min(1, exp^{1 \, - \, \frac{l_o}{|r|}}) \tag{3.24}$$

Here $|r|$ is the number of words in the reference $r \in r(f)$ that has the closest length to system output $o(f)$.

3.4.2.4 METEOR

Meteor is an automatic machine translation metric developed by Banerjee and Lavie (2005) at Carnegie Mellon University. It is based on word-for-word alignment of the system output and the reference translation. This alignment is performed via several steps. First, identical surface words are matched. Then, the algorithm matches words having a common root. Finally, semantically similar words are matched using external resources. These steps of approximate matching are added mainly to address the drawbacks of the exact matching that is used in the BLEU score. The main disadvantage of METEOR is its matching process that is very computationally expensive. This is due to the use of many more parameters (such as the weights for stemming and synonym matches) that need to be tuned (Koehn, 2009).

3.4.2.5 NIST

National Institute of Standards and Technology (NIST) is an automatic measure for evaluating machine translation outputs developed by Doddington (2002). Just like the BLEU Score, NIST estimates the quality of the system output based on the number of shared n-grams with the reference translation. However, it introduces the following improvements:

- NIST introduces the concept of ngram quality, thus it gives more credit to a system when it finds a rare ngram and less credit when it finds a more common one.
- It uses arithmetic mean instead of a geometric mean for ngram precision.
- The brevity penalty equation is also adjusted to make it more robust to a small variation in the output lengths.

3.4.2.6 ROUGE

Recall-Oriented Understudy for Gisting Evaluation (ROUGE) (Lin and Hovy, 2003) is a recall-oriented automated evaluation metric for evaluating automatic summarization and machine translation outputs. It works by comparing the output summary or translation against one or more references using ngram co-occurrence statistics. There are many extensions to this metric that add additional information, such as ROUGE-L (Lin and Och, 2004) that includes statistics about the Longest Common Subsequence (LCS), and ROUGE-S (Lin and Och, 2004) which adds Skip-bigram information that concerns skipping any pair of words in the sentence, allowing for arbitrary gaps.

3.5 Conclusion

In this chapter, we have introduced some basic background concepts regarding machine translation such as language models and word alignment methods. We have presented machine translation paradigms and also the most used MT evaluation methods. In the next chapter, we will talk about the Arabic language, the research studies that have been done in Arabic NLP, and the difficulties that are encountered in its translation.

Chapter 4

Arabic Machine Translation: State-of-the-Art

4.1 Introduction

Arabic is one of the five most spoken languages in the world with more than 300 million native speakers [1] and one of the six official languages of the United Nation (UN) [2]. It is a Semitic language that is well known for its rich and complex morphology which is substantially different from that of Indo-European languages (such as English and French). The Arabic morphology added to other linguistic aspects has made the automatic translation from and to Arabic a lot more challenging. Though a great deal of improvement has been achieved due to the recent advances in data-driven translation paradigms (e.g. statistical and neural methods), these linguistic aspects are still causing many difficulties (Alkhatib and Shaalan, 2018; Habash and Sadat, 2006). In this chapter, we start by presenting the Arabic language and the challenges involved in its translation. Then we provide an exhaustive overview of the important research studies that have been accomplished in the area of Arabic MT. We note that in this chapter as well as in the remainder of this thesis, the Buckwalter transliteration scheme will be used to transcribe Arabic words using Latin characters. Buckwalter uses a simple one-to-one mapping between Arabic and Latin character sets (Figure 4.1[3]).

[1] https://www.conservapedia.com/List_of_languages_by_number_of_speakers
[2] http://www.un.org/en/sections/about-un/official-languages/
[3] Figure 4.1 is taken from the Qamus website http://www.qamus.org/transliteration.htm

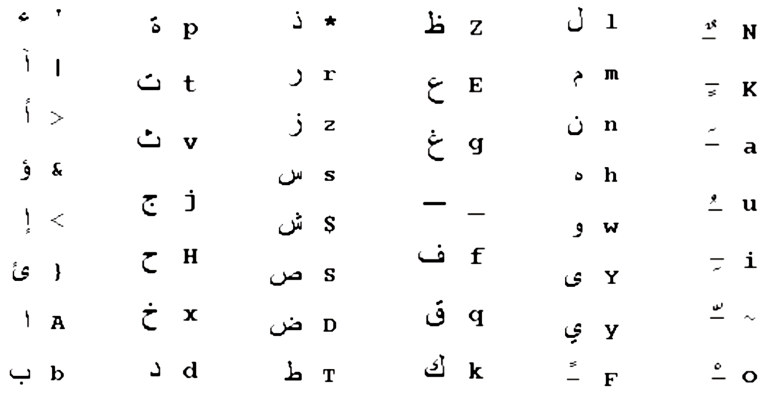


Fig. 4.1: Buckwalter's one-to-one Arabic-Latin mapping scheme

4.2 The Arabic Language: Characteristics and Translation Difficulties

In recent years, the amount of research work that has been devoted to Arabic natural language processing has considerably increased. Due to its rich and complex morphology of Arabic, there has been an increasing demand for highly sophisticated NLP tools that can satisfy its growing needs in several domains and applications.

In this section, we briefly introduce the Arabic language along with the difficulties that are involved in its translation. Given that the main concern of the research community is the task of translation between Arabic and English, this section mainly focuses on the translation difficulties concerning these two languages. We note that this introduction about the Arabic language and its translation challenges merely scratches the surface in this area, thus we refer the reader to Habash (2010) and Ryding (2014) for more detailed overviews about the Arabic language and its characteristics.

4.2.1 Arabic Language Characteristics

In the following subsections we will highlight some orthographic, morphologic, and syntactic aspects of the formal Arabic language (Modern Standard Arabic, MSA). In the same time, we will try to compare each linguistic aspect of the language to those of some other languages such as English.

4.2.1.1 Arabic Orthography

Orthography studies the spelling system of a language. This generally involves a set of symbols that are used to write the language and a set of rules that indicate their

usage. Arabic contains 28 letters which make its orthographic complexity comparable to that of English, and much simpler than the Chinese orthography which contains around 10,000 logographic characters [4] (Habash, 2010). The shape of a single Arabic letter may change slightly depending on its position within the Arabic word (beginning, middle, or at the end). Unlike Latin languages, Arabic texts are written from right to left and do not use special letters to represent vowels. Instead, the Arabic writing system uses diacritics which are small marks that can be added above or below the letters. The presence of these diacritical marks is known as "Vocalization". When the vocalization is present, it adds more information about the correct pronunciation and meanings of the words which solves many lexical and semantic ambiguities. The most common diacritical marks (also known as short vowels) in the Arabic language are Fathah as in أَ (makes an "a" sound as in "Adam"), Dammah as in أُ (makes an "oo" sound as in "took"), and Kasrah as in إِ (makes an "i" sound as in "in").

4.2.1.2 Arabic Morphology

Morphology studies the internal structure of words and emphasizes the way morphemes[5] fuse with each other to form new words and express new meanings. The Arabic morphology involves two main operations: inflection and derivation (Ryding, 2014).

Inflection is the mechanism that allows the creation of new words (inflected words) from other words by adding inflectional morphemes that express specific grammatical properties (e.g., gender, person, number, aspect, mood, time, etc.). This process generally retains both the meaning and the syntactic category of the base word. Arabic is known to be a highly inflected language. Its verbal inflection usually follows very regular patterns with hardly any exceptions (Habash, 2010). It has two gender values: masculine (ex. يَسْمَعُ, yasomaEu, he listens) and feminine (ex. تَسْمَعُ, tasomaEu, she listens); three number values: singular (ex. يَسْمَعُ, yasomaEu, he listens), dual (ex. يَسْمَعَانِ, yasomaEaAni, they listen) and plural (ex. يَسْمَعُونَ, yasomaEuwna, they listen); three tenses: the past (ex. سَمِعَ, samiEa, he listened), the present (ex. يَسْمَعُ, yasomaEu, he is listing), the future (ex. سَيَسْمَعُ, sayasomaEu, he will listen); and two voices: passive (ex. سُمِعَ, sumiEa, it was listened) and active (ex. سَمِعَ, samiEa, he listened). Arabic nominal morphology is a lot more complex than the verbal one (Habash, 2010); it includes gender, number, state, and case inflections. It has two gender values:

[4] A Logogram is a single character that can represent a morpheme, a word, or even an entire phrase (Ho and Bryant, 1997; Tan et al., 2005).

[5] A Morpheme is the smallest part of a word that has a significant meaning in a specific language.

masculine (ex. سَرِيعٌ, sariyEN, fast) and feminine (ex. سَرِيعَةٌ, sariyEapN, fast); three number values: singular (ex. مُعَلِمٌ, muEalimN, teacher), dual (ex. مُعَلِمَانِ, muEalimaAni, teachers) and plural (ex. مُعَلِمُونَ, muEalimuwna, teachers); three states: definite (ex. الرَجُلُ, Alrajulu, the man), indefinite (ex. رَجُلٌ, rajulN, man), and construct (ex. رَجُلُ الكَهْفِ, rajulu Alkahofi, caveman); and three case values: nominative (ex. المَدْرَسَةُ, Almadorasapu, the school), accusative (ex. المَدْرَسَةَ, Almadorasapa, the school), genitive (ex. المَدْرَسَةِ, Almadorasapi, the school). Unlike Arabic, English is not a highly inflected language. Its verbs inflect mainly for number (ex. the lion **eats**), the past tense (ex. helped) and the present tense (ex. looks). On the other hand, its nouns inflect only for the plural (ex. lions) and the possession (ex. John's).

Derivation is the mechanism that allows the creation of new words (inflected words) from other words by inserting one or multiple affixes. This process generally modifies not only the core meaning of the word but also its base syntactic category. The English language supports only very basic derivations. These derivations are generally obtained by adding prefixes (ex. do/**un**do, do/**re**do, way/**sub**way), suffixes (ex. reason/reason**able**, way/ way**ward**) or both (ex. reason/**un**reason**able**). Arabic derivational morphology is different from that of Indo-European languages which rely mostly on the concatenation of stems and affixes. Indeed, Arabic, among other Semitic languages, is based on a root-pattern system which uses two important components to compose words: roots and patterns. A Root also known in Arabic as الجِذر ("Alji*r") holds the basic semantic information (or an abstract meaning) that is shared between all the words that can be derived from it. We note that the Arabic roots are categorized based on the number of their unvocalized letters into triliteral (containing three letters), quadriliteral (containing four letters) and quintiliteral (containing five letters) roots (Habash, 2010). Patterns are abstract templates that take the basic root word and allow the creation of derivative words that have a certain syllabic structure and hold syntactic and semantic information. Table 4.1 presents a few derivations of the word "كَتَبَ" (kataba, meaning "to write").

The Arabic root or one of its derivations can also be combined with other prefixes and suffixes resulting in very morphologically rich words that can hold a considerable amount of information. Indeed, a single Arabic word can have the meaning of multiple words or even a whole sentence in other languages. For instance, the Arabic word "أفنلزمكموها" (">fnlzmkmwhA") has the meaning of the whole sentence "shall we then force you to do/accept it".

Arabic Derivation	Buckwalter transliteration	Meaning
كِتَابَةٌ	kitaAbapN	writing
كَاتِبٌ	kaAtibN	a writer
مَكْتُوبٌ	makotuwbN	written
مَكْتَبَةٌ	makotabapN	a library
كِتَابٌ	kitaAbN	a book

Table 4.1: An example illustrating some derivations of the Arabic word "كَتَبَ" (kataba)

4.2.1.3 Arabic Syntax

Syntax studies the mechanisms responsible for arranging words within a language to create meaningful phrases and sentences. When dealing with a morphologically rich language such as Arabic these mechanisms become heavily related to the morphological level. Thus, several syntactic aspects are not expressed uniquely via word order but also through morphology (Habash, 2010).

The Arabic language admits two types of sentences: verbal and nominal. A verbal sentence in Arabic starts with a verb. In its most basic form, it must contain a verb and a subject within a Verb-Subject word order. For example, the sentence "خرج الولد" (the boy went out) is a simple Verb-Subject sentence in which the verb is "خرج" (went out) and its subject is "الولد" (the boy). The object can also be present after the subject as in the sentence "كتب التلميذ الدرس" (the pupil wrote the lesson) which is a Verb-Subject-Object sentence in which the verb is "كتب" (wrote), its subject is "التلميذ" (the pupil), and the object is "الدرس" (the lesson). This is quite different from the English sentence structure which generally follows a Subject-Verb-Object word order.

A nominal sentence in Arabic does not require a verb. This kind of sentence is not present in English given that each sentence in the English language requires at least one verb. Nominal sentences in Arabic start with a noun and have two parts: a subject and a predicate. The subject (topic) needs to be a noun and the predicate can be a noun, an adjective, a pseudo-sentence [6], or a verbal sentence. For example, the sentence "الشمس مشرقة" (the sun is shining) is a simple Noun-Adjective nominal

[6] A pseudo-sentence in Arabic ("شبه الجملة") refers simply to a phrase that contains a preposition followed by a noun.

sentence resulting from the concatenation of two words: "الشمس" (the sun) and "مشرقة" (shining).

An important feature of Arabic syntax is the fairly loose word order. For instance, the different orderings of the three Arabic words أكلَ (ate), الولدُ (the boy), and التفاحةَ (the Apple) convey slightly different meanings[7]:

1. "أكلَ الولدُ التفاحةَ" in a Verb-Subject-Object order has the meaning of "the boy ate the Apple".
2. "الولدُ أكلَ التفاحةَ" in a Subject-Verb-Object order has the meaning of "(it is) the boy (who) ate the Apple".
3. "التفاحةَ أكلَ الولدُ" in an Object-Verb-Subject order has the meaning of "(it is) the Apple (that) the boy ate".
4. "أكلَ التفاحةَ الولدُ" in a Verb-Object-Subject order has the meaning of "the boy is the one who ate the Apple".

Even Though all the above possibilities are accepted in the Arabic language, the first one is the most common. This kind of flexibility is not found in the English language.

4.2.2 Arabic Machine Translation Challenges

Besides the above morphological and syntactic characteristics of the Arabic language that complicate the task of its translation, many other known problems pose serious challenges that are heavily studied by the Arabic MT community. We will briefly highlight the most important ones in the following subsections [8].

4.2.2.1 Arabic Vocalization

Arabic texts can be fully, partially, or not vocalized at all which causes many challenges to Arabic natural language processing applications (Habash and Rambow, 2007). A common method to address this problem is to perform a preprocessing step that removes all the diacritical marks and produces a fully unvocalized Arabic text.

Using unvocalized Arabic texts reduces the number of word forms (vocabulary size) which is often a positive thing for MT (Diab et al., 2007). However, the absence of vocalization may cause serious ambiguities, especially when dealing with short texts

[7]This Arabic word ordering example is inspired from the one given in (Alqudsi et al., 2014).

[8]For a detailed overview of Arabic MT difficulties we refer the reader to (Abuelyaman et al., 2014; Alkhatib and Shaalan, 2018; Farghaly and Shaalan, 2009).

with a limited context. Indeed, a single unvocalized Arabic word can have multiple meanings. For instance, depending on its vocalization, the Arabic word "ولد" (wld) can have all the meanings presented in Table 4.2.

Word	Buckwalter transliteration	Meaning
وُلِدَ	wulida	he was born
وَلَدَ	walada	he gave birth to
وَلَدُ	waladu	the son of
وَلَدٌ	waladN	a boy
وَلَّدَ	wala~da	helped someone else give birth

Table 4.2: An example illustrating multiple meanings that are driven from the same unvocalized Arabic word "ولد" (wld)

Even in the presence of some contextual words, the unvocalized Arabic words can still be ambiguous, or at least demand a considerable automatic natural language analysis to disambiguate them. For instance, if we consider the following simple Arabic sentence:

"ولد ذلك الشخص الذي رأيناه فار من العدالة"

(the son of that man that we have seen, is fleeing from justice)

The word "ولد" in this context means "the son of" ("وَلَدُ"), yet it is quite difficult for MT systems to figure this out, thus, they often fail to translate it. For example, the Google Translation Service [9] translates it as "that person whom we saw was born out of justice" which is wrong, as it assumes that "ولد" has the meaning of "وُلِدَ" (he was born), instead of "وَلَدُ" (the son of).

4.2.2.2 Arabic Words Polysemy

A polyseme refers to a word (or a phrase) that has multiple senses (meanings) (Fillmore and Atkins, 2000). The task of identifying the correct sense of a given word that has multiple meanings (an ambiguous word) in a given sentence (or text) is known as Word Sense Disambiguation (WSD). As we illustrated in the previous section, the vocalization of a word can help solve some lexical and semantical ambiguities in Arabic, yet even if the word is fully vocalized, it can still have several meanings depending on how it is used. For instance, the word "عَيْنُ" ("Eayonu") in Arabic has multiple meanings in English as shown in Table 4.3.

[9] https://translate.google.com

Arabic	English
عَيْنُ الماء	water source
عَيْنُ الرجل	man's eye
عَيْنُ المكان	the place itself

Table 4.3: An example illustrating a vocalized Arabic word that has several possible meanings

As shown in Table 4.3 the meaning of a single Arabic word can change depending on its context. For MT, this polysemy problem is not limited to Arabic but to both the source and the target languages that are involved. Indeed, if we consider the task of translation between English and Arabic then the problem of polysemy may also come from the English language. For example, the English word "take" has several meanings in Arabic as shown in Table 4.4, where its meaning changes when associated with different prepositions (e.g. out, off, up, down, back, on, in, by, etc.).

English	Arabic
take me to a place	خذني إلى مكان
take them down	إقضِ عليهم
take care of it	إهتم بالأمر
take a long time to load	يستغرق وقتًا طويلاً للتحميل
take off from the airport	تقلع من المطار
take back what you said	تراجع عن كلامك

Table 4.4: An example illustrating several meanings that the word "take" can express depending on its context

4.2.2.3 Arabic Multiword Expressions

A Multiword Expression (MWE) refers to a collocation of two or more words that appear together but whose meaning is not deducible or is only partially deducible from the semantic meanings of their constituents (Kordoni and Simova, 2014; Sag et al., 2002). Arabic MWEs are mainly idiomatic expressions that are frequently used by Arabic speakers in different contexts and whose translation is not evident. Some examples of Arabic idioms are given in Table 4.5.

MWEs have several aspects that make them difficult to translate (Constant et al., 2017), from which we cite the two main ones:

Arabic idioms	Literal words meanings	idiomatic meaning
ما باليد حيلة	no trick in hand	cannot be helped
بشق الأنفس	by cracking the souls	with enormous difficulty
خفيف الظل	with/having a light shadow	funny/pleasant

Table 4.5: Some examples of Arabic idioms

1. Non-compositionality: The overall expression of semantics is not related to individual pieces or words. This is the case, for example, if we consider the Arabic expression "طويل اللسان" which has the literal meaning of "long tongue" in English, where the individual words "طويل" and "اللسان" translate to "long" and "the tongue", respectively. However, this expression means a rude or indecent person.

2. MWE and non-MWE expressions ambiguity: It is hard to determine if the MWE is intended for its literal or idiomatic meaning. For instance, if we consider the Arabic phrase "حيوان طويل اللسان" (an animal that has a long tongue), the MWE "طويل اللسان" here has its literal meaning, not its idiomatic one.

4.2.2.4 Arabic Named Entities

Named entities (NEs) refer to all the entities that can be referenced by using proper names (Martin and Jurafsky, 2009), such as persons, locations, organizations, quantities, values, etc. The task of identification, extraction, and classification of these entities is known as Named Entity Recognition (NER). Depending on the application, this task generally considers four main categories: person names (PER) such as "أحمد" (Ahmed), locations (LOC) such as "الجزائر" (Algeria), organizations (ORG) such as "سامسونج" (Samsung), and miscellaneous (MISC) which includes all remaining types of entities (Shaalan, 2014). The task of NER is known to be much more difficult when performed on the Arabic language in comparison to most of the Latin languages due to many specific linguistic aspects that characterize Arabic. The difficulty is mainly due to the lack of capitalization, the rich lexical variations, and the lack of uniformity in writing Arabic named entities (Shaalan, 2014). In the field of MT, a proper handling of named entities is crucial to produce a reliable translation result. Indeed, named entities need to be addressed carefully as two possible treatments can be adopted depending on either a meaning-based translation or a phoneme-based transliteration (Shaalan, 2014):

1. Arabic meaning-based Translation: In this case, entities are translated according to their meaning. For example "الولايات المتحدة الأمريكية" (AlwlAyAt AlmtHdp Al>mrykyp) gets translated to "The United States of America".

2. Arabic Transliteration: Here entities (e.g. personal names) are transliterated. For example "نيويورك" (nywywrk) gets transliterated to "New York".

4.3 Arabic Machine Translation: Research work

The main focus of the Arabic machine translation community has been devoted to translating Arabic into English. Less interest has been given to the English-to-Arabic translation direction and even fewer research studies have been devoted to translating Arabic to other languages than English. The research studies developed in the field of Arabic MT mainly fall under the statistical machine translation paradigm; the other translation approaches have received less attention. In this section, we attempt to classify the Arabic MT research work based on the main translation approaches that have been followed (e.g. rule-based, statistical, etc.). Given that most of these research studies fall under the SMT and NMT paradigms, we have considered further categorizing them based on the main contribution of each research work. Our detailed classification of the Arabic MT studies is presented in Figure 4.2.

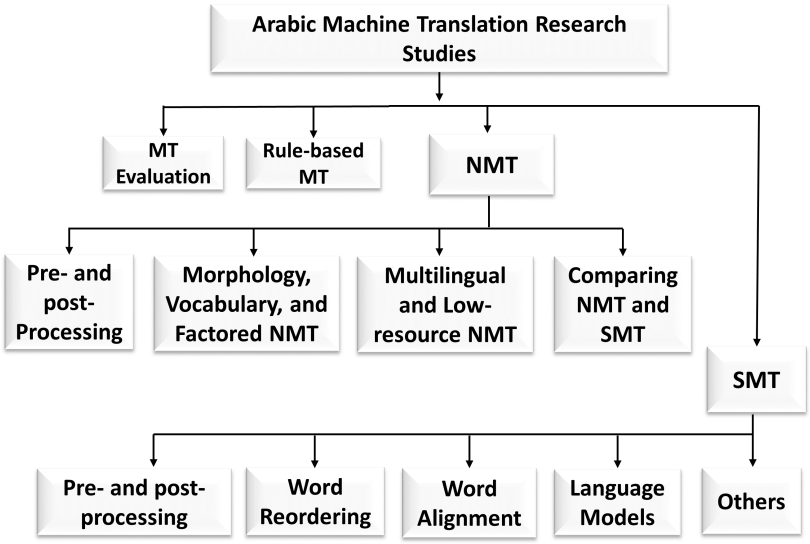


Fig. 4.2: Classification of Arabic MT research work

Table 4.6 presents the main contributions that have been made in the field of Arabic MT in a manner that follows our proposed classification (Fig. 4.2). The studies will be detailed in the remainder of this section.

Main Research Area	Sub-research Area	Main Research Studies
SMT	Pre- and Post-processing	**Arabic-to-English:** Lee (2004), Habash and Sadat (2006), Sadat and Habash (2006), Diab et al. (2007), Hasan et al. (2012) **English-to-Arabic:** Al-Haj and Lavie (2012), Badr et al. (2008), El Kholy and Habash (2012)
	Word Reordering	**Arabic-to-English:** Chen et al. (2006), Habash (2007), Carpuat et al. (2010), Carpuat et al. (2012), Bisazza et al. (2012) **English-to-Arabic:** Elming and Habash (2009), Elming (2008), Badr et al. (2009) **Arabic-Others:** Sadat and Mohamed (2013), Mohamed and Sadat (2015), Alqudsi et al. (2019)
	Word Alignment	**Arabic-to-English:** Ittycheriah and Roukos (2005), Fossum et al. (2008), Hermjakob (2009), Gao et al. (2010), Riesa and Marcu (2010), Khemakhem et al. (2015) **English-to-Arabic:** Ellouze et al. (2018), Berrichi and Mazroui (2018)
	Language Models	**Arabic-to-English:** Brants et al. (2007), Carter and Monz (2010), Niehues et al. (2011) **English-to-Arabic:** Khemakhem et al. (2013)
	Other Arabic SMT Studies	**Arabic-to-English:** Habash (2008), Marton et al. (2012) **English-to-Arabic:** Toutanova et al. (2008)
NMT	Pre- and Post-processing	**Between Arabic & English:** Sajjad et al. (2017), Oudah et al. (2019) **Arabic-Others:** Aqlan et al. (2019)
	Morphology, Vocabulary, and Factored NMT	**Between Arabic & English:** Ding et al. (2019), Ataman et al. (2020), Ataman et al. (2019), Liu et al. (2019), Shapiro and Duh (2018) **Arabic-Others:** Belinkov et al. (2017), Ataman and Federico (2018), García-Martínez et al. (2020)
	Multilingual & Low-resource	**Between Arabic & English:** Nishimura et al. (2019), Tan et al. (2019), Liu et al. (2020), Aharoni et al. (2019) **Arabic-Others:** Almansor and Al-Ani (2018), Ji et al. (2020)
	Comparing NMT & SMT	**Between Arabic & English:** Almahairi et al. (2016), Junczys-Dowmunt et al. (2016) **Arabic-Others:** Belinkov and Glass (2016)
Rule-based MT	-	**Arabic-to-English:** Salem et al. (2008) Shirko et al. (2010) **English-to-Arabic:** Soudi et al. (2002), Shaalan et al. (2004), Al Dam and Guessoum (2010)
MT Evaluation	-	**Arabic-to-English:** Hadla et al. (2014) **English-to-Arabic:** Guessoum and Zantout (2001), Guessoum and Zantout (2004), Adly and Al Ansary (2009), Al-Rukban and Saudagar (2017), Guzmán et al. (2016), El Marouani et al. (2018)

Table 4.6: The main contributions in Arabic MT

4.3.1 Research Studies on Arabic Statistical Machine Translation

Research work on statistical Arabic machine translation systems have primarily focused on one of the following aspects:

1. Improving the quality of statistical MT systems by performing a basic morphological pre- or/and post-processing of the Arabic language.

2. Using a syntactic reordering process to decrease the syntactic gap between the source and the target languages.

3. Improving the word alignment quality as a means of increasing the overall SMT translation performance.

4. Incorporating rich linguistic features via additional language models.

4.3.1.1 Morphological Pre- and Post-Processing

The Arabic language is known to have a very rich and complex morphology. To tackle this, the majority of the research studies on Arabic statistical machine translation have primarily focused on performing morphological pre- and/or post-processing of the language before the translation task.

Arabic-to-English Lee (2004) proposed a model that attempts to establish a syntactic symmetry between Arabic and English languages which have a very asymmetrical morphology. Her method aligns a word-segmented and POS-tagged Arabic corpus to a symbol-tokenized English corpus using IBM word alignment model 1 (Och and Ney, 2000). She tested her proposal on an Arabic-to-English phrase-based SMT baseline with parallel corpora of different sizes and reported that the morphological preprocessing is helpful only when dealing with a small corpus. As the size of the training data increased, the benefit from performing such a preprocessing decreased. Habash and Sadat (2006) and Sadat and Habash (2006) studied the impact of several Arabic morphological preprocessing schemes on the task of Arabic-to-English phrase-based statistical MT. They found that splitting off only the conjunction and particles gives the best performance when using a large amount of training data (they called this scheme D2). However, if the amount of training data is limited, then more sophisticated morphological analysis and disambiguation are needed to achieve better performance. They concluded that an appropriate preprocessing produces a significant increase in the BLEU score, especially when dealing with test data that is not fairly similar to the training one. Diab et al. (2007) investigated the effect of using different diacritization schemes that range from partial to full diacritization. They tested these schemes on an Arabic-to-English SMT baseline which involves no diacritization and found that the partial diacritization schemes do not have any significant impact on the baseline performance. They also found that a full diacritization scheme produces significantly worse results than the SMT baseline. Hasan et al. (2012) investigated the effectiveness of using a rule-based FST segmenter and an SVM-based statistical segmenter (from

the MADA toolkit[10]) on the source Arabic language. They tested their proposal on an Arabic-to-English statistical machine translation task and found that the rule-based segmenter performs poorly when dealing with large translation tasks and that the use of the MADA statistical tool, has yielded better performance but at the cost of a much slower speed.

English-to-Arabic Al-Haj and Lavie (2012) also investigated the impact of using Arabic morphological preprocessing. They tested several segmentation schemes ranging from fully unsegmented to fully segmented Arabic forms on an English-to-Arabic phrase-based SMT baseline and found that the main gain comes from splitting up the pronominal enclitics in their investigated schemes. Badr et al. (2008) and El Kholy and Habash (2012) tested the impact of pre- and post-processing the Arabic target language in English-to-Arabic statistical machine translation. The Arabic texts were first decomposed morphologically in a preprocessing phase, then recombined in a post-processing step via several recombination techniques. The authors showed that morphological processing leads to a significant improvement, especially when using a small parallel training corpus.

4.3.1.2 Syntactic Word Reordering

Some works have used a specific kind of preprocessing known as "syntactic reordering" which aims at decreasing the syntactic differences between the source and the target languages by using syntactic reordering rules.

Arabic-to-English Chen et al. (2006) proposed a reordering method that automatically extracts reordering rules from a parallel corpus and uses them to address the reordering phenomena in phrase-based SMT. They tested the use of either a fully lexicalized or unlexicalized set of reordering rules on the task of machine translation between several language pairs including Arabic-to-English. The authors reported a very small improvement on the IWSLT 2004 and 2005 Arabic-to-English evaluation test sets. Habash (2007) presented a reordering method for the Arabic-to-English MT task. He used a source Arabic dependency parse tree along with word alignment to automatically extract syntactic-level reordering rules from a parallel corpus. These rules have been used to reorder the Arabic training and testing data to match the target-side order. He investigated various alignment strategies and parsing representations

[10]MADA (Habash et al., 2009b) is an Arabic preprocessing toolkit that handles tokenization, diacritization, morphological disambiguation, POS tagging, stemming, and lemmatization.

and provided a comparative analysis of the different combinations of the investigated strategies. His approach gave a significant gain of 25% relative BLEU score when tested on an Arabic-to-English phrase-based machine translation baseline. Carpuat et al. (2010, 2012) used an Arabic noisy syntactic dependency parser to reorder verb-subject constructions into a pre-verbal subject-verb order. They tested their approach on an Arabic-to-English SMT baseline and reported a noticeable improvement in terms of both BLEU and TER scores. Bisazza et al. (2012) tried to address the syntactic disfluencies that are found in the Arabic-to-English phrase-based SMT systems. They proposed a chunk-based reordering method that automatically reorders the Arabic verbs of the source-side sentences that follow a Verb–Subject–Object word order. Their method uses a feature-rich discriminative model to predict the likelihood of each possible verb reordering for a given Arabic source-side sentence. They reported a 1 BLEU point increase on the NIST-MT 2009 Arabic–English translation benchmark. Alqudsi et al. (2019) proposed a method to handle word ordering problem in the context of Arabic-to-English MT. Their proposed method combines Rule-based MT (RBMT) with the Expectation-Maximization (EM) algorithm. They used parallel data from the United Nations (Arabic–English) corpus. They trained their model using 632 sentence pairs and reserved 271 sentence pairs for testing. Their results showed an increase of up to 0.89 BLEU points over their RBMT baseline system.

English-to-Arabic Elming and Habash (2009) investigated the effectiveness of applying the reordering approach which was initially proposed to translate from English-to-Danish by Elming (2008) on the task of English-to-Arabic machine translation. This approach attempts to learn probabilistic rules from a parallel corpus in a fully automatic way. They achieved an improvement in both manual and automatic evaluations. They also found that the rules that are learned from automatic alignments are more useful than those learned from the manual ones. Badr et al. (2009) proposed a source-based syntactic reordering for the task of English-to-Arabic translation. Their reordering is carried out on the English source parse tree via handcrafted reordering rules that have been built based on their knowledge about the linguistic transformations that need to be accounted for when translating from English to Arabic. They reported some improvements in their phrase-based statistical MT baseline system.

Between Arabic and Other Languages Sadat and Mohamed (2013) investigated several Arabic preprocessing schemes based on POS tagging and morphological segmentation on the task of Arabic-to-French machine translation. They used morphological

rules to reduce the Arabic morphology to a level that makes it similar to the French one. They also used some reordering rules to match the source Arabic language with the target one on the syntactic level. Their tests on an Arabic–French statistical machine translation baseline showed that their morphological preprocessing was indeed useful. Their syntactic reordering rules, however, did not result in any significant improvement. Mohamed and Sadat (2015) used handcrafted morphological reordering rules to reorder the source-side Arabic sentences in an Arabic-to-French translation task. Their rules attempt to reorder both the pronouns and verbs of the source-side Arabic sentences in a way that matches the target French language. They reported an improvement of around 1 BLEU point over their baseline statistical MT system.

4.3.1.3 Word Alignment

Some research studies attempted to improve the quality of word alignment as a means of improving the overall SMT quality.

Arabic-to-English Ittycheriah and Roukos (2005) presented a maximum entropy word alignment model for the task of Arabic-to-English machine translation. Their model was trained in a supervised way using annotated training data and compared to several other state-of-the-art word alignment algorithms such as IBM alignment model 1 (Moore, 2004; Och and Ney, 2000) and the HMM algorithm (Vogel et al., 1996). Even though a noticeable improvement has been obtained on the task of word alignment, it did not led to any significant gain in the overall machine translation results. Gao et al. (2010) proposed a semi-supervised word alignment algorithm that combines a discriminative and a generative alignment model which they named EMDC (Expectation-Maximization Discrimination Constraint). The discriminative model finds high precision partial alignments and the generative one uses an Expectation-Maximization algorithm to impose additional constraints. Their tests on Chinese-to-English and Arabic-to-English translation tasks reported a consistent gain in the translation quality. Riesa and Marcu (2010) presented a hierarchical search algorithm to address the problem of automatic word alignment. Their proposal introduces a forest of alignments from which they identify the best alignment points using a linear discriminative model that incorporates hundreds of features. Their test results on the task of Arabic-English word alignment showed a significant increase of 6.3 points in F-measure over a GIZA++ Model-4 baseline and a 1.1 BLEU score gain over a syntax-based statistical machine translation system. Hermjakob (2009) proposed a method to improve the task of Arabic-English word alignment and translation by

combining statistical and linguistic knowledge. Their method uses a bilingual lexicon produced by a statistical word aligner and a set of heuristic alignment rules generalized from a development corpus. Their proposal outperformed GIZA++ and LEAF aligners in terms of F-measure and produced a 1.3 BLEU score increase over a state-of-the-art syntax-based statistical machine translation baseline. Fossum et al. (2008) presented a link deletion method to improve the word alignment produced by the GIZA++ toolkit. They used lexical, structural, and syntactic features to detect and delete weak alignment links produced by the GIZA++ aligner. Their tests on the tasks of Chinese-to-English and Arabic-to-English word alignment gave a significant F-measure increase, yet the gain in the overall translation results have not been as significant. Khemakhem et al. (2015) proposed a method to improve the quality of Arabic-to-English statistical machine translation by including semantic-level knowledge. They first used a semantic word clustering method on the English side of the corpus to obtain semantic word classes. Then they linked these classes to Arabic using Arabic-English word alignment information. Their test results on the IWSLT 2008 and 2010 Arabic-English translation tasks showed up to a 1.4 BLEU score improvement over their statistical MT baseline.

English-to-Arabic Ellouze et al. (2018) proposed a hybrid approach to improve the alignment results of the GIZA toolkit. Their proposal uses linguistic features such as morpho-syntactic tags, syntactic patterns, and statistical features such as mutual information and harmonic mean. They trained their alignment model using an English-Arabic medical corpus and tested it on the Cambridge dictionary. They stated that their results showed an improvement of both the alignment and the translation quality. Berrichi and Mazroui (2018) presented an alignment approach that uses morphosyntactic features (stem, lemma, and POS tags) for the task of English-to-Arabic MT. They evaluated their approach using a phrase-based SMT system as their baseline and reported a significant improvement in both the word alignment quality and the overall BLEU score results.

4.3.1.4 Language Models

A few research studies have focused on investigating the effectiveness of a language model that incorporates new morphological, syntactic, or semantic features.

Arabic-to-English Brants et al. (2007) investigated the impact of using a very large statistical English language model on the task of Arabic-to-English statistical machine translation. Their language model was trained on over 2 trillion tokens and yielding up

to 300 billion n-grams. Their test results reported that the overall translation quality kept increasing gradually with the increase in the size of the language model even when reaching the largest size that they considered. Carter and Monz (2010) used a large-scale discriminative language model to re-rank the n-best list translations generated by an SMT system. Their test results performed on NIST's Arabic-to-English MT-Eval benchmarks reported an improvement of 0.4 BLEU points over the state-of-the-art phrase-based SMT baseline that they considered. Niehues et al. (2011) investigated the effect of using a bilingual language model that extends the translation model of a phrase-based SMT by including bilingual word context on the task of statistical machine translation. They tested their proposal on the tasks of German-to-English and Arabic-to-English machine translation and they reported an overall improvement of up to 1.7 BLEU points on the Arabic-to-English task.

English-to-Arabic Khemakhem et al. (2013) built an Arabic statistical feature-based language model that allows the incorporation of several grammatical features about each Arabic word. Their proposal was used to enhance the performance of an English-to-Arabic statistical machine translation system and they reported over 1-point BLEU score increase over the state-of-the-art phrase-based SMT baseline.

4.3.1.5 Other Arabic SMT Research Studies

Few researchers have investigated other morphological, syntactic, and semantic aspects to improve the quality of Arabic SMT systems.

Arabic-to-English Habash (2008) proposed several methods to address the problem of Out-of-Vocabulary (OOV) words in the task of Arabic-English SMT. His method used morphological, spelling, and dictionary enhancement; he also used a method for proper names transliteration. He reported a noticeable improvement over the state-of-the-art phrase-based SMT baseline in terms of BLEU score and a manual evaluation. Marton et al. (2012) investigated the effect of adding soft constituent-level constraints to the Arabic source parse tree on the task of Arabic-to-English machine translation. They also used a feature weight optimization technique to handle the problem of selecting the best features. Their tests on an Arabic-to-English hierarchical phrase-based translation system showed substantial gains in performance.

English-to-Arabic Toutanova et al. (2008) used a specific inflection generation model to predict the correct inflections of a specific target language stems based

on morphological and syntactic features extracted from both the source and target languages. Their proposed model was trained separately from the SMT baseline and tested on the task of translation of English to both Russian and Arabic. They reported an improvement of about 2 BLEU points on the task of English-to-Arabic translation.

4.3.2 Research Studies on Arabic Neural Machine Translation

The amount of research studies that have been devoted to the neural machine translation paradigm has seen a very significant increase in the last few years. In this section, we classify the current research studies that have been accomplished in regards to Arabic NMT into three main categories:

1. Pre- and post-processing: attempt to improve the quality of NMT systems by using pre- or/and post-processing treatments.
2. Morphology, vocabulary, and factored NMT: investigates the incorporation of different linguistic knowledge sources into baseline NMT systems.
3. Multilingual and low-resource translation: attempt to use multilingual neural machine translation under both rich- and low-resource settings.

4.3.2.1 Pre- and post-processing

Several research studies have been devoted to studying the effect of performing a pre- and/or post-processing treatments on Arabic NMT baselines.

Between Arabic and English Sajjad et al. (2017) investigated the effectiveness of three language-independent segmentations, namely: (1) Byte-Pair Encoding (BPE) (Sennrich et al., 2015), (2) Character-level Encoding (Ling et al., 2015b), and (3) Character CNN (Kim et al., 2016). They tested these segmentations on the Arabic-to-English and English-to-Arabic MT tasks and reported that the BPE segmentation produced the best results and even outperformed the state-of-the-art morphological segmentation (MADAMIRA[11]) on the Arabic-to-English translation direction by a slim margin of 0.2 BLEU points. They also found that the character-level encoding methods perform drastically worse than both the BPE and the morphological segmentation ones lagging behind by more than two BLEU points on the English-to-Arabic evaluation tests

[11]https://camel.abudhabi.nyu.edu/madamira/

that they performed. Oudah et al. (2019) compared the effect of different segmentation schemes on neural and statistical Arabic-English MT models. Their results showed that the effect of the segmentation scheme is closely related to the type of the used translation model. They found that a morphology-based segmentation scheme such as the one used by the Arabic Treebank (ATB) has been beneficial to both the NMT and SMT models. However, the improvement was higher for the SMT models (reaching an increase of up to 3 BLEU points). They also found that the combination of the ATB with the Byte-Pair Encoding (BPE) segmentation gave the best results for the SMT models but did not lead to an increase over the ATB segmentation for the NMT models. The overall conclusions were that ATB was the most useful segmentation for both SMT and NMT and that for SMT combining it with BPE can lead to an additional slight improvement.

Between Arabic and Other Languages Aqlan et al. (2019) proposed a romanization system that converts Arabic scripts to subword units to deal with the unknown words problem on the task of machine translation between Arabic and Chinese. They investigated the effect of their approach on the NMT performance while using various segmentation scenarios. They performed extensive experiments on Arabic-to-Chinese and Chinese-to-Arabic translation tasks and showed that their proposed approach can effectively tackle the unknown words problem and improve the translation quality by up to 4 BLEU points.

4.3.2.2 Morphology, Vocabulary, and Factored NMT

This section summarizes the research studies that have investigated the possibility of improving baseline NMT models via the incorporation of different linguistic knowledge sources.

Between Arabic and English Ding et al. (2019) tried to find the optimal vocabulary size for NMT models that uses subword units. They performed a wide range of experiments in which they varied the vocabulary size (the number of BPE merged operations) across several pairs of languages and reported the obtained results in terms of BLEU score. They found that for the Transformer-based Arabic-to-English and English-to-Arabic architectures the highest BLEU scores are obtained when the vocabulary size contains less than 1000 subword units. They reported a major drop in performance when the vocabulary size contains more than 8000 subword units. They also noted that the difference between the best and the worst performance is about

3 BLEU points. Ataman et al. (2020) proposed a novel NMT decoding method that models word-formation via a hierarchical latent variable that simulates morphological inflection. Their method is aimed at morphologically rich and low-resource languages such as Arabic. Their proposal generates words one character at a time by combining two latent variables; the first is used to represent the lemmas and the second one for the inflectional features. They compared their proposal to subword and character-level decoding methods on the task of translation from English into three morphologically rich languages: Arabic, Czech, and Turkish. They reported a slight improvement of 0.51 BLEU points over the best performing baseline on the task English-to-Arabic translation. Ataman et al. (2019) proposed a hierarchical decoding method for NMT that considers both words and characters when generating the translation. They compared their proposed method with different open-vocabulary subword-level techniques such as BPE across five pairs of languages with distinct morphological typologies. They showed that their hierarchical decoding model can give similar or even better results than the subword-level NMT models while using significantly fewer parameters. For the English-to-Arabic translation task, they reported an increase of up to 1.3 BLEU points over the baseline BPE subword-based NMT model. Liu et al. (2019) proposed a novel method that allows the sharing of source and target word embedding features in the context of a neural machine translation system. Their word embeddings are composed of two parts: shared features which are bilingual features used to improve the NMT's attention mechanism and private features that are used to capture the monolingual words features. The experiments that they have performed on five language pairs including Arabic-English showed significant performance increase over the Transformer baseline (an increase of up to 1.5 BLEU points for the Arabic-to-English translation direction) while using fewer model parameters. Shapiro and Duh (2018) proposed a method that extends the word2vec word embeddings model by allowing it to include morphological lemmas from a language-specific morphological analyzer (MADAMIRA[12]). They showed that their proposed model outperformed word2vec on an Arabic word similarity task. They also performed experiments on Arabic-to-English translation tasks using the TED Talks data and found that the usage of morphological word embedding led to an improvement of 0.4 BLEU points over the original word2vec embeddings model and more than two BLEU points when compared to the random initialization.

[12]https://camel.abudhabi.nyu.edu/madamira/

Between Arabic and Other Languages Belinkov et al. (2017) performed several tests regarding neural machine translation systems to identify the best possible morphological language-related representations. Their tests with different morphological processing on several languages such as French, German, Czech, Hebrew, and Arabic revealed that character-based representations tend to be better at learning morphology and that translating from a rich to a morphologically-poor language generally leads to better source-side representation. For instance, the character-based segmentation showed an improvement of more than 3 BLEU points over the word-based one on the Arabic-to-English translation direction, while on the English-to-Arabic their results were very similar. Ataman and Federico (2018) investigated the use of vocabulary reduction techniques to improve the quality of Neural machine translation (NMT) when dealing with morphologically-rich languages. They tested two unsupervised vocabulary reduction methods: Byte-Pair Encoding (BPE) (Sennrich et al., 2015) and Linguistically-Motivated Vocabulary Reduction (LMVR) (Ataman et al., 2017). They compared the two methods on ten translation directions that involve English and five morphologically-rich languages: Arabic, Czech, German, Italian, and Turkish. They showed that the performance of the subword segmentation method was better for the majority of the tested language pairs. As to the Arabic language, they found that the LMVR segmentation was better than the BPE for both Arabic-to-English and English-to-Arabic by around one BLEU point. García-Martínez et al. (2020) investigated the effect of using linguistic factors on the target-side of an Arabic-to-French factored NMT model [13]. Two pieces of information were predicted by their FNMT model at decoding time, the lemma and the concatenation of the following factors: POS tag, tense, gender, number, person, and the case information. Their training was done using a small or large parallel training dataset to simulate low-resource and rich-resource behaviors, respectively. They also investigated the usage of BPE segmentation for both their Factored and standard NMT architectures. Their evaluation results on several test sets showed that the factored NMT models were far better under low-resource conditions by an improvement of around 3 to 6 BLEU points over the baseline NMT. They also found that combining factors with subword BPE units achieved the best performance when trained under their rich-resource settings.

[13] Factored NMT architectures consider several linguistic factors (e.g. part-of-speech, case, number, gender) of the source and/or the target language to improve the overall NMT quality (Burlot et al., 2017; Sennrich and Haddow, 2016).

4.3.2.3 Multilingual and Low-resource Translation

Some research studies were interested in the usage of multilingual neural machine translation under both rich- and low-resource settings.

Between Arabic and English Nishimura et al. (2019) examined the usefulness of multi-source neural machine translation which incorporates multiple source inputs (from different languages). They also presented a method to use incomplete multilingual parallel corpora in which some source or target translations can be missing. They used UN6WAY multilingual corpus from which they selected Spanish, French, and Arabic as the source languages and English as the target language. Their proposed multi-source NMT model achieved an increase of up to 5.6 BLEU points over the best one-to-one NMT baseline. Tan et al. (2019) proposed a framework in which several languages are grouped into different clusters, each one trained as a multilingual model. They tested two approaches for language clustering: (1) using human-knowledge, thus clustering languages according to their families; and (2) using language embeddings, thus, representing each language via an embedding vector and then clustering them according to those embeddings. They tested their two clustering methods on the task of MT from 23 languages (including Arabic) to English. Their first clustering method placed Arabic and Hebrew in the same "Afroasiatic" cluster and the second one placed Arabic, Persian, and Hebrew in the same cluster. Their test results showed that the second clustering method was better almost in all scenarios, leading to an improvement of 0.25 BLEU points in the case of Arabic-to-English. Liu et al. (2020) presented mBART (Multilingual BART, which is an extension of the original BART[14]) a denoising auto-encoder that they pre-trained on several monolingual language corpora. They have shown that using mBART has led to an increase in performance of up to 12 BLEU points for some low resource sentence-level translation tasks and an increase of around 5 BLEU points for several document-level translation tasks. As far as Arabic is concerned, their mBART25 (pretrained on 25 languages) has led to an increase of 10.1 BLEU points on the task of Arabic-English MT. Aharoni et al. (2019) presented a large multilingual NMT translating 102 languages to and from English. Their test results have been reported on the TED talks multilingual test set, and they showed that their system has been particularly effective under low resource settings. As far as Arabic is concerned, their multilingual NMT achieved a BLEU score increase of 2 to 3 points

[14]BART (Lewis et al., 2020) is a denoising sequence-to-sequence autoencoder trained to denoise and reconstruct texts that have been corrupted by using arbitrary noising functions.

over the one-to-one NMT models on both English-to-Arabic and Arabic-to-English translation directions.

Between Arabic and Other Languages Almansor and Al-Ani (2018) presented a character-based hybrid NMT model that combines both recurrent and convolutional neural networks. They trained their model on a very small portion of the TED parallel corpora containing only 90K sentence pairs. They tested their model on the IWSLT 2016 Arabic-to-English and English-to-Vietnamese evaluation sets and they reported noticeable improvements in comparison to a standard word-based NMT model. For the case of English-to-Arabic translation, the improvement in BLEU score exceeded 10 BLEU points while the word-based NMT model completely failed to train using their very small parallel training corpus. Ji et al. (2020) proposed a transfer learning approach to handle the translation of low-resource languages based on cross-lingual pretraining. Their method trains a universal encoder on several source languages using a shared feature space. Once the universal encoder is pretrained on the monolingual source languages data, the whole NMT model will then be trained using parallel data and used in zero-shot translation scenarios. Their tests on Europarl (involving French, English, Spanish, German, and Romanian languages) and MultiUN (involving Arabic, Spanish, and Russian) test sets showed that their approach significantly outperforms both pivot-based and multilingual NMT baselines. As far as Arabic is concerned, their experiments on the MultiUN test set for the tasks of Spanish and Russian translation from and to Arabic reported an improvement that ranges from 1 to 3 BLEU points over their multilingual NMT baseline model.

4.3.2.4 Comparing Neural and Statistical MT Performance

Some research studies attempted to compare the performance of neural and statistical Arabic machine translation systems.

Between Arabic and English Almahairi et al. (2016) developed a neural machine translation system for the task of Arabic-to-English translation and compared its performance to that of a phrase-based SMT system. They performed extensive tests using several Arabic preprocessing configurations and found that the phrase-based and neural translation systems give similar results and that a proper Arabic preprocessing has a positive impact on both of them. They also observed that neural machine translation performs significantly better when evaluated on out-of-domain test sets. Junczys-Dowmunt et al. (2016) performed a large comparison between phrase-based

and neural MT systems for several language pairs including the Arabic-to-English and English-to-Arabic translation directions. They reported that the NMT results were on par or better than those of the phrase-based SMT. For some languages, a very slight increase has been observed; however, when the Arabic language was involved, noticeable increases that range between 1 and 9 BLEU points were reported over the phrase-based SMT systems. For both Arabic-to-English and English-to-Arabic a 3-point increase in BLEU score has been observed.

Between Arabic and Other Languages Belinkov and Glass (2016) compared the performance of phrase-based and neural MT systems on the task of Arabic-to-Hebrew translation. They tested the effect of tokenization on both NMT and PSMT systems and they also tested the impact of character-level models on the NMT system. The preprocessing step resulted in a significant gain for both NMT and PSMT. They reported that their NMT gave better results when compared to phrase-based MT and that char-based models led to a one BLEU point improvement over their baseline NMT system.

4.3.3 Research Studies on Arabic Rule-based Machine Translation

Very limited research studies were developed using the classical rule-based methods to address the task of Arabic machine translation.

Arabic-to-English Salem et al. (2008) investigated the process of developing a rule-based lexical framework specialized in Arabic language translation by using Role and Reference Grammar (RRG) models. They described the difficulty of incorporating the Arabic language characteristics in such a framework when developing an Arabic-to-English translation system. Shirko et al. (2010) developed a machine translation system that translates Arabic noun phrases into English using a transfer-based approach. They tested their system by translating 88 thesis and journal paper titles from the computer science domain and reported an accuracy of 94.6%.

English-to-Arabic Soudi et al. (2002) described a research project aiming to build an English-to-Arabic interlingual machine translation system. They proposed a mapping procedure that allows the mapping of different semantic concepts from English to Arabic in the interlingual representation. They also addressed some of the differences between English and Arabic such as agreement in number. They provided a simple example

illustrating their proposal. Shaalan et al. (2004) proposed a method to construct a transfer-based English-to-Arabic MT system specialized in translating complex English noun phrases into Arabic. Their system follows the analysis, transfer, and generation steps to transform the English noun phrases into Arabic. Their proposal was tested on the task of translating theses titles taken from the computer science domain, and they reported a 92% translation accuracy on a holdout test set. Al Dam and Guessoum (2010) investigated the effectiveness of using an artificial neural network (ANN) in a transfer-based English-to-Arabic MT system. They used a feed-forward neural network with two hidden layers as a transfer module which learns to transfer a tagged English sentence into a tagged Arabic one. Their tests showed that 56% of the test sentences were perfectly transferred and that 64.5% of them had at least 60% correct tags.

4.3.4 Research Studies on the Evaluation of Arabic MT Systems

Some researchers proposed new methods to better evaluate the quality of Arabic MT systems.

Arabic-to-English Hadla et al. (2014) evaluated two Arabic-to-English machine translation systems namely Google Translate and Babylon. They used a corpus of more than 1000 Arabic-English sentence pairs which associate two reference English translations for each source Arabic sentence. Their Arabic sentences were distributed among four sentence functions: declarative, interrogative, exclamatory, and imperative. Their experimental results reported in terms of the BLEU-score showed that Google's translation quality was better than Babylon's.

English-to-Arabic Guessoum and Zantout (2001) proposed a methodology for performing a semi-automatic evaluation of lexicons in the context of machine translation. Their method takes into consideration the importance of a given word in each possible domain; thus they gave an importance weight to each word in the lexicon based on its morphological properties and its specific sense. They used their proposed methodology to test the lexicons of three English-to-Arabic MT systems: Al-Mutarjim Al-Arabey, Arabtrans, and Al-Wafy[15]. They reported that all the tested systems gave a lexical coverage of more than 93% which they described as "acceptably good". In their later work, Guessoum and Zantout (2004) proposed a generalization to their lexicon-based

[15]Al-Wafi and Al-Mutarjim Al-Arabey are commercial Arabic MT systems developed by ATA Software Technology Inc, and Arabtrans is a commercial Arabic MT system developed by ArabNet.

evaluation in which the central idea of word sense weights was generalized to account for grammatical and semantic correctness. Their methodology was tested on four English-to-Arabic MT systems ATA, Arabtrans, Ajeeb, and Al-Nakel[16]. Their test results showed poor performance for all the evaluated systems that vary between 32% and 64% correctness on grammatical coverage, and between 51% and 84% on semantic correctness. Adly and Al Ansary (2009) proposed an Interlingua-based approach for the evaluation of English-to-Arabic machine translation systems. They used the Universal Networking Language (UNL), a formal declarative language that represents textual data in a semantic way. They compared their evaluation method with some commonly used automatic evaluation metrics such as BLEU, F-1, and F-mean and reported that their proposal was better especially when dealing with sentences that have a complex structure. Al-Rukban and Saudagar (2017) compared the performance of some English-to-Arabic translation systems using two metrics: BLEU and General Text Matcher (GTM). They considered three systems: Google Translator, Bing Translator, and Golden Alwafi. Their tests showed that Golden Alwafi was better in terms of quality as measured using BLEU, but that Google Translator surpassed it when using the GTM metric. El Marouani et al. (2018) investigated the impact of using linguistic features to evaluate the Arabic output of English-to-Arabic translation systems. Their proposed features consider semantic aspects from both the source and the target languages. The tests performed on a medium-sized corpus showed that their features helped improve the correlation between automatic and manual assessments. Guzmán et al. (2016) investigated the effect of incorporating some embedding features that have been obtained from different lexical and morpho-syntactic linguistic representations on the task of machine translation evaluation. They used a pairwise feed-forward neural network that takes the linguistically motivated embeddings of two possible translations and picks the best one among them. Their tests on the task of machine translation from English to Arabic showed that their proposal outperforms state-of-the-art evaluation metrics by an increase of over 75% in the correlation with human judgment as reported on a pairwise MT evaluation quality task.

[16]Al-Nakel is a commercial Arabic MT system developed by CIMOS.

4.4 Arabic MT Resources and Tools

The availability of linguistic corpora and tools has a huge impact on the overall machine translation quality. In this section, we will try to highlight the most substantial tools and resources that are available in the field of Arabic MT[17].

4.4.1 Arabic MT Parallel Training Datasets

Currently, a large amount of parallel corpora exists for the task of translation between Arabic and most of the top spoken languages in the world such as English, Chinese, French, Spanish, etc. The largest collections of freely available parallel corpora can be obtained from the OPUS website[18]. In the following, we will cite the two largest OPUS parallel corpora that include the Arabic language. We encourage the readers to visit the OPUS website to check their full collection of parallel Arabic corpora.

1. The United Nations Parallel Corpus v1.0 (Ziemski et al., 2016)[19]: a freely available parallel corpus built from manually translated UN documents between the years 1990 and 2014 for the six official UN languages: Arabic, Chinese, English, French, Russian, and Spanish. The corpus contains around 20 million sentence pairs for each translation direction between the concerned UN languages.

2. OpenSubtitles (Lison and Tiedemann, 2016)[20]: a large corpus built from movie and TV subtitles gathered from 60 languages including Arabic. The corpus contains more than 30 million English-Arabic sentence pairs and around 20 million sentence pairs for the tasks of translation between Arabic and most of the major spoken-languages such as Chinese, French, Russian, Spanish, etc.

Another major source that offers large collections of corpora is the Linguistic Data Consortium (LDC)[21]. LDC offers large translation data sets of Arabic newswire, weblog, forums, broadcast transcripts, and newsgroup texts. The majority of their data sets are translated manually validated rigorously by translation experts. In the following, we will cite some important LDC parallel corpora that include the Arabic language, but we note that there are so many, thus we encourage the readers to visit their website.

[17] We note that all the details and statistics regarding the mentioned data sets can be found in their provided links.
[18] http://opus.nlpl.eu/
[19] http://opus.nlpl.eu/UNPC-v1.0.php
[20] http://opus.nlpl.eu/OpenSubtitles-v2018.php
[21] https://www.ldc.upenn.edu/

1. Arabic Broadcast News Parallel Text (LDC2007T24, LDC2008T09, and LDC2015T07): contains English translations of several hours of Arabic broadcast news (more than 100k Arabic words).

2. Arabic Newsgroup Parallel Text (LDC2009T03 and LDC2009T09): contains English translation of texts (more than 300k Arabic words) driven from forums posts, discussion groups, etc.

3. Arabic Newswire English Translation Collection (LDC2009T22): contains texts from Arabic Newswire: Agence France Presse (France), An Nahar (Lebanon), and Assabah (Tunisia), along with their English translations (more than 500k Arabic words).

4. TRAD Arabic-French Parallel Text (LDC2018T13, LDC2018T21): contains French translations of a subset of more than 30k Arabic words developed under the PEA-Trad project[22].

4.4.2 Arabic MT Evaluation Datasets

Both free and paid test sets are available for the evaluation of Arabic MT systems. The free test sets generally provide only one reference translation for each source sentence, while the paid test sets provide multiple reference translations for each source sentence which allows a more reliable and robust evaluation. In the following, we will cite the most used and freely available Arabic MT evaluation test sets.

1. Arab-Acquis Dataset (Habash et al., 2017)[23]: is an evaluation set created from the European Union's Acquis Communautaire corpus which involves law-related textual data. Arab-Acquis dataset allows the evaluation of MT systems that translate between Arabic and 22 European languages.

2. International Workshop on Spoken Language Translation (Cettolo et al., 2012)[24]: also known as IWSLT, is a yearly scientific workshop that offers an evaluation campaign for spoken language translation. They released several test sets for the evaluation of Arabic-to-English and English-to-Arabic MT systems which are IWSLT 2012, IWSLT 2013, IWSLT 2014, IWSLT 2015, IWSLT 2016, and IWSLT 2017.

[22]http://www.elra.info/en/projects/archived-projects/pea-trad/
[23]https://camel.abudhabi.nyu.edu/arabacquis/
[24]https://wit3.fbk.eu/

3. The United Nations Parallel Corpus v1.0 Test Set (Ziemski et al., 2016)[25]: is an evaluation data set driven from the United Nations Parallel Corpus that allows the evaluation of translation between the six official UN languages: Arabic, Chinese, English, French, Russian, and Spanish.

Besides the aforementioned free test sets, many paid test sets are available for Arabic MT. In the following, we will cite the most important ones among them.

1. NIST Open Machine Translation Evaluation: offers a large collection of test sets (that can be obtained from the LDC website) intended for the evaluation of MT systems quality across several languages such as Arabic, English, Korean, Farsi, Urdu, Chinese, etc. The NIST MT evaluation test sets that involve the Arabic language are the following:

    - For the tasks of Chinese-to-English and Arabic-to-English translation: NIST OpenMT 2002 (LDC2010T10), NIST OpenMT 2003 (LDC2010T11), NIST OpenMT 2004 (LDC2010T12), NIST OpenMT 2005 (LDC2010T14), NIST OpenMT 2006 (LDC2010T17), NIST OpenMT 2008 (LDC2010T21).
    - For the tasks of Arabic-to-English and Urdu-to-English translation: NIST OpenMT 2009 (LDC2010T23).
    - For the tasks of Arabic-to-English, Chinese-to-English, Dari-to-English, Farsi-to-English, and Korean-to-English translation: NIST OpenMT 2012 (LDC2013T03).

2. TRAD Arabic-French Newspaper Parallel Test set[26]: an Arabic-French evaluation test set created from "Le Monde Diplomatique"[27] articles of the year 2012. It contains two parts: TRAD Arabic-French Newspaper Parallel corpus Test set 1 (ELRA-W0098) and TRAD Arabic-French Newspaper Parallel corpus Test set 2 (ELRA-W0100).

4.4.3 Monolingual Arabic Datasets

Monolingual corpora are very helpful for the task of MT and the field of NLP in general. They can be used to train word and sentence embedding models, language models, subword segmentation models, etc. Due to the importance of the Arabic language

[25]https://conferences.unite.un.org/UNCorpus/
[26]http://catalog.elra.info/
[27]https://www.monde-diplomatique.fr/

(one of the five most spoken languages in the world[28]), a good number of large-sized monolingual Arabic text corpora are currently available. In the following, we will try to mention the most substantial ones among them.

- Arabic Gigaword (LDC2003T12, LDC2006T02, LDC2007T40, and LDC2009T30): the four parts of the Arabic Gigaword contain a total of around two trillion Arabic tokens (more than 20GB of raw Arabic texts) gathered from newswire agencies such as Agence France Presse, Al Hayat News Agency, and Al Nahar News Agency.
- Arabic Newswire Part 1 (LDC2001T55): contains Arabic articles (a total of 76 million tokens) gathered from the French Press Agency.

Besides the LDC paid corpora, a large number of freely monolingual data sets are also available.

- Tashkeela (Zerrouki and Balla, 2017)[29]: a corpus containing 75.6 million vocalized Arabic words.
- KSUCCA Corpus (Alrabiah et al., 2013)[30]: a collection of classical Arabic texts concerning the period from the pre-Islamic era to the fourth Hijri century.
- The International Corpus of Arabic (Alansary and Nagi, 2014): 100 million words Arabic corpus extracted from different sources such as newspapers and web articles; it covers different domains such as science, literature, and politics.
- A collection of Arabic Corpora[31]: a website providing free access to multiple Arabic monolingual corpora such as "Ajdir Corpora" (113 million words), "Open Source Arabic Corpora" (20 million words), "Watan corpus" (12 million words) and "Khaleej corpus" (3 million words) (Abbas and Smaili, 2005; Abbas et al., 2011).

We note that besides the mentioned monolingual corpora, it is also possible to build web crawlers to extract Arabic texts automatically from Arabic news websites, Wikipedia, and other online sources. Also, all the aforementioned parallel Arabic corpora can be used as monolingual ones by just considering the sentences of their Arabic-side.

[28]https://www.conservapedia.com/List_of_languages_by_number_of_speakers
[29]https://sourceforge.net/projects/tashkeela/
[30]https://sourceforge.net/projects/ksucca-corpus/
[31]http://aracorpus.e3rab.com/index.php?content=english

4.4.4 Arabic Treebanks

Treebanks are fully parsed corpora that are linguistically annotated at both the sentence and word levels. Treebanks are very crucial for many NLP tasks such as Part-of-Speech tagging, Named Entity Recognition, Dependency and Constituency Parsing, Relation Extraction, Machine Translation, and Question Answering. In the following (Table 4.7), we will try to cite the most substantial Arabic Treebanks that are currently available.

Name	Ownership	Availability	Number of tokens
Arabic Treebank Part 1 v 4.1 (LDC2010T13)	LDC	Paid	145,386
Arabic Treebank: Part 2 v 3.1 (LDC2011T09)	LDC	Paid	144,199
Arabic Treebank: Part 3 v 3.2 (LDC2010T08)	LDC	Paid	339,710
Arabic Treebank: Part 4 v 1.0 (LDC2005T30)	LDC	Paid	1,000,000
Prague Arabic Dependency Treebank 1.0 (LDC2004T23)	LDC	Paid	113,500
OntoNotes Release 5.0 (LDC2013T19)	LDC	Free	300,000
The Quranic Arabic Corpus	Open Source	Free	77,430
Universal Dependencies Treebank	Open Source	Free	738,889

Table 4.7: Arabic Treebanks

The LDC Treebanks listed in Table 4.7 are all available on the LDC website[32]. The Universal Dependencies Treebank can be obtained from the Universal Dependencies website[33]. We note that at the time of writing this thesis the annotation of the Quranic Arabic Corpus[34] was not completed. A total of 30,895 out of the 77,430 words (around 40%) have been annotated, and the remaining part is still undergoing the annotation process.

4.4.5 Arabic MT Tools

In this section, we will present the most relevant tools that are important for the training and evaluation of both statistical and neural machine translation systems.

Table 4.8 summarizes some of the most substantial MT tools that can help train and evaluate MT systems. However, given that the field of NLP is currently seeing an unprecedented rise in the number of available tools and utilities that keep appearing

[32]https://www.ldc.upenn.edu/
[33]https://universaldependencies.org/treebanks/ar_nyuad/index.html
[34]http://corpus.quran.com/

Task	Tool's Name	Developed using	Description
End-to-end NMT	OpenNMT[35]	Python/Pytorch and TensorFlow	OpenNMT (Klein et al., 2018) is an open-source framework for sequence learning and neural machine translation.
	Fairseq[36]	Python/Pytorch	Fairseq (Ott et al., 2019) is a sequence modeling toolkit that implements different models for translation, language modeling, summarization, and text generation.
	Tensor2Tensor[37]	Python / TensorFlow	Tensor2Tensor (Vaswani et al., 2018) is a library that implements deep learning models for Machine Translation and several other NLP tasks.
End-to-end SMT	Moses[38]	C++ and Perl	Moses (Koehn et al., 2007) is a framework for training and testing statistical machine translation systems.
	Phrasal[39]	Java	Phrasal (Green et al., 2014) is a framework that aims for the fast training of both traditional machine translation models and large-scale discriminative translation models.
Word Segmentation	Subword-nmt[40]	Python	Subword-nmt is a GitHub repository that implements a set of preprocessing scripts for the training of subword unit (BPE) (Sennrich et al., 2016) segmentation models.
	SentencePiece[41]	Python	SentencePiece (Sennrich et al., 2016) is an unsupervised text segmentation and desegmentation tool that implements several subword units algorithms.
Word Alignment	GIZA++[42]	C++	GIZA++ (Och and Ney, 2003) is a tool for learning statistical word-to-word alignment models from parallel corpora.
	FastText Multilingual[43]	Python	FastText Multilingual Smith et al. (2017) is a tool that can be used to align two language vocabularies form their respective monolingual pretrained embeddings.
Word Embedding	Gensim[44]	Python	Gensim (Řehůřek and Sojka, 2010) is a library that offers a parallel implementation of many algorithms that can be used to train word embedding models.
	FastText[45]	Python	FastText (Bojanowski et al., 2017) is a GitHub repository that provides pretrained word embeddings for many languages including Arabic.
MT Evaluation	NLG Evaluation[46]	Python	NLG Evaluation (Sharma et al., 2017) is a framework that implements several MT evaluation metrics such as BLEU, METEOR, and ROUGE.

Table 4.8: Some relevant tools for training statistical and neural machine translation systems

[35]https://opennmt.net/
[36]https://github.com/pytorch/fairseq
[37]https://github.com/tensorflow/tensor2tensor
[38]http://www.statmt.org/moses/
[39]https://github.com/stanfordnlp/phrasal
[40]https://github.com/rsennrich/subword-nmt
[41]https://github.com/google/sentencepiece
[42]https://github.com/moses-smt/giza-pp
[43]https://github.com/Babylonpartners/fastText_multilingual
[44]https://radimrehurek.com/gensim/
[45]https://github.com/facebookresearch/fastText/blob/master/docs/crawl-vectors.md
[46]https://github.com/Maluuba/nlg-eval

each day, we encourage the reader to always visit GitHub and check for new tools and updates.

4.5 Discussion

In Section 4.3, we summarized the most relevant research studies that have been developed in the field of Arabic MT (see Table 4.6). These studies have been categorized according to the translation approach they used along with the MT problem/sub-problem that they tackled. In this section, we will highlight the most important research directions that have been investigated in the studies that have been made in regards to Arabic MT and give our remarks, observations, and comments about them.

After a careful analysis of what has been done in the field of Arabic MT from its early days up until now, we can make the following observations:

- The Arabic machine translation studies that have been accomplished so far have primarily been devoted to translating Arabic to English; English to Arabic translation has been of secondary importance. As to other languages than English, there are only very few studies (for some languages no research studies can be found at all) even though parallel corpora are currently available for the task of MT between Arabic and many other languages.

- When MT was mainly rule-based, work on Arabic was very bare; by the time interest grew for Arabic MT, MT had largely moved to data-driven translation methods. Currently, the MT research studies are focused heavily on the data-based approaches (mainly SMT and NMT) that do not require any linguistic expertise, thus, rule-based MT methods which are extremely demanding in both cost and human effort (Salem et al., 2008; Shaalan et al., 2004; Shirko et al., 2010) keep getting less and less attention over time.

- Arabic morphological analysis has been one of the most studied aspects in the field of Arabic MT. Indeed, the main concern when dealing with the Arabic language is its complex and rich morphology which is substantially different from that of Indo-European languages (such as English). These studies showed that Arabic tokenization and morphological segmentation can lead to some significant improvements in the overall translation results (Al-Haj and Lavie, 2012; Habash and Sadat, 2006; Sadat and Habash, 2006).

- Syntactic word reordering is also another very heavily studied aspect in the context of Arabic SMT. Indeed, the Arabic language is known to have a free word order which permits several possible word orderings, a thing that is not always permitted in other languages that tend to require a specific ordering. The majority of these reordering methods reported some clear gains in the overall translation results (Bisazza et al., 2012; Habash, 2007).

- Word alignment is also a subtask of SMT; it has been investigated by the Arabic MT research community. The proposed approaches attempted to improve it by introducing new linguistic features to boost the alignment quality. Even though some studies have managed to significantly improve the alignment results the impact of this gain on the overall translation results has not always been as significant (Fossum et al., 2008; Ittycheriah and Roukos, 2005).

- Feature-rich language models also have been investigated in the context of Arabic MT, yet the few studies that have adopted this approach have used different features and integration methodologies. Thus, drawing a clear conclusion about the effectiveness of this path is not trivial since the impact of these models is completely dependent on the considered features.

- Neural machine translation is the newest emerging paradigm for MT; yet, a considerable amount of research studies have been recently made with regard to it for the Arabic language. These studies have reported very encouraging results which were often better than the SMT-based ones, especially when tested on out-of-domain data (Almahairi et al., 2016; Junczys-Dowmunt et al., 2016).

- Multilingual machine translation approaches that use a single model to translate between multiple languages have shown very promising translation results. Indeed, recent studies demonstrated their effectiveness not only for low-resource languages but also for languages that have large parallel data such as Arabic. The studies also showed that these models were capable of capturing shared representational features across languages, thus offering better transfer capabilities which lead to larger gains in translation quality (Aharoni et al., 2019; Nishimura et al., 2019; Tan et al., 2019).

4.6 Conclusion

In this chapter, we have provided a comprehensive overview of the different research studies that have been proposed in the field of Arabic MT. To summarize, we first introduced the Arabic language, its characteristics along with some of its translation difficulties. Then, we presented the MT paradigms and highlighted some of their strengths and weaknesses. Next, we provided an overview of the important research studies that have been made so far on Arabic MT in a categorized way that allows the reader to distinguish with ease the different research axes and contributions that have been proposed. Finally, we provided a quick summary of the most relevant studies in each distinct area and discussed their effectiveness and shortcomings. In the next chapter, we will talk about the contributions we have made to pre- and post-processing in the context of statistical machine translation.

Chapter 5

Improving English-to-Arabic SMT via Syntactic Reordering

5.1 Introduction

When translating between two languages that are noticeably different in terms of their grammatical structures, the task of producing a high-quality translation in a correct word order becomes a serious challenge. Finding a better way to model these grammatical transformations or what is known as reordering phenomena, was and still is a long-standing problem which receives a great deal of attention from the machine translation community. The classic Phrase-based Statistical Machine Translation System (PSMT) Brown et al. (1990a) has two means for word reordering: it can either learn the whole bi-phrase as an entry in the phrase table (generally a bi-phrase length does not exceed a certain limit) or via the distortion model which allows limited-phrase movements (reorderings) in the output, but with a certain penalty. These means cannot address reorderings that involve a long-distance jump or what is known as long-distance reordering. This problem represents a well-known limitation for the standard PSMT system, hence the need to provide a more sophisticated model to solve it.

Syntactic reordering in MT is a research area that aims to find more efficient solutions so as to handle both short- and long-distance reordering problems. One common way to perform reordering as a standalone process is known as preordering. Preordering is a preprocessing step that precedes the PSMT phase; its goal is to minimize the syntactic gap between languages and make their grammatical construction as close as possible. Preordering is commonly used to address the long-distance reordering problems; it has the advantage of being easy to use and independent from the used translation system.

Figure 5.1 gives an example of short- and long-distance word reordering phenomena performed on an aligned English-to-Arabic sentence pair. Both the English and Arabic texts are written from left-to-right to keep the alignment order consistent. In the first example (a) the word "announced" appears at the end of the English sentence, aligned to the Arabic word "أعلن" by means of the 6^{th} alignment link. Preordering will attempt to swap the position of the first and the 6^{th} alignment links, which moves the word "announced" to the beginning of the English sentence to match the Arabic sentence order as shown in the second example (b). Since these two links are separated by a large margin we call this reordering a long-distance word reordering. Short reordering cases such as the one involving the second and the third links are called short-distance reorderings.

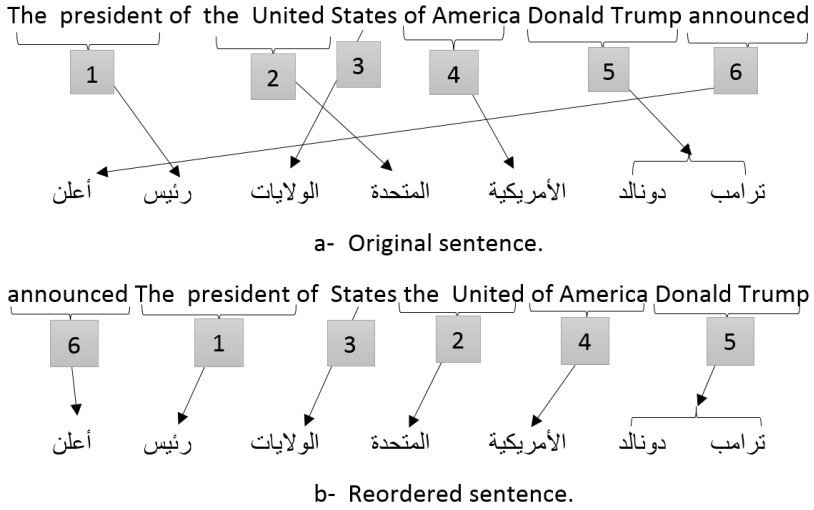


Fig. 5.1: An example illustrating the process of word reordering performed on an English-to-Arabic word-aligned sentence pair. (The English and Arabic texts are written from left-to-right to keep the alignment order consistent)

The goal of word preordering is to perform a reordering process on the English side of the corpus prior to the translation phase. This is done by finding and applying a set of syntactic rules which helps tweak the grammatical construction of the English-side language making it as close as possible to the Arabic-side language structure, which can also be seen as the task of minimizing the alignment links, as shown in Fig. 5.1 (b).

In this chapter, we present our first contribution which addresses the problem of English-to-Arabic word reordering by proposing a preordering method that can efficiently handle both long- and short- distance word reorderings. The reordering rules

are learned automatically from a parallel corpus using word alignment and a basic part-of-speech source language tagging. The test results showed a noticeable improvement over the baseline PSMT system which proves the consistency and adequacy of our proposal. The remainder of this chapter is organized as follows: the next section gives an overview of the state-of-the-art preordering methods. Section 5.3 presents our proposed reordering system and explains the details of each of its components. In Sect. 5.4, we present and discuss the tests we have done and the results we have obtained. Finally, in Section 5.5, we conclude and highlight some possible future improvements.

5.2 Researches Studies on Word Preordering

Preordering methods can be classified into two main categories: the deterministic approaches which provide the decoder with only one optimal reordering; and the non-deterministic methods which feed the decoder with multiple candidate sentences in the form of a weighted lattice, and it is then up to the decoder to find the best choice among them.

In terms of deterministic methods, one of the earlier works was done by Xia and McCord (2004), whose system deals with the task of French-to-English MT. They automatically extracted syntactic rules (which they called rewrite patterns) from a bilingual corpus, using syntactic parsers of the source and target languages along with word alignment. They reported a 10% relative improvement in the Bleu Score. Habash (2007) proposed a preordering method for Arabic-to-English translation. He used word alignment and a source dependency parse tree to automatically extract syntactic reordering rules. The extracted rules were used to reorder the Arabic training and testing data. He investigated various alignment strategies and parsing representations and provided a comparative analysis of the different combinations of the investigated strategies. Genzel (2010) defined the reordering task as a dependency parse tree transformation, in which the goal is to find the best children order for each internal node that has more than two children. He proposed a number of metrics for rule quality estimation which allows the filtering and selection of higher quality reordering rules. His proposal was tested on the task of translation from English to various other languages. In a similar fashion, Yang et al. (2012) performed the reordering task on a dependency parse tree by reordering the children of each internal node. They handled the position of each node as its rank making the reordering a ranking problem in which the task is to find a certain function f that determines the best rank of each child. Then, the children get sorted according to their ranks. For the task of translation

from Chinese to Japanese, Sudoh and Nagata (2016) used a learning-to-rank model based on a pairwise classification method to predict the target Japanese word order. In the same spirit, Jehl et al. (2014) proposed a feature-based reordering model for English-to-Japanese and English-to-Korean translation. Their model predicts whether a pair of sibling nodes on the source-side of the parse tree needs to be swapped. Based on the node swapping probabilities, a global branch-and-bound search is applied to find the best ordering of the children. Fuji et al. (2016) proposed a global reordering model that captures language-specific sentence structure directly from non-annotated corpora and used it to boost the performance of a conventional syntactic reordering system.

In terms of non-deterministic methods, Zhang et al. (2007) presented a preordering strategy for Chinese-to-English translation using chunk-based syntactic rules. They used a source-reordering lattice instead of a single best reordering, and a reordering source language model as an additional feature to score each path in the lattice. Elming (2008) presented a preordering approach for English-to-Danish translation. His proposed approach automatically learns probabilistic rules from a parallel corpus. The reordered sentences are fed via a lattice to the SMT decoder. He reported an absolute improvement in the translation quality of 1.1% in Bleu Score.

Despite the existing work on word preordering, to the best of our knowledge, no strategy has appeared to give convincing reordering results, hence the continuing efforts to improve them. This study aims to introduce a new, efficient way for both rule identification and application, along with a method for estimating rules usefulness in the reordering process.

5.3 Preordering System

Our proposed preordering system automatically learns syntactic reordering rules and uses them to change the grammatical structure of the English source-side sentences making them as close as possible to the target Arabic one.

Figure 5.2 shows the architecture of our proposed preordering system. There are two main steps: first, a set of reordering rules will be extracted from a parallel corpus, then each rule will be evaluated using a specific rule evaluation mechanism. The second step applies the selected rules to reorder both the training and test data prior to their exploitation in the PSMT.

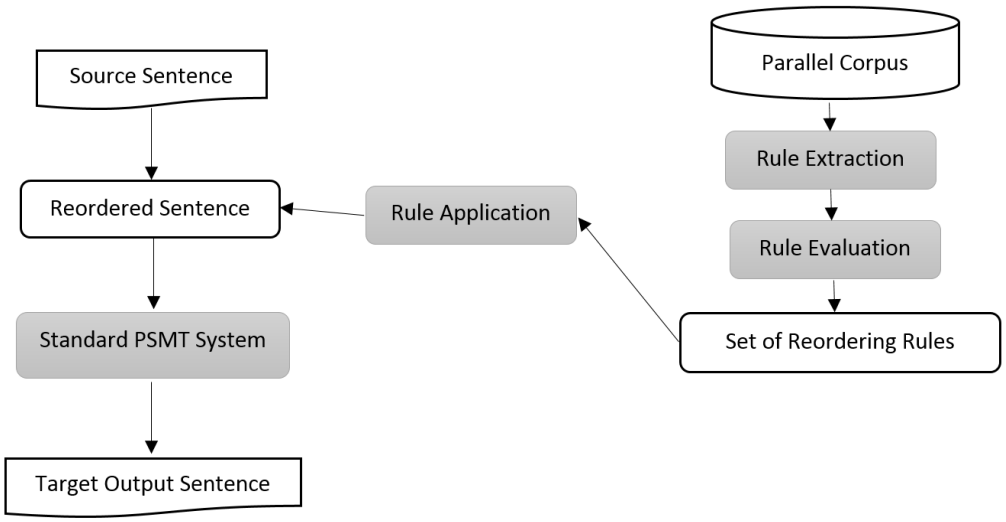


Fig. 5.2: Architecture of the preordering framework

In the following we will define our syntactic rules and show how to extract and apply them. Then we show how to evaluate the quality of each reordering rule and select only the useful ones among them.

5.3.1 Reordering Rules Definition

In this study, the reordering rules are composed of part-of-speech tags only; as such all the used rules are unlexicalized. Two tagsets are considered: the English Penn Treebank (PTB) tagset that uses 48 tags (Marcus et al., 1993), and the English Universal (Univ) tagset that uses 17 tags (Petrov et al., 2012).

Using high-level tags (more general tag classes) will result in more general rules while using more specific tags will allow the rules to capture more accurate contextual information albeit with a low generalization ability. The intuition behind using these two different tagsets is to investigate in a practical way the impact of the tags fineness on the reordering performance.

Our reordering rules are composed of three parts: the condition, the reordering, and an optional context. A rule condition may have more than one possible reordering, each reordering having its own specific context.

Table 5.1 shows an example of reordering rules with PTB part-of-speech tags. The second column presents the condition part of the rule and the third column shows all its corresponding reorderings. The context is presented as a pair (previous tag, next tag) which need to appear before and after the condition part of the rule. For example, for the first rule, the sequence of tags “DT NNP NNPS” needs to be present in the

Rule Number	Rule Condition	Rule Action (Reordering)	Rule Context
1	DT NNP NNPS	2, 0, 1	(IN, IN)
2	IN DT NNP NNPS	3, 0, 1, 2	(NN, IN)
3	DT NN IN ... VBD	10, 0, 1, ..., 9	(Non, Non)

Table 5.1: An example of reordering rules

sentence. Additionally, the left and right contextual tags "**IN** DT NNP NNPS **IN**" also need to appear before and after the condition. In such a case, the reordering "2, 0, 1" can be applied to reorder the tags producing a new order "NNPS DT NNP".

5.3.2 Reordering Rules Extraction

The reordering rules are extracted using only word alignment and a tagged source text of the parallel corpus. Figure 5.3 shows the overall process of rules extraction.

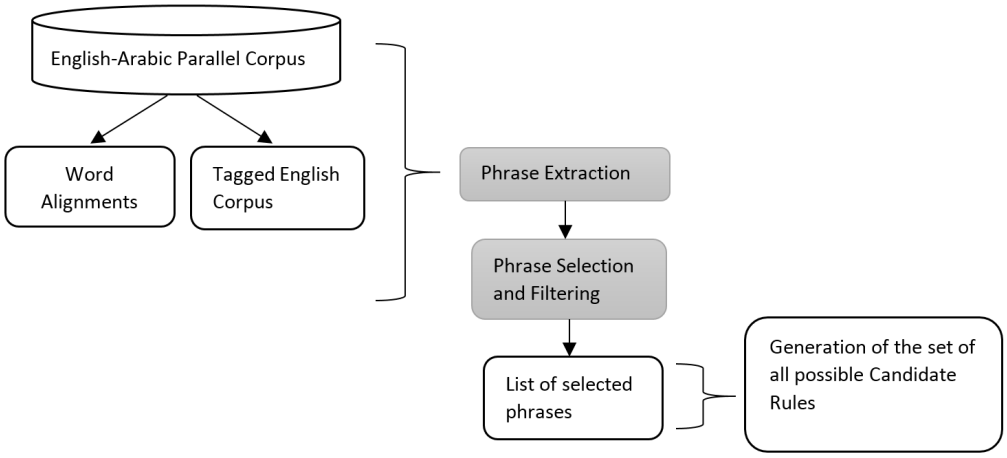


Fig. 5.3: Rules extraction mechanism

First, the bi-phrases are extracted using word alignment with a tagged source text; the phrases are then filtered by imposing some restrictions; and, finally, the candidate rules are formed from the selected phrases.

Figure 5.4 shows an example of word alignment with a tagged source text. The Univ and PTB tags are presented for each word in the English source sentence. In the first step, phrase extraction is done using the standard phrase extraction algorithm described by Koehn (2009), which uses word alignment to extract bi-phrases from a parallel corpus. From our previous example of Fig. 5.4, the phrase extraction algorithm extracts a total of 25 bi-phrases, some of which are shown in Table 5.2.

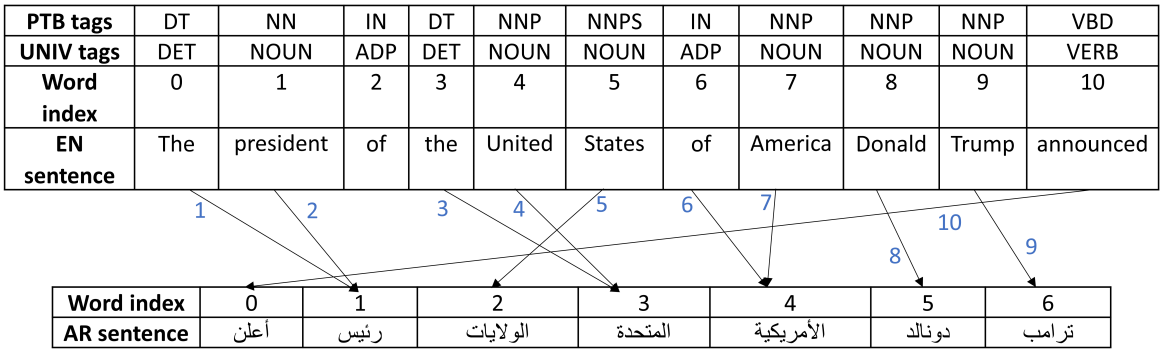


Fig. 5.4: Word alignment with tagged source-side, in which both the English and Arabic texts are written from left-to-right to keep the alignment order consistent

English Phrase	Arabic Phrase
Donald Trump	دونالد ترامب
the United States	الولايات المتحدة
announced	أعلن
The president announced	أعلن رئيس
The United States of America	الولايات المتحدة الأمريكية

Table 5.2: Some extracted bi-phrases from the example given in Fig. 5.4 using the standard phrase extraction algorithm described in (Koehn, 2009)

Having a set of English-Arabic bi-phrases, we select the pair of bi-phrases that contain a crossing. For instance, in Fig. 5.4 the phrases denoted by the links 4 and 5 crosses each other (by abuse of language, since the links cross each other); thus swapping them will minimize the number of crossing links; which makes the English sentence structure more similar to the Arabic one. We will be using this kind of crossings of bi-phrases in-order to form our syntactic rules.

Given two bi-phrases $p_1 = (s_1, t_1)$ and $p_2 = (s_2, t_2)$ where s_i and t_i are phrases from the source and target sentences, respectively, the two bi-phrases p_1 and p_2 are considered valid to form a syntactic rule if they satisfy the following conditions:

1. If s_1 precedes s_2 in the source sentence, then t_2 must precede t_1 in the target sentence. In other words, the two bi-phrases must cross each other.

2. The two bi-phrases must be consecutive in both the source and the target sentences. In other words, s_2 must follow s_1 and t_2 must follow t_1.

For example in Fig. 5.4, the two bi-phrases denoted by the links 5 and 4 respect these two conditions. The two bi-phrases 5 and 3 do not respect the second condition (they are not consecutive in the English text).

The set of selected phrases are then used to generate unlexicalized syntactic rules; the left and right tags that precede and follow the two phrases are used as context.

5.3.3 Reordering Rules Evaluation

The extracted rules are not always useful for reordering. In fact, most of them are very specific, which makes their coverage quite limited. Another issue resides in the errors introduced by the automatic word alignment which increases the rate of incorrect rules.

To tackle these problems, a number of metrics that estimate rules quality have been proposed. The metric that is most used is the Crossing Score (CS) (Genzel, 2010) which determines the quality of a rule based on the decrease in crossing alignment links after its application.

In practice, the quality of a rule is tested on the whole training corpus by applying the rule to each of its sentences and evaluating the change in the number of crossing alignments. This should give a solid estimation of the rule quality.

Applying this kind of metric directly will be computationally expensive since each rule is generally evaluated separately. Another issue is to perform the reordering task when given a set of rules. This involves finding all the applicable rules and determining the best order for their application. To this end, we build an index that accelerates both rules lookup and rules application.

5.3.3.1 Index Construction

We build an index, which is a compact Trie (De La Briandais, 1959). To reduce rule lookup time, this index will be used for the tasks of rules evaluation and application.

Formally, given a set of rules R, with their conditions driven from a set of tags G, such that $|R| = n$ and $|G| = m$. Each rule r in R, is a tuple (c, a, x), where c is the condition, a is the action and x is the rule context.

We construct a compact Trie T on R which has the following characteristics:

- It has a root node and n leaves (since each rule condition c ends at a leaf node which contains its corresponding action a and context x).
- For each internal node, its descendants have the same prefix (the same condition).
- Two branches leaving the same node cannot start with the same prefix.

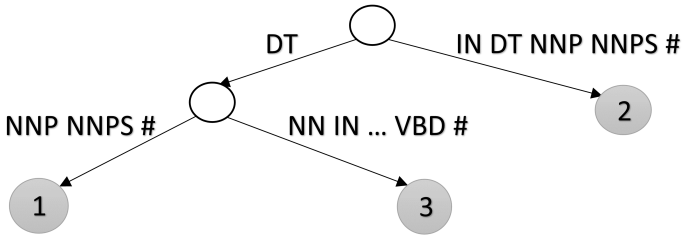


Fig. 5.5: Syntactic rules indexing via a compact Trie

Figure 5.5 shows a compact Trie constructed over the tag-sequences of the previous set of rules from Table 5.1. The leaf nodes are numbered according to their corresponding rules and the labels are printed on the edges. A special end tag # is added to ensure that each sequence terminates at the level of a leaf node. The reordering and context for each rule are kept in the leaf node that corresponds to it.

5.3.3.2 Efficient Search for Applicable Rules

The task of finding all the applicable rules for a given sentence is very challenging given the number of rules and the variations in their corresponding part-of-speech tags. Algorithm 1 presents an easy and efficient way to identify all the applicable rules for a given sentence using a compact Trie representation for all the reordering rules. In-order

Algorithm 1: Algorithm *FindAll* that finds all the applicable rules for a given sentence

Input : T: a Trie constructed over a set of rules R.
$S_t = t_1, t_2, \ldots t_k$: the part-of-speech tags of the English sentence S where k is the length of S.
Output : *candidate_rules*: a list that contains all the applicable rules for S.
Function `FindAll(`R, T, S_t`):`

```
    foreach suffix s_t in S_t starting at postion i do
        rules_i = find all applicable rules for s_t in T;
        foreach rule r in rules_i do
            candidate_rules.add ((r, i))
        end
    end
return candidate_rules;
```

to find all the applicable rules for a sentence S, Algorithm 1 finds all the applicable rules for each suffix (each position) in the tagged source sentence S_t; this is done by traversing the Trie starting from the root node and following the path led by each suffix in s_t. The rules found for each suffix are accumulated in *candidate_rules* and

returned when the algorithm terminates. All the applicable rules for S can be found in $O(k^2)$ time, where k is the length of S.

5.3.3.3 Best Applicable Rule Selection

Having the set of applicable rules for a given sentence, we need to determine the best one among them. This is done in two steps:

1. Sort the obtained candidate rules in decreasing order (from the longest to the smallest) according to the length of their condition parts. This is done to ensure that the rules concerning long-distance reorderings are applied first.

2. Select the best rule from the top k [1] sorted rules that have the longest condition parts based on their evaluation scores. The process of rule scoring will be detailed in the following section.

5.3.3.4 Rule Quality Evaluation

As mentioned in Sect. 5.3.3, since the majority of rules are not useful for reordering purposes, a good method for rule quality estimation is needed. Algorithm 2 scores the rules which can be applied to a given sentence using the CS metric (refer to section 5.3.3).

Algorithm 2 starts by identifying the best applicable rule for a given sentence S as described in Sect. 5.3.3.3. A close list is then used to prevent the rules from being reused in the same position. The best rule is then applied to reorder the word-aligned sentence, and the number of crossing alignments is then estimated using the CS metric. The score of the applied rule is then updated based on the CS difference [2]. This process is repeated for several iterations as indicated by the *max_iterations* variable.

After scoring all the rules, we estimate the usefulness of a given rule r by taking the ratio of the number of times the rule gave a positive impact on the reordering task and the total number of its applications:

$$usefulness(r) = \frac{positive(r)}{usage(r)} \tag{5.1}$$

In case the rule usefulness surpasses a certain threshold, it will be considered useful and selected for reordering. Applying this equation on the whole corpus will select a

[1] The value of k is selected experimentally.

[2] $FindCS$ is a simple method that finds the number of crossing alignments (CS) for a given aligned sentence.

Algorithm 2: Algorithm *UpdateSent* that updates the evaluation score for each syntactic rule

Input : R: a set of rules.
T: a Trie constructed over R.
$S_t = t_1, t_2, \ldots t_k$: S part-of-speech tags for the sentence S.
S_a: alignment points for the sentence S and its target translation.
close: a close list.
Output : Updates the scores for each applicable rule in T for the sentence S.

Function `UpdateSent(`R, T, S_t, S_a, *close*`):`

```
original_CS = findCS(S_a)
while i < max_iterations do
    candidate_rules = FindAll(R, T, S_t) that are not present in close;
    r_best = find the best rule in candidate_rules;
    insert r_best in close;
    S'_a = reorder S_a using r_best;
    new_CS = findCS(S'_a);
    r_best.usage+=1;
    if new_CS < original_CS then
        r_best.positive+=1;
    end
    else if new_CS > original_CS then
        r_best.negative+=1;
    end
    else
        r_best.neutral+=1;
    end
    i = i + 1;
end
```

subset of useful reordering rules (since not all the rules can be applied in all possible contexts). This process of rules usefulness estimation is repeated for several iterations until the subset of the useful rules stabilizes, which indicates that a convergence point has been reached.

Figure 5.6 shows the variation of the number of selected rules when estimating the rules' usefulness.

The number of useful rules (Fig. 5.6 (a)) increases with the number of epochs and at the same time, the count of non-useful rules (Fig. 5.6 (b)) decreases until a convergence is achieved.

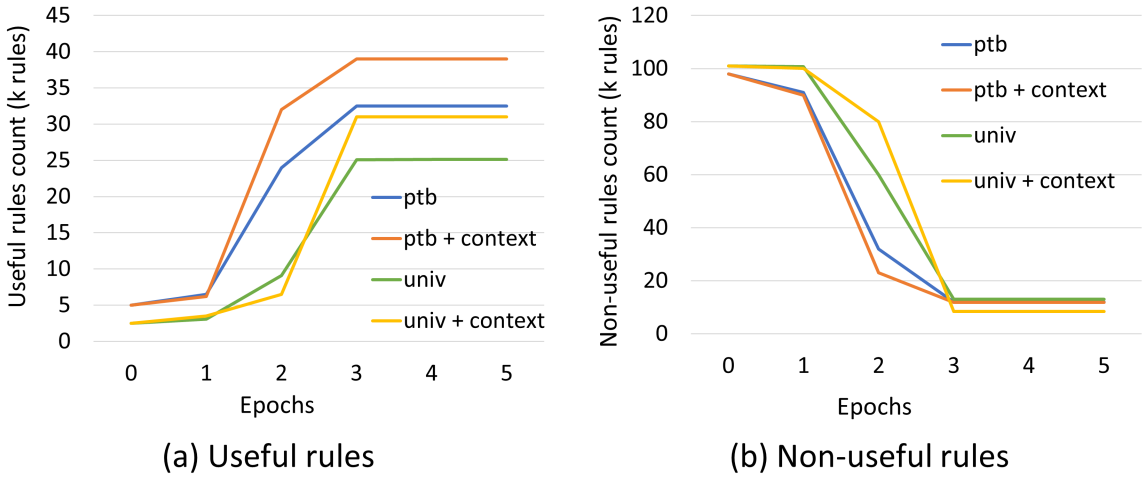


Fig. 5.6: Rules count variation when applying the process of rules filtering

5.4 Experiments

To test our approach, we have used the English-Arabic parallel corpus provided by the IWSLT2016 [3] evaluation campaign which offers a complete testing framework which includes: training, development, and evaluation data. Our results have been obtained on the IWSLT 2010, 2011, 2012, 2013 and 2014 test sets. Tagging is done using the Stanford English Log-linear Part-Of-Speech Tagger (Toutanova et al., 2003). The Univ part-of-speech tags are obtained by converting the PTB tagset using a simple tag mapping method [4].

5.4.1 Preprocessing

For the Arabic language, our preprocessing includes diacritic sign removal, Arabic character normalization, and word segmentation by means of the AMIRA toolkit (Diab, 2009) using the default tokenization scheme in which conjunctions, prepositions, determinants, suffixes, and future markers are all individually separated. For the English side, only word tokenization is performed using the Python NLTK toolkit [5]. We have also added a number <*nbr*> and a link tags <*url*> to all numbers and links found in the parallel corpus respectively. Sentence length has been limited to 40 words; "bad" sentence pairs, i.e. whose length difference exceeds a certain threshold were also removed.

[3] http://workshop2016.iwslt.org/59.php

[4] The conversion table can be found in the following link http://universaldependencies.org/tagset-conversion/en-penn-uposf.html

[5] http://www.nltk.org/

Table 5.3 shows some statistics about the resulting data from the preprocessing step.

	English	Arabic
Sentences	110 549	110 549
Words	1692394	1910968
Unique words	26574	37539

Table 5.3: Statistics about the training corpus

5.4.2 Evaluation of Translation Quality

We have investigated the use of two tagsets and the presence/absence of part-of-speech contexts. This leads to four systems:

1. Reordering with Univ tagset without context.
2. Reordering with PTB tagset without context.
3. Reordering with Univ tagset with context.
4. Reordering with PTB tagset with context.

All our systems have been tested using the Moses PSMT framework (Koehn et al., 2007). We have used a 6-gram language model instead of the default tri-gram model to ensure a better language modeling for the segmented Arabic language. The rest of the parameters are kept unchanged. Our test results have been reported using the Bleu Score Metric (Papineni et al., 2002).

Test set	PSMT-Baseline	PSMT-MSE-Bi
IWSLT2010	17.24	17.33 (+0.09)
IWSLT2011	17.28	17.54 (+0.26)
IWSLT2012	19.30	19.48 (+0.18)
IWSLT2013	18.67	18.64 (-0.03)
IWSLT2014	16.16	16.82 (+0.66)

Table 5.4: Bleu Score results for the PSMT baseline and the MSE-bidirectional reordering model

Table 5.4 shows the Bleu Score results obtained by the Moses baseline with and without its default MSE-bidirectional reordering model (Koehn et al., 2005). The values in parentheses indicate the gain in Bleu score with respect to the PSMT baseline

system. A slight increase in Bleu Score is obtained when the default Moses reordering model was turned on.

Test set	Base-UNIV	Base-PTB
IWSLT2010	17.03 (-0.21)	17.52 (+0.28)
IWSLT2011	17.62 (+0.34)	18.18 (+0.90)
IWSLT2012	19.80 (+0.50)	19.95 (+0.65)
IWSLT2013	18.73 (+0.06)	19.11 (+0.44)
IWSLT2014	17.22 (+1.06)	17.51 (+1.35)

Table 5.5: Bleu Scores using the PTB and the Univ part-of-speech tags without including the context

Table 5.5 shows the reordering results obtained when using the PTB tags and the Univ tags without including the context. The obtained results when using the PTB was slightly better than the one obtained with the Univ tags. The maximum gain in Bleu Score was 1.35 point compared to the Baseline PSMT system.

Test set	CONTEXT-UNIV	CONTEXT-PTB
IWSLT2010	17.72 (+0.48)	17.71 (+0.47)
IWSLT2011	18.31 (+1.03)	18.34 (+1.06)
IWSLT2012	19.90 (+0.6)	20.09 (+0.79)
IWSLT2013	19.13 (+0.46)	19.24 (+0.57)
IWSLT2014	17.61 (+1.45)	17.58 (+1.42)

Table 5.6: Bleu Scores using the PTB tags and the Univ tags when including the context

Table 5.6 shows the reordering results obtained using the PTB tags and the Univ tags when the context is included. The obtained results for the two tagsets were very similar with a maximum increase of about 1.5 in the Bleu Score over the PSMT baseline. These results prove the importance of using the context to enhance the accuracy of the syntactic rules. Indeed, the more specific the rules, the better. Another thing to note is the effect of the POS-tag fineness: we can see that the use of PTB tags yields better results than with Univ tags, especially when no context is used. This suggests that more tag fineness will lead to a more accurate reordering.

5.4.3 Evaluation of Alignment Ambiguity

We have also used the Normalize Crossing Links Score (NCS) (Genzel, 2010) to measure the quality of the different investigated reordering systems. The NCS metric

formula is the following:

$$NCS = \frac{C}{S} \tag{5.2}$$

where C is the number of cross links in the aligned corpus and S is the number of words in the source text of the corpus.

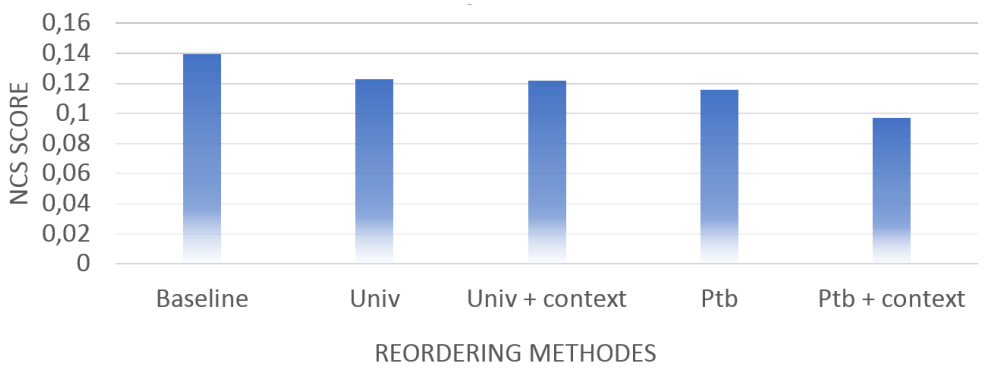


Fig. 5.7: The NCS scores for the different reordering methods

For this formula the smallest the NCS score, the better. An ideal score will be zero, which means that the corpus is completely monotonic [6]. The NCS scores are shown in Fig. 5.7 for the different reordering methods. The results indicate that using the PTB tagset with the contextual information produce less ambiguous alignments, hence better translation results.

5.5 Conclusion

In this chapter, we have presented our first contribution which is a general framework for word preordering in which reordering rules are extracted from a parallel corpus using only word alignments and basic part-of-speech tagging. Rule quality is estimated using the CS metric, which allows the selection of only the best applicable rules. Our proposal has been evaluated in terms of translation quality using the Bleu Score, and the change in alignment ambiguity has been investigated using the NCS metric. We have found out that using the PTB tags yields a more noticeable improvement over the baseline PSMT system. This suggests that the higher the tag fineness the better the effect of the part-of-speech preordering methods. In the next chapter, we will present our second contribution regarding named entity transliteration.

[6] We mean by a monotonic corpus, a corpus in which the alignment does not contain any crossing links.

Chapter 6

Improving Named Entity Transliteration via Deep Learning Methods

6.1 Introduction

Machine Transliteration (Deselaers et al., 2009; Jadidinejad, 2016) is the process of transforming a given word from one alphabet to another while preserving the phonetic and orthographic aspects of the transliterated word. Even though the task of transliteration may appear to be trivial at first glance, it turns out to be a complicated one. The main reason for its difficulty is the absence of some phonetic character correspondences between the source and target languages. In such situations, these kinds of characters will need to be either omitted or even approximated depending on their context of occurrence. For instance, in the task of transliteration from Arabic to English, some Arabic letters such as "ط", "ظ" and "ث" do not have any direct single-letter correspondences in the English alphabet; thus the system will need to transliterate them in a way that best preserves their phonetic aspects.

The accurate transliteration of named entities can be very crucial for many applications. For instance, it can be used to handle Out-Of-Vocabulary (OOV) words in Machine Translation systems (Habash, 2008; Hermjakob et al., 2008), and incorporated to handle proper names transliteration in Cross-lingual Information Retrieval (IR) (Fujii and Ishikawa, 2001; Virga and Khudanpur, 2003). With the emergence of deep learning, sequence-to-sequence models have seen a significant improvement (Cho et al., 2014b). Given the importance of the latter in many applications, several attempts

have been made toward improving them using deep-learning models (Deselaers et al., 2009; Jadidinejad, 2016); yet only little research has been conducted in this direction for the Arabic language.

In this chapter, we first present our methodology for creating a large supervised training data from raw English-Arabic parallel corpora that we name "Arabic Named Entity Transliteration and Classification Dataset"(ANETAC) [1]. Then we present our attention-based encoder-decoder transliteration model. The proposed system is evaluated on the task of transliteration from both English-to-Arabic and Arabic-to-English. The results obtained show a noticeable improvement in the transliteration accuracy over some previous works on these language pairs, which proves the adequacy of our proposal.

The remainder of this chapter is organized as follows. Section 6.2 presents works done on machine transliteration. The methodology we have followed to construct the transliteration corpus is presented in Section 6.3. The details of our proposed transliteration system is then described in section 6.4. In Section 6.5, we present and discuss the tests performed and the results we have obtained. In Section 6.6, we conclude our contribution and highlight some possible future improvements and research directions.

6.2 Research Studies on Machine Transliteration

Machine Transliteration approaches can be classified into two main categories: Direct-mapping methods and Model-based methods. The Direct-mapping methods (Kaur and Singh, 2014) are a simple one-to-one direct and fully-reversible character mapping (bijection) that associates to each letter in the source alphabet a unique letter from the target alphabet, without necessarily respecting its phonetic and orthographic aspects. This kind of methods is not useful at handling named entity transliteration since it does not preserve the phonological structure of words. An example of such a mapping is the scheme proposed by Buckwalter (Habash, 2010) [2] for the task of transliteration between Arabic and Latin. The Model-based methods also known as Phoneme-based Transliteration (Kaur and Singh, 2014), attempt to train a model that can accurately predict the transliteration of a given word while preserving its phonetic and orthographic aspects. A considerable amount of work was developed in

[1] https://github.com/MohamedHadjAmeur/ANETAC

[2] The Arabic transliteration table is provided in the following link http://www.qamus.org/transliteration.htm

the area of machine transliteration. Shao and Nivre (2016) used a Convolutional Neural Network (CNN) for the task of transliteration between English and Chinese. They compared their system with a phrase-based Statistical Machine Translation (SMT) system and found that the accuracy they obtained was slightly below that of the SMT system. They justified these results by the higher-order language model incorporated in the SMT framework. Finch et al. (2016) proposed a method that exploits the agreement between a pair of target-bidirectional Recurrent Neural Networks (RNNs). Their experimental results carried out on various language pairs showed that their proposal performs similar, or even better than the Phrase-based Statistical Machine Translation system (PSMT) on many language pairs. Jadidinejad (2016) proposed a sequence-to-sequence model consisting of a bidirectional encoder and an attention-based decoder. They used their proposal for the task of transliteration between several language pairs. Their experimental results showed that their system outperformed the classical PSMT system. Jiang et al. (2007) proposed a Maximum Entropy Model (MaxEnt) for named entity transliteration from English to Chinese. Their model ranks the transliteration candidates by combining pronunciation similarities and bilingual co-occurrences. They compared their system with some rule-based approaches and reported a slight improvement in the overall accuracy. In the context of the Arabic language, Arbabi et al. (1994) presented a hybrid algorithm for English-to-Arabic transliteration that uses both Feed-forward Neural Networks (FFNs) and Knowledge-based Systems (KBSs). Deselaers et al. (2009) used Deep Belief Networks (DBNs) which consist of multiple Restricted Boltzmann Machine (RBM) layers. They used their system for the Arabic-to-English city names transcription task. The results they obtained showed that the PSMT system clearly outperforms their proposal. Despite that, the authors stated that their system can easily be combined with other state-of-the-art systems to achieve better results. AbdulJaleel and Larkey (2003) presented a Phrase-based Statistical system for English-to-Arabic transliteration using unigram and bigram character-based translation models. Their results showed a higher accuracy when the bigram model was incorporated. Rosca and Breuel (2016) used a neural sequence-to-sequence model for the task of machine transliteration between several languages including Arabic-to-English, in which they achieved 77.1% Word Error Rate (WER) accuracy.

6.3 Building a Transliteration Corpus

As pointed out by Rosca and Breuel (2016), there is a shortage of Arabic machine transliteration corpora (that are freely available). In this section, we present our methodology for the automatic creation of a supervised named entity transliteration corpus (ANETAC) from a raw parallel textual data. We note that the corpus we have constructed will be made freely available to the NLP research community [3].

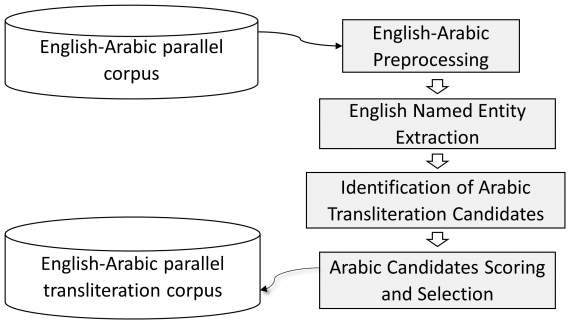


Fig. 6.1: The architecture of the proposed parallel English-Arabic Named entity extraction system.

the extraction system (Fig. 6.1) uses a parallel corpus in order to automatically extract bilingual named entities. The system starts by a preprocessing step in which all the sentences in the English and Arabic languages are normalized and tokenized. Then, the English named entities will be extracted from each sentence of the English-side of the parallel corpus. A set of Arabic transliteration candidates will then be assigned to each extracted English named entity. Finally, the best Arabic transliteration will be selected for each English named entity to form the resulting transliteration corpus. The detail of each step will be provided in the remainder of this section.

6.3.1 Parallel Named Entity Extraction

We recall that our goal is to obtain the Arabic transliteration for each English named entity found in the parallel corpus. To this end, we first need to extract the named entities contained in each English sentence in the parallel corpus. Formally, given a set of English-Arabic parallel sentences $S = \{(e_1, a_1), ..., (e_n, a_n)\}$ we perform an English Named Entity recognition to find all the named entities present in the English-side of the corpus $N = \{n_1, n_2, ..., n_k\}$ where k is the total number of named entities. Just

[3]The Arabic Named Entity Transliteration and Classification Dataset (ANETAC) is available at https://github.com/MohamedHadjAmeur/ANETAC and also at the TALAA website https://lria.usthb.dz/TALAATeam/index.html

like (Rosca and Breuel, 2016), we transform all the named entities containing multiple words to several singleton entities, each one as a standalone named entity. We do this to avoid keeping phrases (having multiple words) in the training corpus since every single English word can always be transliterated without needing any additional information about its preceding words (history). For each English named entity n_i belonging to a sentence e_j, we keep track of its corresponding Arabic sentence a_j. We end up with a list N of pairs (n_i, a_j) denoting that the i^{th} English named entity (singleton word) is associated with the j^{th} Arabic sentence.

6.3.2 Candidates Extraction and Scoring

From the previous step, we end up with a set of pairs (n_i, a_j), where n_i is the English named entity (word) and a_j is the Arabic sentence containing its transliteration. To identify the correct transliterated word in the Arabic sentence a_j, we first remove all the frequent Arabic words from it using a vocabulary containing the n most frequent Arabic words. This ensures that most of the remaining words in the Arabic sentence a_j are rare words (hopefully only named entities). All the remaining words in a_j are considered as transliteration candidates $C(a_j) = \{c_{j1}, c_{j2}, ..., c_{jt}\}$, where c_{ji} denotes the i^{th} candidate word found in the j^{th} Arabic sentence, and t is the total number of Arabic candidates in $C(a_j)$. An external multilingual transliteration tool [4] is used to obtain an approximate Arabic transliteration t_i of each English named entity n_i. For each English named entity n_i having the approximate transliteration t_i and the list of Arabic candidates $C(a_j)$, the score of each candidate is estimated using the following three functions:

1. The number of common characters: this function is used to find the number of shared characters between each candidate c_{ji} and the approximate transliteration t_i.

2. Longest common sequence: this function is used to compute the length of the longest shared sequence of characters between each candidate c_{ji} and the approximate transliteration t_i.

3. Length difference penalty: this function is used to penalize the length difference between each candidate c_{ji} and the approximate transliteration t_i.

The final score for each candidate is then estimated as the average sum of all the considered scoring functions. The candidate having the highest score is then selected if

[4]The details of all the used tools will be provided in the test and experiment section 6.5.

its corresponding final score surpasses a certain confidence threshold. Some examples of the extracted English-Arabic named entities are provided in Table 6.1. The reader should recall that the Arabic language has no letters for the English sound "v", "p" and "g".

Entity class	English	Arabic
PERSON	Villalon	فيلالون (fylAlwn)
LOCATION	Nampa	نامبا (nAmbA)
ORGANIZATION	Soogrim	سوغريم (swgrym)

Table 6.1: Some examples of the extracted English-Arabic named entities

6.4 The Transliteration System

This section describes our proposed model for machine transliteration. Our model uses the Encoder-decoder architecture proposed by Cho et al. (2014b) which has been proven to perform extremely well in many sequence-to-sequence applications (Sutskever et al., 2014).

Given an input sequence $x = \{x_1, x_2, ..., x_d\}$ and an output sequence $y = \{y_1, y_2, ..., y_d\}$, where each x_t and y_t represent the input and output tokens at time step t respectively, and d is the maximum sequence length[5], the basic Encoder-decoder uses two Recurrent Neural Networks (RNNs) that will be trained jointly to generate the target sequence y given the input sequence x. The encoder RNN encodes the input sequence into a hidden representation $h = \{h_1^e, h_2^e, ..., h_k^e\}$, where h_t^e is the encoder hidden state at time step t and k is the dimension of the hidden representation $\mathbb{R}^k$ (Eq. 6.1).

$$h_t^e = \sigma(W_e x_t + U_e h_{t-1}^e + b_e) \tag{6.1}$$

where W_e and U_e are the encoder weight matrices, b_e is the encoder bias vector and σ is the logistic sigmoid activation function. The last hidden state h_k^e will be the summary of all the input sequence x.

The decoder RNN takes the encoder summary h_k^e and a target token o_t at each time step t along with its previous decoder hidden state h_{t-1}^d to estimate the value of its current state h_t^d using Eq. 6.2:

[5]Padding is used to pad all the input sequences into the same length.

$$h_t^d = \sigma(W_d o_t + U_d h_{t-1}^d + C_d h_k^e + b_d) \tag{6.2}$$

where W_d, U_d and C_d are the decoder weight matrices, b_d is the decoder bias vector and σ is a logistic sigmoid activation function. Then the output sequence is predicted (generated) token by token at each time step t from the target vocabulary using a softmax activation function (Eq. 6.3).

$$o_t^d = Softmax(V_d h_t^d) \tag{6.3}$$

where V_d is the target vocabulary weight matrix. Figure 6.2 shows the global architecture of the encoder-decoder model used for the task of transliteration from English-to-Arabic. The first RNN encodes a padded variable-length English named entity into a fixed hidden representation and the second RNN (the decoder) generates a transliteration output from the hidden representation.

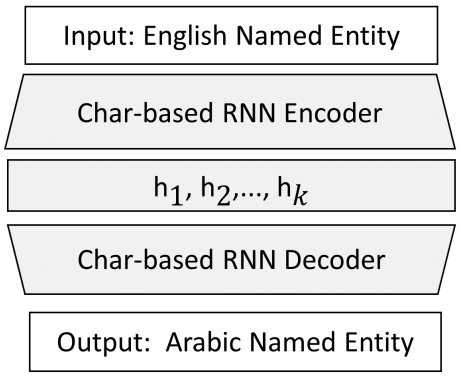


Fig. 6.2: The global architecture of the Encoder-decoder model used for the task of transliteration from English-to-Arabic

6.5 Tests and Analysis of the Results

This section presents an in-depth discussion and the details of the tests we have performed. These tests examine two important aspects. First, we want to investigate the effect of using sequence-to-sequence deep neural network models on the task of transliteration between English and Arabic. The second aspect aims at comparing the performance of our proposal with the Phrase-based SMT system which has been proven to perform very well at sequence-to-sequence prediction tasks. We start this section by presenting some statistics about the data we have used and the preprocessing stage.

Then we provide all the hyperparameters that have been incorporated into our models. Finally, we address the two aforementioned key tests.

6.5.1 Data and Preprocessing

To build our transliteration corpus, we have used a set of four English-Arabic parallel corpora obtained from the "lingfil" website[6]. The statistics of these corpora are provided in Table 6.2.

Corpus	Sentences (in millions)
United Nation	10.6M
Open Subtitles	24.4M
News Commentary	0.2M
IWSLT2016	0.2M
All	35.4M

Table 6.2: Statistics about the used English-Arabic parallel corpora

For the Arabic language, our preprocessing included the removal of diacritic signs, the normalization of Arabic characters and word tokenization using the Python NLTK toolkit [7]. For the English part of the corpus, only a word tokenization is performed using the same NLTK toolkit. English named entities were identified by means of the Stanford Named entity recognition system (Finkel et al., 2005) [8]. A vocabulary containing the $n = 40000$ most frequent words has been used to filter the functional words during our corpus construction phase (Section 6.3). The approximate transliterations were obtained using the polyglot multilingual NLP library (Chen and Skiena, 2016) [9]. Table 6.3 shows the statistics about the count of each named entity class that is found in our constructed transliteration corpus.

The corpus has been divided into training, development, and test data as shown in Table 6.4.

6.5.2 Phrase-based SMT

We have used a Phrase-based Statistical Machine Translation (PSMT) (Brown et al., 1990a) that includes the following components:

[6]The used data along with their descriptions are found on the "lingfil"website http://opus.lingfil.uu.se

[7]http://www.nltk.org/

[8]https://nlp.stanford.edu/software/CRF-NER.shtml

[9]http://polyglot.readthedocs.io

Named entity	Count
Person	61,662
Location	12,679
Organization	5,583
All	79,924

Table 6.3: The count of the Person, Location and Organization named entities present in our constructed transliteration corpus

Sets	Train	Dev	Test
Named entities count	75,898	1004	3013

Table 6.4: Instance counts in the training, development and test datasets of our transliteration corpus

- A phrase translation model with a maximum phrase length of 7 tokens.
- A trigram target language model.

We have used the Moses framework (Koehn et al., 2007) [10] to implement our PSMT system. In our configuration, the distance-based reordering model has been turned off since no character-based target reordering is needed in the transliteration task. The default values have been kept unchanged for all of the remaining Moses hyperparameters.

6.5.3 Encoder-decoder Models

We have investigated the use of both a single GRU encoder and a Bidirectional GRU encoder, along with the presence and absence of the decoder attention mechanism. This led to four different systems:

1. Seq2seq basic: A stranded GRU encoder and decoder.
2. Bi-seq2seq: A Bi-directional encoder with a stranded GRU decoder.
3. Att-seq2seq: A stranded GRU encoder and attention-based decoder.
4. Bi-Att-seq2seq: A Bi-directional encoder and an attention-based decoder.

Hyperparameters: To choose an adequate number of neurons in our encoder and decoder GRU cells, we have investigated the effect of changing the number of neurons by measuring the error rates we have obtained for each neural configuration and all the

[10] http://www.statmt.org/moses/

four encoder-decoder systems. This variation is shown for the Bi-Att-seq2seq model (Fig. 6.3).

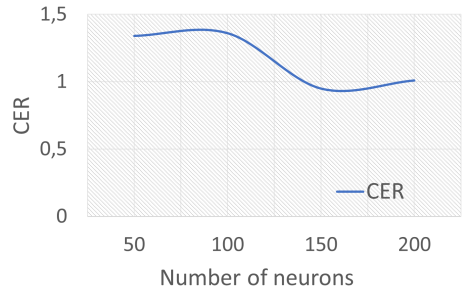


Fig. 6.3: Character Error Rates for the English-to-Arabic transliteration when varying the encoder-decoder hidden sizes

Figure 6.3 shows the effect of varying the network size (the number of neurons in each layer of the encoder-decoder architecture) on the Character Error Rate (CER) for the Bi-Att-seq2seq model. The best performance in terms of CER has been achieved when the number of neurons was fixed to 150. The training perplexity for this same configuration is shown in Fig. 6.4.

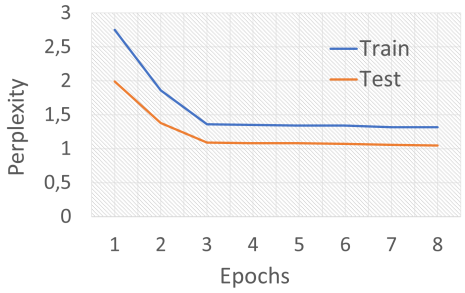


Fig. 6.4: Perplexity variation on the train and test data for the attention bi-directional encoder-decoder model for the English-to-Arabic transliteration task

The train and test perplexity values (Fig. 6.4) decrease with the number of epochs until convergence is reached starting from the fourth epoch. In a similar way, we have fixed all the remaining hyperparameters experimentally. In all our sequence-to-sequence models, an embedding dimension of $\mathbb{R}^{10}$ has been used. The maximal sequence length has been set to 50 characters. A mini-batch of size 128 has been incorporated. The training has been done by means of Stochastic Gradient Descent (SGD) with the Adam optimization function (Kingma and Ba, 2014). Our models have been trained using the Nvidia $GTX1070$ GPU with $8GB$ of $DDR5$ memory. All the encoder-decoder models have been trained using the OpenNMT toolkit (Klein et al., 2017) [11].

[11]http://opennmt.net/

6.5.4 Results

The Word Error Rates (WERs) and Character Error Rates (CERs) obtained by all the investigated sequence-to-sequence models are provided in Table 6.5a and Table 6.5b for the tasks of transliteration from English-to-Arabic and Arabic-to-English respectively.

Models	WER	CER
Seq2seq basic	16.91	3.44
Bi-seq2seq	16.11	3.07
Att-seq2seq	8.81	1.51
Bi-Att-seq2seq	**5.40**	**0.95**
Moses PSMT	6.96	1.03

(a) English-to-Arabic

Models	WER	CER
Seq2seq basic	70,02	19.25
Bi-seq2seq	74,53	21.49
Att-seq2seq	66,49	17.10
Bi-Att-seq2seq	**65,16**	**16.35**
Moses PSMT	68,57	16.61

(b) Arabic-to-English

Table 6.5: Transliteration results

The reported error rates for the English-to-Arabic direction (Table 6.5a) show that the incorporation of the decoder attention mechanism leads to a significant improvement. Indeed, around 10% WER reduction has been achieved over the basic encoder-decoder when the decoder attention mechanism was incorporated. The PSMT model and the Bi-Att-seq2seq models gave the best results of 6.96% and 5.40% in terms of WER respectively, with a slightly better performance in favor of the attention model. The results for Arabic-to-English direction (Table 6.5b) were not as good. This is due to the higher ambiguity that is present in this direction, which has also been pointed out by Rosca and Breuel (2016) and Deselaers et al. (2009). As for the Arabic-to-English test results, the Bi-Att-seq2seq gave the best performance of 65,16% WER followed by the PSMT with a 68.57% WER.

Table 6.6 provides a comparison between our best models, the Bi-Att-seq2seq and the Moses PSMT systems, and some other research works on machine transliteration from Arabic-to-English [12].

The results show that our Bi-Att-seq2seq model gives the lowest error rate, which demonstrates the efficiency of our proposal for the transliteration task [13].

[12]For the other direction (English-to-Arabic) we have not found any research work that uses our same metrics.

[13]We note that the systems we have compared have not been tested on the same test set, thus conclusions should be taken with some caution.

Metrics	CER	WER
Rosca and Breuel (2016)	22.5	77.1
Deselaers et al. (2009)	20,1	-
Our Bi-Att-seq2seq	**16,35**	**65.1**6
Our Moses PSMT	16,61	68.57

Table 6.6: A comparison of our proposed approaches with some other Arabic-to-English deep learning-based transliteration systems

6.5.5 Error Analysis

In this section, we will provide a quick glance at the errors made by our best Bi-Att-seq2seq model in the task of machine transliteration.

Input	Reference	System output
Brandes	براندیس (brAndys)	براندس (brAnds)
Mayhawk	مايهوك (mAyhwk)	مايهاوك (mAyhAwk)

Table 6.7: Example of some errors made in English-to-Arabic transliteration

For the English-to-Arabic direction most of the errors made by the transliteration system are due to conflicting vowels. Some examples that illustrates this are given in Table 6.7. Indeed, this is to be expected since there are no unified ways of transliterating named entities between Arabic and English. Instead, many possible transliterations can be made for the same named entity. For instance, a "ي" can either be considered or omitted in the transliteration of "Brandes" as shown in Table 6.7.

Concerning the Arabic-to-English direction the reported results were much worse then the ones reported for the English-to-Arabic one. This is probably due to the absence of vowels (diacritic signs) on the Arabic side. For instance, words such as "حسين" and "حسن" are transliterated to "**Houc**ine" and "**Hass**an", respectively, (instead of "Hacine"and "Houssn", for instance), even though they share the same prefix "حس". Another problem is the absence of some Arabic character sounds in the English language which often leads to transliteration errors, such as the sounds of the Arabic letters "ط", "ظ" and "ث".

6.6 Conclusion

In this chapter, we have presented our second contribution namely a bidirectional attention-based encoder-decoder model for the task of machine transliteration between

Arabic and English. Our transliteration system has been compared to several sequence-to-sequence models, and its results prove the efficiency of our proposal. This study can be further developed in various directions. One direction is to consider the case of transliteration between Arabic and other languages besides English. Another interesting future direction is to integrate this model into an English-to-Arabic machine translation system to address the problem of named entity transliteration.

In the next chapter, we will present our main contribution regarding n-best list re-ranking in the context of an English-to-Arabic Neural Machine Translation system.

Chapter 7

Improving NMT via N-best List Re-ranking

7.1 Introduction

Even though a great deal of improvement has been achieved in the field of MT, the current translation results are still not perfect. This is due to many factors, one of them being the complexity (or difficulty) of the decoding task. Indeed, the decoder is not always capable of selecting the one best translation from the space of all possible translations (Freitag and Al-Onaizan, 2017). To this end, various solutions have been proposed. One of the most common methods are those which are based on a re-ranking process. Re-ranking can be seen as a two-pass procedure (Duh et al., 2010). First, the decoder is used to generate a list containing the top n best translations known as "the n-best list" [1]. Then a re-ranking methodology is used to select the best translation from the n-best list by re-ranking these translations according to a rich set of features.

Re-ranking the decoder's n-best lists is an effective approach to improve the overall quality of the machine translation system for three reasons:

1. It allows the incorporation of various additional features to select the best translation from the n-best list.

2. The search space becomes significantly smaller given that it will be limited to only the translations found in the n-best list.

3. Re-ranking is a standalone model that can easily be incorporated into any other translation system.

[1] Each translation in the n-best list is known as a translation candidate or a translation hypothesis.

In this chapter, we present our third and main contribution which is a method for n-best list re-ranking in the context on an NMT system that uses a set of sophisticated features. The features set that we propose covers the lexical, syntactic and semantic aspects of the translation candidates (the n-best list candidates) and can be grouped into five classes: (1) Translation-based Features: these features are related to a translation model and can be helpful for promoting translation adequacy; (2) Fluency Features: these features are used to promote syntactic fluency by incorporating language models; (3) Length-based Features: these features are used to promote the n-best list candidates according to the likelihood of their lengths; (4) N-best list Features: these features are extracted directly from the n-best list and are used to promote the most likely candidates in it; and (5) Embedding Features: these are based on bilingual word embeddings and are used to cover the semantic aspect of the translation candidates. For the problem of feature weights optimization, we present a methodology that is fairly similar to the one presented by Farzi and Faili (2015). It uses a Quantum-behaved Particle Swarm Optimization (QPSO) which guarantees the global convergence of the optimization process. Though our overall re-ranking framework is closely related to the one proposed by Farzi and Faili (2015), we diverge from them in various aspects:

- This study proposes several new re-ranking features, namely:
  - Position-based feature (section 7.5.1.1).
  - N-best alignment feature (section 7.5.1.2).
  - Word-to-word alignment feature (section 7.5.3.3).
  - Estimated length feature (section 7.5.2.1).
  - Global semantic similarity (section 7.5.5.1).
  - Alignment-based semantic similarity (section 7.5.5.2).
- All the features proposed in this study are language-independent, thus they can be used for any other languages.
- A new class of embedding-based features is proposed to take into account the semantic aspects of the translation candidates via bilingual word embeddings.
- Unlike Farzi and Faili (2015), our proposed objective function is based on the corpus-level, not the sentence-level, BLEU Score. Thus, feature weights are updated only if an improvement is achieved on the whole development corpus (section 7.4.2).

- An in-depth investigation about the features and classes impact is performed along with a discussion about their individual and combined effectiveness.
- This study focuses on two MT directions: English-to-Arabic and Arabic-to-English, in which very few n-best list re-ranking studies are performed.

The remainder of this chapter is organized as follows. Section 7.2 presents the research studies that have been made on the problem of n-best list Re-ranking. The necessary background information needed to understand this chapter is provided in Section 7.3. Section 7.4 explains the overall system design. The details concerning all the incorporated reordering features are provided in Section 7.5. The evaluation methodology and all the experimentations are provided in Section 7.6. Section 7.7 gives a brief discussion about the obtained results. Finally, Section 7.8 gives a conclusion as well as a listing of some possible research directions.

7.2 Researches Studies on n-best list Re-ranking

Many interesting ideas have been proposed to address the problem of n-best list re-ranking in MT. In the following, we will try to categorize the most important methods that have been proposed according to the way in which they address the re-ranking problem.

7.2.1 Re-ranking by Optimizing the Decoder Feature Weights

One research direction investigated the possibility of replacing the linear decoder scoring method with a more efficient one. Arun and Koehn (2007) investigated discriminative training of a phrase-based SMT system using millions of features for the task of n-best list re-ranking. Their model parameters were optimized using two online learning algorithms, the structured perceptron and Margin-Infused Relaxed Algorithm (cf. (Watanabe et al., 2007)). Their experiments on Czech-English translation showed that the two methods produce very similar results while the perceptron has a more rapid convergence. Duh and Kirchhoff (2008) presented a boosting algorithm which they called BoostedMERT; it uses Minimum Error Rate Training (MERT) (Och, 2003) to boost the BLEU score on the n-best list re-ranking task. The results they reported on the IWSLT 2007 Arabic-to-English translation task showed an absolute improvement of 0.8 BLEU points over the baseline PSMT system. In their later work, Duh et al. (2010) addressed the problem of n-best list re-ranking as a multi-task learning problem

in which each n-best list is taken as a distinct task. First, they used a meta-algorithm that discovers the common feature representations across the n-best lists via multi-task learning. Then they used a conventional re-ranker to reorder the n-best list. They reported a 0.5 improvement in the overall BLEU score on the English-to-Japanese translation task. Sokolov et al. (2012) proposed an approach that uses a non-linear scoring function instead of the conventional PSMT linear scoring function via a Boosting algorithm. The experiments they carried out on the WMT'10, WMT'11 and WMT'12 test sets resulted in a performance boost of about 0.4 BLEU points.

7.2.2 Re-ranking by Including Additional Features

The simplest way to improve the translation output is to include additional language models and use them to select the best translation from the n-best decoding list. Following this research direction, Kirchhoff and Yang (2005) investigated the effect of including additional language models on n-best list re-scoring. They used a 4-gram word-based language model with modified Kneser-Ney smoothing (Kneser and Ney, 1995) and interpolation and a factored feature-based trigram language model. Their experiment results reported on the ACL05 Shared MT task for four language pairs (translation from Finnish, German, Spanish and French into English) showed that using additional language models did not result in a significant increase in the overall PSMT performance. Carter and Monz (2010) applied a large-scale discriminative language model to re-rank the n-best list translations generated by an SMT system. They reported an improvement of up to 0.4 BLEU points on the NIST Arabic-to-English translation benchmarks. Luong and Popescu-Belis (2016) proposed a method to better handle the translation of pronouns between English and French. They used a Pronoun-aware Language Model (PLM) which encodes the likelihood of generating a target pronoun given the gender and number of the nouns preceding it. They combined the PSMT decoder score and their PLM model score to re-rank the translation candidates and reported a 5% relative accuracy improvement in pronouns prediction over the PSMT baseline.

Various researchers have used several linguistic features to boost the performance of a translation system as a standalone post-processing phase. Following this direction, Och et al. (2004) presented a method for n-best list re-ranking that uses a large number of features with different levels of syntactic representation. The features were combined using the log-linear model and their weights were optimized directly against the BLEU score using MERT on held-out data. They reported a significant improvement of 1.3% BLEU score on the task of Chinese-to-English translation. Hasan et al. (2007)

investigated the usefulness of increasing the size of the n-best list produced by a statistical MT system. They showed that although it is possible to generate many distinct translation candidates, starting from a certain value of n, the increase in the overall system performance is very minimal. They showed that going above n = 100 does not yield a significant improvement while noticeably increasing the decoding time. Specia et al. (2008) used Word Sense Disambiguation features to re-rank the n-best list translations generated by a statistical MT system. Experiments with English-to-Portuguese translation showed a significant improvement that varied between 1.5 and 2.5 absolute BLEU points. Farzi and Faili (2015) used a set of non-syntactical features to re-rank the n-best translation candidates generated by a PSMT system. They investigated several feature weights optimization algorithms such as Particle Swarm Optimization (PSO), Quantum-behaved Particle Swarm Optimization (QPSO), Genetic Algorithms (GA), Perceptron and Averaged Perceptron. They reported an improvement of 1.09 and 1.73 points in BLEU score for English-to-Persian and German-to-English, respectively. They concluded that QPSO was better suited for weight optimization than the other investigated alternatives. Tong et al. (2016) investigated the use of new semantic and syntactic features in the re-ranking framework. The representations they used are basically sentence-embeddings that are learned using the recursive auto-encoder. They evaluated their proposal on the WMT2015 French-to-English translation data and reported a noticeable improvement of about 1.23 BLEU points over the baseline SMT system.

7.2.3 Re-ranking via System Hybridization

Some researchers have tried to re-rank the n-best list via the hybridization of different types of MT systems. In this spirit, Xiao et al. (2013) proposed an ensemble learning-based approach in which they first generate an ensemble of weak translation systems from a single SMT engine, and then they learn a strong translation system from that ensemble. One of their proposed system combination methods is a sentence-level one in which the best translation is selected from the union of all the n-best lists from the weak MT systems. They tested their approach on the NIST Chinese-to-English translation task and reported a significant improvement for all three state-of-the-art statistical MT systems that they tested (PSMT, hierarchical phrase-based system, and syntax-based system). Neubig et al. (2015) described the results of applying neural MT (NMT) re-ranking to a baseline syntax-based MT system. Their tests on the WAT2015 translation task between English, Japanese and Chinese yielded a noticeable improvement in the overall BLEU score over the baseline system. Stahlberg

et al. (2016) investigated the use of hierarchical PSMT lattices in an end-to-end NMT system. They evaluated their proposed system on the English-to-German and English-to-French WMT 2014 news tests and found that the hybridization yielded a noticeable improvement over the individual models. Zhang et al. (2017) proposed a method that uses an existing PSMT model to compute the phrase-based decoding cost for a given NMT output, which is then used to re-rank the n-best list generated by their NMT system. They tested their proposal on several language pairs: English-to-Chinese, English-to-Japanese, English-to-German, and English-to-French. They reported some noticeable improvements over their NMT baseline.

7.2.4 Re-ranking by Improving the Decoder Search Strategy

A few studies have attempted to address the main drawback of the beam search decoder which is the lack of diversity of its generated candidates. Indeed, n-best lists generated via beam search are generally very similar and differ only slightly from each other. To address this problem, a branch of research has focused on diversifying the decoder n-best list candidates. Vijayakumar et al. (2016) proposed a Diverse Beam Search decoder for NMT as an attempt to generate more diverse n-best list candidates than those that can be obtained from a classical beam search decoder. Their proposal optimizes a diversity-augmented objective function that divides the beam search space into smaller groups and promotes the diversity between them. Their test results on an English-German news test dataset showed a gain of up to 0.6 points in BLEU score over the classical beam search decoder. Li and Jurafsky (2016) proposed a diversification heuristic that prevents the beam search decoder from producing highly similar candidates, thus implicitly increasing the diversity of the n-best list produced. Their test results on WMT German-to-English and French-to-English translation tasks showed a consistent performance boost over their NMT baseline. Some other studies attempted to recombine the hypotheses generated by a beam search decoder to create new ones. Zhang et al. (2018) introduced a recombination method for NMT decoding based on the equivalence of partial hypotheses. They used an n-gram suffix-based heuristic approximation to determine partially equivalent hypothesis in the beam search space. They tested their proposal on two translation tasks: NIST Chinese-English and WMT English-German and reported a very small gain in BLEU score on both tasks. Tromble et al. (2008) presented a Minimum Bayes-Risk (MBR) decoding over a translation lattice that can encode a large number of translation candidates in a compact way. Their tests on Arabic-to-English, Chinese-to-English and English-to-

Chinese translation tasks showed moderate gains in translation performance over classical N-best MBR decoding.

7.3 Background

This section introduces some concepts that will be important for a better understanding of this chapter.

7.3.1 Beam Search Decoder

Beam Search (Koehn, 2009; Russell and Norvig, 2016) is a heuristic search algorithm which can be seen as an improvement to the classical Greedy Search (GS) algorithm. Unlike the GS algorithm which keeps track of only the best next step as the solution sequence is constructed, the beam search algorithm expands all possible next steps (tokens) and chooses the B most likely ones resulting in B best partial solutions; where B is the unique parameter of the beam search algorithm known as the "beam width".

In the case of NMT, given a source sentence X, the beam search algorithm can be used to generate the B best target translations of X. At each time step t, the beam search decoder expands each of the B partial candidates with all the possible words $y_t \in V$, where V is the target vocabulary. Each newly constructed partial candidate $C_t = \{y_1, y_2, ..., y_t\}$ will be scored using Eq. (7.1):

$$P(C_t) = P(y_1, y_2, ..., y_t|X) = P(y_1, .., y_{t-1}|X) * P(y_t|X, y_1, .., y_{t-1}) \tag{7.1}$$

where $P(y_t|X, y_1, .., y_{t-1})$ is the probability of generating the target word y_t at time step t by the NMT decoder. The B partial solutions with the most likely probabilities will be selected at each time step t. This same process will be repeated until the end of the sequence is reached.

7.3.2 Minimum Bayes Risk

Minimum Bayes Risk (MBR) is a decoding method that finds the candidate with the least expected loss (González-Rubio et al., 2011; Kumar and Byrne, 2004; Shu and Nakayama, 2017),[2]. The Bayes risk of a given candidate y can be estimated using Eq. (7.2):

$$R(y) = \sum_{y' \in E} \triangle(y, y') P(y'|x) \tag{7.2}$$

[2]We follow the MBR re-ranking method given by González-Rubio et al. (2011).

where E refers to the evidence space,[3] $P(y'|x)$ is the probability of generating the candidate y' as the translation of the source sentence x by the NMT decoder, and $\triangle(y, y')$ is the level of discrepancy between the two candidates y and y' which can be estimated using Eq. (7.3):

$$\triangle(y, y') = 1 - SBLEU(y, y') \tag{7.3}$$

The function $SBLEU$ refers to the third smoothed sentence-level similarity metric proposed by Chen and Cherry (2014). The intuition behind this method is to select the candidate sharing the highest similarity with the candidates of the evidence space E (González-Rubio et al., 2011).

7.4 System Design

The global architecture of our system is presented in Figure 7.1. Its functioning mechanism involves two main consecutive steps: (i) a re-ranking model is built; (ii) this re-ranking model is used to re-rank the translation candidates in the n-best list generated by the NMT decoder.

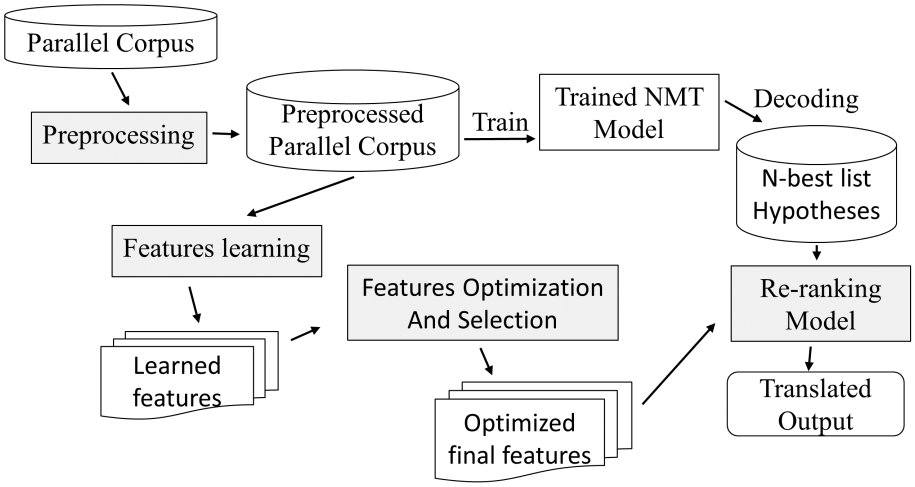


Fig. 7.1: Global architecture of the proposed re-ranking system

As shown in Figure 7.1, we first start by applying a preprocessing step to both the source and target sentences that are found in the parallel corpus. Then, a re-ranking model is built by using a set of predefined sentence-level linguistic features which are

[3]In this study the list of n-best list candidates is considered as the evidence space.

directly learned and optimized from the preprocessed parallel corpus. Finally, the re-ranking model is used to reorder the candidates in the n-best list generated by the NMT baseline decoder.

For the remainder of this section, we start by explaining in more detail the functionality of our re-ranking system. Then, we explain the feature weights optimization task along with the swarm-based algorithm that we incorporate to solve it.

7.4.1 The Re-Ranking Process

Our final goal is to re-rank the candidates in the n-best list in a better way that allows us to pick the best translation among them. To do so a set of sentence-level features is defined; each feature will be used to assign a score in the interval [0, 1] to each individual candidate (translation) in the n-best list. We denote the n-best translation list of a given source sentence x by $H(x) = (h_1, h_2, ..., h_n)$, where n is the size of the n-best list and h_i is the i^{th} candidate translation of sentence x. We denote the feature set by $F = \{f_1, f_2, ..., f_k\}$, where k is the number of features and each feature f_i is a function that takes as argument a translation candidate from the n-best list and assigns a probability to it. For example, $f_1(h_{12}) = 0.9$ means that the score of the 12^{th} candidate in the n-best list $H(x)$ according to the first feature f_1 is equal to 0.9.

By scoring each translation candidate using all the features defined in F, each candidate from the n-best list $H(x)$ will end up with multiple scores, one score per feature. An example is given in Table 7.1.

$H(x)$	f_1	f_2	f_3	...	f_k
h_1	0.38	0.11	0.25	...	0.14
h_2	0.17	0.28	0.98	...	0.65
...	...	...	...	...	...
h_n	0.49	0.32	0.78	...	0.55

Table 7.1: An example showing the process of scoring the candidates in the n-best list via a set of predefined features

To re-rank the n-best list candidates we need to assign a single score to each candidate and re-rank them according to it. To this end, we need to define a way of combining all the individual feature scores assigned to a given candidate (Table 7.1) into a single score. One possible way to do that is to use a weighted linear combination method as in Eq. (7.4):

$$S(h_i) = w_1 f_1(h_i) + w_2 f_2(h_i) + ... + w_k f_k(h_i) \tag{7.4}$$

$$with \qquad \sum_{j=1}^{k} w_j = 1 \tag{7.5}$$

where $w_j \in W$ is the weight (coefficient) of the j^{th} feature of F, h_i is the i^{th} candidate in the n-best list $H(x)$, and the total sum of all the feature weights w_j is always equal to one (Eq. (7.5)). Then, the best translation of the source sentence x is selected from the n-best list $H(x)$ by simply picking the translation candidate that has the highest score as shown in Eq. (7.6):

$$best_{trans}(x|\ W = \{w_1, .., w_k\}) = argmax_{(h_i \in H(x))} S(h_i) \tag{7.6}$$

Using Eq. (7.6) we can determine the best translation candidate (the one with the highest score) from the n-best list of the source sentence x when the weight of each feature is provided.

7.4.2 The Task of Feature Weights Optimization

Our goal is to find the optimal feature weight vector $W = \{w_1, w_2.., w_k\}$ that gives the best re-ranking results. To this end, we use a development corpus which contains a set of source sentences $X = \{x_1, x_2, ..., x_t\}$, where each sentence $x_i \in X$ is associated with its n-best translation list $H(x_i) = (h_1, h_2, ..., h_n)$ generated by the NMT decoder. We define our objective function as a corpus-based BLEU score that considers the firs best translation (using Eq. (7.6)) from each n-best list $H(x_i)$. Changing the feature weights will automatically lead to a change in the order of the n-best list candidates which in turn will lead to a change in the corpus-based BLEU score of the development corpus as shown in Figure 7.2.

As shown in Figure 7.2, if we do not consider any feature by setting all the weights to zero (the top-right of the figure), then the original decoder ranking remains the same. Modifying the feature weights (the second and the third examples in the same figure) changes the ranks of the n-best list translation candidates, thus changing the overall corpus-based BLEU score. This highlights the search space which consists of all the possible combinations of feature weight values. Each one of them can lead to a different BLEU score on the development corpus.

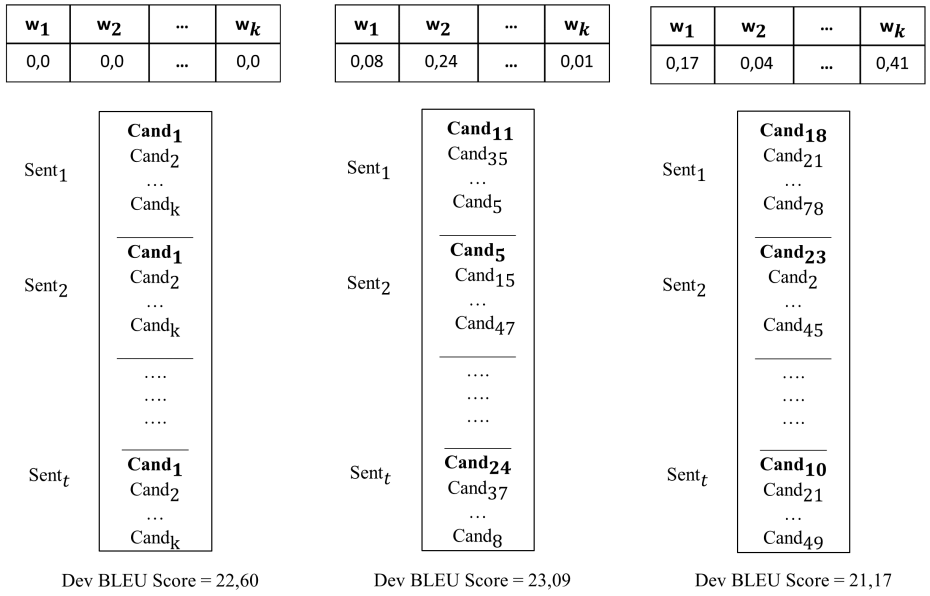


Fig. 7.2: The functioning mechanism of the weight-optimization process in the re-ranking system

7.4.3 Quantum-behaved Particle Swarm Optimization

Many methods have been proposed to tackle the feature weights optimization problem, such as the Perceptron algorithm (Carter and Monz, 2011), Particle Swarm Optimization and Quantum-behaved Particle Swarm Optimization (Farzi and Faili, 2015). The later work of Farzi and Faili (2015) reported that QPSO gives better re-ranking results in comparison to the other algorithms mentioned. Motivated by the work of Farzi and Faili (2015), we also make use of the QPSO algorithm.

QPSO (Sun et al., 2004) uses a swarm of m particles which attempt to find the optimal solution in a d-dimensional space. Each particle i is characterized by its position vector $X_i(t) = \{x_{i1}(t), x_{i2}(t), ..., x_{id}(t)\}$ and its best previous position *pbest* "Personal best". The information about the global best position $gbest(t)$ achieved by the m particles at each time t is also kept. Each particle moves according to the following equation:

$$X_i(t+1) = \begin{cases} P_i + \beta \cdot (mbest - X_i(t)) \cdot ln(\frac{1}{u}),\ if\ h > 0.5 \\ P_i - \beta \cdot (mbest - X_i(t)) \cdot\ ln(\frac{1}{u}),\ otherwise \end{cases} \tag{7.7}$$

$$P_i = \varphi\ pbest_i + (\varphi - 1)\ gbest_i \tag{7.8}$$

$$mbest = \sum_{i=1}^{m} \frac{pbest_i}{m} \tag{7.9}$$

where P_i is the "Inclination Point" which combines the personal best *pbest* and the global best *gbest* as shown in Eq. (7.8), *mbest* is the "Mean Best Position"; the center of gravity of all particles' best positions, φ, h and u are random numbers uniformly distributed on [0, 1], and β known as the "Contraction Expansion Coefficient" is the unique parameter of the QPSO algorithm which is used to control the convergence speed of the algorithm.[4]

The high-level steps that explain the functioning mechanism of the QPSO-based weights optimization algorithm that we have incorporated in our re-ranking system are presented in Algorithm 3. The steps that we used are similar to the ones proposed by Farzi and Faili (2015). We take as input the development corpus and a set of features. Then we find the optimal set of feature weights that gives the highest corpus-based BLEU score on the development set (as explained earlier in Figure 7.2).

Algorithm 3: Feature weights optimization algorithm

Input : A parallel development corpus
A set of features $F = \{F_1, F_2, ..., F_k\}$
Output : The optimal feature weights
Pseudo Algorithm:
begin
- For each particle i in the population, randomly initialize its position vector W_i (the feature weights vector) and set its personal best $pbest_i$ to W_i.
- **while** *no termination condition is met* **do**
  - Estimate the corpus-based BLEU score (objective function) based on the position vector W_i of each particle i in the population (as shown in Section 7.4.2).
  - Update the personal best $pbest_i$ of each particle i, then, update the global best *gbest* of the whole population.
  - For each particle, update its position vector W_i (Eq. 7.7).
- **end**
- Return the optimal feature weights of the global best *gbest*.

end

The algorithm starts by randomly initializing the position vectors that contain the feature weights for each particle. Then it updates the personal and global bests

[4]The value of the contraction-expansion coefficient parameter is generally set to 0.75 as recommended by Sun et al. (2012).

according to the BLEU score achieved by each particle in the swarm. Then the particles move toward the optimal solution using Eq. (7.7) and Eq. (7.8). This process is repeated until one of the following termination conditions is met:

1. Reaching the maximum number of iterations.
2. Reaching the maximum execution time limit.
3. Meeting the early stopping condition, which applies if no improvement is achieved for a certain number of iterations.

7.5 Proposed Features

As mentioned earlier, we do not want to make the NMT system dependent on any language-specific tool. As such, we have tried to avoid using any external NLP tools (such as part-of-speech taggers, named entity recognizers, parsers, etc.) and we have used only the features that can be automatically extracted from the bilingual corpus. We have proposed five categories (classes) of features, each of which contains a set of sophisticated features which serve different purposes.

For the remainder of this section, the following notations will be respected. The source English sentence is denoted by x with d being its length. The n-best translation list of x is denoted by $H(x) = (h_1, h_2, ..., h_n)$ where h_i is the i^{th} Arabic translation candidate.

To illustrate the functioning mechanism of each feature we will consider the English source sentence $x =$ "something stiffened inside me", which together with the notation will be used throughout this section.

Let us assume that the NMT decoder generated an n-best list that contains three candidates $H(x) = (h_1, h_2, h_3)$ as follows:

- $h_1 =$ "هناك شيء ب داخل ي"
- $h_2 =$ "هناك شيء تشدد ب داخل ي"
- $h_3 =$ "هناك شيء ما في ال داخل"

We also consider that the NMT decoder alignments for those three candidates h_1, h_2 and h_3 are those provided in Figure 7.3. We note that the Arabic language candidates are written from left-to-right only to match the English language writing direction as indicated by the numbering of their words (Figure 7.3).

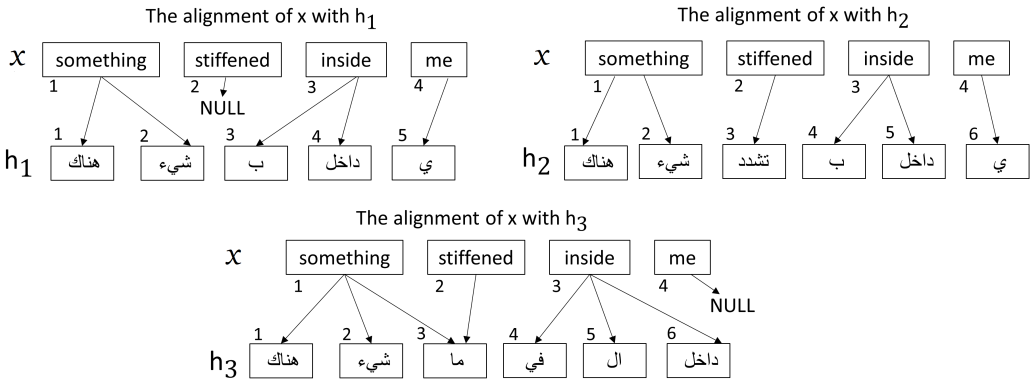


Fig. 7.3: The NMT decoder source-to-candidate alignments for all the n-best list candidates

We denote the decoder alignment concerning the source sentence x and its candidate translation h by $Alig(x,h) = \{A_s(h[1]), A_s(h[2]), ..., A_s(h[t])\}$, where t is the length of the candidate h and $A_s(h[j])$ is a list of all word positions of x that are aligned to the word $h[j]$ found at position j of h. For example, $A_s(h_3[3]) = A_s(\text{ما}) = [1,2]$ because the first and second words of the source sentence x are both aligned to the third word of the candidate h_3. We note that we have obtained these word-to-word alignments from the NMT decoder source-to-candidate attention weights. For each source word from the source sentence we select the target word that has the highest attention weight as its alignment. An example demonstrating how word-to-word alignments are extracted from the NMT decoder attention weights is provided in Fig. 7.4.

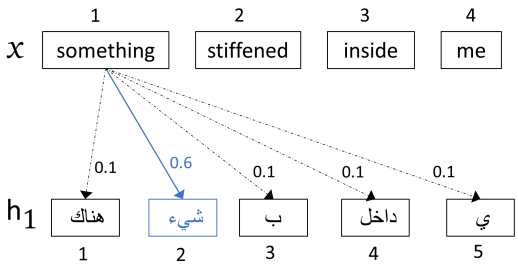


Fig. 7.4: An example showing the process of extracting word-to-word alignments from the NMT decoder attention weights

The first source word “something" is aligned with the target word “شيء" given that it has the highest attention weight 0.6. This same principle is applied to the remaining words of x to obtain their word-to-word alignments.

7.5.1 N-best list Features

This class of features relies on the knowledge that can be extracted from the n-best list to promote the most promising candidates in it. It includes two features: Position-based Score and N-best Alignment Score.

7.5.1.1 Position-based Score

This feature is a generalization of the one proposed by Farzi and Faili (2015), where a feature was proposed that promotes the candidates having the most probable word distribution in the n-best list. First, they estimate $P(h,i)$ the probability of appearance of the i^{th} word of candidate h in the i^{th} position of h, as in Eq. (7.10):

$$P(h,i) = \frac{\sum_{h' \in H(x)} E(h[i], h'[i])}{n} \tag{7.10}$$

where $E(h[i], h'[i])$ is the equality function that returns 1 if $h[i] = h'[i]$ and 0 otherwise. Then the total probability of candidate h is estimated as the product of the probability at each position i in h using Eq. (7.11):

$$P(h) = \prod_{i=1}^{t} P(h,i) \tag{7.11}$$

With respect to this feature we propose a generalization of that defined in Farzi and Faili (2015), in that we consider n-grams (up to four grams) instead of unigrams. Thus, we propose the new equation (7.12) instead of Eq. (7.10):

$$P(h,i) = \sum_{k=1}^{4} \frac{1}{k} \frac{\sum_{h' \in H(x)} E(h[i:i+k], h'[i:i+k])}{n} \tag{7.12}$$

Equation (7.12) considers the case of n-grams with $k \in \{1,2,3,4\}$ (for $k = 1$, this equation is exactly the same as Eq. (7.10)). To estimate the final probability of candidate h (the score of candidate h), we make use of the same equation (Eq. 7.11) proposed by Farzi and Faili (2015).

Example 1. Following the example given at the beginning of Section 7.5, to estimate the probability of the first candidate h_1 according to Farzi and Faili (2015) (Eq. (7.10)), we first estimate $P(h_1,i)$, the probability of appearance of the i^{th} word of h_1 in the i^{th} position of h_1, as follows:
$P(h_1,1) = \text{P}(\text{هناك}) = 3/3 = 1$

$P(h_1,2) = \text{P}(\text{شيء}) = 3/3 = 1$
$P(h_1,3) = \text{P}(\text{ب}) = 1/3$
$P(h_1,4) = \text{P}(\text{داخل}) = 1/3$
$P(h_1,5) = \text{P}(\text{ي}) = 1/3$

Then, the final probability of h_1 is estimated as the product of these quantities:
$P(h_1) = P(h_1,1) * P(h_1,2) * ... * P(h_1,5) = 1/27$

The only difference in our proposed feature is that we use n-grams instead of unigrams, thus we estimate the probability of finding the i^{th} n-gram of h_1 in the i^{th} position of h_1 as follows:
$P(h_1,1) = \frac{1}{4}\ \text{P}(\text{هناك}) + \frac{1}{4}\ \text{P}(\text{هناك شيء}) + \frac{1}{4}\ \text{P}(\text{هناك شيء ب}) + \frac{1}{4}\ \text{P}(\text{هناك شيء ب داخل})$
$P(h_1,1) = \frac{1}{4} * \frac{3}{3} + \frac{1}{4} * \frac{3}{3} + \frac{1}{4} * \frac{1}{3} + \frac{1}{4} * \frac{1}{3} = 2/3$
$P(h_1,2) = ...$
...
$P(h_1,5) = ...$

Then, the final probability of h_1 is estimated as we have done previously:
$S(h_1) = P(h_1) = P(h_1,1) * P(h_1,2) * ... * P(h_1,5)$

As illustrated in this example, the generalization that we have proposed in this feature allows us to make decisions based on the likelihood of the appearance of entire segments (n-grams) on a given position instead of solely relying on individual words. We believe that this addition increases the usefulness of this feature as it takes into consideration the context of each word (the words that surround it) at each given position of the translation candidate.

7.5.1.2 N-best Alignment Score

The second feature that we propose in this class promotes the candidates that have the most probable alignments in the n-best list. First, for each word $h[j]$ found at position j in h we estimate the correctness of its source-words alignments list $A_s(h[j])$ based on its likelihood of appearance in the n-best list, as given in Eq. (7.13):

$$P(h,j) = \frac{\sum_{h' \in H(x)} E(A_s(h[j]), A_s(h'[j]))}{n} \tag{7.13}$$

where $E(A_s(h[j]), A_s(h'[j]))$ is the equality function that returns 1 if the list of source word positions aligned to $h[j]$ is equal to that of $h'[j]$, and 0 otherwise. Then the final probability (the score) of the candidate h is estimated as the product of the probabilities that we calculate for each position j in h, as in Eq. (7.14):

$$S(h) = P(h) = \prod_{j=1}^{t} P(h, j) \tag{7.14}$$

Example 2. With the same example introduced at the beginning of this section, to estimate the probability of the first translation candidate h_1, we first estimate $P(h_1, j)$, the likelihood of having the j^{th} word of h_1 aligned to the list of positions $A_s(h[j])$ in the n-best list. This is done as follows:
$P(h_1, 1) = \frac{3}{3} = 1$, we obtained this probability because the alignment link found between the first word of h_1 and the first word of x is also present in the two other candidates h_2 and h_3.
$P(h_1, 2) = \frac{3}{3} = 1$, we obtained this probability for the same reason as for $P(h_1, 1)$.
$P(h_1, 3) = \frac{1}{3}$, we obtained this probability because the alignment link found between the third word of h_1 and the third word of x is not present in any other candidate.
And in a similar way we estimate the remaining probabilities:
$P(h_1, 4) = \frac{3}{3} = 1$
$P(h_1, 5) = \frac{1}{3}$
Then, the final probability of h_1 is estimated as the product of the probabilities:
$S(h_1) = P(h_1) = P(h_1, 1) * P(h_1, 2) * ... * P(h_1, 5) = 1/9$

7.5.2 Length-based Features

This class of features is used to promote a candidate translation based on the likelihood of its length. Thus, a candidate in the n-best list having a highly probable length will be given a high score and one having a less probable length will be penalized (by being given a lower score).

7.5.2.1 Estimated Length Score

The first feature in this class of length-based features is used to estimate the probability of generating a target translation h of length t for a given source sentence x of length d. This is done by estimating the likelihood of finding a source sentence of length d

aligned to a target sentence of length t in the parallel corpus as shown in Eq. (7.15):

$$S(h) = P(h) = p(len(h) = t \mid len(x) = d) = \frac{c(d,t)}{c(d)} \tag{7.15}$$

where $c(d)$ is the number of times a sentence of length d appears in the source side of the parallel corpus, and $c(d,t)$ is the number of times a source sentence of length d is found aligned to a target sentence of length t in the parallel corpus.

Example 3. Using the example introduced at the beginning of Section 7.5:
$d = len(x) = 5$
$t_1 = len(h_1) = 5, \quad t_2 = len(h_2) = 6, \quad t_3 = len(h_3) = 6$

Let us assume that the statistics that we obtain from the parallel corpus tell us that: $c(d = 5, t = 5) = 2540$, $c(d = 5, t = 6) = 1200$ and $c(d = 5) = 5105$.
Then, the probability of each candidate will be estimated as follows:
$P(h_1) = \frac{c(d=5,t=5)}{c(d=5)} = \frac{2540}{5105} = 0.49$
$P(h_2) = P(h_3) = \frac{c(d=5,t=6)}{c(d=5)} = \frac{1200}{5105} = 0.23$
The first candidate h_1 has the more probable length according to the considered statistics.

7.5.2.2 Length-based Penalty

The second feature of this class is used to penalize the candidates based on their distance from the most probable candidate's length in the n-best list. Equation (7.16) defines the most probable candidate's length in the n-best list as the most frequent one:

$$f_{len}(H(x)) = \underset{len(h), h \in H(x)}{\arg\max} \; (freq(len(h))) \tag{7.16}$$

Then, the score of each candidate $h \in H(x)$ is estimated in a way that penalizes the candidates based on their absolute distance from $f_{len}(H(x))$, as in Eq. (7.17):

$$S(h) = \frac{1}{|len(h) - f_{len}(H(x))| + 1} \tag{7.17}$$

Example 4. Using the example introduced at the beginning of Section 7.5:
$len(h_1) = 5, \quad len(h_2) = 6, \quad len(h_3) = 6$
$f_{len(H(x))} = 6$, because the most frequent candidate length in our case is 6.
The score of each candidate will then be estimated as follows:

$S(h_1) = \frac{1}{|5-6|+1} = \frac{1}{2} = 0.5$
$S(h_2) = S(h_3) = \frac{1}{|6-6|+1} = \frac{1}{1} = 1$

The first candidate h_1 received a slight penalty because of its distance from the most frequent candidate's length. In contrast, the other two candidates h_2 and h_3 were not penalized because their lengths are equal to the most frequent one.

7.5.3 Translation-based Features

This class incorporates some features related to the translation model. These features are used to promote the candidates that have the highest translation adequacy.

7.5.3.1 Reverse Translation Score

This feature is used to prioritize candidates based on the quality of their reverse translation. We denote the direct NMT model that translates from source to target as $\overrightarrow{NMT}$ and the reverse target-to-source model as $\overleftarrow{NMT}$. Each candidate h will be scored using the reverse translation model (target-to-source model) in Eq. (7.18):

$$S(h) = \overleftarrow{NMT}(h) \tag{7.18}$$

where $\overleftarrow{NMT}(h)$ is the score given to the candidate h by the reversed target-to-source NMT translation model.

7.5.3.2 Original Rank Score

The second feature of this class is added to take into consideration the original n-best list order produced by the NMT decoder. To this end, we propose a function that gives a high score to any candidate that is highly ranked in the n-best list and a low score otherwise, as in Eq. (7.19):

$$S(h) = \frac{1}{\log_2(rank(h)+1)} \tag{7.19}$$

This function produces a score of 1 if the candidate is ranked first in the n-best list ($rank(h) = 1$). The assigned score will decrease as the rank of the candidate in the n-best list increases. The score converges towards zero when $rank(h)$ converges towards infinity.

Example 5. Using the above example, we suppose that the n-best list candidates were generated by the NMT decoder in the following order $H(x) = (h_1, h_2, h_3)$. Thus we have:
$rank(h_1) = 1, \quad rank(h_2) = 2, \quad rank(h_3) = 3$
The score of each candidate will then be estimated as follows:

$S(h_1) = \frac{1}{log_2(1+1)} = \frac{1}{1} = 1$
$S(h_2) = \frac{1}{log_2(2+1)} = \frac{1}{1.58} = 0.63$
$S(h_3) = \frac{1}{log_2(3+1)} = \frac{1}{2} = 0.5$

The first candidate was not penalized given that it is ranked first by the decoder. The other two candidates have been penalized based on their ranks.

7.5.3.3 Word-to-word Alignment Score

The third feature of this class is proposed to promote the n-best list candidates that have the most probable alignments. We measure the quality of a candidate alignment by using a word-to-word alignment model. The model is trained using a parallel corpus via the IBM alignment algorithms 1 to 5 (Brown et al., 1993).

To estimate the probability of aligning a candidate word to a source word, we just rely on the IBM word-to-word alignment statistics obtained from the parallel corpus. If we suppose that the j^{th} word of the candidate h was aligned to the i^{th} word of the source sentence x, then their alignment probability is estimated as in Eq. (7.20):

$$P(h[j] \mid x[i]) = \frac{c(h[j], x[i])}{c(h[j])} \tag{7.20}$$

where $c(h[j], x[i])$ is the number of times a target word $h[j]$ was aligned to a source word $x[i]$, and $c(h[j])$ is the count of the target word $h[j]$ in the parallel corpus, respectively.

To address the general case in which a candidate word $h[j]$ can be aligned to multiple source words $A_s(h[j])$ (such as the 3^{rd} word of h_3 in Figure 7.3), we just take their average word-to-word alignment probability as shown in Eq. (7.21):

$$P(h, j) = \frac{\sum_{i \in A_s(h[j])} P(h[j] \mid x[i])}{|A_s(h[j])|} \tag{7.21}$$

where $|A_s(h[j])|$ is the number of source words that $h[j]$ is aligned to. Then, the probability of the whole candidate h (the score of h) is estimated as the product of all the alignment probabilities estimated at each position j of h as shown in Eq. (7.22):

$$S(h) = P(h) = \prod_{j=1}^{t} P(h,j) \tag{7.22}$$

We note that the case of null probability ($P(h,j) = 0$) is handled by assigning a very small value to it, in our case $\epsilon = 10^{-5}$.

Example 6. Still using the example introduced at the beginning of Section 7.5 and the alignment given in Figure 7.3, to estimate the probability of the first candidate h_1 according to this feature, we first estimate $P(h_1, j)$ the likelihood of the alignment link of the j^{th} word of h_1 as follows from the parallel corpus
$P(h_1, 1)$ = P(هناك | something) = c(هناك, something) / c(هناك)
$P(h_1, 2)$ = P(شيء | something) = c(شيء, something) / c(شيء)
The remaining probabilities $P(h_1, 3)$, $P(h_1, 4)$ and $P(h_1, 5)$ can be estimated in a similar manner.
Then, the final probability of h_1 is estimated as the product:
$S(h_1) = P(h_1) = P(h_1, 1) * P(h_1, 2) * ... * P(h_1, 5)$

7.5.3.4 Right-to-left Translation Score

Unlike standard NMT models which generate the translation from left-to-right, a right-to-left (R2L) NMT model generates the translation from right-to-left (in the reverse order). Some recent studies have shown that a R2L NMT model can be beneficial for the task of n-best list re-ranking (Hassan et al., 2018; Liu et al., 2016, 2018). The R2L model is trained in a similar way to the L2R model, the only difference being that the target-side sentences of the parallel corpus need to be reversed. After training the model, each translation candidate in the n-best list is first reversed, and then the R2L NMT is used to assign a score to it as shown in Eq. (7.23):

$$S(h) = P(h) = NMT_{R2L}(reverse(h)) \tag{7.23}$$

where h is the translation candidate and $reverse(h)$ is h in its reversed order.

7.5.4 Fluency Features

This class of features has been included to promote the candidates based on their level of fluency. To this end, two models have been used: a recurrent neural network and an n-gram language model.

7.5.4.1 *n*-gram Language Model Score

An *n*-gram language model assigns a probability to a candidate sentence based on the likelihood of occurrence of each word in it given a few previous words (history). We use this feature to assign a score (probability) to each candidate in our n-best list based on a statistical *n*-gram Markov language model (Kirchhoff and Yang, 2005).

7.5.4.2 RNN Language Model Score

An RNN language model (Mikolov et al., 2010) can keep track of long-term dependencies and is often shown to be very effective in practice. We incorporate this feature to assign a score to each candidate in the n-best list.

7.5.5 Embedding Features

In this class, we use bilingual word embedding features to take into account the semantic and syntactic aspects of the translated candidates. To build a bilingual word embeddings vector, we followed the approach proposed by Smith et al. (2017). First, monolingual embeddings are built from the source and the target sides of the parallel corpus. Then the source and target embeddings are aligned using a small bilingual word-to-word translation dictionary. The alignment is performed via a linear transformation between the two embedding distributions (Smith et al., 2017). Under this bilingual word embedding model, the similarity between a source and a target word-embedding vectors reflects the degree of their semantic correspondence. This interesting property constitutes the core of all the features that we will be presenting under this class.

We define $ES(x[i])$ as the source embedding vector of the word $x[i]$ and $ET(h[j])$ as the target embedding vector of the word $h[j]$.

7.5.5.1 Global Semantic Similarity

This first feature is used to consider the global semantic similarity between the source sentence and its candidate translation in a bag-of-words fashion. The idea is simple: we first need to represent the whole source sentence as a fixed-sized vector by taking the average of all its word embeddings. Then, in a similar manner, we estimate the average vector representation of each translation candidate in the n-best list. Having these fixed-sized vector representations for the source sentence and each one of its

translation candidates, we can easily estimate the semantic similarity (cosine similarity) between the source sentence and one of its translation candidates.

To estimate the similarity between the source sentence x and one of its candidate translations h, we first estimate the average embeddings of all their words as follows:

$$ES_{avg}(x) = \frac{\sum_{i=1}^{d} ES(x[i])}{d} \tag{7.24}$$

$$ET_{avg}(h) = \frac{\sum_{j=1}^{t} ET(h[j])}{t} \tag{7.25}$$

where d is the length of the source sentence x and t is the length of its candidate translation h. Then, the global similarity between x and h is estimated using Eq. (7.26):

$$S(h) = sim(x,h) = \frac{1 + Cosine(ES_{avg}(x), ET_{avg}(h))}{2} \tag{7.26}$$

where $Cosine(ES_{avg}(x), ET_{avg}(h))$ is the Cosine similarity between the average embedding vectors of x and h. We add one to the nominator and divide by two only to map the Cosine result from the interval $[-1, 1]$ to $[0, 1]$.

Example 7. Using the example introduced at the beginning of Section 7.5, to estimate the score of the first candidate h_1 according to this feature, we first calculate $ES_{avg}(x)$ and $ET_{avg}(h_1)$ the average word embedding of the source sentence x and the average word embedding of the first candidate h_1, respectively, as follows:
$ES_{avg}(x)$ = (ES(something) + ES(stiffened) + ES(inside) + ES(me))/4
$ET_{avg}(h_1)$ = (ET(هناك) + ET(شيء) + ET(ب) + ET(داخل) + ET(ي))/5
Then, the final score of h_1 is estimated as the cosine similarity between x and h_1.
$S(h_1) = sim(x, h_1) = \frac{1+Cosine(ES_{avg}(x), ET_{avg}(h_1))}{2}$

7.5.5.2 Alignment-based Semantic Similarity

The second feature of this class is similar to the feature presented in Section 7.5.3.3; the only difference is that this feature relies on bilingual word embeddings instead of a word-to-word alignment.

As previously stated in Section 7.5.3.3, the NMT decoder generates a word-to-word alignment for the input source sentence x and each one of its translation candidates. This feature uses this alignment and the bilingual word embedding information to assign a score to each candidate. Figure 7.5 takes the previous example provided at

the beginning of this section and illustrates its first part which involves the alignment of the source sentence x and its translation candidate h_1.

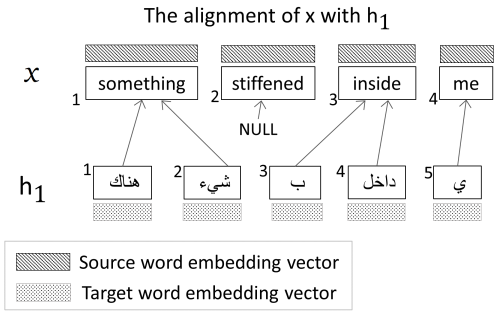


Fig. 7.5: An example illustrating the functioning mechanism of the alignment-based semantic similarity feature

As shown in Figure 7.5, to estimate the score of the candidate h_1, we go through each word of h_1 and measure its semantic similarity to the source words aligned to it. For instance, if we take the first word of h_1, we need to estimate its similarity to the English word “something" (because it is aligned to it). Then for the second word of h_1, we also estimate its similarity with “something" given that it is also aligned to it. Since a single target word (candidate word) of h can be aligned to multiple words of x, we propose Eq. (7.27) to estimate the average embedding vector of all the source words that the target word $h[j]$ is aligned to.

$$ES_{avg}(j) = \frac{\sum_{i \in A_s(h[j])} ES(x[i])}{|A_s(h[j])|} \tag{7.27}$$

Here $A_s(h[j])$ is the list of source word positions of x that the target word $h[j]$ is aligned to, and $|A_s(h[j])|$ is the size of that list. Then Eq. (7.28) is used to estimate the probability of aligning $h[j]$ to its list of source word positions $A_s(h[j])$:

$$P(h,j) = max(\frac{1 + Cosine(ES_{avg}(j), ET(h[j]))}{2}, \varepsilon) \tag{7.28}$$

where ε is a very small value, e.g. 10^{-5}, that is used to avoid having null probabilities. We add one to the nominator and divide by two as we did for the previous feature to simply map the Cosine result from the interval $[-1, 1]$ to $[0, 1]$. Then the score of the candidate sentence h is estimated using Eq. (7.29):

$$S(h) = P(h) = \prod_{j=1}^{t} P(h,j) \tag{7.29}$$

Example 8. Using the same example from the beginning of Section 7.5 and the alignment given in Figure 7.3, to estimate the score of the candidate h_1 according to this feature, we first calculate the alignment probability $P(h_1, j)$ at each position j of the candidate sentence h_1 as follows:
$P(h_1, 1) = \max(\ 1 + \text{Cosine}(\text{ET}(\text{هناك}), \text{ES}(\text{something}))/2\ , \varepsilon)$
$P(h_1, 2) = \max(\ 1 + \text{Cosine}(\text{ET}(\text{شيء}), \text{ES}(\text{something}))/2\ , \varepsilon)$
...
$P(h_1, 5) = \max(\ 1 + \text{Cosine}(\text{ET}(\text{ي}), \text{ES}(\text{me}))/2\ , \varepsilon)$
Then, the final score of h_1 is estimated as the product of all the individual probabilities:
$S(h_1) = P(h_1) = P(h_1, 1) * P(h_1, 2) * ... * P(h_1, 5)$

7.6 Experimentation and Evaluation

In this section, we start by presenting the software and hardware that we have used along with the data that we have incorporated. Then, we present the different tests that we have performed.

7.6.1 Software and Hardware Setup

For all our tests, a Desktop Computer with the following characteristics was used: an Intel Core I5 6500 Skylake Quad-Core processor with a frequency of 3.20 GHz, 16 GB of DDR4 system RAM and a Gigabyte GeForce GTX 1070 GPU with 8GB of GDDR5 memory.

The following software packages were used:

- GIZA toolkit: we used GIZA++ V2 (Och and Ney, 2003)[5] which performs word alignment by means of IBM Models 1 to 5 (Koehn, 2009). We used it for the word-to-word alignment feature of the translation-based feature class (see Sect. 7.5.3.3).

- KenLM toolkit: We used the KenLM toolkit (Heafield, 2011)[6] to train a 6-gram language model feature (see Sect. 7.5.4.1) from the monolingual target-side (Arabic) of the parallel training corpus. Modified Kneser-Ney smoothing was incorporated to handle the case of unseen words.

[5]https://github.com/moses-smt/giza-pp/tree/master/GIZA%2B%2B-v2
[6]https://github.com/kpu/kenlm

- OpenNMT toolkit: We used the OpenNMT toolkit (Klein et al., 2017)[7] to train the RNN language model feature (see Sect. 7.5.4.2) and also to train our NMT baseline.

- SentencePiece toolkit: We used the SentencePiece toolkit[8] to perform a byte-pair-encoding (BPE) language-independent segmentation on both the English and Arabic training data.

- FastText Multilingual toolkit (Smith et al., 2017):[9] This is a toolkit to learn multilingual word embeddings via a linear transformation from their monolingual word embeddings. We used it to learn our bilingual word embeddings from an English-Arabic parallel corpus to use it in our Embedding Features (see Sect. 7.5.5).

- NLG evaluation toolkit (Sharma et al., 2017): We used the nlg-eval toolkit[10] to estimate the BLEU and METEOR[11] (Denkowski and Lavie, 2014) scores.

7.6.2 Data and Preprocessing

To train our baseline NMT model we used the English-Arabic United Nations parallel corpora, which can be obtained for free from the "lingfil" website.[12] We devised our tests into two groups:

1. In-domain tests: For these tests, we used the UNv1.0 English-to-Arabic and Arabic-to-English development and test sets (Ziemski et al., 2016).[13]

2. Out-of-domain tests: For these tests, we used the IWSLT 2015 and 2016 English-to-Arabic and Arabic-to-English test sets.[14]

We kept only sentence pairs with both the source and the target sentence lengths having less than 50 words. "Bad" sentence pairs, i.e. for which the length difference between their source and target exceeds a certain threshold, were removed. We also removed all sentence pairs that contain more than 20% out-of-vocabulary words. The

[7]http://opennmt.net/
[8]https://github.com/google/sentencepiece
[9]https://github.com/Babylonpartners/fastText_multilingual
[10]https://github.com/Maluuba/nlg-eval
[11]For the remainder of this section "MET" is used as an abbreviation for "METEOR".
[12]http://opus.lingfil.uu.se
[13]https://cms.unov.org/UNCorpus/
[14]http://workshop2016.iwslt.org/59.php

tag <*nbr*> was used to group all the numbers that are present in the corpus, such as 12.1, 124, etc.

For the Arabic language, a language-specific preprocessing was applied, in which all the Arabic diacritics such as "Fathah", "Dammah" and "Kasrah" were removed. This is done to decrease the vocabulary size (i.e. decrease the number of Arabic unique words), which eases the training process.[15] For example, the two words رَأيتَ ("you saw") and رَأيتُ ("I saw") will be joined under the same unvocalized word رأيت. For the English side of the parallel corpus, only word-tokenization is performed using the Moses tokenizer available with the Python NLTK toolkit[16].

After applying all the aforementioned preprocessing to the training data, BPE (Sennrich et al., 2015) was used to segment the Arabic and English words into smaller units. We set the BPE vocabulary size to 40k subword symbols for both the English and Arabic languages. This BPE segmentation allows us to simulate an open vocabulary with a limited set of subword symbols, which elegantly addresses the unknown word problem. The statistics about the corpus that we obtained before and after BPE segmentation are provided in Table 7.2.

	Sentences	Unique words	Total words
English	1,273,841	91,843	24,596,431
Arabic	1,273,841	211,221	22,125,765
English-BPE	1,273,841	38,581	25,668,988
Arabic-BPE	1,273,841	39,882	22,973,496

Table 7.2: Statistics of the parallel training corpus used

7.6.3 Baseline Model

Our baseline model follows Google's NMT architecture (Wu et al., 2016) which is an extension of the standard attention-based NMT model (Bahdanau et al., 2014). We used the Adam function (Kingma and Ba, 2014) for the Stochastic Gradient Descent learning rate adjustment. We used 2 GRU encoders and 2 GRU decoders with 500 units each, and a dropout layer with a dropout-rate of 0.3. We set the size of the word embedding vectors to 300. Our model was trained for 13 epochs (which is the default

[15]The effectiveness of this preprocessing step has been investigated in the work of Habash and Sadat (2006).

[16]http://www.nltk.org/

parameter suggested by the OpenNMT toolkit). All the hyperparameters hav been fixed empirically.

7.6.4 Classes/Features Impact

For the remainder of this section, B is used to denote the NMT baseline system, and C_i and F_j denote the i^{th} class and the j^{th} feature, respectively, as follows:

1. N-best list Features (C_1): This includes the Position-based Score (f_1) and N-best Alignment Score (f_2).

2. Length-based Features (C_2): This includes the Estimated Length Score (f_3) and Length-based Penalty (f_4).

3. Translation-based Features (C_3): This includes the Reverse Translation Score (f_5), Original Rank Score (f_6), Word-to-word Alignment Score (f_7), and Right-to-left Translation Score (f_8).

4. Fluency Features (C_4): This includes the n-gram Language Model Score (f_9) and RNN Language Model Score (f_{10}).

5. Embedding Features (C_5): This includes the Global Semantic Similarity (f_{11}) and Alignment-based Semantic Similarity (f_{12}).

The test results incorporating each individual class of features to re-rank the n-best list generated by the NMT baseline decoder are presented in Table 7.3. The feature weights of each individual class are optimized using the QPSO swarm optimization algorithm over the development (DEV) dataset. The MBR results are also reported for both the EN-AR and the AR-EN translation directions.

As shown in Table 7.3, adding a new class of features to the re-ranking system can result in either an increase or a decrease in the overall NMT baseline performance. Indeed, Classes C_2 (length-based features) and C_4 (fluency features) gave a negative impact by decreasing the baseline NMT system results in terms of both BLEU and METEOR scores. This is to be expected since relying solely on the length of the n-best list candidates (C_2) or on their degree of fluency (C_4) does not provide us with the necessary information to re-rank them properly. In contrast, using classes C_1 (N-best list Features), C_3 (Translation-based Features) and C_5 (Embedding Features) led to an increase in the overall baseline performance. This indicates that each of these three classes holds enough information to make good re-ranking decisions, which enables the

AR-EN	Dev-UN		Test-UN		IWSLT15		IWSLT16	
	BLEU	MET	BLEU	MET	BLEU	MET	BLEU	MET
B	42.03	46.48	42.50	47.09	28.13	40.59	28.79	39.27
B+C1	42.51	46.81	**42.82**	46.95	29.02	**42.40**	**30.21**	42.09
B+C2	39.61	45.83	41.29	45.57	27.11	38.83	26.35	39.30
B+C3	42.89	47.07	42.59	47.54	28.97	41.55	30.11	**42.39**
B+C4	40.21	46.13	41.83	47.20	26.42	40.49	27.07	39.30
B+C5	**43.03**	**47.21**	42.41	47.91	28.40	41.50	29.83	42.13
MBR	42.07	46.78	42.47	**48.05**	**29.21**	42.29	29.64	42.28
EN-AR	Dev-UN		Test-UN		IWSLT15		IWSLT16	
	BLEU	MET	BLEU	MET	BLEU	MET	BLEU	MET
B	33.13	43.65	34.11	44.60	19.01	40.15	19.21	39.31
B+C1	33.67	**44.98**	**34.68**	45.81	19.52	**41.81**	19.95	42.29
B+C2	33.03	43.79	33.15	44.52	17.73	39.11	18.63	40.15
B+C3	**33.72**	44.87	34.59	45.87	19.51	41.59	**20.49**	41.97
B+C4	32.07	42.19	31.33	44.21	17.97	39.88	17.31	40.25
B+C5	33.61	44.82	34.47	45.32	**19.59**	41.34	20.35	41.43
MBR	32.23	44.51	34.50	**46.13**	19.22	41.47	19.82	**42.31**

Table 7.3: Translation results for the individual feature classes

selection of the best translation among the n-best list candidates. Even though the difference between them is very minor, C_1 and C_3 gave the highest increase, followed by C_5. The MBR re-ranking method also gave very close results to those three feature classes (but still slightly worse overall in terms of BLEU and METEOR).

Our second experiment aims at testing the effect of removing each feature class. First, we present the results of using all the feature classes (the abbreviation "all"), then we remove one class at a time and see how this affects the overall re-ranking performance. We note that each time a class of features is removed, the QPSO algorithm is incorporated to optimize the weights of the remaining features.

The results of this experiment are provided in Table 7.4. As can be seen, removing the C_1 and C_3 classes led to a noticeable decrease in re-ranking performance. Removing the C_4 and C_5 classes also decreased the overall re-ranking performance. However, the decrease was not as substantial as for C_1 and C_3. Removing the C_2 feature class did almost no damage at all to the re-ranking performance and, in some cases, slightly improved it.

The third experiment that we performed investigated the effect of combining various feature classes, by adding one feature class at a time. As we did for the first experiment,

AR-EN	Dev-UN		Test-UN		IWSLT15		IWSLT16	
	BLEU	MET	BLEU	MET	BLEU	MET	BLEU	MET
All	43.11	48.68	**43.31**	47.90	29.76	**42.62**	31.08	43.17
All - C1	42.71	48.60	42.89	47.11	**29.88**	42.22	30.61	42.07
All - C2	43.02	**48.73**	43.22	47.95	29.81	42.55	30.98	42.23
All - C3	42.91	48.61	42.81	47.77	29.11	41.95	31.10	42.15
All - C4	**43.22**	48.50	43.12	47.51	29.66	42.59	**31.13**	42.10
All - C5	42.81	47.42	43.27	**48.01**	29.59	42.40	31.11	**43.19**
EN-AR	Dev-UN		Test-UN		IWSLT15		IWSLT16	
	BLEU	MET	BLEU	MET	BLEU	MET	BLEU	MET
All	34.72	**45.82**	**35.71**	46.49	**20.41**	**42.48**	20.48	42.47
All - C1	34.52	45.80	35.39	46.29	20.07	42.11	20.22	41.97
All - C2	**34.83**	45.79	35.61	46.53	20.53	42.39	**20.51**	**42.62**
All - C3	33.98	45.29	34.92	45.89	19.83	42.22	20.32	42.49
All - C4	34.68	45.77	35.53	46.42	20.39	42.35	20.49	42.41
All - C5	34.81	45.73	35.63	**46.69**	20.11	42.41	20.19	42.35

Table 7.4: Translation results of removing each individual feature class

we include the MBR results (that were presented previously in Table 7.3) for comparison purposes. We note that each time a new class of features is added to the re-ranking system, a feature weight optimization is performed by means of the QPSO algorithm.

The results of this experiment are provided in Table 7.5. As can be seen, the impact of accumulating the feature classes is generally positive. We can see that even though the fluency feature class (C_4) had no positive impact when used individually, it still yields a very slight increase when used along with other classes. As stated earlier, the MBR re-ranking results were similar to those obtained from the C_1 feature class (yet slightly below it).

The weights (importance values) that were assigned to each feature in our final re-ranking system using the QPSO optimization algorithm are shown in Figure 7.6. The

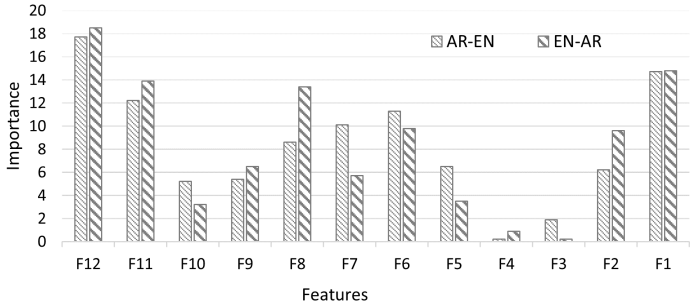


Fig. 7.6: Importance of Features in the Re-ranking System

AR-EN	Dev-UN		Test-UN		IWSLT15		IWSLT16	
	BLEU	MET	BLEU	MET	BLEU	MET	BLEU	MET
B	42.03	46.48	42.50	47.09	28.13	40.59	28.79	39.27
B+C1	42.51	46.81	42.82	46.95	29.02	42.40	30.21	42.09
B+C12	42.48	47.50	43.07	47.30	29.21	42.33	30.49	43.01
B+C123	42.91	47.69	43.11	47.25	29.30	42.64	30.81	**43.21**
B+C1234	42.81	47.42	43.27	48.01	29.59	42.40	**31.11**	43.19
B+C12345	**43.11**	**48.68**	**43.31**	47.90	**29.76**	**42.62**	31.08	43.17
MBR	42.07	46.78	42.47	**48.05**	29.21	42.29	29.64	42.28
EN-AR	Dev-UN		Test-UN		IWSLT15		IWSLT16	
	BLEU	MET	BLEU	MET	BLEU	MET	BLEU	MET
B	33.13	43.65	34.11	44.60	19.01	40.15	19.21	39.31
B+C1	33.67	44.98	34.68	45.81	19.52	41.81	19.95	42.29
B+C12	33.73	44.92	34.51	45.88	19.61	41.79	19.93	42.25
B+C123	34.13	45.33	34.79	45.91	19.79	42.17	20.11	42.37
B+C1234	**34.81**	45.73	35.63	**46.69**	20.11	42.41	20.19	42.35
B+C12345	34.72	**45.82**	**35.71**	46.49	**20.41**	**42.48**	**20.48**	**42.47**
MBR	32.23	44.51	34.50	46.13	19.22	41.47	19.82	42.31

Table 7.5: Translation results for the accumulated feature classes

QPSO algorithm gave a very high importance to the 1^{st} (Position-based Probability) and the 12^{th} (Alignment-based Semantic Similarity) features, meaning that their influence on the re-ranking process is considerable. The estimated length score (F_3) and length-based penalty (F_4) were given very low importance, which indicates that their impact on the re-ranking task was very minimal. All remaining features played an important role in determining the best candidate in the re-ranking system.

Figure 7.7 presents the results of the QPSO feature weight optimization algorithm per class. These results were obtained simply by summing up the feature weights belonging to the same class (from Figure 7.6).

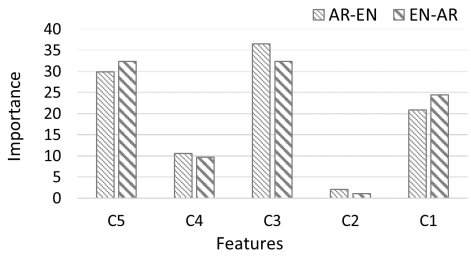


Fig. 7.7: Importance of Classes in the Re-ranking System

As shown in Figure 7.7, a high importance was assigned to the C_3 (Translation-based Features), C_5 (Embedding Features), and C_1 (N-best list Features) classes. Less importance was given to the 4^{th} (Fluency Features) class. Very little importance was assigned to the 2^{sd} (Length-based Features) class. These results suggest that the length-based features are not very helpful for the task of n-best list re-ranking. The fluency features (language models) had a small positive impact on the re-ranking performance.

7.6.5 N-best List Impact on the Decoding Time

We conducted this test to investigate the rate of increase of the decoding time as a function of the n-best list size. The results of this experiment are reported on the UN development set. The effect of increasing the n-best list size on the decoding time is shown in Figure 7.8.

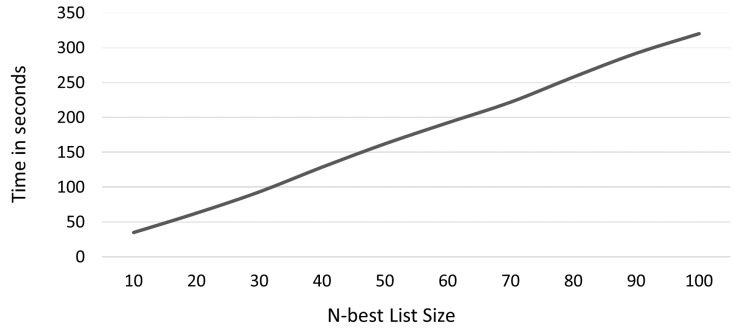


Fig. 7.8: The effect of the n-best list size on the decoding time

We can see that the increase in the decoding time is almost linear with respect to the increase in the n-best list size. It increases from $50s$ for $n = 10$ to $350s$ for $n = 100$.

7.6.6 N-best List Candidates Selection

When working with an n-best list of size n, the re-ranking system tends to pick candidates that are present in certain positions of the n-best list more than others. Figure 7.9 shows the results of the experiment that we have performed on our development set. In this experiment, the selection rate of each position (rank) from the n-best list by our re-ranking system is reported for an n-best list of size $n = 100$.

As shown in Figure 7.9, the best candidate translation is chosen from the first 10 positions of the n-best list almost 70% of the time (for both directions). This is to be expected since the decoder's original ordering is reasonably well-founded which means

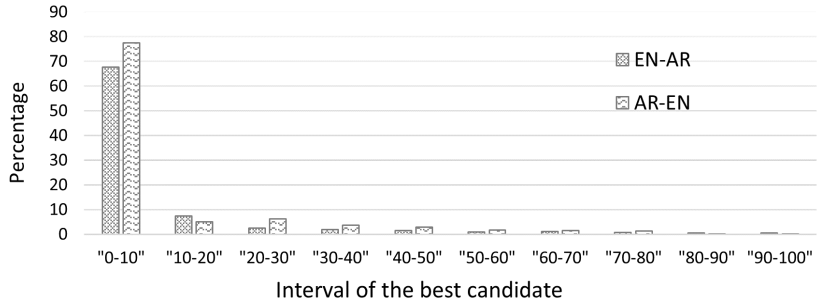


Fig. 7.9: Statistics about the ranks selected by the re-ranking system

that the best candidate is more likely to be there. As we go further towards $n = 100$, the rate of selection of the best candidate keeps decreasing until it becomes almost null when reaching $n = 100$. This result confirms that using a larger n-best list (more than $n = 100$) will not lead to any significant improvement as also suggested by the work of Hasan et al. (2007).

7.7 Discussion

In this chapter, we have tackled the n-best list re-ranking problem by incorporating several features whose weights were optimized using a swarm-based optimization algorithm. The test results that were presented in the previous section show that the features belonging to the translation (C_3), the embedding (C_5), and the n-best list (C_1) feature classes were the most effective for the task of n-best list re-ranking. Indeed, these feature classes alone managed to achieve more than 80% importance value (cf. Figure 7.7) among all the other feature classes incorporated in this study. The two most useful individual features were the alignment-based semantic similarity (F_{12}) and the position-based feature (F_1). The usefulness of the embedding features suggests that the semantic knowledge that can be automatically learned from a bilingual word embedding is indeed helpful for the re-ranking process. With the exception of the length-based features, all the remaining features had mid-to-high importance values (Figure 7.7). Another observation to note is that the language model features did not have a very noticeable impact on the re-ranking performance. We believe this is due to the nature of the NMT encoder-decoder model which has a built-in capacity for learning language model information, i.e. for producing more fluent translations. For these reasons, we believe that additional language model features are not very helpful for the re-ranking process especially when using an NMT baseline system.

Overall, our proposal has some clear advantages, though it still suffers from some limitations as discussed hereafter:

- Strengths:
  - The features proposed in this paper are automatically learned from the parallel corpus without requiring any external tools. Thus our proposal is general enough to be used to tackle the translation task between any pair of languages.
  - This study has addressed the semantic level by introducing two features that use bilingual word embeddings. The effectiveness of these features has been demonstrated via practical tests.
  - Our method can automatically estimate the effectiveness of each feature and assign an importance value to it via a swarm-based weight optimization algorithm. Thus, our proposal can deal with marginally useful features by decreasing their importance.
  - Even though our proposal does not use external tools, the improvement that we have achieved was still very noticeable, which encourages us to explore future improvements in this same direction.
- Limitations:
  - Even though general-purpose features are interesting in the task of re-ranking, it is always helpful to use language-specific features to address linguistic phenomena that are related to a specific language such as Arabic.
  - Our system is able to ignore irrelevant features by giving them weights of very minor importance. However, having a more sophisticated filtering method that can remove the features when they are deemed irrelevant would be a valuable improvement.

7.8 Conclusion

In this chapter, we have presented our third and main contribution which is a re-ranking system that uses a set of new sophisticated features to effectively reorder n-best lists of translation candidates. All our proposed features can be extracted directly from the parallel corpus without the need for any language-specific NLP tools. We also present a method for feature weight optimization that uses a Quantum-behaved Particle Swarm

Optimization algorithm which guarantees the global convergence of the optimization process. The effectiveness of our re-ranking system has been tested on the UN and the IWSLT evaluation benchmarks, and the results obtained have shown a noticeable increase in the overall translation performance.

The contributions of this study can be summarized as follows:

- We proposed a re-ranking system that does not have any language-specific tool dependencies.
- We proposed a new feature class that can capture the semantic level of the translation candidates via bilingual word embeddings.
- We used a swarm-based QPSO optimization algorithm that has been trained to maximize the BLEU score on a held-out data set.
- We tested our proposal on an English-to-Arabic and Arabic-to-English NMT systems.

This contribution can be developed further in various directions. One such direction is to introduce a feature filtering mechanism that detects and removes the non-useful features. Furthermore, merging the n-best lists of multiple translation systems prior to the re-ranking task may yield better results.

Chapter 8

Conclusions

In this thesis, we have started by presenting the concept of deep learning along with its applications in the field of NLP. Then, we have presented all the paradigms that have been proposed in the field of MT and the different ways that can be used to evaluate the performance of MT systems. We have also introduced the Arabic language, its characteristics, and the difficulties involved in its translation and gave a complete review of all the research studies that have been done in regard to Arabic MT along with a discussion about the current limitations that are still present. Finally, we have presented and detailed all our contributions which we will summarize below and also discuss future research directions.

8.1 Main Contributions

This thesis has addressed the task of English-to-Arabic MT. We have investigated the state-of-the-art methods that have been applied for this purpose and proposed several contributions.

- One contribution we have made is to improve the English-to-Arabic word reordering task by presenting a new preordering system that can efficiently handle both long- and short-distance word reorderings. The system is able to learn reordering rules automatically from a parallel corpus using word alignment and a basic part-of-speech source language tagging. We showed that our system brings a noticeable improvement to a baseline English-to-Arabic PSMT system of up to 1.45 BLEU points.
- A second contribution we have made concerns named entity handing by proposing a transliteration system that transliterates English named entities from and into

Arabic. The system that we proposed uses a deep learning neural network that is an attention-based encoder-decoder. We have obtained very promising results; for the English-to-Arabic direction, we have achieved a CER of 0.95 and a WER of 5.40, and for the Arabic-to-English direction, though the results were not as good (CER 16.35 of and WER of only 65.16). Compared to the previous studies that have been done in this transliteration direction, this is still a promising start.

- A third and most important contribution is a system we proposed for n-best list re-scoring in the context of English-to-Arabic NMT that works by using a set of sophisticated features. The features set that we have used covers the lexical, syntactic and semantic aspects of the translation candidates (the n-best list candidates) and can be grouped into five classes: (1) Translation-based Features: these features are related to a translation model and can be helpful for promoting translation adequacy; (2) Fluency Features: these features are used to promote syntactic fluency by incorporating language models; (3) Length-based Features: these features are used to promote the n-best list candidates according to the likelihood of their lengths; (4) N-best list Features: these features are extracted directly from the n-best list and are used to promote the most likely candidates in it, and (5) Embedding Features: these are based on bilingual word embeddings and are used to cover the semantic aspect of the translation candidates. The weights of these features are optimized automatically via a swarm-based optimization algorithm. Our proposal manages to achieve an improvement of up to 1.5 BLEU points over the baseline NMT results.

8.2 Future Work

We have presented in this thesis several contributions in the context of English-to-Arabic MT (also applied to Arabic-to-English MT). These contributions can be further improved in several ways, and the room is still open to many new ideas and thoughts.

- Our approach regarding n-best list re-ranking can be further developed in various directions. One such direction is to introduce a feature filtering mechanism that detects and removes the non-useful features. Also merging the n-best lists of multiple translation systems prior to the re-ranking task may yield better results. Additional features can also be added to appropriately handle the problem of pronoun resolution.

- Our work on named entity transliteration can be further developed in various directions. One direction is to consider the case of transliteration between Arabic and other languages besides English. Another interesting future direction is to integrate this model into an English-to-Arabic machine translation system to address the problem of named entity transliteration.
- Our work on word reordering can be improved by exploring tree structures (dependency and constituency trees). The coupling of both preordering and post-ordering strategies in the same framework can also be interesting to explore.
- Some other thoughts that are not related to our contributions and that we believe to be promising in the field of Arabic machine translation in general are:
  - Finding new ways of merging the advantages of neural and statistical machine translation in a single framework.
  - Finding new ways of training the NMT models with less data which is very helpful when dealing with low-resource languages (e.g. Arabic).
  - Finding new approaches of dealing with the document-level translation which allows the translation system to consider linguistic phenomena that go beyond the sentence-level boundaries such as pronoun resolution.
  - Finding efficient methods for dealing with out-of-vocabulary (OOV) words especially for the languages that have rich morphology (e.g. Arabic).
  - Dealing with Arabic-related linguistic problems such as the translation of Arabic named entities, idiomatic expressions, ambiguous words, etc.

References

Abbas, M. and Smaili, K. (2005). Comparison of topic identification methods for arabic language. In *Proceedings of International Conference on Recent Advances in Natural Language Processing, RANLP*, pages 14–17.

Abbas, M., Smaïli, K., and Berkani, D. (2011). Evaluation of topic identification methods on arabic corpora. *J. Digit. Inf. Manag.*, 9(5):185–192.

AbdulJaleel, N. and Larkey, L. (2003). English to arabic transliteration for information retrieval: A statistical approach. *Center for Intelligent Information Retrieval Computer Science, University of Massachusetts*.

Abuelyaman, E., Rahmatallah, L., Mukhtar, W., and Elagabani, M. (2014). Machine translation of Arabic language: challenges and keys. In *Intelligent Systems, Modelling and Simulation (ISMS), 2014 5th International Conference on*, pages 111–116. IEEE.

Adly, N. and Al Ansary, S. (2009). Evaluation of Arabic machine translation system based on the universal networking language. In *International Conference on Application of Natural Language to Information Systems*, pages 243–257. Springer.

Agrawal, R., Srikant, R., et al. (1994). Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499.

Aharoni, R., Johnson, M., and Firat, O. (2019). Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884.

Al Dam, R. and Guessoum, A. (2010). Building a neural network-based english-to-arabic transfer module from an unrestricted domain. In *Machine and Web Intelligence (ICMWI), 2010 International Conference on*, pages 94–101. IEEE.

Al-Haj, H. and Lavie, A. (2012). The impact of Arabic morphological segmentation on broad-coverage English-to-Arabic statistical machine translation. *Machine translation*, 26(1-2):3–24.

Al-Rukban, A. and Saudagar, A. K. J. (2017). Evaluation of English to Arabic machine translation systems using bleu and gtm. In *Proceedings of the 2017 9th International Conference on Education Technology and Computers*, pages 228–232. ACM.

Alansary, S. and Nagi, M. (2014). The international corpus of arabic: Compilation, analysis and evaluation. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 8–17.

Alkhatib, M. and Shaalan, K. (2018). The key challenges for Arabic machine translation. In *Intelligent Natural Language Processing: Trends and Applications*, pages 139–156. Springer.

Almahairi, A., Cho, K., Habash, N., and Courville, A. C. (2016). First result on arabic neural machine translation. *CoRR*, abs/1606.02680.

Almansor, E. H. and Al-Ani, A. (2018). A hybrid neural machine translation technique for translating low resource languages. In *International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 347–356. Springer.

Alqudsi, A., Omar, N., and Shaker, K. (2014). Arabic machine translation: a survey. *Artif. Intell. Rev.*, 42(4):549–572.

Alqudsi, A., Omar, N., and Shaker, K. (2019). A hybrid rules and statistical method for arabic to english machine translation. In *2019 2nd International Conference on Computer Applications & Information Security (ICCAIS)*, pages 1–7. IEEE.

Alrabiah, M., Al-Salman, A., and Atwell, E. (2013). The design and construction of the 50 million words ksucca. In *Proceedings of WACL'2 Second Workshop on Arabic Corpus Linguistics*, pages 5–8. The University of Leeds.

Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov, S., and Collins, M. (2016). Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2442–2452.

Aouichat, A., Hadj Ameur, M. S., and Geussoum, A. (2018). Arabic question classification using support vector machines and convolutional neural networks. In *International Conference on Applications of Natural Language to Information Systems*, pages 113–125. Springer.

Aqlan, F., Fan, X., Alqwbani, A., and Al-Mansoub, A. (2019). Arabic–chinese neural machine translation: Romanized arabic as subword unit for arabic-sourced translation. *IEEE Access*, 7:133122–133135.

Arbabi, M., Fischthal, S. M., Cheng, V. C., and Bart, E. (1994). Algorithms for arabic name transliteration. *IBM Journal of research and Development*, 38(2):183–194.

Arun, A. and Koehn, P. (2007). Online learning methods for discriminative training of phrase based statistical machine translation. *Proc. of MT Summit XI*, 2(5):29.

Ataman, D., Aziz, W., and Birch, A. (2020). A latent morphology model for open-vocabulary neural machine translation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Ataman, D. and Federico, M. (2018). An evaluation of two vocabulary reduction methods for neural machine translation. In *Proceedings of the The 13th Conference of The Association for Machine Translation in the Americas, Boston, USA*, pages 97–110.

Ataman, D., Firat, O., Di Gangi, M. A., Federico, M., and Birch, A. (2019). On the importance of word boundaries in character-level neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 187–193.

Ataman, D., Negri, M., Turchi, M., and Federico, M. (2017). Linguistically motivated vocabulary reduction for neural machine translation from turkish to english. *The Prague Bulletin of Mathematical Linguistics*, 108(1):331–342.

Ayan, N. F. and Dorr, B. J. (2006). A maximum entropy approach to combining word alignments. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 96–103. Association for Computational Linguistics.

Ayan, N. F., Dorr, B. J., and Monz, C. (2005). Neuralign: Combining word alignments using neural networks. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 65–72, Vancouver, British Columbia, Canada. Association for Computational Linguistics, Association for Computational Linguistics.

Badr, I., Zbib, R., and Glass, J. (2008). Segmentation for english-to-arabic statistical machine translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 153–156, Stroudsburg, PA, USA. Association for Computational Linguistics.

Badr, I., Zbib, R., and Glass, J. (2009). Syntactic phrase reordering for english-to-arabic statistical machine translation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 86–93, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Banerjee, S. and Lavie, A. (2005). Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, volume 29, pages 65–72.

Belinkov, Y., Durrani, N., Dalvi, F., Sajjad, H., and Glass, J. R. (2017). What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 861–872.

Belinkov, Y. and Glass, J. R. (2016). Large-scale machine translation between arabic and hebrew: Available corpora and initial results. *CoRR*, abs/1609.07701.

Bengio, Y., Goodfellow, I. J., and Courville, A. (2015). Deep learning, book in preparation for mit press. *Disponıvel em http://www. iro. umontreal. ca/bengioy/dlbook*.

Berrichi, S. and Mazroui, A. (2018). Benefits of morphosyntactic features on english-arabic statistical machine translation. In *2018 IEEE 5th International Congress on Information Science and Technology (CiSt)*, pages 244–248. IEEE.

Bisazza, A., Pighin, D., and Federico, M. (2012). Chunk-lattices for verb reordering in Arabic-English statistical machine translation. *Machine translation*, 26(1-2):85–103.

Blunsom, P. and Cohn, T. (2006). Discriminative word alignment with conditional random fields. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 65–72. Association for Computational Linguistics.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Bordes, A., Usunier, N., Chopra, S., and Weston, J. (2015). Large-scale simple question answering with memory networks. *CoRR*, abs/1506.02075.

Brants, T., Popat, A. C., Xu, P., Och, F. J., and Dean, J. (2007). Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Britz, D. (2016). Recurrent neural network tutorial.

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

Brown, P. F., Cocke, J., Della-Pietra, S. A., Della-Pietra, V. J., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Rossin, P. (1990a). A statistical approach to machine translation. *Computational Linguistics*, 16(2):76–85.

Brown, P. F., Cocke, J., Della-Pietra, S. A., Della-Pietra, V. J., Jelinek, F., Mercer, R. L., and Rossin, P. (1988b). A statistical approach to language translation. In *Proceedings of the International Conference on Computational Linguistics (COLING)*.

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. (1990b). A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85.

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

Burlot, F., García-Martínez, M., Barrault, L., Bougares, F., and Yvon, F. (2017). Word representations in factored neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 20–31, Copenhagen, Denmark. Association for Computational Linguistics.

Carpuat, M., Marton, Y., and Habash, N. (2010). Improving Arabic-to-English statistical machine translation by reordering post-verbal subjects for alignment. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 178–183. Association for Computational Linguistics.

Carpuat, M., Marton, Y., and Habash, N. (2012). Improved Arabic-to-English statistical machine translation by reordering post-verbal subjects for word alignment. *Machine Translation*, 26(1-2):105–120.

Carter, S. and Monz, C. (2010). Discriminative syntactic reranking for statistical machine translation. In *Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010), Denver, CO, USA*.

Carter, S. and Monz, C. (2011). Syntactic discriminative language model rerankers for statistical machine translation. *Machine translation*, 25(4):317–339.

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

Chen, B., Cettolo, M., and Federico, M. (2006). Reordering rules for phrase-based statistical machine translation. In *International Workshop on Spoken Language Translation (IWSLT) 2006*.

Chen, B. and Cherry, C. (2014). A systematic comparison of smoothing techniques for sentence-level bleu. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 362–367.

Chen, D. and Manning, C. (2014). A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 740–750.

Chen, Y. and Skiena, S. (2016). False-friend detection and entity matching via unsupervised transliteration. *arXiv preprint arXiv:1611.06722*.

Cheng, Y. (1995). Mean shift, mode seeking, and clustering. *IEEE transactions on pattern analysis and machine intelligence*, 17(8):790–799.

Cherry, C. and Lin, D. (2006). Soft syntactic constraints for word alignment through discriminative training. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 105–112. Association for Computational Linguistics.

Chiu, J. and Nichols, E. (2016). Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association of Computational Linguistics*, 4(1):357–370.

Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014a). On the properties of neural machine translation: Encoder-decoder approaches. In *Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014*, pages 103–111.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014b). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734.

Collins, M. (2011). Statistical machine translation: Ibm models 1 and 2.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Constant, M., Eryiğit, G., Monti, J., Van Der Plas, L., Ramisch, C., Rosner, M., and Todirascu, A. (2017). Multiword expression processing: A survey. *Computational Linguistics*, 43(4):837–892.

Costa-Jussa, M. R. and Fonollosa, J. A. (2015). Latest trends in hybrid machine translation and its applications. *Computer Speech & Language*, 32(1):3–10.

Coughlin, D. (2003). Correlating automated and human assessments of machine translation quality. In *Proceedings of MT summit IX*, pages 63–70.

Dai, A. M., Olah, C., and Le, Q. V. (2015). Document embedding with paragraph vectors. *CoRR*, abs/1507.07998.

Day, W. H. and Edelsbrunner, H. (1984). Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of classification*, 1(1):7–24.

De La Briandais, R. (1959). File searching using variable length keys. In *Papers presented at the the March 3-5, 1959, western joint computer conference*, pages 295–298. ACM.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38.

Deng, L. and Liu, Y. (2018). *Deep Learning in Natural Language Processing*. Springer.

Denkowski, M. and Lavie, A. (2014). Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pages 376–380.

Deselaers, T., Hasan, S., Bender, O., and Ney, H. (2009). A deep learning approach to machine transliteration. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 233–241. Association for Computational Linguistics.

Diab, M. (2009). Second generation amira tools for arabic processing: Fast and robust tokenization, pos tagging, and base phrase chunking. In *2nd International Conference on Arabic Language Resources and Tools*, volume 110.

Diab, M., Ghoneim, M., and Habash, N. (2007). Arabic diacritization in the context of statistical machine translation. In *Proceedings of MT-Summit*.

Ding, S., Renduchintala, A., and Duh, K. (2019). A call for prudent choice of subword merge operations in neural machine translation. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pages 204–213, Dublin, Ireland. European Association for Machine Translation.

Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145. Morgan Kaufmann Publishers Inc.

Duh, K. and Kirchhoff, K. (2008). Beyond log-linear models: boosted minimum error rate training for n-best re-ranking. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 37–40. Association for Computational Linguistics.

Duh, K., Sudoh, K., Tsukada, H., Isozaki, H., and Nagata, M. (2010). N-best reranking by multitask learning. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 375–383. Association for Computational Linguistics.

El Kholy, A. and Habash, N. (2012). Orthographic and morphological processing for English-Arabic statistical machine translation. *Machine Translation*, 26(1-2):25–45.

El Marouani, M., Boudaa, T., and Enneya, N. (2018). Incorporation of linguistic features in machine translation evaluation of Arabic. In *International Conference on Big Data, Cloud and Applications*, pages 500–511. Springer.

Ellouze, M., Neifar, W., and Belguith, L. H. (2018). Word alignment applied on english-arabic parallel corpus. In *LPKM*.

Elming, J. (2008). Syntactic reordering integrated with phrase-based smt. In *Proceedings of the Second Workshop on Syntax and Structure in Statistical Translation*, pages 46–54. Association for Computational Linguistics.

Elming, J. and Habash, N. (2009). Syntactic reordering for English-Arabic phrase-based machine translation. In *Proceedings of the EACL 2009 Workshop on Computational Approaches to Semitic Languages*, pages 69–77. Association for Computational Linguistics.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). Density-based spatial clustering of applications with noise. In *Int. Conf. Knowledge Discovery and Data Mining*, volume 240.

Farghaly, A. and Shaalan, K. (2009). Arabic natural language processing: Challenges and solutions. *ACM Transactions on Asian Language Information Processing (TALIP)*, 8(4):14.

Farzi, S. and Faili, H. (2015). A swarm-inspired re-ranker system for statistical machine translation. *Computer Speech & Language*, 29(1):45–62.

Fillmore, C. J. and Atkins, B. T. (2000). Describing polysemy: The case of 'crawl'. *Polysemy: Theoretical and computational approaches*, pages 91–110.

Finch, A., Liu, L., Wang, X., and Sumita, E. (2016). Target-bidirectional neural models for machine transliteration. *ACL 2016*, page 78.

Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 363–370. Association for Computational Linguistics.

Fossum, V., Knight, K., and Abney, S. (2008). Using syntax to improve word alignment precision for syntax-based machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 44–52. Association for Computational Linguistics.

Freitag, M. and Al-Onaizan, Y. (2017). Beam search strategies for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 56–60.

Fuji, M., Utiyama, M., Sumita, E., and Matsumoto, Y. (2016). Global pre-ordering for improving sublanguage translation. *WAT 2016*, page 84.

Fujii, A. and Ishikawa, T. (2001). Japanese/english cross-language information retrieval: Exploration of query translation and transliteration. *Computers and the Humanities*, 35(4):389–420.

Gao, Q., Guzman, F., and Vogel, S. (2010). Emdc: a semi-supervised approach for word alignment. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 349–357. Association for Computational Linguistics.

García-Martínez, M., Aransa, W., Bougares, F., and Barrault, L. (2020). Addressing data sparsity for neural machine translation between morphologically rich languages. *Machine Translation*, 34(1):1–20.

Genzel, D. (2010). Automatically learning source-side reordering rules for large scale machine translation. In *Proceedings of the 23rd international conference on computational linguistics*, pages 376–384. Association for Computational Linguistics.

Goller, C. and Kuchler, A. (1996). Learning task-dependent distributed representations by backpropagation through structure. In *Neural Networks, 1996., IEEE International Conference on*, volume 1, pages 347–352. IEEE.

González-Rubio, J., Juan, A., and Casacuberta, F. (2011). Minimum bayes-risk system combination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1268–1277. Association for Computational Linguistics.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

Green, S., Cer, D., and Manning, C. D. (2014). Phrasal: A toolkit for new directions in statistical machine translation. In *In Proceddings of the Ninth Workshop on Statistical Machine Translation*.

Groves, D. and Way, A. (2005). Hybrid data-driven models of machine translation. *Machine Translation*, 19(3-4):301–323.

Guessoum, A. and Zantout, R. (2001). A methodology for a semi-automatic evaluation of the lexicons of machine translation systems. *Machine translation*, 16(2):127–149.

Guessoum, A. and Zantout, R. (2004). A methodology for evaluating arabic machine translation systems. *Machine Translation*, 18(4):299–335.

Guzmán, F., Bouamor, H., Baly, R., and Habash, N. (2016). Machine translation evaluation for Arabic using morphologically-enriched embeddings. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1398–1408.

Habash, N. (2007). Syntactic preprocessing for statistical machine translation. *Proceedings of the 11th MT Summit*, 10:10.

Habash, N. (2008). Four techniques for online handling of out-of-vocabulary words in Arabic-English statistical machine translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 57–60. Association for Computational Linguistics.

Habash, N., Dorr, B., and Monz, C. (2009a). Symbolic-to-statistical hybridization: extending generation-heavy machine translation. *Machine Translation*, 23(1):23–63.

Habash, N. and Rambow, O. (2007). Arabic diacritization through full morphological tagging. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 53–56. Association for Computational Linguistics.

Habash, N., Rambow, O., and Roth, R. (2009b). Mada+ tokan: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In *Proceedings of the 2nd international conference on Arabic language resources and tools (MEDAR), Cairo, Egypt*, volume 41, page 62.

Habash, N. and Sadat, F. (2006). Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 49–52. Association for Computational Linguistics.

Habash, N., Zalmout, N., Taji, D., Hoang, H., and Alzate, M. (2017). A parallel corpus for evaluating machine translation between Arabic and European languages. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 235–241, Valencia, Spain. Association for Computational Linguistics.

Habash, N. Y. (2010). Introduction to arabic natural language processing. *Synthesis Lectures on Human Language Technologies*, 3(1):1–187.

Hadj Ameur, M. S., Guessoum, A., and Meziane, F. (2019). Improving arabic neural machine translation via n-best list re-ranking. *Machine Translation*, 33(4):279–314.

Hadj Ameur, M. S., Khadir, A. C., and Guessoum, A. (2017a). An automatic approach for wordnet enrichment applied to arabic wordnet. In *International Conference on Arabic Language Processing*, pages 3–18. Springer.

Hadj Ameur, M. S., Meziane, F., and Guessoum, A. (2017b). Arabic machine transliteration using an attention-based encoder-decoder model. *Procedia Computer Science*, 117:287–297. Arabic Computational Linguistics, Dubai.

Hadj Ameur, M. S., Moulahoum, Y., and Guessoum, A. (2015). Restoration of arabic diacritics using a multilevel statistical model. In Amine, A., Bellatreche, L., Elberrichi, Z., Neuhold, E. J., and Wrembel, R., editors, *Computer Science and Its Applications*, pages 181–192, Cham. Springer International Publishing.

Hadla, L., Hailat, T., and Al-Kabi, M. (2014). Evaluating arabic to english machine translation. *International Journal of Advanced Computer Science and Applications*, 5:68–73.

Han, J., Pei, J., and Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Hasan, S., Mansour, S., and Ney, H. (2012). A comparison of segmentation methods and extended lexicon models for Arabic statistical machine translation. *Machine translation*, 26(1-2):47–65.

Hasan, S., Zens, R., and Ney, H. (2007). Are very large n-best lists useful for smt? In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 57–60. Association for Computational Linguistics.

Hassan, H., Aue, A., Chen, C., Chowdhary, V., Clark, J., Federmann, C., Huang, X., Junczys-Dowmunt, M., Lewis, W., Li, M., et al. (2018). Achieving human parity on automatic chinese to english news translation. *arXiv preprint arXiv:1803.05567*.

Heafield, K. (2011). Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197. Association for Computational Linguistics.

Hermjakob, U. (2009). Improved word alignment with statistics and linguistic heuristics. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 229–237. Association for Computational Linguistics.

Hermjakob, U., Knight, K., and Daumé III, H. (2008). Name translation in statistical machine translation-learning when to transliterate. In *ACL*, pages 389–397.

Ho, C. S.-H. and Bryant, P. (1997). Learning to read Chinese beyond the logographic phase. *Reading research quarterly*, 32(3):276–289.

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

Hubel, D. H. and Wiesel, T. N. (1968). Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, 195(1):215–243.

Hutchins, W. J. (1986). *Machine translation: past, present, future*. Ellis Horwood Chichester.

Hutchins, W. J. (1995). Machine translation: A brief history. In *Concise history of the language sciences*, pages 431–445. Elsevier.

Hutchins, W. J. and Somers, H. L. (1992). *An introduction to machine translation*, volume 362. Academic Press London.

Ittycheriah, A. and Roukos, S. (2005). A maximum entropy word aligner for Arabic-English machine translation. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 89–96. Association for Computational Linguistics.

Iyyer, M., Boyd-Graber, J., Claudino, L., Socher, R., and Daumé III, H. (2014). A neural network for factoid question answering over paragraphs. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 633–644.

Jadidinejad, A. H. (2016). Neural machine transliteration: Preliminary results. *arXiv preprint arXiv:1609.04253*.

Jehl, L., de Gispert, A., Hopkins, M., and Byrne, B. (2014). Source-side preordering for translation using logistic regression and depth-first branch-and-bound search. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 239–248.

Ji, B., Zhang, Z., Duan, X., Zhang, M., Chen, B., and Luo, W. (2020). Cross-lingual pre-training based transfer for zero-shot neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 115–122.

Jiang, L., Zhou, M., Chien, L.-F., and Niu, C. (2007). Named entity translation with web mining and transliteration. In *IJCAI*, volume 7, pages 1629–1634.

Jozefowicz, R., Zaremba, W., and Sutskever, I. (2015). An empirical exploration of recurrent network architectures. In *International Conference on Machine Learning*, pages 2342–2350.

Junczys-Dowmunt, M., Dwojak, T., and Hoang, H. (2016). Is neural machine translation ready for deployment. *A case study on*, 30.

Karpathy, A. (2019). Convolutional neural networks tutorial.

Kaur, K. and Singh, P. (2014). Review of machine transliteration techniques. *International Journal of Computer Applications*, 107(20).

Kelleher, J. D., Mac Namee, B., and D'Arcy, A. (2015). *Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies*. MIT Press.

Khemakhem, I. T., Jamoussi, S., and Hamadou, A. B. (2013). Integrating morpho-syntactic features in English-Arabic statistical machine translation. In *Proceedings of the Second Workshop on Hybrid Approaches to Translation*, pages 74–81.

Khemakhem, I. T., Jamoussi, S., and Hamadou, A. B. (2015). Arabic-English semantic word class alignment to improve statistical machine translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 663–671.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Kim, Y., Jernite, Y., Sontag, D., and Rush, A. M. (2016). Character-aware neural language models. In *AAAI*, pages 2741–2749.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kirchhoff, K. and Yang, M. (2005). Improved language modeling for statistical machine translation. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 125–128. Association for Computational Linguistics.

Klein, G., Kim, Y., Deng, Y., Nguyen, V., Senellart, J., and Rush, A. (2018). OpenNMT: Neural machine translation toolkit. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Papers)*, pages 177–184, Boston, MA. Association for Machine Translation in the Americas.

Klein, G., Kim, Y., Deng, Y., Senellart, J., and Rush, A. M. (2017). Opennmt: Open-source toolkit for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, System Demonstrations*, pages 67–72.

Kneser, R. and Ney, H. (1995). Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184, Detroit, MI, USA.

Koehn, P. (2009). *Statistical machine translation*. Cambridge University Press, New York, NY, USA, 1st edition.

Koehn, P., Axelrod, A., Birch, A., Callison-Burch, C., Osborne, M., Talbot, D., and White, M. (2005). Edinburgh system description for the 2005 iwslt speech translation evaluation. In *IWSLT*, pages 68–75.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics.

Koehn, P. and Monz, C. (2006). Manual and automatic evaluation of machine translation between european languages. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 102–121.

Kordoni, V. and Simova, I. (2014). Multiword expressions in machine translation. In *LREC*, pages 1208–1211.

Kumar, S. and Byrne, W. (2004). Minimum bayes-risk decoding for statistical machine translation. Technical report, JOHNS HOPKINS UNIV BALTIMORE MD CENTER FOR LANGUAGE AND SPEECH PROCESSING (CLSP).

Kusner, M., Sun, Y., Kolkin, N., and Weinberger, K. (2015). From word embeddings to document distances. In *International Conference on Machine Learning*, pages 957–966.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 260–270.

Lee, Y.-S. (2004). Morphological analysis for statistical machine translation. In *Proceedings of HLT-NAACL 2004: Short Papers*, pages 57–60. Association for Computational Linguistics.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Li, J. and Jurafsky, D. (2016). Mutual information and diverse decoding improve neural machine translation. *arXiv preprint arXiv:1601.00372*.

Lin, C.-Y. and Hovy, E. (2003). Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*.

Lin, C.-Y. and Och, F. J. (2004). Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 605. Association for Computational Linguistics.

Ling, W., Dyer, C., Black, A. W., and Trancoso, I. (2015a). Two/too simple adaptations of word2vec for syntax problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1299–1304.

Ling, W., Trancoso, I., Dyer, C., and Black, A. W. (2015b). Character-based neural machine translation. *CoRR*, abs/1511.04586.

Lison, P. and Tiedemann, J. (2016). Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929.

Liu, L., Utiyama, M., Finch, A., and Sumita, E. (2016). Agreement on target-bidirectional neural machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 411–416.

Liu, X., Wong, D. F., Liu, Y., Chao, L. S., Xiao, T., and Zhu, J. (2019). Shared-private bilingual word embeddings for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3613–3622.

Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M., and Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation. *arXiv*, pages arXiv–2001.

Liu, Y., Zhou, L., Wang, Y., Zhao, Y., Zhang, J., and Zong, C. (2018). A comparable study on model averaging, ensembling and reranking in nmt. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 299–308. Springer.

Luong, N. Q. and Popescu-Belis, A. (2016). A contextual language model to improve machine translation of pronouns by re-ranking translation hypotheses. In *Proceedings of the 19th Annual Conference of the European Association for Machine Translation*, pages 292–304.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

Martin, J. H. and Jurafsky, D. (2009). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Pearson/Prentice Hall.

Marton, Y., Chiang, D., and Resnik, P. (2012). Soft syntactic constraints for Arabic-English hierarchical phrase-based translation. *Machine Translation*, 26(1-2):137–157.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*.

Mohamed, E. and Sadat, F. (2015). Hybrid Arabic-French machine translation using syntactic re-ordering and morphological pre-processing. *Computer Speech & Language*, 32(1):135–144.

Moore, R. C. (2004). Improving ibm word-alignment model 1. In *Proceedings of the 42nd annual meeting on association for computational linguistics*, page 518. Association for Computational Linguistics.

Moore, R. C. (2005). A discriminative framework for bilingual word alignment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 81–88. Association for Computational Linguistics.

Moore, R. C., Yih, W.-t., and Bode, A. (2006). Improved discriminative bilingual word alignment. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 513–520. Association for Computational Linguistics.

Nagao, M. (1984). A framework of a mechanical translation between japanese and English by analogy principle. *Artificial and human intelligence*, pages 351–354.

Nallapati, R., Zhou, B., dos Santos, C., Gulcehre, C., and Xiang, B. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290.

Nasrabadi, N. M. (2007). Pattern recognition and machine learning. *Journal of electronic imaging*, 16(4):049901.

Neubig, G., Morishita, M., and Nakamura, S. (2015). Neural reranking improves subjective quality of machine translation: Naist at wat2015. *arXiv preprint arXiv:1510.05203*.

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*, pages 198–206. Association for Computational Linguistics.

Nishimura, Y., Sudoh, K., Neubig, G., and Nakamura, S. (2019). Multi-source neural machine translation with missing data. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:569–580.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics.

Och, F. J., Gildea, D., Khudanpur, S., Sarkar, A., Yamada, K., Fraser, A., Kumar, S., Shen, L., Smith, D., Eng, K., et al. (2004). A smorgasbord of features for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*.

Och, F. J. and Ney, H. (2000). Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 440–447. Association for Computational Linguistics.

Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51.

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. (2019). fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Oudah, M., Almahairi, A., and Habash, N. (2019). The impact of preprocessing on arabic-english statistical and neural machine translation. In Forcada, M. L., Way, A., Haddow, B., and Sennrich, R., editors, *Proceedings of Machine Translation Summit XVII Volume 1: Research Track, MTSummit 2019, Dublin, Ireland, August 19-23, 2019*, pages 214–221. European Association for Machine Translation.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Patterson, J. and Gibson, A. (2017). *Deep Learning: A Practitioner's Approach*. " O'Reilly Media, Inc.".

Petrov, S., Das, D., and McDonald, R. T. (2012). A universal part-of-speech tagset. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, May 23-25, 2012*, pages 2089–2096.

Plank, B., Søgaard, A., and Goldberg, Y. (2016). Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 412–418.

Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. http://is.muni.cz/publication/884893/en.

Riesa, J. and Marcu, D. (2010). Hierarchical search for word alignment. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 157–166. Association for Computational Linguistics.

Rosca, M. and Breuel, T. (2016). Sequence-to-sequence neural network models for transliteration. *CoRR*, abs/1610.09565.

Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

Russell, S. J. and Norvig, P. (2016). *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,.

Ryding, K. C. (2014). *Arabic: A linguistic introduction*. Cambridge University Press.

Sadat, F. and Habash, N. (2006). Combination of Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1–8. Association for Computational Linguistics.

Sadat, F. and Mohamed, E. (2013). Improved Arabic-French machine translation through preprocessing schemes and language analysis. In *Canadian Conference on Artificial Intelligence*, pages 308–314. Springer.

Sag, I. A., Baldwin, T., Bond, F., Copestake, A., and Flickinger, D. (2002). Multiword expressions: A pain in the neck for nlp. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 1–15. Springer.

Sajjad, H., Dalvi, F., Durrani, N., Abdelali, A., Belinkov, Y., and Vogel, S. (2017). Challenging language-dependent segmentation for arabic: An application to machine translation and part-of-speech tagging. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*, pages 601–607.

Salem, Y., Hensman, A., and Nolan, B. (2008). Implementing Arabic-to-English machine translation using the role and reference grammar linguistic model. In *In Proceedings of the Eighth Annual International Conference on Information Technology and Telecommunication (ITT 2008), At Ireland*. Dublin Institute of Technology.

Santos, C. D. and Zadrozny, B. (2014). Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1818–1826.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, 61:85–117.

Schuster, M. and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Sennrich, R. and Haddow, B. (2016). Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 83–91, Berlin, Germany. Association for Computational Linguistics.

Sennrich, R., Haddow, B., and Birch, A. (2015). Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Serban, I. V., Sordoni, A., Bengio, Y., Courville, A. C., and Pineau, J. (2016). Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, volume 16, pages 3776–3784.

Serban, I. V., Sordoni, A., Lowe, R., Charlin, L., Pineau, J., Courville, A. C., and Bengio, Y. (2017). A hierarchical latent variable encoder-decoder model for generating dialogues. In *AAAI*, pages 3295–3301.

Shaalan, K. (2014). A survey of Arabic named entity recognition and classification. *Computational Linguistics*, 40(2):469–510.

Shaalan, K., Rafea, A., Moneim, A. A., and Baraka, H. (2004). Machine translation of English noun phrases into Arabic. *International Journal of Computer Processing of Oriental Languages*, 17(02):121–134.

Shao, Y. and Nivre, J. (2016). Applying neural networks to english-chinese named entitytransliteration. In *Sixth Named Entity Workshop, joint with 54th ACL*.

Shapiro, P. and Duh, K. (2018). Morphological word embeddings for Arabic neural machine translation in low-resource settings. In *Proceedings of the Second Workshop on Subword/Character LEvel Models*, pages 1–11.

Sharma, S., El Asri, L., Schulz, H., and Zumer, J. (2017). Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *CoRR*, abs/1706.09799.

Shirko, O., Omar, N., Arshad, H., and Albared, M. (2010). Machine translation of noun phrases from Arabic to English using transfer-based approach. *Journal of Computer Science*, 6(3):350.

Shu, R. and Nakayama, H. (2017). Later-stage minimum bayes-risk decoding for neural machine translation. *arXiv preprint arXiv:1704.03169*.

Skansi, S. (2018). *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer.

Smith, S. L., Turban, D. H., Hamblin, S., and Hammerla, N. Y. (2017). Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv preprint arXiv:1702.03859*.

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*.

Socher, R., Lin, C. C.-Y., Ng, A. Y., and Manning, C. D. (2011). Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 129–136, USA. Omnipress.

Sokolov, A., Wisniewski, G., and Yvon, F. (2012). Non-linear n-best list reranking with few features. In *Association for Machine Translation in the Americas*.

Soudi, A., Cavalli-Sforza, V., and Jamari, A. (2002). A prototype English-to-Arabic interlingua-based mt system. In *Proceedings of the third international conference on language resources and evaluation: workshop on Arabic language resources and evaluation: status and prospects. Las Palmas de Gran Canaria, Spain*, pages 18–25.

Specia, L., Sankaran, B., and Nunes, M. d. G. V. (2008). N-best reranking for the efficient integration of word sense disambiguation and statistical machine translation. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 399–410. Springer.

Stahlberg, F., Hasler, E., Waite, A., and Byrne, B. (2016). Syntactically guided neural machine translation. *arXiv preprint arXiv:1605.04569*.

Su, K.-Y., Wu, M.-W., and Chang, J.-S. (1992). A new quantitative quality measure for machine translation systems. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 433–439. Association for Computational Linguistics.

Sudoh, K. and Nagata, M. (2016). Chinese-to-japanese patent machine translation based on syntactic pre-ordering for wat 2016. In *Proceedings of the 3rd Workshop on Asian Translation (WAT2016)*, pages 211–215.

Sun, J., Fang, W., Wu, X., Palade, V., and Xu, W. (2012). Quantum-behaved particle swarm optimization: analysis of individual particle behavior and parameter selection. *Evolutionary computation*, 20(3):349–393.

Sun, J., Xu, W., and Feng, B. (2004). A global search strategy of quantum-behaved particle swarm optimization. In *Cybernetics and Intelligent Systems, 2004 IEEE Conference on*, volume 1, pages 111–116. IEEE.

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.

Tan, L. H., Spinks, J. A., Eden, G. F., Perfetti, C. A., and Siok, W. T. (2005). Reading depends on writing, in Chinese. *Proceedings of the National Academy of Sciences*, 102(24):8781–8785.

Tan, X., Chen, J., He, D., Xia, Y., Tao, Q., and Liu, T.-Y. (2019). Multilingual neural machine translation with language clustering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 962–972.

Tong, Y., Wong, D. F., and Chao, L. S. (2016). Exploiting rich feature representation for smt n-best reranking. In *Wavelet Analysis and Pattern Recognition (ICWAPR), 2016 International Conference on*, pages 101–106. IEEE.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics.

Toutanova, K., Suzuki, H., and Ruopp, A. (2008). Applying morphology generation models to machine translation. *Proceedings of ACL-08: HLT*, pages 514–522.

Tromble, R. W., Kumar, S., Och, F., and Macherey, W. (2008). Lattice minimum bayes-risk decoding for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 620–629. Association for Computational Linguistics.

Vapnik, V. (1992). Principles of risk minimization for learning theory. In *Advances in neural information processing systems*, pages 831–838.

Vaswani, A., Bengio, S., Brevdo, E., Chollet, F., Gomez, A. N., Gouws, S., Jones, L., Kaiser, L., Kalchbrenner, N., Parmar, N., Sepassi, R., Shazeer, N., and Uszkoreit, J. (2018). Tensor2tensor for neural machine translation. *CoRR*, abs/1803.07416.

Vijayakumar, A. K., Cogswell, M., Selvaraju, R. R., Sun, Q., Lee, S., Crandall, D., and Batra, D. (2016). Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*.

Virga, P. and Khudanpur, S. (2003). Transliteration of proper names in cross-lingual information retrieval. In *Proceedings of the ACL 2003 workshop on Multilingual and mixed-language named entity recognition-Volume 15*, pages 57–64. Association for Computational Linguistics.

Vogel, S., Ney, H., and Tillmann, C. (1996). Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, pages 836–841. Association for Computational Linguistics.

Wang, P., Qian, Y., Soong, F. K., He, L., and Zhao, H. (2015a). Part-of-speech tagging with bidirectional long short-term memory recurrent neural network. *CoRR*, abs/1510.06168.

Wang, P., Qian, Y., Soong, F. K., He, L., and Zhao, H. (2015b). A unified tagging solution: Bidirectional LSTM recurrent neural network with word embedding. *CoRR*, abs/1511.00215.

Watanabe, T., Suzuki, J., Tsukada, H., and Isozaki, H. (2007). Online large margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 764–773, Prague, Czech Republic.

Williams, R. J. and Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Xia, F. and McCord, M. (2004). Improving a statistical mt system with automatically learned rewrite patterns. In *Proceedings of the 20th international conference on Computational Linguistics*, page 508. Association for Computational Linguistics.

Xiao, T., Zhu, J., and Liu, T. (2013). Bagging and boosting statistical machine translation systems. *Artificial Intelligence*, 195:496–527.

Xiong, D. and Zhang, M. (2016). *Linguistically Motivated Statistical Machine Translation*. Springer.

Yang, N., Li, M., Zhang, D., and Yu, N. (2012). A ranking-based approach to word reordering for statistical machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 912–920. Association for Computational Linguistics.

Zerrouki, T. and Balla, A. (2017). Tashkeela: Novel corpus of arabic vocalized texts, data for auto-diacritization systems. *Data in brief*, 11:147.

Zhang, J., Utiyama, M., Sumita, E., Neubig, G., and Nakamura, S. (2017). Improving neural machine translation through phrase-based forced decoding. *arXiv preprint arXiv:1711.00309*.

Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

Zhang, Y., Zens, R., and Ney, H. (2007). Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. In *Proceedings of the NAACL-HLT 2007/AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 1–8. Association for Computational Linguistics.

Zhang, Z., Wang, R., Utiyama, M., Sumita, E., and Zhao, H. (2018). Exploring recombination for efficient decoding of neural machine translation. *arXiv preprint arXiv:1808.08482*.

Zhou, G., He, T., Zhao, J., and Hu, P. (2015). Learning continuous word embedding with metadata for question retrieval in community question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 250–259.

Ziemski, M., Junczys-Dowmunt, M., and Pouliquen, B. (2016). The united nations parallel corpus v1.0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3530–3534, Portorož, Slovenia. European Language Resources Association (ELRA).