Statistical Machine Translation of the Arabic Language

Walid Aransa

▶ **To cite this version:**

Walid Aransa. Statistical Machine Translation of the Arabic Language. Linguistics. Université du Maine, 2015. English. NNT : 2015LEMA1018 . tel-01316544

HAL Id: tel-01316544
https://theses.hal.science/tel-01316544v1

Submitted on 17 May 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Université du Maine - Ecole Doctorale STIM

Statistical Machine Translation of the Arabic Language

Thèse

présentée et soutenue publiquement le 23 Septembre 2015 au Mans

pour l'obtention du

Doctorat de l'Université du Maine
(spécialité Informatique)

par

Walid Aransa

Membres du jury

Rapporteurs:	Prof. Kamel Smaïli	*Professeur*	LORIA, Université de Nancy 2
	Prof. Nizar Habash	*Professeur*	New York University Abu Dhabi
Directeur de thèse:	Prof. Holger Schwenk	*Professeur*	LIUM, Université du Maine
Co-encadrant:	Dr. Loïc Barrault	*Maître de Conférence*	LIUM, Université du Maine
Examinateur:	Dr. Yaser Al-onaizan	*PhD*	IBM T.J. Watson Research Center

Laboratoire d'Informatique de l'Université du Maine

This thesis is dedicated to
the Egyptian Revolution and the Arab Spring

Acknowledgements

All the praises and thanks be to Allah, who helped me throughout all my life stages and Who helped me to accomplish this work.

I would like to express my gratitude to my thesis supervisor Prof. Holger Schwenk for his guidance, suggestions, support and feedback during my PhD research and during the writing of this thesis. I am deeply indebted to my PhD co-supervisor Dr. Loïc Barrault who provided to me, research guidance, suggestions and support during hard times with a lot of motivation and encouragement. He provided very useful suggestions and feedback during the writing of this thesis.

Beside my supervisors, my sincere thanks also goes to my thesis committee members: Prof. Nizar Habash, Prof. Kamel Smaïli and Dr. Yaser Al-onaizan for their review effort and useful comments and questions during my thesis defense. I am truly grateful to all my jury members for the wonderful defense experience they gave me.

All my thanks to my colleagues in machine translation group in LIUM who have helped me at various stages of my thesis. Special thanks to Huei Chi Lin for the good work that we did together in BOLT project, Haithem Afli for his big effort in helping me in many things including administrative papers for university, apartment and more beside useful research discussions, Fethi Bougares for a lot of research discussions and useful feedback, Mercedes Garca Martnez for her effort in my thesis review and her useful comments, Sahar Ghannay for

all her help especially on the day of my thesis defense, also Alex Ter-Sarkisov for his helpful explanation and review effort. Also thanks to the rest of my colleagues: Mohammed Attik, Kashif Shah, Rahman Ali, Anthony Rousseau, Frédéric Blain, Christophe Servan for their help and all what I learned from them during our work together. Special thanks to Nicolas Coëtmeur who suddenly and sadly passed away in April 2015. Nicolas helped and supported me a lot during my work in CSLM toolkit, he was smart, helpful and professional.

Additionally, I would like to thank Patrik Lambert and David Langlois for their effort in review my PhD progress reports and their useful feedback, suggestions and advices. I would like to record my gratitude to friendly assistance of the kind professors: Dominique Py, Yannick Estve. Also thanks to the academic office and support in LIUM, especially Martine Turmeau, Mélanie Hardy, Etienne Micoulaut and Teva Merlin for their assistance and cooperation.

I would not forget to thank the hidden warriors of my PhD work, my parents and my wife, for their encouragement all the time. I learned a lot from my father and also I can not forget how my mother gives me her support and encouragement when I need them. Also I am grateful to my wife Rihan, whose her believe in me, patience, love helped me to complete this work. I really appreciate this Rihan.

Finally, I can not mention all names of colleagues, friends and family members who helped me in many ways, I am deeply grateful to each one of you, thank you!

Abstract

The Arabic language received a lot of attention in the machine translation community during the last decade. It is the official language of 25 countries and it is spoken by more than 295 million people. Egypt is the largest Arabic speaking country with a population around 90 million. The Egyptian dialect is the main spoken Arabic dialect in Egypt. The interest in Arabic language and its dialects increased more after the Arab spring and the political change in the Arab countries. In this thesis, I worked on improving LIUM's machine translation system for Arabic/Egyptian into English in the frame-work of the BOLT project.

In this thesis, I have extend LIUM's phrase-based statistical machine translation system in many ways. Phrase-based systems are considered to be one of the best performing approaches. Basically, two probabilistic models are used, a translation model and a language model. The translation model is trained on bilingual corpora and is used to model the faithfulness of the translation. The language model is trained on monolingual corpora and is used to improve the fluency of the translation output.

I have been working on improving the translation quality. This is done by focusing on three different aspects. The first aspect is reducing the number of unknown words in the translated output. I concentrate on three types of unknown words. First, words which are not correctly morphologically segmented - this can be corrected by using a better segmentation. Second, the entities like numbers or dates that can

be translated efficiently by some transfer rules. Finally, I have been working on the transliteration of named entities.

The second aspect of my work is the adaptation of the translation model to the domain or genre of the translation task. This is done by weighting different bilingual sub-corpora according to their importance. One technique is weighting of translation models using perplexity optimization. Another way is using a multi-domain translation model architecture. In this architecture, the computation of the translation model probabilities is delayed until decoding time, allowing dynamic instance weighting using optimized weights.

Finally, I have been working on improved language modeling, based on neural network language models, also called continuous space language models. They are used to rescore the n-best translation hypotheses. All the developed techniques have been thoroughly evaluated and I took part in three international evaluations of the BOLT project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, the modern technological advances in communication has turned the world into a small village. It is easy to communicate by phone with any person in any geographical location. It is also possible, using the widely spread mobile devices, to reach any person not only at his address but virtually anywhere. If two persons have an internet connection, beside that they can use text messaging and talking using regular free audio calls, they can also have free video calls if they have a camera device installed. Even though many people now have a mobile phone with 3G or 4G access to the internet, there is still a big communication obstacle between people from different parts of the world. This problem is the language barrier between people speaking different languages. The next mankind hope would be reliable technology that can overcome the language barrier and facilitate the communication between people. This could be instant translation of audio or text from any foreign language to our native language and vice versa.

In the last decade, the need for such automatic translation was driven by the wide spread of the internet and the rapid increase of web content. Many internet users would like to read and have a fair understanding of web sites written in other languages. The continuous increase in the number of users of many internet services like social networks (e.g. Facebook, Google+ and Linkedin), chat and audio/video calls (e.g. Whatsapp and Skype) created a need and a business for automatic translation services. This is because most users prefer to speak, read and write using their own native language. If the user can read in his native

language a web page or a post on Facebook written in another foreign language, this would allow him to communicate effectively in an interactive way. This also means, from service business point of view, more revenue from advertisement and better target audience for the user's native language ads, which means more sales for the advertiser. These great business opportunities were interesting and raised the fund for more machine translation research in big internet companies. Some companies already established an online free automatic translation service like Microsoft Bing (supports 51 languages) and Google translate (supports 90 languages). Facebook integrated an option that allows the user to translate in-place any post written in a different language. They used "Bing" translation service from Microsoft. A Similar option to translate e-mail content is integrated into Gmail, the widely used e-mail service from Google. Another challenge facing these free online automatic translation services is the scalability and the reliability. Due to the interactive nature of such services, internet users expect fast translation and uninterrupted service.

Since early days of computers, scientists tried to build machine translation systems. At that time, they started by focusing on linguistic approaches to address the machine translation problem. They had, with a lot of optimism, the impression that once the vocabulary and the grammar rules are programmed, automatic translation will be an easy task. These approaches use linguistic analysis and generation with different depth. The deeper the analysis, the more abstract is the intermediate representation of the source sentence, which also requires more effort to generate the target sentence from this intermediate representation. The linguistic approach evolved over time, starting from the transfer-based method, to the interlingua method.

Another better approach which makes use of the translations extracted from corpora previously translated by humans is the corpus-based approach. One example of the corpus-based approach is Statistical Machine Translation (SMT), which is based on statistical models trained on bilingual and monolingual corpora. SMT was invented in the IBM Research Lab. Basically, two probabilistic models are used, a translation model which is trained on bilingual corpora and a language

model which is trained on monolingual corpora. SMT has many advantages, it is language independent, easy, cheap and fast to build. Many tools for training and decoding are freely available now. Also the huge bilingual and monolingual corpora needed for training are available for many language pairs. The current state of the art in SMT is the Phrase-based Statistical Machine Translation (PBSMT) because it uses longer translation units than the initial word-based models. By these means, more contextual information is captured by the translation model, which improves the translation quality. It also uses the log linear model which allows the integration of additional features into the model with different weights. The weights are optimized using optimization algorithms.

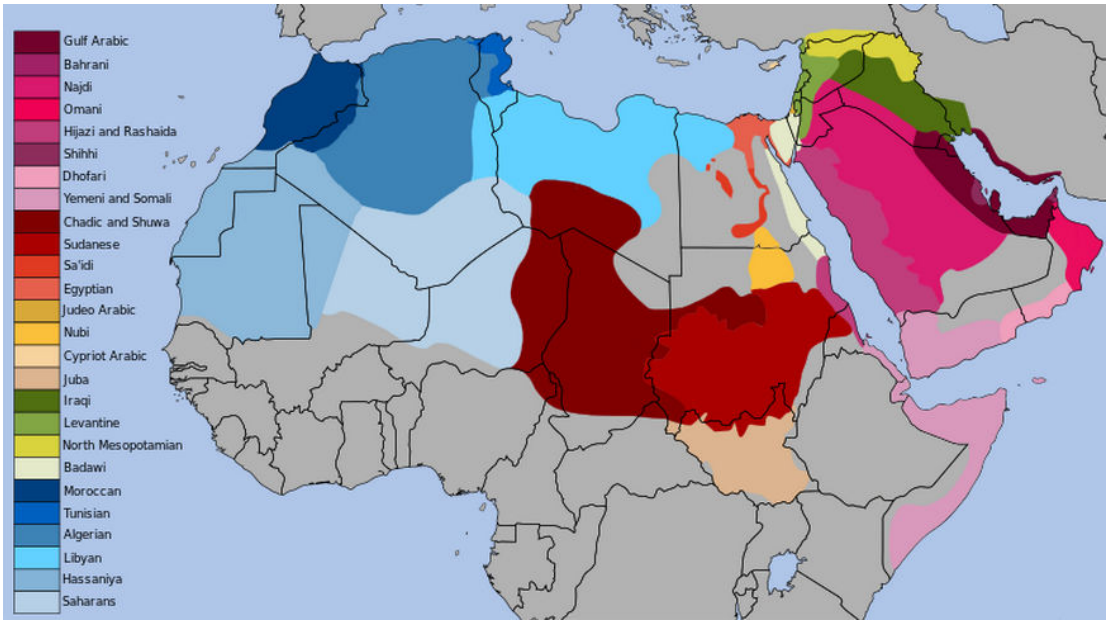

Figure 1.1: Different Arabic dialects in the Middle-East region[1]

The Arabic language received a lot of attention in the machine translation community during the last decade. It is the official language of 25 countries and it is spoken by more than 295 million people. Egypt is the largest Arabic speaking country with a population around 90 million. The Egyptian dialect is the main spoken Arabic dialect in Egypt. A map of different Arabic dialects are shown in

[1]Source: https://en.wikipedia.org/wiki/Varieties_of_Arabic. Image distributed under a CC-BY 3.0 license: http://creativecommons.org/licenses/by/3.0/

Figure 1.1. The interest in Arabic language and its dialects increased more after the Arab spring and the political change in the Arab countries. In this thesis, I worked on improving LIUM's machine translation system for Arabic/Egyptian into English in the frame-work of the BOLT project.

The work in this thesis was part of the Broad Operational Language Translation (BOLT) program funded by Defense Advanced Research Projects Agency (DARPA) in the USA. BOLT focuses on improving machine translation of informal Egyptian dialect and Chinese text into English. In this thesis, I focus only on translation of Egyptian dialect. The following informal text types were in the scope of the project: discussion forums, SMS/chat and conversational telephone speech (CTS) transcription.

The modern standard Arabic (MSA) and the Egyptian dialect have common MT challenges. This is because the Egyptian dialect is a mixture of MSA and additional dialectal words and dialectal structure. Egyptian dialect shares many words, features and grammar with MSA. For examples, missing short vowels, the clitics and the sentence structure . Additionally, the Egyptian dialect has its own special attributes. They can be divided into two categories: general and writing specific. The general category includes: more flexible sentence structure for example the sentence انا مرحتش معاه (i.e. I did not go with him) has a different word order than its equivalent in MSA انا لم أذهب معه. Another attribute is that Egyptian dialect has different or additional morphological forms for some words like مرحتش (i.e. I did not go) which has no equivalent word in MSA. Also Egyptian has different inflection compared to MSA like ماكلتش (i.e. She did not eat) which in MSA لم تأكل. It also replaces some letters by others for sake of easy pronunciation like replacing ث by ت in the MSA word ثلاثة (i.e. three) to be تلاته or ضin ضابط (i.e. officer) to be ظابط and adding additional letters to the MSA word like adding additional alef ا in رجل (i.e. man) to be راجل and in معه (i.e. with him) to be معاه.
The writing specific category includes: various orthographic forms of the same

word due to lack of a standard writing like حيسوق (i.e. he will drive) and هيسوق or معكاش (i.e. you do not have anything) and معكش; a high rate of orthographic mistakes, letter repetitions like راااائع (i.e. wonderful); and omitting of some punctuations and some letters' dots like in كوبرى (i.e. bridge) instead of كوبري; and using of additional vocabulary which are not in MSA ست (i.e. woman), ياريت (i.e. I hope), زي (i.e. like). Some of these characteristics cause the training data to be more sparse or introduce more ambiguity.

In addition to the MSA and Egyptian dialect challenges in MT, there are general MT challenges. One of these challenges is that some words are not translated by the SMT system because they are Out-Of-Vocabulary (OOV) words. One way to deal with OOV words is to automatically identify and transliterate proper nouns. Transliteration is the process of writing a word (mainly proper nouns) from one language in the alphabet of another language. This requires mapping the pronunciation of the word from the original language to the closest possible pronunciation in the target language. Since I am using a statistical approach throughout this thesis, I will need data to train the system. In this case, the training data should be a bilingual list of names in Arabic and English. Since we do not have this training data available, we have to deal with the automatic extraction of this parallel list of names from the available corpora. This is called transliteration mining.

Another challenge is the adaptation of SMT systems to the Egyptian dialect. The available training corpora, in the context of BOLT program, contain MSA, Egyptian, Levantine and Iraqi dialects. One way to benefit from such heterogeneous training corpora is treating different dialects as different domains. This

is done by weighting different translation models according to their importance using perplexity optimization. One of the disadvantages of this technique is that we can adapt the system either to MSA or to the Egyptian dialect but not both together. To overcome this disadvantage, I experimented with a multi-domain translation model architecture. This architecture delays the computation of the translation model features until decoding, allowing dynamic instance weighting using optimized weights from multiple domains (i.e. MSA and Egyptian dialect in our case).

Besides adapting the SMT system to the Egyptian dialect and different genres, I also addressed the translation of ambiguous (i.e. with different meanings) Arabic/Egyptian words. This is achieved by applying a word sense disambiguation (WSD) technique on ambiguous words. I used this technique to help the phrase-based SMT system to better translate ambiguous words.

Finally, another challenge is improving language modeling which plays an important role in MT. It is today acknowledged that neural network language models, also called continuous space language models (CSLMs) outperform n-gram language models. However, CSLMs are usually not used in SMT decoding because of high the computational complexity. CLSMs are usually used to rescore the n-best list of hypotheses. One possible way to improve CSLM is by providing additional information at the input of the neural network. For example, this additional information can be used to train a topic-conditioned CLSM. I experiment with different types of auxiliary features including line length, text genre, vector representations of multiple lines, ... etc. By these means, better domain and context specific LM estimations can be obtained.

1.1 Scientific goals and objectives

The main aim of this PhD thesis is to improve a state of the art PBSMT system of informal Egyptian into English for the three genres in the scope of BOLT program by applying new approaches and techniques.

The following are the main scientific objectives for this work:

- Development and improvement of a PBSMT system for BOLT program.
- Adaptation the PBSMT system on Egyptian dialects and different genres by applying domain adaptation techniques.
- Development of multi-domain (i.e. MSA and Egyptian dialect) dynamic adaptation technique to build a dialect independent PBSMT system.
- Reduction of the number of OOVs in the translated output using different techniques targeting different type of OOVs. The concentration was on three types of unknown words, words which are morphologically segmented incorrectly, entities like numbers or dates and proper nouns.
- Integration of new features and techniques from other disciplines like neural networks, word sense disambiguation into the baseline PBSMT system to improve the translation quality.
- Evaluation of our improvements in the yearly BOLT program evaluation as well as in other international evaluation campaigns like OpenMT organized by National Institute of Standard and Technology (NIST).

1.2 Research contributions

The contributions of this thesis are as follows:

- A novel transliteration mining algorithm using bilingual and monolingual corpora. The results of the transliteration mining is partitioned based on the origin of the name (either from Arabic or English origin) and then used to train a forward and backward transliteration system. These transliteration system can be used to decrease the number of OOVs by transliterate proper nouns.
- A novel CSLM architecture which using additional information at the input of the neural network. This is used to train an auxiliary feature conditioned CLSM. By these means, better domain and context specific LM estimations

can be obtained. The architecture is evaluated using different types of auxiliary features including line length, text genre, vector representations of multiple lines, ... etc..

- Development of dialect independent PBSMT system by using an architecture that delays the computation of the translation model features until decoding, allowing dynamic instance weighting that uses optimized weights from multiple domains (i.e. MSA and Egyptian dialect).
- Evaluating recent well established methods and techniques in the literature by applying them in the context of BOLT program and report the best practices on using them.

1.3 Outline of the thesis

The thesis consists of 6 chapters which are organized as follows: Chapter 2 gives an introduction to machine translation. Chapter 3 covers the work I did in the BOLT program. The details of the work I did in transliteration and transliteration mining is presented in Chapter 4. The improvement of CSLM is presented and discussed in Chapter 5. Finally, the conclusion and future work is presented in Chapter 6.

Chapter 2

Introduction to Machine Translation

In this chapter, I will give a general introduction to machine translation (MT), its history and approaches. I will focus more on statistical machine translation (SMT) since it is the basis of my work in this thesis. I will cover different components of word-based and phrase-based SMT, including the translation model (TM) and the language model (LM). For the language model, I will give a brief introduction to n-gram back-off and neural network language models. Decoding, MT metrics and evaluation will also be covered. The last section of this chapter will give an overview on the challenges of translating the Arabic language and the Egyptian dialect since this is the focus of the experiments in this thesis.

2.1 Machine translation history

Automatic translation, or machine translation as it is generally known, is the attempt to automate all, or part of the process of translation from one human language to another [Arnold et al., 1993].

The motivation behind MT is the ability of fast translation of text or audio from one language into another language regardless of the availability of human translators. MT would also break the language barrier between people. For ex-

ample, currently, online MT services (e.g. Google translate or Microsoft Bing) provide a translation of a text of various quality that allows the users to have a fairly good understanding of the content. MT can also provide an initial draft translation to human translators who have to review and post-edit it. This can decrease the human translation time, effort and hence cost.

Computers were used during the second world war in Britain to break the German Enigma code by considering it as coded English and decode it. This decoding seemed like an apt metaphor for machine translation. From these early days, the view was optimistic and even over-promising researches were going on. For example, in 1954, the Georgetown university and IBM developed jointly an experiment to demonstrate a machine translation system. The experiment involved the automatic translation of about sixty Russian sentences into English. It was claimed that within three to five years the MT problem will be solved. A good amount of funding was provided to machine translation researches around the world guided by these optimistic goals. Many approaches were explored from direct translation with some basic transfer rules to more complex interlingua approaches that use an abstract semantic representation.

In 1966, the Automatic Language Processing Advisory Committee (ALPAC) report was issued, which had a negative impact on MT research funding and almost caused a stop of funding from US agencies. Before it, there were many hopes in the MT research community which had unrealistic targets for possible progress and the ability of machine translation systems at that time. The report basically showed that the cost of machine translation or post editing of automatic translation is higher than human translation. The report observed that there is no shortage of human translators, as well as no big demand of translation of the Russian scientific literature. The report suggested that there is no advantage of using machine translation systems over human translation and recommended to direct the funding to basic linguistic research.

Even though funding was sharply reduced in the USA, research in Europe and other countries continued with funding from the government and commercial

companies. For example, the Systran company was founded in 1968. It developed a Russian-English MT system that was used by the US Air Force since 1970. The university of Montreal developed a fully functional MT system for weather forecasts called Météo which has been used since 1976.

The development of rule-based MT continued during the 80s and 90s. For example, Carnegie Mellon University developed the CATALYST system that use interlingua to represent the sentence meaning in a language-independent form. Other systems were developed by universities (e.g. Pangloss which was developed by the New Mexico State University, the University of Southern California, and CMU).

In 1988, at the second Theoretical and Methodological Issues (TMI) in machine translation conference at Carnegie Mellon University, a new era of MT started when IBM's Peter Brown and his colleagues presented an approach to MT which was purely statistical [Brown et al., 1988], inspired by successes of similar work in speech processing. At that time, most researches were focused on syntax-based and interlingua approaches. The statistical approach started to get more interest during 1990s. This was facilitated by various free tools which implement IBM methods. By 2000, many statistical machine translation researches were on-going by many projects. This was motivated by more funds especially from Defense Advanced Research Projects Agency (DARPA), which is a leading funding agency in the US. DARPA showed great interest in statistical approach for MT and funded large projects: TIDES and Global Autonomous Language Exploitation (GALE). Now, many universities and companies (like Google, IBM, Microsoft and Facebook) are developing statistical machine translation systems. A periodically NIST evaluation workshop is organized by NIST in order to exchange ideas and latest developments and measure progress in the MT field. Today, statistical machine translation represents the state-of-the-art. SMT and other data-driven approaches are widely used because of the increase in computing power and the availability of free tools and resources.

Recently, other approaches are proposed like using neural network based machine translation [Bahdanau et al., 2014; Sutskever et al., 2014], which could be competitive and promising.

2.2 Machine translation approaches

We will divide the MT approaches into linguistic and corpus-based approaches.

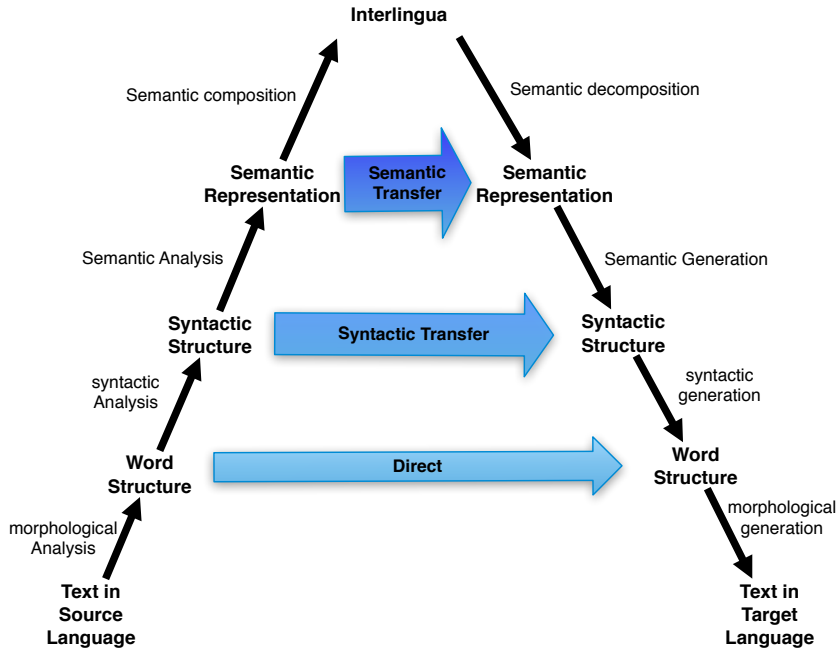

Figure 2.1: The Vauquois triangle for MT [Vauquois, 1968]

2.2.1 Linguistic approach

This approach uses linguistic analysis and generation with different depth. The deeper the analysis, the more abstract is the intermediate representation of the source sentence, which also requires more effort to generate the target sentence from this intermediate representation.

The linguistic approach evolved over time, starting from the transfer-based method, to the interlingua method as shown by the Vauquois triangle in Figure 2.1. Each method is explained in brief in the following sections.

2.2.1.1 Transfer-based method

In direct-transfer translation, the source language words are translated word by word using a bilingual dictionary to the target words. Reordering is performed on the translation output using simple syntactic rules (e.g. move adjective after noun). As shown in Figure 2.1, direct-transfer uses a morphological analysis of words and a complex bilingual dictionary, as well as some simple reordering rules. There is no deep analysis of the source sentence nor complex generation rules for the target translation. This gives fair translation for simple sentence structures if used between languages which are syntactically and semantically close.

In the higher transfer-based method a complex linguistic analysis and generation can be used during translation. This consists of three steps: analysis, transfer and generation [Arnold et al., 1993]. The first step is to perform deeper analysis of the source language text which can be syntactic and/or semantic. In the second step, a transfer from the source sentence syntactic/semantic representation to the target language representation is performed using mapping rules. Finally, a generation of the target sentence from the mapped representation is performed.

Usually this analysis requires a special syntactic parser that only focuses on differences between the source and target language in order to facilitate the mapping step. For the syntactic transfer, several types of transfer rules will be required: syntactic and lexical. The first one will be used to map the sentence syntactic representation from the source language into the target language [Jurafsky and Martin, 2000], while the second one is needed to select the correct word-to-word translation using a bilingual dictionary that could deal with lexical ambiguity. It is possible to resolve lexical ambiguity by performing word sense disambiguation during the source language analysis phase. Semantic transfer can be used to deal with semantic roles in the sentence structure.

2.2.1.2 Interlingua method

As seen in the previous section, the transfer-based method involves source and target language-dependent rules for lexical, syntactic and semantic transfer. If we

want to translate between more than two languages, we have to write a distinct sets of transfer rules for each language-pair. The simple idea of interlingua is to represent the source sentence in a language-independent abstract concept representation that can be generated from any source language, and which is also used to generate the sentence in any target language. This universal representation is called interlingua. As shown in Figure 2.1, more effort is needed to perform the analysis to get the interlingua representation as well as to generate the translation output in the target language than for the other methods below in the pyramid.

One of the advantages of the interlingua method is that it would be easy to support translation from a new source language. This will only require building the analysis modules to get the interlingua representation, then the system will be able to generate the translation from this new source language into all already supported target languages. In this method, there is no need for lexical transfer rules since interlingua is an abstract representation that represents source words in a disambiguated semantic form, that can be used to generate the correct translation just by using the target language generation module. Since interlingua requires deep concept and semantic analysis, it is usually used in simple domains like weather forecast, hotel reservation or air travel domains. One example of such a system is the CATALYST project at Carnegie Mellon University (CMU). It was used to translate technical manuals and documentation at the Caterpillar Tractor company.

2.2.2 Corpus-based approach

Corpus-based approaches are using translations extracted from corpora previously translated by humans. Typical examples of of corpus-based approaches are:

- Example-based MT:
  This method was motivated by the way human translators work when using a bilingual dictionary. The system searches in the parallel corpora to find the closest source example to the source phrase. Finding the best match for a source phrase can involve calculating the closeness to various stored

examples. Target phrases in the correspondence translation examples are extracted and combined to generate the target sentence. This is done based on the probability of the source phrase's alternative translations. More details on example-based MT approach can be found in [Somers, 1999]

- Statistical machine translation:
  Another method of a corpus-based approach is Statistical Machine Translation (SMT), which is based on statistical models trained on bilingual and monolingual corpora. SMT was invented in the IBM Research Labs by [Brown et al., 1990] after the success of using statistical methods in speech recognition in the late 80s. Basically, two probabilistic models are used, a **translation model** which is trained on bilingual corpora and is used to estimate the probability that the source sentence is a translation of the target sentence and a **language model** which is trained on monolingual corpora and is used to improve the fluency of the output translation.

  SMT uses conditional probability theory to find the translation t of the source sentence s that has maximum conditional probability $P(t|s)$. Bayes rule is applied to invert the translation direction to $P(s|t)$ and to integrate a language model P(t). If $s = s_1, \ldots, s_j, \ldots, s_{l_s}$ is the source sentence with length l_s and $t = t_1, \ldots, t_i, \ldots, t_{l_t}$ is the target sentence with length l_t. The best translation t_{best} is the one that has maximum probability using noisy channel model as shown in Equation 2.1.

$$t_{best} = \arg\max_t P(t|s) = \arg\max_t \overbrace{P(s|t)}^{Translation\ Model} \times \overbrace{P(t)}^{Language\ Model} \tag{2.1}$$

  Until today, SMT is widely used and still obtains state-of-the-art results for many language pairs. Since it is the method used in this thesis, I will explain it in more details in Section 2.3.

2.3 Statistical machine translation

SMT has many advantages, it is language independent, easy, cheap and fast to build. Many available tools for training and decoding are freely available. Also SMT training data are available as huge bilingual and monolingual training corpora in many languages. A list of these corpora and tools can be found at http://www.statmt.org.

SMT treats the translation problem as a machine learning problem. It learns how to translate by means of learning a translation model from many examples of human translation (i.e. training corpora). The best translation is the one that has the maximum probability using noisy channel model as shown in Figure 2.2.

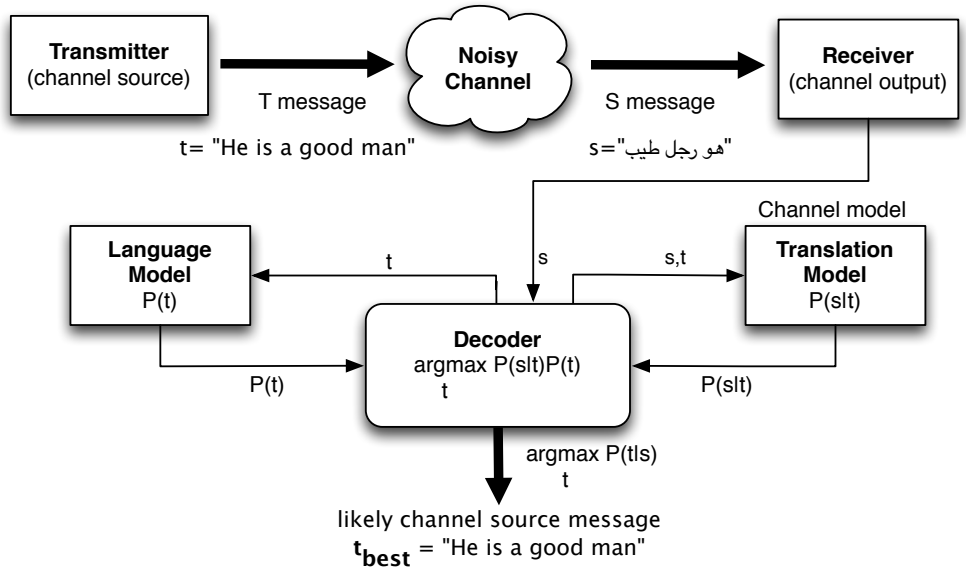

Figure 2.2: Using of the noisy channel model in SMT

The fundamental equation of statistical machine translation is Equation 2.1, which consists of two components, the translation model $P(s|t)$ and the language model $P(t)$. According to this equation, we need to calculate the **reverse** translation probability $P(s|t)$. Maximizing the reversed translation probability

component tries to ensure that the output translation t_{best} corresponds semantically to the source sentence s. While maximizing the language model component ensures that the generated translation is grammatically correct, fluent and commonly used. The process of finding this best translation is called **decoding** and it is performed by a component called the **decoder**. Several decoding algorithms have been used, I will give more information on the decoding process and decoding algorithms in Section 2.3.4.

2.3.1 Word-based translation models

According to Equation 2.1, the inverse translation probability $p(s|t)$ is needed. Many techniques have been developed to calculate it from bilingual corpora. In order to simplify the presentation of these methods, we will assume that we want to calculate $p(t|s)$ where s is the source and t is the target sentence.

Although, the word-based translation model is not the current state-of-the-art, it provides the basis for most current statistical machine translation methods. The IBM models were originally the result of the work at the IBM Watson Research center in the context of Candide project in the early 1990s. Brown et al. [1990] proposed five generative models to calculate the translation model probability $p(t|s)$. These generative models are used to generate a number of different translations for a sentence, each with different probability.

Practically, the translation model cannot be calculated directly by collecting sentences statistics due to sparseness, instead it could be calculated indirectly by decomposing the sentence into a sequence of words, then collect the needed statistics to estimate the probability distribution. The IBM models propose algorithms for estimating the probability that a word in the source sentence will be a translation of a particular word in the target sentence [Brown et al., 1990]. Once such probabilities are estimated they can be used together to align the words in a target sentence with the words in the corresponding source sentence. An example of the alignment of Egyptian Arabic and English sentences is shown in Figure 2.3.

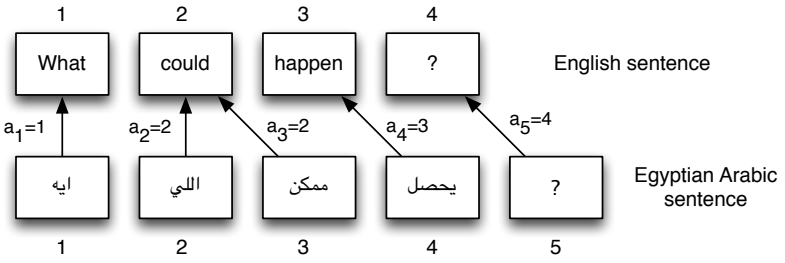

Figure 2.3: A visualization of an alignment between English and Egyptian sentences

Word alignment:

IBM models are defined over a hidden alignment variable a which captures the word-level correspondences between s and t. The conditional probability $p(t|s)$ is expressed as a sum of the probabilities of hidden alignments a between s and t as follows [Brown et al., 1990]:

$$P(t|s) = \sum_{a} P(t, a|s) \tag{2.2}$$

where a is a vector of alignment positions a_i for each word t_i in t.

This word alignment is a mapping function for each sentence pair, which maps each word in the translated sentence at position i to a word at position j in the source sentence $a : i \rightarrow j$.

This alignment function is mapping each source word position to one target word position. So it is not possible to have one-to-many or many-to-many alignments, but many-to-one.

It is normal that in some languages, words in the source sentence have no translation and hence are not aligned to any word in the target sentence. In this case the alignment model will learn to drop such words during translation.
To fully define the alignment function, we need to assign an alignment index for all words in the target sentence. An additional word $s_0 = NULL$ is added to the

source sentence (usually at index 0) which is used as a mapping index for each target word that does not align to any source word (called spurious words). This allows the alignment model to give an alignment position for each target word, even those which are not a direct translation of any word in the source sentence.

2.3.1.1 The five IBM generative models

Brown et al. [1990] proposed five generative models (named IBM model 1 until IBM model 5), each model improves its predecessor by adding or reinterpreting parameters. During training, the Expectation Maximization (EM) algorithm [Dempster et al., 1977] is used to estimate the hidden parameters by maximizing the likelihood probability of the bilingual training corpus which is considered as a set of independent sentence pairs. Two of the widely used toolkit that implements IBM models is GIZA++ [Och and Ney, 2003b] and MGIZA++ [Gao and Vogel, 2008].

2.3.1.2 Hidden Markov Model (HMM), IBM models 1 and 2

These three models are used to estimate the alignment using the lexical translation probability distribution $P(t_i|s_{a_i})$, which is calculated using the count of co-occurrences of aligned word pairs in the bilingual training corpus. All the three models are using the following decomposition equation for $P(t, a|s)$:

$$P(t, a|s) = \prod_{i=1}^{l_t} P(t_i|s_{a_i}) P(a_i|a_{i-1}, i, l_t, l_s) \tag{2.3}$$

where a is a vector of alignment positions, $a_i = j$ for the word t_i in t. The difference between how the three models parameterize the alignment is shown in the following equation:

$$P(a_i|a_{i-1}, i, l_t, l_s) = \begin{cases} \frac{\epsilon}{(l_s+1)^{l_t}} & \text{IBM 1} \\ P(a_i|i, l_s, l_t) & \text{IBM 2} \\ P(a_i - a_{i-1}) & \text{HMM} \end{cases} \tag{2.4}$$

In **IBM model 1**, all alignment are equally likely, so the lexical translation probability is normalized by the source sentence length (including the additional virtual NULL word). In **IBM model 2**, the model depends on the position of the aligned words. In the **HMM model**, the model depends on the shift of the current aligned word position from the previous aligned word position. It is clear that IBM model 1 lacks the ability to model word reordering.

Since the alignment is hidden and unknown, the estimation of the lexical probabilistic model is a kind of incomplete data problem. In machine learning, the incomplete data problem is addressed using the EM algorithm. The EM algorithm is an iterative algorithm that fills the gaps in the data, then trains the model in alternating steps. In summary EM will start with uniform lexical probabilities (i.e. initially the alignments will be equally likely). In the following iterations, EM will use co-occurrence counts of each word pair to learn better lexical probabilities. EM keeps doing this until convergence to good lexical probabilities.

The EM algorithm works as follows:

1. Initialize the model with some lexical translation probability distribution. Uniform distribution can be used.
2. Expectation step: collect sentence level co-occurrence counts of each word pair in the training aligned corpus.
3. Maximization step: re-estimate the lexical translation probabilities based on the new counts.
4. Loop though step 2 and 3 until convergence.

For each iteration, the perplexity is used to evaluate and determine the convergence of the EM algorithm, which will decrease at every iteration. It is calculated

as follows:

$$\log_2 PP = -\sum_s \log_2 p(t_s|s_s) \tag{2.5}$$

For IBM model 1, the EM training is guaranteed to converge to the global minimum, while for IBM Model 2 and HMM model, it will converge to local minimum.

2.3.1.3 IBM models 3, 4 and 5

IBM models 1 and 2 and the HMM model are generative models, which focus on the words in the source sentence to calculate the lexical translation probabilities, while in IBM models 3, 4 and 5, the generative models are focusing on the target sentence, first by choosing the source word fertility (i.e. the number of connections with target words), then the identity of these target words, and finally their position in the target sentence.

IBM Model 3, models the fertility and the NULL tokens insertion. The fertility parameter $P(\phi|s_j)$ is incorporated, where ϕ is the number of target words aligned to the source word s_j. Dropping of source words during translation can be modeled by $\phi = 0$ which is $P(0|s_j)$. The NULL insertion is modeled as a special step after the fertility step, where NULL token is inserted with the probability $p1$ and not inserted with the probability of $p0 = 1 - p1$. Lexical translation is handled using the conditional probability distribution $P(t_i|s_{a_i})$ as in IBM model 1. Distortion is modeled the same way as in IBM model 2 with the probability distribution $P(i|j, l_t, l_s)$.

IBM model 4 provides more improvement over IBM model 3. Since the distortion parameters of model 3 can not realistically be estimated for long source and target sentences due to data sparseness, they are replaced with relative distortion parameters. In this model, the placement of the target translation of a source word is based on the placement of the translation of the preceding source word.

IBM model 5 fixes the deficiency problem in IBM models 3 and 4. The deficient problem happens because in these two models multiple target words can be placed in the same position. Model 5 fixes this problem by keeping track of the available target word positions and allows placement only into these positions.

Limitations of the IBM models:

IBM models have several limitations: they can align each target word to one source word only, while many to many alignments are needed to translate expressions and idioms. Also they do not use any context information to estimate the translation probabilities. These limitations have been overcoming in the phrase-based translation model which is explained in the following section.

2.3.2 Phrase-based translation models

Phrase-based models use longer translation units. If the translation unit is larger than one word, more contextual information is captured by the translation model which leads to better word selection from different translation candidates. This multi-word translation unit is called a **phrase**, however it is not linguistically motivated.

Phrase-based models uses more simple and accurate re-ordering technique which

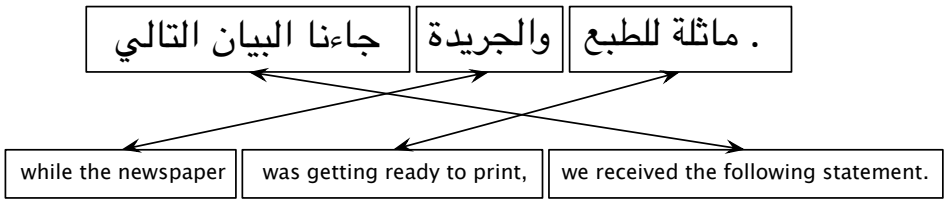

Figure 2.4: Example of Arabic-English aligned phrases

handles phrases instead of words. For example, this can help in local reordering of adjective-noun expressions. IBM models still have a central role in the phrase-based translation models due to their ability to estimate good word-alignment

which is a key step in phrase-based translation models training as we will see in next sections. An example of aligned phrases is shown in Figure 2.4. A graphical example of the word-alignment is shown in Figure 2.5.

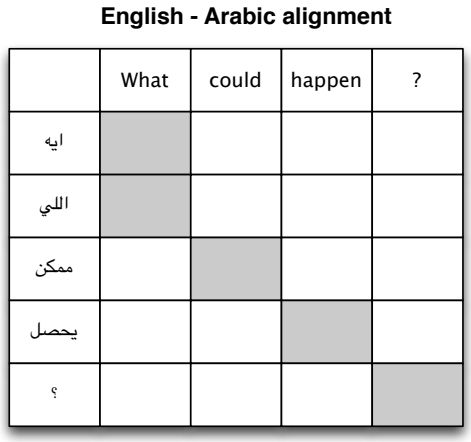

Figure 2.5: A graphical word-alignment

2.3.2.1 Phrase pair extraction

In order to extract phrases during training, IBM models are used to generate word-level alignments, which are used to extract aligned phrase-pairs. The first step is performing asymmetric alignment of the bilingual corpus in both source to target and target to source directions. The second step is getting a high-precision alignment and a high-recall alignment by using the intersection and the union of both alignments respectively. Using heuristics, we start with the high-precision alignment points and add additional alignment points. The phrase extraction is performed by looping over all possible phrases of the target sentence and finding the minimal source phrases that match each of them [Koehn et al., 2003]. Several conditions should be considered while extracting phrase pairs:

1. All alignments points between the phrase-pair should be included. This is because if the extracted phrase-pair contains a word that is translated to two or more words, these words should be included in the target phrase.

Such phrase-pairs are called consistent phrase-pairs as shown in Figure 2.6.

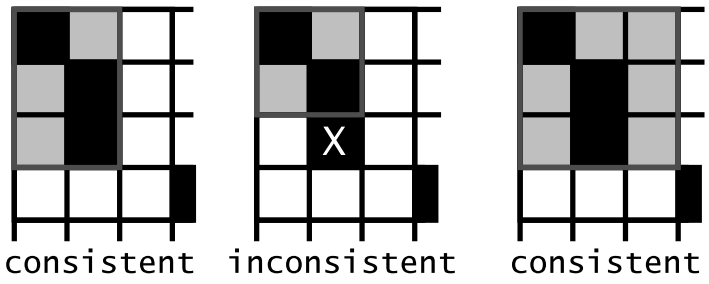

Figure 2.6: Consistent and non-consistent phrase-pairs (from [Koehn, 2010]).

2. Any extracted phrase pair should contain at least one alignment point.

3. More phrase-pairs can be extracted by by including more unaligned words near its boundaries.

The alignment we explained in Section 2.3.1.1 is called asymmetric alignment because it is restricted to map each output word to only one input word. In order to overcome this problem, a method called symmetrizing is used. The symmetrizing method consists of: train the alignment in two directions, source-to-target and target-to-source directions separately to get two alignment matrices, then combine these two alignment matrices. One way to combine them is to take the intersection of them to get the alignment points that exists in both of them (i.e. the high-precision alignment) as shown in Figure 2.7. A phrase-pairs extraction can use this high-precision alignment matrix to extract consistent phrase-pairs.

2.3.2.2 Phrase-based translation model

If the source sentence s is broken up into I phrases, the reverse translation model $P(s|t)$ is calculated as follows:

$$P(s|t) = \prod_{i=1}^{I} \phi(\bar{s}_i|\bar{t}_i) d(a_i - b_{i-1} - 1) \tag{2.6}$$

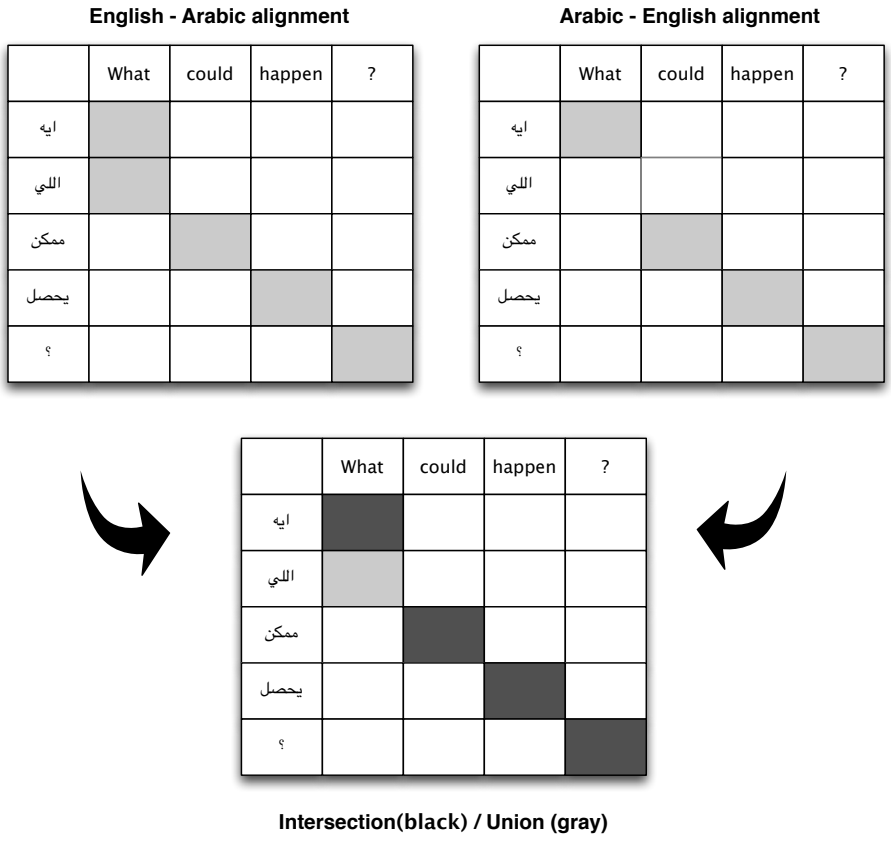

Figure 2.7: A visualization of symmetrization of IBM alignments by taking the intersection of source-to-target and target-to-source alignments to get a high-precision alignment, the union of both alignments is used to extract phrases.

The first part $\phi(\bar{s}_i, \bar{t}_i)$ in Equation 2.6 is the phrase translation probability that the phrase $\bar{s}_i$ is the translation of the phrase $\bar{t}_i$. It is modeled as a translation from target to source and is calculated by collecting the counts from the training data as follows:

$$\phi(\bar{s}_i|\bar{t}_i) = \frac{count(\bar{s}_i, \bar{t}_i)}{\sum_{\bar{s}} count(\bar{s}, \bar{t}_i)} \tag{2.7}$$

The second part is a **distance-based reordering model**. a_i is the start position of the source phrase which is the translation of the target phrase i, and b_{i-1} is the last word in the previous phrase. Hence reordering distance is calculated as $(a_i - b_{i-1} - 1)$. The distortion function can be $d(a_i - b_{i-1} - 1) = \alpha^{|a_i - b_{i-1} - 1|}$. which will penalizes large distortion by giving them lower probability. Equation 2.6 is considered to be the calculation of the translation model for standard phrase-based SMT. However phrase-based translation system usually uses log-linear model, since it allows using more features instead of just using translation model and language model probabilities as in noisy-channel model. We will cover log-linear model in more details in Section 2.3.2.3.

2.3.2.3 Log-linear models

As we saw before, the standard phrase-based model has two components, the translation model and the language model. However the translation model actually can be split into two models, the phrase-translation model and the distortion or reordering model. Using the noisy-channel model Equation 2.1 and the reverse translation model $P(s|t)$ Equation 2.6, we can get the translation output as follows:

$$t_{best} = \arg\max_{t} \prod_{i=1}^{I} \phi(\bar{s}_i|\bar{t}_i) d(a_i - b_{i-1} - 1) P(t) \tag{2.8}$$

This equation is actually a multiplication of the phrase translation model, the reordering model and the language model, all getting the same uniform weight which is 1. It would be better to give different weight for each model as in the following equation and then find a way to calculate the best weights.

$$t_{best} = \arg\max_{t} \prod_{i=1}^{I} \phi(\bar{s}_i|\bar{t}_i)^{\lambda_\phi} d(a_i - b_{i-1} - 1)^{\lambda_d} \prod_{i=1}^{|t|} P(t_i|t_1...t_{i-1})^{\lambda_{LM}} \tag{2.9}$$

where $(\lambda_\phi, \lambda_d, \lambda_{LM})$ are the weights that can be chosen for the contribution of each model.

if $h_1 = \log \prod_{i=1}^{I} \phi(\bar{s}_i|\bar{t}_i) = \sum_{i=1}^{I} \log \phi(\bar{s}_i|\bar{t}_i)$,
and $h_2 = \log \prod_{i=1}^{I} d(a_i - b_{i-1} - 1) = \sum_{i=1}^{I} \log d(a_i - b_{i-1} - 1)$,
and $h_3 = \log \prod_{i=1}^{|t|} P(t_i|t_1...t_{i-1}) = \sum_{i=1}^{|t|} \log P(t_i|t_1...t_{i-1})$
we will get

$$t_{best} = \arg\max_{t} exp(\lambda_\phi h_1 + \lambda_d h_2 + \lambda_{LM} h_3) \tag{2.10}$$

Assume that $n = 3$, $\lambda_1 = \lambda_\phi$,$\lambda_2 = \lambda_d$, $\lambda_3 = \lambda_{LM}$, in Equation 2.10 we will get the following:

$$t_{best} = \arg\max_{t} exp \sum_{i=1}^{n} \lambda_i h_i(s, t, a, b) \tag{2.11}$$

which is using the basic form of a log-linear model:

$$p(x) = exp \sum_{i=1}^{n} \lambda_i h_i(x) \tag{2.12}$$

Using a log-linear model gives us two advantages over the noisy-channel model. First, we can give different weights to each component model. The second advantage is that one can add more component models, also called feature functions. Usually the weights in a log-linear model are optimized using Minimum Error Rate Training (MERT) to maximize the overall system translation quality using a translation evaluation metric [Och, 2003]. I will explain MERT in more details in Section 2.3.6. The following are the common used feature functions in the state-of-the-art phrase-based systems:

- LM probability.
- Bidirectional (i.e. source to target and target to source) phrase translation

probabilities.

- Bidirectional lexical probabilities.
- Phrase reordering model.
- Word/phrase penalty.
- Operation Sequence Model features.

2.3.3 Language models

An LM is an important component in many natural language processing tasks. In SMT, the LM is responsible of the fluency of the translation output as a feature function in the log-linear model in Equation 2.11. The LM is trained on a monolingual corpus in order to be able to estimate the probability of a sequence of words. In the next sections, I will cover the n-gram LM, neural network LM and the evaluation of LMs using perplexity.

2.3.3.1 N-gram language models

The joint probability a $P(w_1, \ldots, w_m)$ of a sequence of words $w_1, \ldots, w_m$ is computed using the chain rule as a multiplication of the conditional probabilities of each word w_i as shown in Equation 2.13.

$$P(w_1, \ldots, w_m) = \prod_{i=1}^{m} P(w_i | w_1, \ldots, w_{i-1}) \tag{2.13}$$

Using a Markov chain, this can be approximated by limiting the history of the preceding words to $n-1$ words as in the following equation:

$$P(w_1, \ldots, w_m) \approx \prod_{i=1}^{m} P(w_i | w_{i-(n-1)}, \ldots, w_{i-1}) \tag{2.14}$$

This is called n-gram LM with order n. An n-gram LM estimates the conditional probability for a word given the previous $n-1$ words. The words' conditional probabilities are multiplied to estimate the joint probability of the whole sentence.

If $n = 1$, the n-gram is called a unigram, if $n = 2$, the n-gram is called a bigram and if $n = 3$ the n-gram is called trigram.

The n-gram conditional probability is estimated using Maximum Likelihood Estimation (MLE) by collecting frequency counts as follows:

$$P(w_i|w_{i-(n-1)}, \ldots, w_{i-1}) = \frac{count(w_{i-(n-1)}, \ldots, w_{i-1}, w_i)}{count(w_{i-(n-1)}, \ldots, w_{i-1})} \tag{2.15}$$

One major problem in estimating the n-gram model using MLE is the fact that many possible n-grams are not observed in the training data. This can lead to zero probability (numerator is zero) or an undefined value (denominator is zero). Many smoothing techniques have been proposed in the literature (e.g. add-one smoothing, Laplace Smoothing, Good-Turing Discounting or Kneyser-Ney smoothing). A good overview of $n-$gram smoothing techniques is presented in [Chen and Goodman, 1996]. In the following sections I will cover LM interpolation and back-off techniques.

Interpolation :
Interpolation is a linear composition of lower and higher order n-gram LMs. It is motivated by the idea that lower order n-gram models are less sparse than higher order n-gram models. Each n-gram model contributes with a specific weight λ_i to the total probability estimation as follows:

$$P_{intr}(w_n|w_1, \ldots, w_{n-1}) = \lambda_1 P_1(w_n) + \lambda_2 P_2(w_n|w_{n-1}) + \ldots + \lambda_n P_n(w_n|w_1, \ldots, w_{n-1}) \tag{2.16}$$

where P_i is an i-gram language model and $0 \leq \lambda_i \leq 1$. $\sum_i \lambda_i = 1$ to ensure that P_{intr} is a proper probability distribution. One way to find the best weights is using the EM algorithm on a held-out set. It converges on locally optimal weights.

Back-off LM:
Like interpolation, back-off is used to address the problem of unseen n-grams. The difference is that in a back-off model, we only use the higher order n-gram probability if it is available, otherwise we back off to a lower order LM to get the

probability as follows:

$$P_n^{BO}(w_i|w_{i-(n-1)},\ldots,w_{i-1}) = \begin{cases} d_n(w_{i-(n-1)},\ldots,w_{i-1})P_n(w_i|w_{i-(n-1)},\ldots,w_{i-1}) \\ \text{if } count_n(w_{i-(n-1)},\ldots,w_{i-1}) > 0 \\ \alpha_n(w_{i-(n-1)},\ldots,w_{i-1})P_{n-1}^{BO}(w_i|w_{i-n+2},\ldots,w_{i-1}) \\ \text{otherwise} \end{cases} \tag{2.17}$$

A discounting function d is used to make sure that all probabilities add up to 1. The lower order probabilities are multiplied by a discounting factor α between 0 and 1 in order to ensure that only the probability mass set aside by the discounting step is distributed to the lower-order n-grams. More details on back off LM can be found in [Katz, 1987]

LM Evaluation and perplexity:

We can measure the LM quality using two ways. The first way is an end-to-end evaluation. In this method, the performance of different LMs is evaluated in the framework of the full system (i.e. a MT system in our case). This is the best evaluation but it is more expensive. The second way is to calculate an independent LM quality measure on an development set. The standard metric is the **perplexity(PP)**. Perplexity is based on the concept of **entropy** $H(p)$, which measures uncertainty in a probability distribution as defined below:

$$H(p) = -\sum_x p(x)\log_2 p(x) \tag{2.18}$$

The perplexity is a simple transformation of **cross-entropy**. Given an evaluation set $(w_1, w_2 \ldots, w_m)$, the language model P_{LM}, the cross-entropy $H(P_{LM})$ is defined as follows:

$$H(P_{LM}) = -\frac{1}{m}\sum_{i=1}^{m}\log_2 P_{LM}(w_i|w_1,\ldots,w_{i-1}) \tag{2.19}$$

and the perplexity is defined as follows:

$$PP = 2^{H(P_{LM})} \tag{2.20}$$

The PP is a positive number. The smaller the value, the better the language model is. It is important to note that the PP of two LMs are only directly comparable if they use the same vocabulary.

2.3.3.2 Neural network language models

The neural network LM (also known as continuous space LM or CSLM) tries to overcome the disadvantages of back-off n-gram LMs. One of these disadvantages is that the probabilities are estimated in a discrete space which does not allow directly the estimation of non-observed n-gram in the training data. In a neural network LM, the words are projected into a continuous space during the training. Bengio et al. [2003] proposes a multi-layer neural network model that jointly learns the word projection and the probability estimation. The basic architecture of this neural network is shown in Figure 2.8.

The inputs of the neural network are $h_j = w_{j-(n-1)}, \ldots, w_{j-2}, w_{j-1}$ which are the previous $n-1$ words. For each input word an 1-of-n encoding is used (i.e. for the word w_i in the vocabulary, set the element i of the input vector to 1 and the remaining elements to zeros). P, N and H are the sizes of one projection, one hidden layer and the output layer respectively. The continuous representation (i.e. embedding) of the word w_i is at the ith row in the projection matrix which has a dimension of N x P. The outputs of the neural network are the posterior probabilities of all words of the vocabulary as follows:

$$P(w_j = i|h_j) \quad \forall i \in [1, N] \tag{2.21}$$

If m_{jl} and v_{ij} are the weights of the hidden and output layers, b_j and k_i are the corresponding biases, c_l the projections, d_j the hidden layer activities, o_i the outputs and p_i their softmax normalization, then the neural network calculates the following:

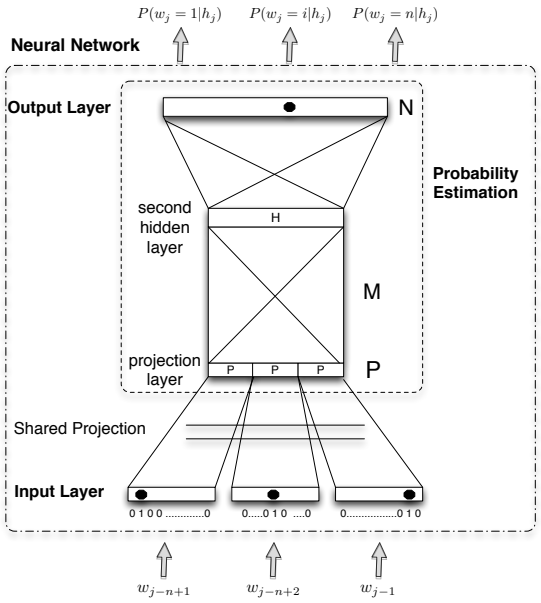

Figure 2.8: The neural network language model architecture. h_j denotes the context $w_{j-(n-1)}^{j-1}$. P, N and H are the size of one projection, one hidden layer and the output layer respectively.

$$d_j = \tanh\left(\sum_l m_{jl} c_l + b_j\right) \tag{2.22}$$

$$o_i = \sum_j v_{ij} d_j + k_i \tag{2.23}$$

$$p_i = e^{o_i} / \sum_{r=1}^{N} e^{o_r} \tag{2.24}$$

p_i will be the probability $P(w_j = i|h_j)$.

The neural network is trained using the standard back-propagation algorithm to minimize the following error function:

$$E = \sum_{i=1}^{N} t_i \log p_i + \beta\left(\sum_{jl} m_{jl}^2 + \sum_{ij} v_{ij}^2\right) \tag{2.25}$$

where t_i is the target output (i.e. the probability 1 for the next word and 0 for the rest). $\sum_{i=1}^{N} t_i \log p_i$ is the cross-entropy between the output and the target probability distributions, and the second part of Equation 2.25 is a weight decay which is used to prevent the model from over-fitting the training data. The value of the parameter β is set experimentally.

The computation complexity of a CSLM is higher than for an n-gram back-off LM because of the high dimension output layer. One way to decrease its complexity is to use a short list instead of the full vocabulary at the output layer. The short list will be limited to the most frequent words, the remaining words will be predicted by a standard back-off LM Schwenk [2004]. At the input layer, all words are modeled.

A CSLM has many advantages, it can be used to estimate the probability of long n-gram (also short n-gram) which can not be directly estimated using n-gram back-off LMs. Also, it can be trained using longer context with just small increase in the complexity which is not possible for n-gram back-off LMs.

The CSLM was successfully applied to large vocabulary speech recognition. It is usually used to rescore lattices and improvement of the word error rate by

about one point were obtained for many languages and domains, for instance [Lamel et al., 2011; Park et al., 2010; Schwenk, 2007; Schwenk et al., 2002]. More recently, the CSLM was also successfully applied to statistical machine translation [Le et al., 2011; Schwenk, 2008a, 2010; Schwenk et al., 2006].

I will present more details on neural network language models and their use in SMT in chapter 5.

2.3.4 Decoding in SMT

The goal of the decoder is to find the best target sentence that maximize the translation probability $P(t|s)$ as expressed in the log-linear Equation 2.11. Several decoders are publicly available like Jane [Freitag et al., 2014], Cdec [Dyer et al., 2010] and Moses [Koehn et al., 2007b]. Moses is an open source SMT toolkit and implements a beam search decoder.

SMT decoding is NP-complete [Knight, 1999], however heuristic techniques work well. Decoding for word-based SMT had a higher complexity because of the possible reordering of individual words compared to phrase-based SMT which use larger translation units (i.e. phrases). The decoding algorithm for word-based SMT could be implemented using optimal A* search [Och et al., 2001], integer programming [Germann et al., 2001] or greedy search algorithms [Wang and Waibel, 1998].

In phrase-based SMT, the most commonly used decoding algorithm is **beam-search stack decoding**, other algorithms like Beam search based on converge stacks, A* search, Greedy Hill-Climbing decoding and Finite state transducer decoding which have been proposed in the literature.

In beam search decoding, the decoder starts by looking for all possible translations in the phrase table. This includes the possible translations of all possible phrases of a given source sentence as shown in the upper part of Figure 2.9.

Decoding of a source sentence starts with an initial empty hypothesis, then the translation output hypotheses are constructed from left to right. The hypotheses are expanded by picking the available translation options as shown in the lower part of Figure 2.9. The decoder then updates the source translation

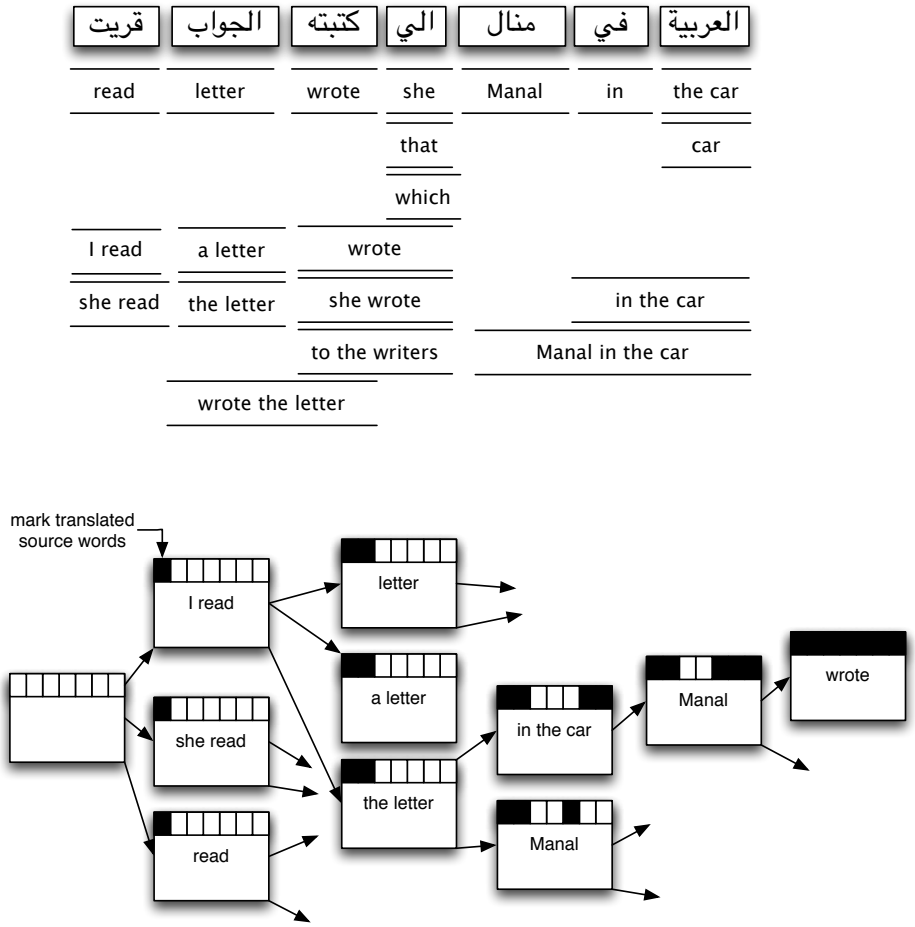

Figure 2.9: Decoding process: start with empty hypothesis, hypotheses are expanded by picking translation options

coverage vector for these new expanded hypotheses. It incrementally computes the translation probability of each of them. Several techniques are used to limit the exponential explosion of the search space. These techniques include hypotheses recombination (i.e. combine similar hypotheses which cover the same source translation but have different scores), pruning out bad hypotheses with worse scores from the hypotheses stack, estimating hypotheses future cost to prevent pruning out good future hypotheses. The expansion process of each remaining hypothesis continues until all source words are covered. These hypotheses are called completed hypotheses. If there are no more incompleted hypotheses, the decoder selects the hypothesis with the highest probability from the completed hypotheses as the most likely translation t_{best}.

2.3.5 MT evaluation metrics

MT evaluation is needed in order to know how good the automatic translation output is. MT evaluation can be done by a human given the source sentence or using a human translated reference(s). It can be also done automatically by a software tool given one or more human reference translations. Traditionally, human judgment is based on two factors, the **adequacy** and the **fluency**. Adequacy measures the degree that the information contained in the reference(s) are presented in the translation. This can be measured as a score which varies from 5 when full meaning in the source sentence is conveyed in the translation to 1 if none of the meaning is conveyed. Fluency measures how fluent the translation is. This can be measured as a score which varies from 5 for a fluent sentence to 1 for an incomprehensible sentence. Since human judgment is expensive in terms of time and cost, automatic evaluation is usually used during MT system development.

Automatic evaluation uses the evaluation metrics that are found to be correlated with human judgment. Usually automatic metrics are useful in measuring the relative translation performance of the MT system from version to version.

One of the first and still frequently used MT evaluation metric is BLEU, short for *Bilingual Evaluation Under Study* [Papineni et al., 2002]. This metric works by measuring the n-gram co-occurrence between a given translation and the set of reference translations and then taking the weighted geometric mean. BLEU is a precision oriented metric as it considers the number of n-gram matches as a fraction of the total number of n-grams in the output sentence.

A variant of BLEU score is the **NIST** evaluation metric [Doddington, 2002], which also calculates how informative a particular n-gram is, the rarer a correct n-gram, the more weight it is given. The NIST score also differs in its calculation of the brevity penalty.

Another metric that I used in this thesis, the Translation Edit Rate (TER) [Snover et al., 2006]. TER is defined as the minimum number of edits needed to change a hypothesis so that it exactly matches one of the references, normalized by the average number of references words. Possible edits include the insertion, deletion, and substitution of single words as well as shifts of word sequences. A shift moves a contiguous sequence of words within the hypothesis to another location within the hypothesis. All edits, including shifts of any number of words, by any distance, have equal cost. In addition, mis-capitalization is counted as an edit in follows:

$$TER = \frac{number\ of\ edits}{average\ number\ of\ reference\ words} \tag{2.26}$$

Also, Snover et al. [2006] proposed Human-targeted Translation Edit Rate (HTER) that employs human annotation to make TER a more accurate measure of translation quality. They proposed creating targeted references to accurately measuring the number of edits needed to transform a hypothesis into a fluent target language sentence with the same meaning as the references. This is done by human editing of the system hypothesis translation to generate the target reference that has the same meaning as the original references. Then, measure HTER by computing TER with this single targeted reference as a new human reference.

Other evaluation metrics are Word Error Rate (WER) [Och et al., 1999], ME-

TEOR [Lavie and Agarwal, 2007] or Translation edit rate plus (TERp) [Snover et al., 2009].

2.3.6 Minimum error rate training

The log-linear model gives us two advantages over the noisy-channel model: the first one is that we can give different weights to different component models. The second advantage is the possibility to easily add new components (also called feature functions). Usually the weights λ_i in the log-linear model (Equation 2.11) are optimized using the MERT algorithm proposed by [Och, 2003]. MERT is an efficient supervised algorithm used to maximize the translation quality on a held-out set as measured by an automatic metric.

MERT works as follows:

- Initialization : initialize λ_i randomly or based on some heuristics.
- Translation: n-best translation of the development set with current λ_i
- Comparison: compare the objective score (such as BLEU) of the n-best translation with previous run
- Re-estimation: Re-estimate the weights λ_i
- Iterate: Iterate until weights have converged

MERT does not scale well to large number of feature functions [Ittycheriah et al., 2007], so other tuning algorithms have been proposed to overcome this issue like MIRA tuning algorithm [Chiang, 2012; Hasler et al., 2011] and the pairwise ranked optimization (PRO) [Hopkins and May, 2011].

2.4 Challenges for Arabic MT

Machine translation from and into Arabic faces the same challenges as human translation between any other two languages as well as some specific issues related

to the Arabic language (like missing of diacritic or short vowels). According to Arnold et al. [1993], the challenges and difficulties of MT in general can be categorized into three main categories: 1) problems of ambiguity; 2) problems arising from structural and lexical difference between languages; 3) multiword units like idioms and collocations. Jurafsky and Martin [2000] explained that the translation difficulty is caused by the differences between human languages and hence the translation between similar languages could be more easier than the translation between non-similar languages. If we also consider the translation challenges of web content that is written by internet users and the problems related to human mistakes and online writing styles and this thesis focus on Arabic and Egyptian dialect, I divide the challenges and difficulties of Arabic translation into the following five main categories: Ambiguity problems, Degree of similarity of languages, Human related challenges, Arabic vs. Egyptian dialect differences and MT approach related challenges.

2.4.1 Ambiguity problems

2.4.1.1 Lexical ambiguity

Lexical ambiguity means that the word can have more than one meaning. One case for lexical ambiguity is that the word has two or more lexical categories (e.g. *fly* as noun vs. a *fly* as verb). In this case one possibility to disambiguate these words is by using a part of speech (POS) tagger. A word has two or more meanings within the same lexical category (e.g. the noun *bank* as a financial institution vs. the noun *bank* as in a river bank).
In Arabic, one of the reasons of increasing the lexical ambiguity is the omitting of short vowels (diacritics) and sometimes dots for the Yaa and Taa-Marboota letters. However, native speakers can still understand the correct meaning (i.e.

the correct diacritics) using the context. For example the word جد can mean "grandfather" or "serious". If this word is used in the context "تحدثت مع جد أحمد اليوم" (i.e. I talked to Ahmed's grandfather today), the right meaning will be "grandfather".
In all these cases, the translation process will need to solve these problems using word sense disambiguation techniques either implicitly or explicitly. One way is to translate a sequence of words which contains larger word-context like what happens in phrase-based SMT. This solution assumes that the source phrase has been seen before in the training data, otherwise the phrase-based SMT system will not be able to generate the right translation since it will back-off to translate shorter phrases or even individual words.

2.4.1.2 Lexical divergences

An example of lexical divergence is the translation of the English word watch could translate into Arabic as "ساعة يد" or "مراقبة" or "يشاهد". The translation often requires solving the same problems as word sense disambiguation. Another example is the English word "know" which can be translated into Arabic as "يعلم" or "يعرف".
Another example is the translation of a verb from English into Arabic, since Arabic verbs are inflected by the subject's gender (e.g. اكتبي). The translation of such verb into Arabic will require deciding the gender of the subject in order to be able to translate it correctly into Arabic.

2.4.1.3 Structural ambiguity

Structural ambiguity is the case when a sentence can have two or more different structure interpretation. For example, in this Egyptian Arabic sentence: "احمد قرى الجواب اللي كتبته ساره في العربية" (i.e. "Ahmed read the letter which Sara wrote in the car"), it is not clear if "Ahmed read the letter in the car" or "Sara wrote the letter in the car".
Sometimes, this case of ambiguity is difficult to resolve, even for human translators. In this case, larger context like a paragraph context could be useful to pick the right structure interpretation, then possibly re-phrase the sentence to remove the ambiguity.

2.4.2 Degree of similarity of languages

Several characteristics can be used to determine the degree of similarity between any language pair. The first category of these characteristics are related to the morphology, syntax and structure. The second category is related to idioms, collocations and similar issues. I am giving more details on these categories in the following sections.

2.4.2.1 Systematic differences across languages

These differences can be divided into three categories:

A) Morphology:
There are some languages with rich morphology like Arabic, while others have a simple morphology. Human languages can differ in:

- Number of morphemes per word:
  In some languages, each word has one morpheme like Vietnamese. These languages are called isolating languages. While in other languages, like Arabic, each word may have many morphemes. These languages are called polysynthetic languages [Jurafsky and Martin, 2000].

- Difficulty to segment the word into morphemes:
  In some languages, the morphemes have clear boundaries, while in others,

a single affix may conflate multiple morphemes. These languages are called agglutinative languages and fusion languages respectively. Arabic is considered to be an agglutinative language.

Like other semitic languages, Arabic language has a rich morphology. It has also complex morphological inflections. Some morphemes like the prepositions: ف in فذهب, personal object ـه in أكلته and possessive pronoun ي in أبي are affixed the word stems. The corpus of rich morphological language, like Arabic, is sparser than the equivalent English corpus because the average number of observed instances of an Arabic word in surface form (without morphological segmentation) will be lower, than the average number of observed instances of the words in the English corpus [Abdelhadi Soudi, 2012].

Preprocessing of training, tune, development and test sets aims at reducing the morphological differences between source and target languages. Morphological segmentation is used to segment the word into its different morphemes. This helps the translation model to get better alignment and hence improve the translation quality. For Arabic, this includes the segmentations of prepositions, possessive pronouns, subject pronoun, object pronoun, and other types of morphemes.
Another problem typical for the Arabic language specifically is the omittion of short vowels (diacritics). Sometimes, the only difference between two morphological forms is the diacritics. If they are missing, it is not possible to understand which one is used without the context. For example the word "كتب" (i.e. *wrote* or *had been written*) in the following two sentences has different diacritics and hence different morphological form but the diacritics are omitted:
"كتب الطالب في الكتاب" (i.e. the student wrote in the book)

"كتب في الكتاب" (i.e. the book had been written) .
Native speakers use the context to decide the form with the correct diacritics. This issue is actually resolved the same way by phrase-based SMT systems since the phrases as translation units have larger context and hence the translation of the phrase will usually be correct, however translating correctly a single word will still be a challenge.

B) Syntax:
Languages can have different sentence structure. For example, in English, the sentence structure is Subject(S)-Verb(V)-Object(O) while the Arabic language has a more flexible syntactic order which could be SVO or VSO or VOS or even S-Predicate(P). Other languages like Japanese, the sentence structure is SOV. Languages similar in their syntactic structure usually have similar characteristics. For example, languages with SVO sentence structure, usually have preposition, while languages with SOV sentence structure, usually have postposition.
It is clear that different syntactic structure orders will need more effort during translation; more specifically, more reordering of the translation is needed to match the target language syntactic structure. It is even more difficult if the source language structure has different grammatical components than the target language structure. For example, the translation of an Arabic sentence with structure S-P to the English sentence SVO.
For the easier case, when just orders are different, one way to overcome these syntactic differences is to perform some preprocessing on the source language sentences to reorder it to be closer to the syntactic structure order of the target language. This needs a parser to process the source language sentences and reorder the words to match the target language syntactic structure with some hand crafted rules. For example, for translating Arabic into English, we need to re-order the Arabic sentence from VSO to SVO, which is the English sentence structure. This could help in increasing word alignment coverage and significantly improve the translation performance scores as shown in [Carpuat et al., 2010], who reordered Arabic VS into SV when translating from Arabic into English. Similar methods were used to perform word reordering to make the Chinese sentences closer to the English sentence order [Way and Du, 2010], and they reported

significant improvement in the translation performance scores.

C) Argument structure and linking:

In this category, there are three types of differences between languages regarding argument structure and linking:

1. Relation location marking between the head and its dependents
   Languages have different location of the relation marking between the head and its dependents. In Head-marking languages, the relation mark is on the head, while in Dependent-marking language, the relation mark is on the dependent [Jurafsky and Martin, 2000].

2. The verb manner and motion direction
   In some languages, the direction of motion is marked on the verb leaving the satellites to mark the manner of motion. They are called Verb-Framed languages. Other languages, mark the direction of motion on the satellite and leave the verb to mark the manner of motion. They are called Satellite-framed languages [Jurafsky and Martin, 2000].

3. Referential density and pro-drop
   In some languages like Arabic, the pronoun can be dropped when talking about a referent that is given in the discourse, these are called pro-drop languages, while for other non-pro-drop like English, it is required to use explicit pronoun. Even pro-drop languages vary in the frequency of omission, which is called referential density of the language. Languages which use more pronouns are more referentially dense than those use less pronouns.

2.4.2.2 Idiosyncratic differences including multiword units like idioms and collocations

The following subset of idiosyncratic differences are part of the translation challenges between languages:

1. Adjective-noun order
   Some languages like English, the adjective precedes nouns, while in other languages like Arabic languages, the adjective follows the noun.

2. Idioms
   Idioms are expressions whose meaning cannot be completely understood from the meaning of the component parts [Arnold et al., 1993].
   For example in English the idiom "kick the bucket" means "dies". It is difficult to know the meaning of the idioms from the individual words in it. This is a real challenge for SMT and word alignment model. Idioms should be translated as a single unit, otherwise the translation will be wrong.

3. Collocations
   In collocations, the sentence meaning can be understood from the meanings of individual words, but the correct word choice is not predictable [Arnold et al., 1993]. The collocation problem is less significant than idioms since the selection of the right word is predictable from other word(s), while with idioms it is not possible to know the meaning from any part of the sentence. Using phrases as translation units as well a good LM can fix the collocations problem by helping selecting the right word among different hypotheses.

4. Dates and time format/calendars
   Different languages usually have different date and time formats. Sometimes even for the same language, there are different date and time formats. For example, the date format used in the UK is different from the date format used in USA. Another challenge for machine translation is the use of different calendars. It is a challenge to translate the Islamic Hijri date for example to Gregorian date. This issue is usually addressed in the preprocessing of the corpora, by detecting the format used and translating or re-ordering the date parts as required in the target language.

2.4.3 Human related challenges

Nowadays, one of the common sources of corpora is the web. Some collected texts are written by internet users who have different backgrounds and education levels. People can make spelling mistakes and they also can have their own writing style like stressing on some letters by repeating them or by using some punctuations for other purposes like emotional expression or for text decorations. So we can have two categories of these problems:

1. Orthographic errors.
2. Writing behavior on digital media.

2.4.3.1 Orthographic errors

One of the challenges of translating text are the orthographic errors. This increases significantly when translating text written by internet users like news comments, forums, social media posts and comments, chat and tweets. Because of various education levels it is possible that the Arabic internet users substitute some letters with others which are close to them in pronunciation like "ذ" (i.e Zal) with "ز" (i.e. Zay) also missing shadda "ّ" or using "ى" (i.e. alf-maksoura) instead of "ي" (i.e. Yaa). For example using "مرسى" instead of "مرسي" or vice versa.
For SMT such spelling mistakes will impact the word alignment, translation and language models. In an SMT system, we can deal with this challenge either by training our system on such data and allowing it to learn to translate words with mistakes OR we can do a pre-processing step on training, tuning and dev, and testing data to correct the spelling mistakes. The decision usually depends on the

performance of the translation performance of the two SMT systems with and without spelling correction.

2.4.3.2 Writing behavior on digital media

Internet users have some writing behavior, for example Arabic users are used to repeat one letter as a kind of stressing a word (e.g. "رااااااااااائع" or "wonderful") It is also possible to repeat (haa letter in Arabic or h in English) to express the laughing action (e.g. ههههههههههه). Usually more repetition of the letter means longer laugh. Another example, Arabic users can use some punctuations for text decoration instead of the normal purpose like the following:
(ـ ■■ ـ ■■ ـ ■■ ـ ■■ ـ ■■ ـ مصر الحبيبة ـ ■■ ـ ■■ ـ ■■ ـ ■■ ـ ■)).
This kind of writing behavior introduces another challenge for SMT and even for human translators since sometimes it may be needed to reflect (or use) this writing behavior in the translation output, while some other times, we can do some preprocessing and normalization to help SMT produce better translation independent of the writing behavior.

2.4.4 Arabic vs. Egyptian dialect differences

The modern standard Arabic (MSA) and Egyptian dialect have a common MT challenges. Since Egyptian dialect is a mixture of MSA and additional dialectal words and dialectal structure, it shares many words, features and grammar of MSA. Some examples of such common attributes: the missing short vowels, the clitics and the sentence structure.

Additionally, the Egyptian dialect has its own special attributes which can be divided into two main categories: general and writing specific.

1. The general category includes:

   - More flexible sentence structure for example the sentence الستات دول (i.e. these women) has a different word order than its equivalent in MSA هؤلاء النساء. Additional examples shown in Table 2.1.

Linguistic	Differences		Example		
	MAS	Egyptian	MSA	Translation	Egyptian
Present tense	Starts with Hamza أ or others	Start with Baa ب	أنا آكل	I am easting	باكل
Future tense	Uses Seen س	Use Haa or Heh هـ حـ	سأسافر	I will travel	حسافر
Passive form	Uses wazn فُعل	Starts with Alef إ or Alef+Taa ات	أُكل	have been eaten	إتاكل
Negation	Uses Lam لم	Uses two parts مـ..ش	لم أسافر	I did not travel	مسفرتش

Table 2.1: *Comparison of some linguistic forms used in MSA and Egyptian dialect*

   - Different or additional morphological form for some words like مرحتش (i.e. I did not go) which has no equivalent one word in MSA.
   - Different inflection compared to MSA like the Egyptian specific negation (ما..ش) in ماكلتش (i.e. She did not eat) which does not exist in MSA and instead it use لم negation لم تأكل.
   - Replacing some letters by others for sake of easy pronunciation like replace tha ث by taa ت in the MSA word ثلاثة (i.e. three) to be تلاته . Additional examples shown in Table 2.2.
   - Adding additional letters to the MSA word like adding additional alef

Arabic letter	Egyptian		Example		
	Pronunciation	Writing	MSA	Translation	Egyptian
Hamza on Yaa ئ	Yaa ي	Seen ي	ذئب	Wolf	ديب
Daad ض	Zaa ظ	Zaa ظ	ضابط	Officer	ظابط
Qaaf ق	Hamza ء	Qaaf ق	قمر	Moon	قمر
Thaa ث	Taa ت/Seen س	Taa ت/Seen س	ثانوية	Secondary	سانوية
Zaal ذ	Daal د/Zaay ز	Daal د/Zaay ز	ذرة	Corn	درة
Zaa ظ	Daad ض	Daad ض	ظل	Shadow	ضل
Hamza at the end ء	omitted	omitted	صحراء	Desert	صحرا

Table 2.2: *Some examples of how Egyptian dialect replaces some Arabic letters by others in pronunciation and most time in writing*

ا in رجل (i.e. man) to be راجل and in معه (i.e. with him) to be معاه.

2. The writing specific category includes:

   - Various orthography of the same word due to lack of standard writing like حيسوق (i.e. he will drive) and هيسوق or معكاش (i.e. you do not have anything) and معكش.
   - High rate of orthographic mistakes.
   - Letter repetition like رااائع (i.e. wonderful).
   - Omitting of some punctuations and some letters' dots like in كوبرى (i.e. bridge) instead of كوبري.
   - Using of additional vocabulary which are not in MSA ست (i.e. woman),

ياريت (i.e. I hope), زي (i.e. like).

Some of these attributes causes the training data to be more sparse or introduce more ambiguity.

2.4.5 MT approach related challenges

In this category, the problems are specific to the MT approach or method. For example, in corpus-based approaches, we use specific bilingual and monolingual corpora and hence closed vocabulary. This leads to several problems as follows:

1. Some source words will not be translated by the MT system because they are unknown to the translation model. These are called Out-Of-Vocabulary (OOV) words. Examples of such unknown words are proper nouns, verbs with different morphological form, words with different inflection form and entities like number or dates. Transliteration of proper nouns can be used to decrease the number of OOVs in the translation output.

2. Unknown target words to the language model.

3. Mismatch between the domain or the style of the bilingual and monolingual training corpora and the translation task. For example when the MT system is trained on modern standard Arabic and formal corpora, but it is used to translate Egyptian dialectal and informal text.

4. Segmentation errors: words are wrongly segmented instead of being left unprocessed or unsegmented words.

5. Low resource languages: small bilingual corpora mostly will lead to a bad translation model and a lot of OOVs, while small monolingual corpora could lead to non-fluent translations and bad formed target sentences.

6. Pre-ordering and inflection of languages with flexible sentence components is a challenge since several orders can be correct and acceptable but inflec-

tion could be different in each order (e.g. اكل أحمد التفاح vs. e.g. التفاح أكله أحمد)

7. Limited data resources causes data sparseness problem. How often the word occurs in the training data correlates with the machine translation quality. if the word (or phrase) occurs rarely, it causes problems in word alignment, calculation of the translation probabilities and other statistical modeling training. If the word never occur, this causes the problem of OOVs which we discussed in the first point above. The data sparseness problem is generally addressed by using more data which help in a better word alignment, a better estimation of the words and phrases translation probabilities as well as additional context for PBSMT.

2.5 Conclusion

In this chapter, I have briefly explained an introduction to machine translation (MT), its history and approaches. Since the SMT is the bases of this thesis, I focused on explaining the basics of SMT and covered different components of word-based and phrase-based SMT, including the translation model and the language model. I introduced the current state-of-the-art in language modeling in a full section that covers n-gram back-off and neural network language models. I also explained the decoding algorithm in PBSMT, then gave more details on the machine translation metrics and evaluation. Finally, a full section was dedicated to an overview of the challenges of translating Arabic and Egyptian dialect into English, since this is the focus of this thesis in the context of BOLT program.

Chapter 3

BOLT Project

3.1 Introduction

The Arabic language received a lot of attention in the machine translation community during the last decade. It is the official language of 25 countries and it is spoken by more than 295 million people. The interest in Arabic language and its dialects increased more after the Arab spring and the political change in the Arab countries. There are several research projects with adequate funds focusing on Arabic MT research. Our research group in LIUM is partner in many national and international projects that work on MT. One of these projects is the Broad Operational Language Translation (BOLT) program.

In order to address the need to develop a technology for the task of handling informal language, in October 2011, DARPA launched BOLT program to focus on developing new methods, tools and technology for machine translation and linguistic analysis which mainly address the informal genres of text and speech common in online and personal communication.

As stated on DARPA website[1], BOLT is aimed at enabling communication with non-English-speaking populations and identifying important information in foreign-language sources by:

1. Allowing English-speakers to understand foreign-language sources of all gen-

[1]source: http://www.darpa.mil/program/broad-operational-language-translation

res, including chat, messaging and informal conversation.

2. Providing English-speakers the ability to quickly identify targeted information in foreign-language sources using natural-language queries.

3. Enabling multi-turn communication in text and speech with non-English speakers. If successful, BOLT will deliver all capabilities free from domain or genre limitations.

BOLT project consists of three phases, started in October 2011 and finished by December 2014. LIUM was partner with IBM and other universities including RWTH Aachen University, Stanford University, Cambridge University, University of Maryland and MIT, working on machine translation research in BOLT delphi team, leaded by IBM. This chapter covers the activities, different techniques and research that were performed during this project. These techniques were used to improve the translation quality of Arabic into English MT system, but in most cases they can be adapted to other languages with small effort. This includes addressing some of the Arabic machine translation challenges presented in Section 2.4. I also present the LIUM systems evaluation results in each phase of this project.

3.2 Resources description

Genres:
During this project we have developed three systems for Egyptian dialect. Each system has been adapted for one genre of the following:

- Discussion forum (DF).
- SMS/Chat system.
- Conversational telephone speech (CTS) transcript.

Even though, the BOLT project focuses on Egyptian and the genres above, corpora in other dialects and genres were available to use for system training. For

easy reference of each dialect/genre I assigned an ID for each of them as shown in table 3.1.

Genre ID	Description
MSA_NW_WB	Modern Standard Arabic (MSA) (includes Broadcast News, Broadcast Conversation, Newswire, Newsgroups and Weblogs)
EGY_DF	Informal text in Egyptian dialect (threads, posts collected from online discussion forums)
IRQ_DF	Iraqi Arabic dialect
LEV_DF	Levantine Arabic dialect
MSA_FORMAL	Formal MSA (document collections from the United Nations)
EGY_SMS_CHAT	Egyptian dialect (collected naturally occurring SMS and Chat data in Egyptian Arabic)
EGY_CTS	Egyptian dialect (CTS transcript, which is supplied from LDC's multilingual CALLHOME and CALLFRIEND collections.)
EN_DF	Collected threaded posts from online discussion forums in English language.

Table 3.1: List of the genres and dialects used in BOLT project and the assigned IDs used in this thesis.

Bilingual corpora description:

The bilingual training corpora used in BOLT project are listed in Table 3.2. The list of tune, development and test sets with some short meaningful names is shown in Table 3.3.

Evaluation metric:

The official phase evaluation in BOLT program is performed by NIST using human evaluation HTER, but during system development, teams use automatic evaluation metrics. Normally, we should use TER [Snover et al., 2006] since it is similar to HTER but this obtains worse BLEU [Papineni et al., 2002] score, so we used (TER - BLEU)/2 metric [Servan and Schwenk, 2011] which we refer to as TB2 through this thesis. For some experiments, BLEU metric has been used and it was clearly stated, otherwise TB2 should be assumed. More details on HTER, TER and BLEU are available in Section 2.3.5.

corpus	genre	release phase	Ar tokens	En tokens
bolt	EGY_DF	1	1.70m	2.05m
thy		1	282k	362k
bbnturk		1	1.52m	1.58m
bbnegy		1	514k	588k
gale	MSA	1	4.28m	5.01 m
fouo		1	717 k	791k
ummah		1	3.61m	3.72m
e103		1	4.44m	4.45m
isi		1	35.44m	34.71m
fix		1	1.22m	1.43m
iraq	IRQ_DF	1	1m	1.14m
bbnlev	LEV_DF	1	1.59m	1.81m
un	MSA_FORMAL	1	134.88m	127.71m
smschat	EGY_SMS_CHAT	2	648k	845k
cts1	EGY_CTS	3	430k	522k
cts2		3	804k	931k
Total	-	-	193.13m	187.69m

Table 3.2: The sizes and the genres of bilingual training corpora in BOLT project.

Set	Genre	Tune Ar/En tokens	Dev Ar/En tokens	Test Ar tokens
d10 (3 references)	MSA_NW_WB	42k/ R1=49.4k R2=46.8k R3= 50k	42.5k/ R1=49.7k R2= 47k R3=50.3k	43k
d12	EGY_DF	17.7k/21.4k	27.2k/32.6k	19k
p1r6		52.5k/67.3k	18k/22.3k	21.4k
cts-asr	EGY_CTS	17.6k/24k	21k/29.6k	39k
cts		20.3k/24k	25k/29.6k	44k
smschat (3arrib)	EGY_SMS_CHAT	19.7k/25.6k	19.4k/24.6k	18.5k
smschat (trans)		19.3k/25.6k	19.4k/24.6	18.5k

Table 3.3: The size and the genre of tune, dev and test sets used in BOLT project.

3.3 Baseline Systems

All LIUM BOLT systems are built using the standard phrase-based SMT with Moses toolkit [Koehn et al., 2007a] and the alignment performed using GIZA++ [Och and Ney, 2003c]. We use 4-gram LM trained using Kneser-Ney smoothing as implemented in the SRILM toolkit [Stolcke, 2002] and is converted to KenLM LM [Heafield, 2011] in order to decrease the required memory and improve the speed of CSLM training and re-scoring. For CSLM training and rescoring, LIUM open source CSLM toolkit [Schwenk, 2007, 2010, 2012] is used. Log-linear features' weights are optimized using MERT [Och, 2003] [Bertoldi et al., 2009]. XenC [Rousseau, 2013], the LIUM open source tool is used for data selection. For Arabic segmentation, MADA/TOKAN [Habash and Rambow, 2005], MADA-ARZ version 0.4 [Habash et al., 2013] or data segmented by IBM using IBM internal tools.

Since BOLT program had 3 phases, we had several baselines either internally in LIUM or externally based on the previous phase delivered PBSMT systems. Table 3.4 summarizes each phase baseline system and the evolvement of the baseline systems from phase to phase.

Phase	Genre ID	Baseline ID	Delivered ID	Applied technique or experimental work
Phase 1	EGY_DF	EGY_DF_BL1	EGY_DF_P1 June 2012	+LM adaptation(data selection) +TM adaptation(data selection) +find best set for MERT optimization *Evaluate MADA segmentation schemes +CSLM rescoring
Phase 2	EGY_DF	EGY_DF_P1	EGY_DF_P2 Sept 2013	+TM adaptation(instance weighting) +Fill-up/Backoff phrase tables +Arabic/Egyptian preprocessing +TM light-supervised training *Muti-domain Adaptation *Entity-based translation *IBM ATB vs. MADA-ARZ ATB
Phase 3	EGY_DF	EGY_DF_P2	EGY_DF_P3 Dec 2014	+Operation sequence models(OSM) +Using combined CSLM models *Using word embedding for WSD *LM lightly-supervised training
	EGY_SMS_CHAT	EGY_SMS_BL1	EGY_SMS_P3 Dec 2014	+New EGY_SMSCHAT bitext +LM adaptation(data selection) +TM adaptation(data selection) +TM adaptation(instance weighting) +Operation sequence model *Arabic/Egyptian preprocessing +CSLM rescoring +Using combined CSLM models
	EGY_CTS	EGY_CTS_BL1	EGY_CTS_P3 Dec 2014	+New EGY_CTS bitext +same same techniques as SMS

Table 3.4: Description of the baseline systems for each BOLT phase and the techniques applied or experimental work (+ means applied to the baseline, * means experimental)

3.4 Evaluation Results

The summaries of the results of LIUM systems in the three international evaluations of the BOLT project are shown in Table 3.5 and Table 3.6.

Table 3.4 lists the techniques and methods we applied and integrated in the delivered system during the phase (i.e. marked by +) as well as the experimental and research work that had not been integrated due to its modest results (i.e. marked by *).

Set	type	P1	P2	P3
d10	test	1.45	2.61	1.25
d12	dev	16.93	15.88	15.20
d12	test	16.15	14.39	13.74
P1R6	dev	15.75	14.76	14.71
P1R6	test	15.84	15.39	15.28
P1Prog	dev	17.86	17.77	17.63
P1Prog	test	10.50	10.50	9.75

Table 3.5: *LIUM systems evaluation results during the three phases of the* BOLT *project for EGY_DF genre (scores in* TB2*).*

Set	type	P3 baseline	P3
smschat	dev	19.24	15.46
smschat	test	16.83	12.67
smschat 3arrib	dev	19.87	15.74
smschat 3arrib	test	17.81	12.70
cts	dev	16.89	15.91
cts-asr	dev	27.91	25.46
cts	test	18.09	17.97
cts-asr	test	29.79	27.04

Table 3.6: *LIUM systems evaluation results compared to initial baseline during the* BOLT *project phase three for EGY_SMS_CHAT and EGY_CTS genres (scores in* TB2*).*

3.5 General improvements and Arabic specific improvements

Several general and Arabic related techniques have been implemented. One of these techniques is adapting our SMT systems to the Egyptian dialect. Since the available training corpora, in the context of BOLT project, contains MSA, and several dialects (i.e. Egyptian, Levantine and Iraqi). We improved the system performance by using domain adaptations techniques and treating different dialects as different domains. We use four adaptation techniques to adapt our system on the Egyptian dialect and the system genre. The first technique is using instance weighting of translation models to improve the translation quality by giving more weights to Egyptian than MSA or other Arabic dialects. More details can be found in Section 3.7.3 and 3.7.4. Since our training corpora have various genres (i.e. NEWS, WEB, UN, DF, SMSCHAT and CTS), we adapt our systems by using data selection techniques. Two techniques are used, the first one is used to select the relevant sentences from monolingual corpora to improve and adapt the LMs, while the second one is used to select the most relevant sentences from the bilingual corpora to improve the TMs. These two techniques are detailed in Sections 3.7.1 and 3.7.2 respectively. We also apply another method for the adaptation of SMT systems to Egyptian using the so-called "lightly supervised" training. This is explained in Section 3.7.5.

Since Arabic is a morphologically rich language, the selection of the suitable Arabic morphological segmentation is one of the important preprocessing steps in MT research. There are many morphological schemes that can be used to segment the Arabic words. I evaluated various Arabic segmentation schemes from full word form to fully segmented form to explore the effect on the system performance and translation quality. More details can be found in Section 3.6.1.

In order to address ambiguous Arabic/Egyptian words translation errors, I worked on applying word sense disambiguation technique on them using their context. I integrated this technique into a phrase-based SMT system in order to improve the system performance in translating ambiguous words. This research

was conducted during my 3 months internship at IBM T.J. Watson Research Center in 2014 and is covered in Section 3.9.

Another challenge in MT research is dealing with the Out-Of-Vocabulary (OOV) words. I have performed research on several methods to decrease the OOV rate by proper noun transliteration. More details can be found in chapter 4.

Finally, some OOVs are actually numbers, dates .. etc. which can be translated to target language using some rules. This problem is more critical between languages using different writing scripts like Arabic and English than between French and English for example. Since there is no integrated method to handle such entities translation, I developed a method to detect numbers, dates and other entities and then transform them from the source language to the target language. This also allows us to have class-based SMT systems with less language model and translation model size. More details can be found in Section 3.6.2.

3.6 Preprocessing techniques

The following preprocessing techniques were evaluated and used:

3.6.1 Arabic segmentation schemes

The scheme is used to define the desired target tokenization. Each scheme specifies what to split (i.e. segmentation) and what form to represent the various parts (i.e. regularization) [Habash, 2010]. The selection of the suitable Arabic morphological segmentation scheme is one of the challenges and opportunities in MT research for MSA [El Kholy and Habash, 2010, 2012; Habash, 2008] and also for Arabic dialects [Salloum and Habash, 2011, 2013; Zbib et al., 2012]. Since Arabic is a morphologically rich language, the selection of the suitable Arabic morphological segmentation is a very important preprocessing step of MT data. This selection is proved to have a significant impact on the translation quality [Al-Haj and Lavie, 2012; Sadat and Habash, 2006; Zollmann et al., 2006]. The segmentation scheme should be consistent across all train, tune and test sets.

For example, the wrong segmentation of the Arabic word غزه (Gaza) to غز (kill by penknife) and ه (-his) can lead to translate it into *"kill"* instead of the city name *"Gaza"*.

There are many morphological schemes that can be used to segment the Arabic words. I evaluated various Arabic segmentation schemes from a full surface form to a fully segmented form to explore the effect on the system performance and the impact on the translation quality.

In this work, initially, I used MADA/TOKAN to perform the segmentation. The same corpus with a different segmentation is used to build SMT systems. I used two baseline systems, the first baseline system is built using raw Arabic unsegmented training data and the other baseline is built using Arabic data segmented with an IBM in-house tool following the Arabic Tree Bank (ATB) schema. The motivation of using the first baseline is to emphasize the importance of segmenting Arabic text and to show the large impact on machine translation performance. To limit the time needed to perform a large set of experiments, we used the gale corpus only and tried different schemes using MADA/TOKAN. Also, when I performed these experiments, the MADA version that supports the segmentation of Egyptian dialect was not released yet. The results of these experiments are shown in Table 3.7. The details of each schema are shown in Table B.1 on page 148. We concluded that ATB outperforms other Arabic segmentation scheme in the context of MT. Also, in these experiments, MADA/TOKAN ATB tokenization slightly out-performed IBM ATB tokenization.

During the second phase of BOLT, we did a full system comparison using our EGY_DF system which is using IBM ATB and re-built the whole system using

MADA ATB using MADA-ARZ. The TB2 scores of both systems are shown in Table 3.8. The motivation of this experiment was to build a different system to benefit system combination across BOLT delphi team. IBM-based system outperformed MADA-ARZ-system on Egyptian dialect, but the later one outperformed former one on MSA by 0.3. Based on these results, we decided to continue using IBM ATB segmentation especially that MADA-ARZ-based system did not benefit the system combination task that involve systems from LIUM, IBM and other universities in BOLT delphi team.

Bitext	Arabic Segmentation Scheme	d10 tune	d10 dev
GALE	Raw (baseline)	24.78	23.13
	IBM ATB (baseline)	**33.75**	**36.16**
	MADA-ATB	**34.30**	**36.56**
	MADA-D1	32.29	34.56
	MADA-D2	33.37	35.69
	MADA-D3	32.96	35.35
	MADA-S1	33.00	**36.16**
	MADA-S2	33.30	35.65
	MADA-ATB+POS	**33.99**	35.50
	MADA-OLD-ATB	33.41	35.68
	MADA-ATB4MT	32.32	34.42
	MADA-D34MT	31.67	34.20
	MADA-DIAC	29.21	31.39

Table 3.7: BLEU scores of GALE training corpus with different Arabic segmentation schemes.

System	d10 test	d12 dev	d12 test	p1r6 dev	p1r6 test
IBM ATB	2.73	16.20	**15.11**	**15.75**	**15.84**
MADA ATB	**2.43**	**16.02**	15.16	16.59	16.52

Table 3.8: TB2 *results of EGY_DF system built using IBM ATB and MADA-ARZ ATB segmentation.*

Egyptian dialect preprocessing

We noticed that in forum genre corpora, there are many repeated letters. The Arabic internet users usually use repeated letters as a way to emphasize a word or

letters	Preprocessing
Tatweel ـ	Remove
غ ض ح ج ث ة ئ ؤ ء	Normalize repeated letters to just one letter
ي ى و ص ظ ق ط ذ ص ز ر د ن ب	Normalize repeated letters to max of 2 letters
ك ف م ل ع ش هـ خ ي و ا إ أ آ	Normalize repeated letters to max of 3 letters
و ا إ أ آ	Normalize repeated letters at the end of the word to just one letter
ي	Normalize repeated letters at the end of the word to max two letters

Table 3.9: *Pre-processing rules for EGY_DF genre development/test sets*

to express the amount of emotions or enthusiasm. So I applied some preprocessing rules to normalize these repeated letters as summarized in Table 3.9. This leads to a gain between 0.15 and 0.35 on TB2 on development set as shown in Table 3.10. This gain is only observed with EGY_DF genre.

System	d10	d12	p1r6
Baseline(BL)	5.42	17.75	16.02
BL+pre-processing	**5.19**	**17.40**	**15.87**

Table 3.10: *Results of development set preprocessing. (scores in* TB2*)*

3.6.2 Entity translation

I focused on number, date, email and URLs entities. Numbers and dates are part of the cultural preference of any language and country. For example date format in France is different than the date format used in the USA or the UK (e.g. day/month/year vs. month/day/year). We can also observed a difference in format of numbers (e.g. 2 450,30 in France vs. 2,450.34 in the USA). It is important to translate them by phrase entries in the phrase table, but this is not always possible because usually they have many variations. Unknown entities are

considered OOVs which their translation should not be a complex task. They can be translated to target language using some rules. This problem is more critical between languages using different writing scripts like Arabic and English than between French and English for example. Since there is no integrated method to handle such entities translation, I developed a procedure to detect numbers, dates and other entities and then transform them from the source language to the target language.

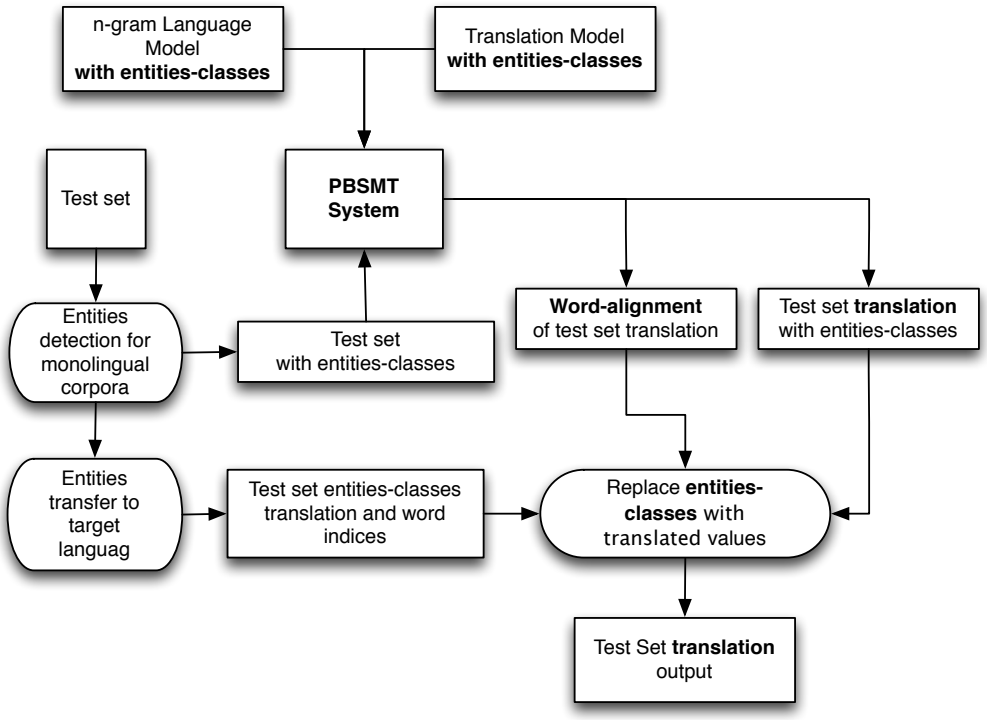

Figure 3.1: The support of externalization of entities values translation in entity-based PBSMT system.

This work aims in developing pre-processing and post-processing engines to manage such entities as shown in Figure 3.1. The value (e.g. 1 Jan 2015) of each entity is substituted by a placeholder. The preprocessing engine uses the detection and transformation rules and apply them on the provided text. The separation of the rules in a separate file makes the change of the detection and translation rules more flexible. One post-processing and three preprocessing tools were developed.

The preprocessing tools were applied to all kind of corpora, namely bilingual and monolingual training corpora.

All preprocessed text do not contain the entities' values (numbers or dates, etc...), but it only contains the placeholder of each entity. This helps decreasing data sparseness and decreasing the size of the translation model (i.e. the phrase table) and the language model. A post-processing tool is responsible for replacing the placeholders in the translation output by their translated values using the source to target alignments provided by the decoder. One advantage of this technique is that we can keep the MT system independent of the source and target languages cultural preferences. At decoding time, we have the flexibility to select the required cultural preference needed for the translation task. For example, the same SMT system can be used to translate text from UK or USA by specifying the input type to the entities handling engine.

3.7 Domain adaptation

3.7.1 Monolingual corpora data selection

As seen in Table 3.11, the available monolingual corpora provided by LDC is more than 5.6 billion tokens including the English Gigaword corpora. Most of these data are news (i.e. formal) data, while BOLT project focuses on informal text as mentioned before. We can adapt our LM by selecting a small portion of the most relevant data to our task from these huge monolingual corpora. This selected data is used as additional training data for our LM. We performed data selection using the method of Moore and Lewis [2010]. Their method is based on comparing the cross-entropy (i.e. Equation 2.19 on Page 30), according to in-domain and out-of-domain language models for each sentence of the text.

We used XenC, the LIUM open source tool for data selection which implements the cross-entropy monolingual data selection proposed by [Moore and Lewis, 2010].

If IN and OUT are the in-domain and out-of-domain corpora respectively. Firstly, XenC creates the in-domain language model LM_{IN} and out-of-domain

language model LM_{OUT}. Secondly, XenC calculates, for each line s in OUT, the cross-entropy $H_{IN}(s)$ given by LM_{IN} and $H_{OUT}(s)$ given by LM_{OUT}. Then, XenC will add for each line a score which is calculated using the cross-entropy difference as in following Equation:

$$Score_s = H_{IN}(s) - H_{OUT}(s) \tag{3.1}$$

Then select lines based on a score cutoff optimized on held-out in-domain data. The selected portions from all corpora are used as additional corpora to train the adapted language model. The advantages of this method are obtaining an adapted smaller LM that better matches the in-domain data and requires less training data.
The results of the data selection technique for EGY_DF genre is shown in Table 3.11. We observed that a good portion (i.e. 73%) of the corpus e103 was selected, this was expected since this corpus contains English text that was translated by human not native text. A LM is trained using these selected sentences in addition to the target side of BOLT bilingual corpora. The data selected from each corpus as well as other corpora are used to build individual 4-gram back-off LM using modified Kneser-Ney smoothing implemented in the SRILM toolkit. The final LM is built by interpolating these individual LMs. The interpolation coefficients are calculated to minimize perplexity using EM procedure on the English side of the concatenation of the EGY_DF tune sets (d12 + p1r6). The same technique was used to adapt the LM for EGY_SMS and EGY_CTS genres.

3.7.2 Bilingual corpora data selection

The data amount provided by LDC is approximately 193 million words of bilingual corpora. We performed system domain adaptation by using only a portion of these huge bilingual corpora that is most relevant to our task. We used XenC, which implements the cross-entropy bilingual data selection proposed by [Axelrod et al., 2011].

XenC is used to select a subset of parallel sentences which are the most relevant

corpus	full size En tokens	selected %	selected size En tokens
e103	4.45m	73	3.2m
isi	34.7	8	2.7m
un	127.7m	2	2.5m
forum3_1k	666.4m	5	33.3m
cna	44.1m	3	1.3m
ltw	326.9m	5	16.3m
wpb	20.9m	12	2.5m
nyt9x	772.4m	4	30.8m
nyt2xa	554.8m	3	16.6m
nyt2xb	385.9m	4	15.4m
apw9x_2k	392.2m	4	15.6m
apw2xa	550.1m	3	16.5m
apw2xb	482.1m	3	14.4m
afp9x	156.8m	4	6.2m
afp2xa	311.4m	3	9.3m
afp2xb	399.6m	4	15.9m
xin9x	106.8m	3	3.2m
xin2x	270.3m	3	8.1m
Total	5625.3m	-	231.7m

Table 3.11: The size and percentage of the selected data from monolingual corpora (including the English gigaword) for EGY_FORUM system.

to the task. If In_{source} and Out_{source} are the in-domain and out-of-domain corpora of the source language. And, In_{target} and Out_{target} are the in-domain and out-of-domain corpora of the target language. Firstly, XenC creates two in-domain LMs ($LM_{In_{source}}$ and $LM_{In_{target}}$) using the in-domain source and target corpora. Secondly, it creates two out-of-domain LMs ($LM_{Out_{source}}$ and $LM_{Out_{target}}$) using the out-of-domain source and target corpora. For each parallel lines s_s and s_t in Out_{source} and Out_{target} respectively, the monolingual cross-entropy differences are calculated (i.e. $H_{In_{source}}(s_s) - H_{Out_{source}}(s_s)$ and $H_{In_{target}}(s_t) - H_{Out_{target}}(s_t)$) using the cross-entropy Equation 2.19 on Page 30. Finally, the score of each line is calculated by the sum between the two cross-entropy differences, as in the following equation:

$$Score_{(s_s,s_t)} = [H_{In_{source}}(s_s) - H_{Out_{source}}(s_s)] + [H_{In_{target}}(s_t) - H_{Out_{target}}(s_t)] \quad (3.2)$$

Then XenC selects lines based on a score cutoff optimized on held-out in-domain data. This method was particularly effective for the large generic corpora like UN corpus: only about 3% of the data was preserved for EGY_DF genre and 1% for both EGY_SMS_CHAT and EGY_CTS genres. The final EGY_DF genre system was trained on 20M words from different genres using this method. The result of data selection on each genre is shown in Table 3.12. These models perform better than those trained on all available data and they are much smaller.

system genre	corpus	corpus genre	full size Ar/En tokens	selected %	selected size Ar/En tokens
EGY_DF	ummah	MSA_NW_WB	3.61m/3.72m	85	3.03m/3.16m
	un	FORMAL_UN	134.88m/127.71m	3	4.03m/3.83m
EGY_SMS_CHAT	gale	MSA_NW_WB	4.28m/5.01m	3	128k/158k
	bolt	EGY_DF	1.70m/2.05m	8	136k/165k
	e103	MSA_NW_WB	4.44m/4.45m	1	44k/46k
	isi	MSA_NW_WB	35.44m/34.71m	1	354k/348k
	bbnturk	EGY_DF	1.52m/1.58m	11	167k/177k
	bbnlev	LEV_DF	1.59m/1.81m	7	111k/124k
	fix	MSA_NW_WB	1.22m/1.43m	6	73k/84k
	ummah	MSA_NW_WB	3.61m/3.72m	1	36k/37k
	un	FORMAL_UN	134.88m/127.71m	1	1.34m/1.27m
EGY_CTS	smschat	EGY_SMS_CHAT	650k/841k	89	579k/749k
	gale	MSA_NW_WB	4.28m/5.01m	3	128k/157k
	bolt	EGY_DF	1.70m/2.05m	6	102k/123k
	e103	MSA_NW_WB	4.45m	1	44k/46k
	isi	MSA_NW_WB	35.44m/34.71m	1	354k/350k
	bbnturk	EGY_DF	1.52m/1.58m	10	152k/162k
	bbnegy	EGY_DF	514k/588k	67	344k/390k
	iraq	IRQ_DF	1m/1.14m	2	20k/23k
	bbnlev	LEV_DF	1.59m/1.81m	4	63k/70k
	fix	MSA_NW_WB	1.22m/1.43m	2	24k/27k
	ummah	MSA_NW_WB	3.61m/3.72m	1	36k/37k
	un	FORMAL_UN	134.88m/127.71m	1	1.34m/ 1.30m

Table 3.12: The size and percentage of the selected data from bilingual corpora for different system genres.

3.7.3 Translation model domain adaptation

We used the method called perplexity minimization for translation model domain adaptation which is proposed in [Sennrich, 2012]. This method performs instance weighting of translation models, based on the sufficient statistics. It separately optimizes the four features of the log-linear translation model through perplexity optimization. This is done using perplexity minimization for weighted counts, and a modified implementation of linear interpolation. Sennrich [2012] proposed performing perplexity minimization independently for the four features of the standard moses SMT translation model: the phrase translation probabilities $p(\overline{s}|\overline{t})$ and $p(\overline{t}|\overline{s})$, and the lexical weights $lex(\overline{s}|\overline{t})$ and $lex(\overline{t}|\overline{s})$. $\overline{s}$ and $\overline{t}$ denote the source and target phrases.

Traditionally, the phrase translation probabilities $p(\overline{s}|\overline{t})$ and $p(\overline{t}|\overline{s})$ are estimated through unsmoothed maximum likelihood estimation (MLE) by Equation 2.7 on page 26.

In order to combines statistics from a vector of n component corpora, Sennrich [2012] uses a weighted version of equation 2.7, by adding a weight vector λ of length n :

$$p(\bar{s}_i|\bar{t}_i;\lambda) = \frac{\sum_{i=1}^{n} \lambda_i c_i(\bar{s}_i,\bar{t}_i)}{\sum_{i=1}^{n}\sum_{\bar{s}} \lambda_i c_i(\bar{s},\bar{t}_i)} \tag{3.3}$$

An objective function is need in order to perform the translation model perplexity minimization and find the optimized weights for the different TM components in mixture modelling. The used objective function is the minimization of the cross-entropy with the weight vector λ as argument to get the best weight vector $\hat{\lambda}$ as in the following equation:

$$\hat{\lambda} = \underset{\lambda}{\operatorname{argmin}} - \sum_{x,y} \hat{p}(x,y) \log_2 p(x|y;\lambda) \tag{3.4}$$

A development set is needed to train a model with the same word alignment and phrase extraction tools that were used for training (i.e. Giza++). We can then, extract the phrase pair (x,y) and get their empirical probability $\hat{p}$ from the development set translation model that we trained. p is the model probability.

Sennrich [2012] uses L-BFGS with numerically approximated gradients [Byrd

et al., 1995] to perform the optimization.

System	tune set	d10 dev	d12 dev	p1r6 dev
Baseline	-	5.45	18.16	16.61
Adapted_d12	d12 (EGY_DF)	5.45	**17.77**	**16.31**
Adapted_p1r6	p1r6 (EGY_DF)	6.27	18.69	17.04
Adapted_d10	d10 (MSA_NW_WB)	**4.71**	18.25	17.54
Adapted_d12+p1r6	d12+p1r6(EGY_DF)	6.36	18.75	17.33
Adapted using LM interpolation coefs	manual weights	5.79	18.25	16.53

Table 3.13: TB2 *scores for several systems' translation models adapted on different tune sets (or LM interpolation coefs)*

In a group of experiments, this method is used to adapt the system on different tune sets from different genres (i.e. MSA_NW_WB and EGY_DF). The experimental results obtained are shown in Table 3.13. This shows the effectiveness of translation model adaptation on d10 (i.e. MSA_NW_WB), with a gain up to 0.74 for d10 dev on TB2 over unadapted baseline system, and when adapter on d12 (i.e. EGY_DF) a gain up to 0.39 for d12 dev over unadapted baseline system. I also used another method to calculate the translation models weights by creating individual LM for each source side corpus of the bilingual corpora, then interpolated them and optimized the coefficients on d12+p1r6.

I used the LMs interpolation coefficients as weights for the four features of the translation models (i.e. $p(\overline{s}|\overline{t})$ and $p(\overline{t}|\overline{s})$, $lex(\overline{s}|\overline{t})$ and $lex(\overline{t}|\overline{s})$). The TB2 score of using these weights was relatively better for p1r6 (i.e. EGY_DF) but worse for the other two sets with a loss of 0.34 on d10 set (i.e. MSA_NW_WB). We can understand the reason of these results by comparing the LMs interpolation weights to the best score weights (i.e. adapted_d12) as shown in Figure 3.2. We observed that the bolt model got the similar weight which explains the slightly better score on p1r6, but for the rest of the models, they have relatively lower weights (i.e. thy, iraq, bbnegy, bbnlev and fouo) which did not allow the final model to benefit from these bilingual corpora. This could explain the loss in d10 and d12 sets. Since we focus on improving Egyptian dialect translation without degrading the translation of MSA data, we chose to adapt our translation model

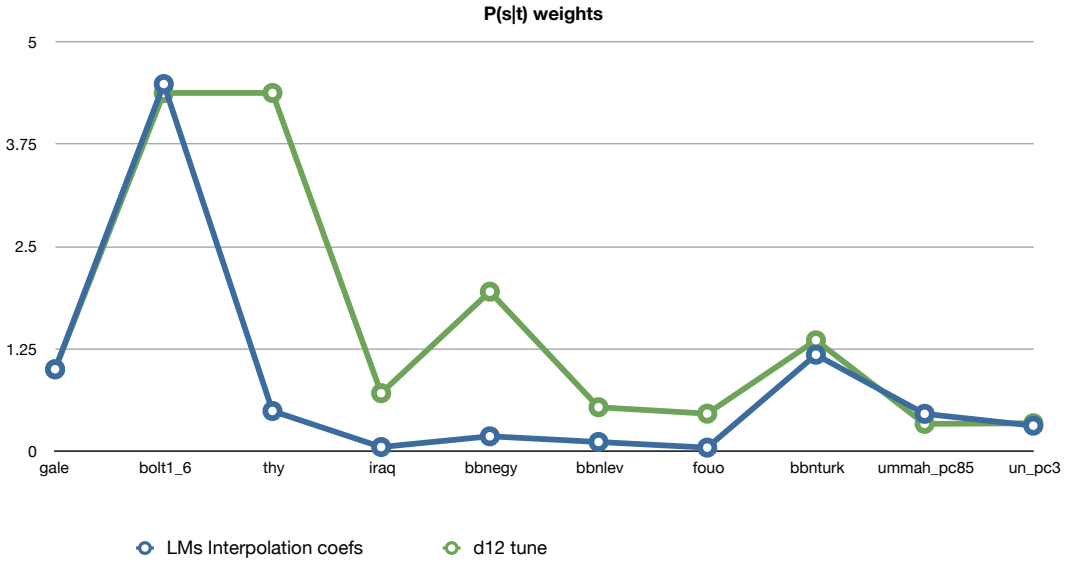

Figure 3.2: Comparing the automatically assigned weights for feature $P(s|t)$ with the weights calculated using LMs interpolation technique.

on d12 tune set which has the best scores for d12 and p1r6 dev sets. Some examples of the translation output are shown in Table 3.14

I performed some analysis on the results by study the weights assigned to each feature. The weights assigned automatically for each translation model feature are shown in Figures 3.3, 3.4, 3.5 and 3.6. We observed that using d12 tune set (i.e. the yellow line) to optimize the weights gives reasonable weights for all the four features especially for bolt and thy bitext, while using p1r6 or the combination of d12+p1r6 gives more weight on thy and much less weight on bolt bitext. We concluded that d12 tune set has better correlation, than p1r6, with bolt corpus which is the main training data of Egyptian dialect in BOLT project. This conclusion is also confirmed by the better TB2 scores when the same set (i.e. d12 tune) is used to optimize the system log-linear features weights using MERT.

One of the disadvantages of this technique is that we can adapt the SMT system on one genre only. If we would like to adapt the system on two dialects like Iraqi and Egyptian, we have to build two separate systems, each system

description	sentence
source	لكن في التطبيق ما # فيش كده خالص .
reference	but in application there is nothing like that at all .
unadapted system	but in practice what fish like this at all .
adapted system	but in practice there is nothing like this at all .
source	كبري دماغ +ك من +هم .
reference	never mind them
unadapted system	the largest your brain from them .
adapted system	widen your mind from them .
source	هاتوا سينا تاني ..
reference	bring sinai again ..
unadapted system	get sina again .
adapted system	get sinai again .
source	اما ب # النسب +ه ل # مستر حمدين صباحي فا هو .
reference	as for mr. hamdein sabbahy , he is
unadapted system	as for his lineage to mr hamdeen sabahi , so it is .
adapted system	as the percentages to mr hamdin sabbahi , so it is .
source	سوري بس تخنق مش كلم +ه دقيقه .
reference	sorry , but suffocating is not an accurate word .
unadapted system	syrian but choke do n't talk to him a minute .
adapted system	sorry , but really annoying not call him a minute .

Table 3.14: *First 3 examples for improved translation and last two examples of not improved translation when TM adaptation is used*

would be adapted on one dialect. We can overcome this problem by using a multi-domain architecture explained in the next section.

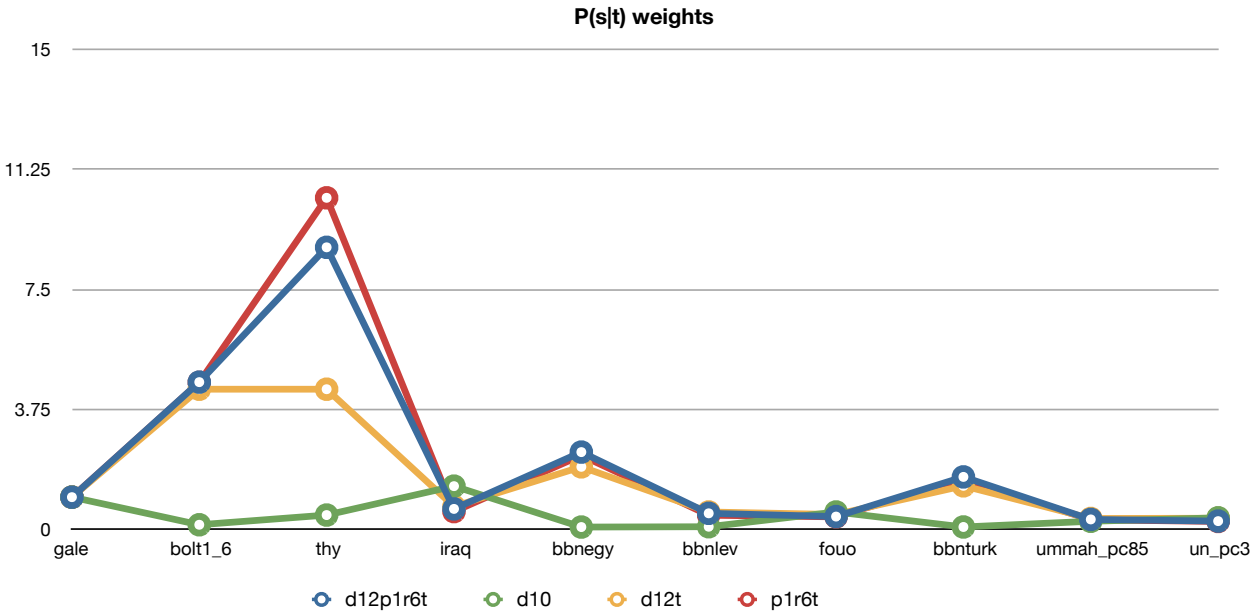

Figure 3.3: Automatically assigned weights for feature $P(s|t)$ for different translation models.

3.7.4 Multi-domain translation model

Domain adaptation techniques for SMT have proven to be effective at improving translation quality as explained in Section 3.7.3, but their usage in a multi-domain environment is often limited because of the computational and human costs of developing and maintaining multiple systems adapted to different domains.

In [Sennrich et al., 2013], we present an architecture that delays the computation of translation model features until decoding, allowing dynamic instance weighting using optimized weights. Also a method for unsupervised adaptation with development and test data from multiple domains (i.e. MSA and Egyptian dialect in our case) is described. An unsupervised method to cluster the sentences of the development set is presented. This is done by train a language model on the source language side of each of the n component bitexts, and compute an n-

Figure 3.4: Automatically assigned weights for feature $P(t|s)$ for different translation models.

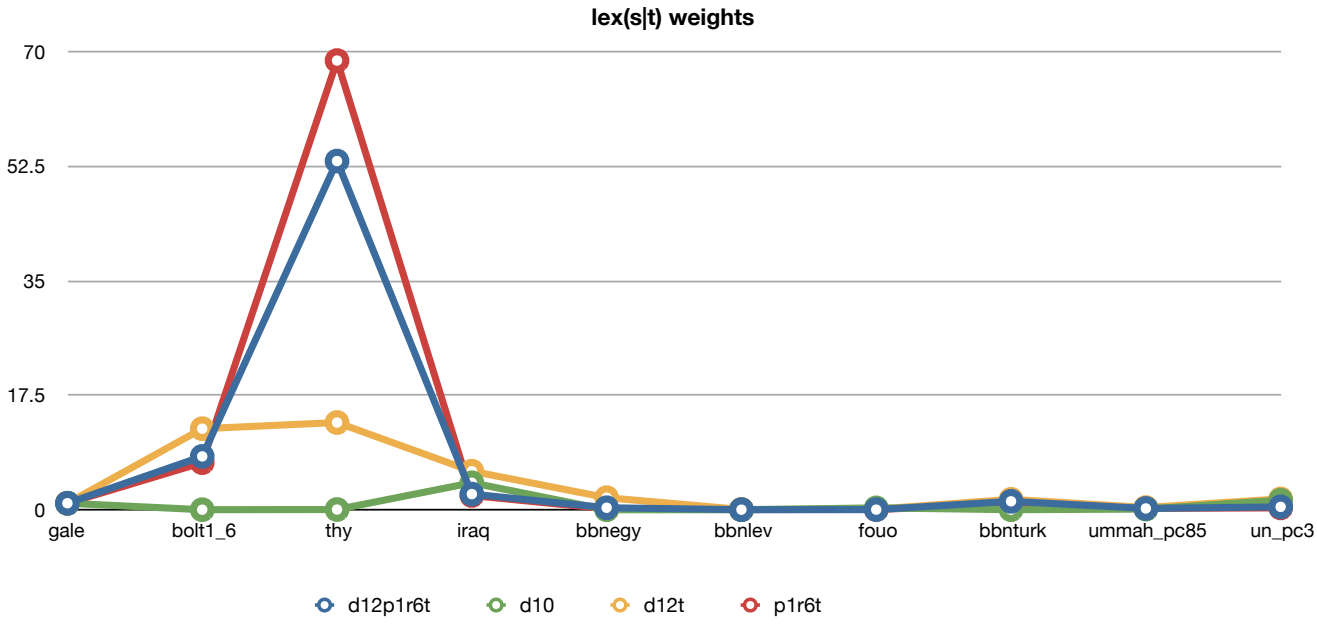

Figure 3.5: Automatically assigned weights for feature $lex(s|t)$ for different translation models.

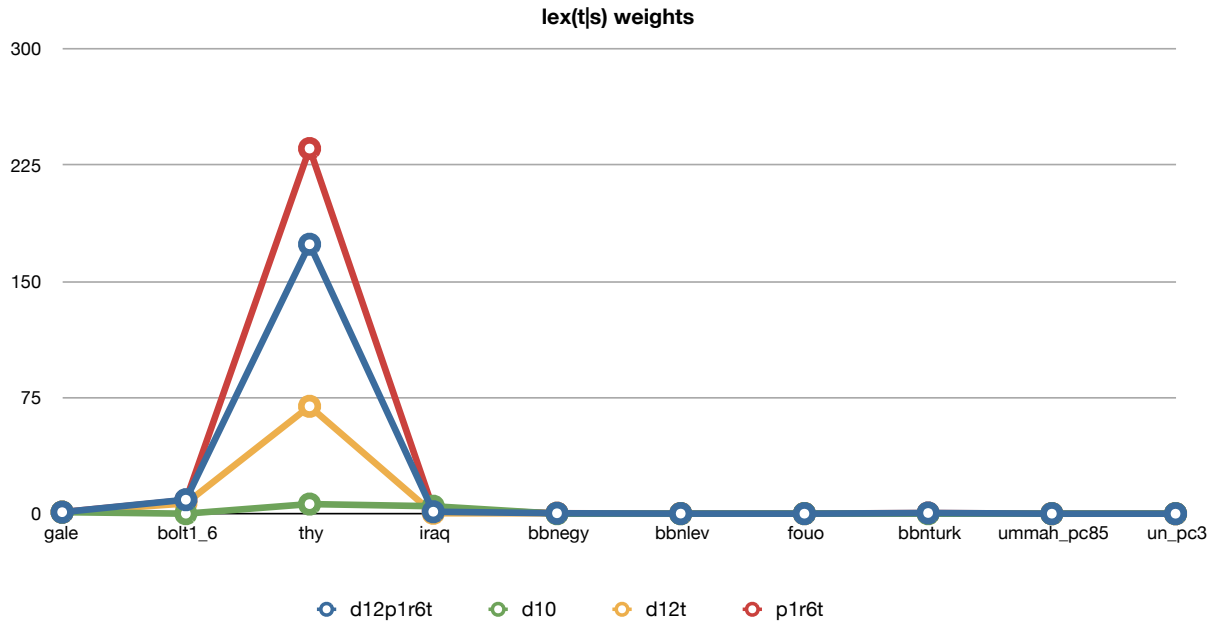

Figure 3.6: Automatically assigned weights for feature $lex(t|s)$ for different translation models.

dimensional vector for each sentence by computing its entropy with each language model. A k-means clustering algorithm is used to cluster these vectors using cosine similarity measure. A bitext for each cluster is obtained, which is used to optimize the model weights using the same method presented in Section 3.7.3. At decoding time for test set, a cluster and its associated optimized weight vector are assigned to each sentence. Cosine distance of the sentence n-dimensional vector and each centroid are used to find the closest cluster. This allows the adaptation even with unlabeled and heterogeneous test data.

3.7.4.1 Translation model architecture

The architecture has two goals: move the calculation of translation model features to the decoding phase, and allow for multiple knowledge sources (e.g. bitexts or user-provided data) to contribute to their calculation.

In order to compute the translation model features online, a number of suf-

ficient statistics need to be accessible at decoding time. For $p(\overline{s}|\overline{t})$ and $p(\overline{t}|\overline{s})$, the statistics $c(\overline{s})$, $c(\overline{t})$ and $c(\overline{s},\overline{t})$ are required. For accessing them during decoding, they are simply stored in the decoder's data structure, rather than storing pre-computed translation model features.

The statistics are accessed when the decoder collects all translation options for a phrase $\overline{s}$ in the source sentence. Then, all translation options for each component table are accessed, obtaining a vector of statistics $c(\overline{s})$ for the source phrase, and $c(\overline{t})$ and $c(\overline{s},\overline{t})$ for each potential target phrase. For phrase pairs which are not found, $c(\overline{s},\overline{t})$ and $c(\overline{t})$ are initially set to 0.

For $lex(\overline{s}|\overline{t})$, we require an alignment a, plus $c(t_j)$ and $c(s_i,t_j)$ for all pairs (i,j) in a. $lex(\overline{t}|\overline{s})$ can be based on the same alignment a (with the exception of NULL alignments, which can be added online), but uses statistics $c(s_j)$ and $c(t_i,s_j)$.

The architecture can thus be used as a drop-in replacement for a baseline system that is trained on concatenated training data, with non-uniform weights only being used for texts for which better weights have been established. This can be done either using domain labels or unsupervised methods as described in the next section.

This architecture supports decoding each sentence with a separate weight vector of size $4n$, n the number of TM components and 4 is the number of translation model features whose computation can be weighted.

The good weights are automatically selected for each sentence by optimizing instance weights using a set of phrase pairs automatically extracted from a parallel development set.

The basic idea consists of three steps:

1. Cluster a development set into k clusters.
2. Optimize translation model weights for each cluster.
3. For each sentence in the test set, assign it to the nearest cluster and use the translation model weights associated with this cluster.

For step 2, we use the algorithm by Sennrich [2012] as detailed in Section 3.7.3, implemented in the decoder to allow for a quick optimization of a running system.

Next section gives more detail on steps 1 and 3.

3.7.4.2 Clustering the tune set

The development set is clustered using k-means clustering algorithm. A language model on the source language side of each of the n component bitexts are trained, n-dimensional vector for each sentence are computed by computing its entropy with each language model. We used the measure of cosine similarity since we would like to cluster on the basis of relative differences between the language model entropies.

The result of the development set clustering to k clusters is obtaining a bitext for each cluster. Each bitext is used to optimize the model weights. The centroid of each cluster is calculated. During decoding, each test set sentence is assigned to the centroid that is closest to it in the vector space using cosine similarity measure.

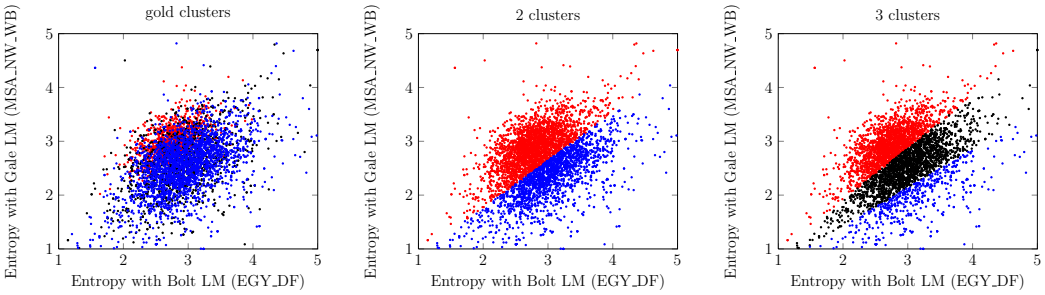

Figure 3.7: Clustering of d10+d12+p1r6 tune set which contains sentences from two domains: MSA_NW_WB and EGY_DF. Comparison between gold segmentation, and clustering with cosine similarity/distance measures. red: MSA_NW_WB; bleu: EGY_DF; black: mixed MSA_NW_WB and EGY_DF.

The weight vector is set globally, but can be overridden on a per-sentence basis. A chart of the gold clusters vs. two and three automatic clusters of the tune set d10+d12+p1r6 are shown in Figure 3.7. For illustration purpose only, in this figure I presented a 2D chart of the clusters using only two bitext (i.e. Gale and Bolt) and comparing them to the gold clusters which are both MSA_NW_WB (i.e. d10) and EGY_DF (i.e. d12 and p1r6) genres in this case. We observed that the clustering technique was able to cluster the concatenated tune sets to two and

three clusters; each cluster is closer to either EGY_DF in blue or MSA_NW_WB in red. I will show more result analysis in the next section.

3.7.4.3 Experiments and results

The experiments are done using two phases: offline phase and online phase. In the offline phase, we train the individual component models, cluster the tune set and compute the optimal weight vector for each cluster (with perplexity minimization). In the online phase, we assigned each sentence in the test set to the closest cluster, translated it using the cluster's corresponding weight vector and evaluated the output using TB2 metric.

I used this architecture to build a multi-domain system on both MSA_NW_WB and EGY_DF. I experimented with several tune sets (i.e. d10+d12+p1r6 and d12+p1r6) as the tune set to be clustered. The experimental results are shown in Table 3.15. The results show the effectiveness of multi-domain translation model, with a TB2 gain of up to 0.58 for MSA_NW_WB and a TB2 gain of up to 0.6 for Egyptian dialect over unadapted baseline system. If we compared it to the adapted baseline we can observe a similar gain of up to 0.58 for MSA_NW_WB and a small gain of up to 0.15 for EGY_DF (i.e. just for d12) on TB2. The best scores on both MSA_NW_WB and EGY_DF are obtained by clustering d12 tune set to 32 clusters.

In order to do some result analysis, we need to look at the simple case of two automatic clusters and the automatic weights assigned to each one. Since we have good weights for baseline2 (i.e. in second row in the table) using the method detailed in Section 3.7.3 on d12 tune set, it would be good to compare them to the new assigned weights of each cluster. In Figure 3.8, for illustration purpose only, I presented a 2D chart of the clusters of d12+p1r6 tune set using only two bitext (i.e. Gale and Bolt) and comparing them to the gold clusters which are all EGY_DF genre in this case. Even though the two sets are informal EGY_DF, the clustering technique is still able to cluster them to two clusters; each cluster is closer to either EGY_DF or MSA_NW_WB. We confirmed this observation by

System	clustered tune set	#clusters	d10	d12	p1r6
Baseline	unadapted		5.45	18.16	16.61
Adapted_d12 (Baseline2)	d12 (Instance weighting)	-	5.45	17.77	**16.31**
SYS1_C2	set1	2	5.25	17.75	**16.48**
SYS3_C2	d12		5.15	**17.62**	**16.44**
SYS1_C3	set1	3	5.05	17.89	16.50
SYS2_C3	set2		5.12	17.79	**16.47**
SYS3_C3	d12		4.99	17.74	**16.43**
SYS1_C4	set1	4	5.06	17.72	16.52
SYS1_C5	set1	5	5.03	17.74	**16.46**
SYS2_C5	set2		5.28	17.72	16.52
SYS1_C8	set1	8	4.94	**17.66**	**16.44**
SYS2_C8	set2		4.96	17.69	16.56
SYS1_C16	set1	16	4.91	17.76	16.52
SYS2_C16	set2		4.99	17.73	16.52
SYS1_C32	set1	32	**4.87**	**17.66**	**16.50**
SYS2_C32	set2		4.94	**17.62**	**16.48**
SYS3_C32	d12		4.98	**17.56**	**16.29**
SYS1_C40	set1	40	**4.86**	17.72	16.59
SYS2_C40	set2		**4.79**	17.76	16.71
SYS3_C40	d12		**4.89**	**17.58**	16.64

Table 3.15: *Using the tune sets: set1= d12+p1r6, set2= d10+d12+p1r6 and d12 with different number of clusters in multi-domain adaptation (scores in* TB2*)*

comparing the automatically assigned weights for each cluster (using all bilingual corpora as in our experiments) as shown in Figure 3.9. I observed that cluster 2 has larger weights for the main EGY_DF corpora (i.e. bolt, thy, bbnegy and bbnturk), while cluster 1 has lower weights for them.

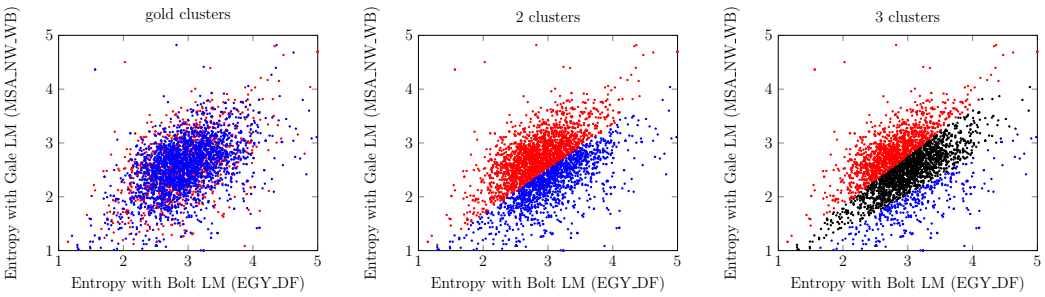

Figure 3.8: Clustering of d12+p1r6 tune set which contains sentences from two domains: MSA_NW_WB and EGY_DF. Comparison between gold segmentation, and clustering with cosine similarity/distance measures. red: MSA_NW_WB; bleu: EGY_DF; black: mixed.

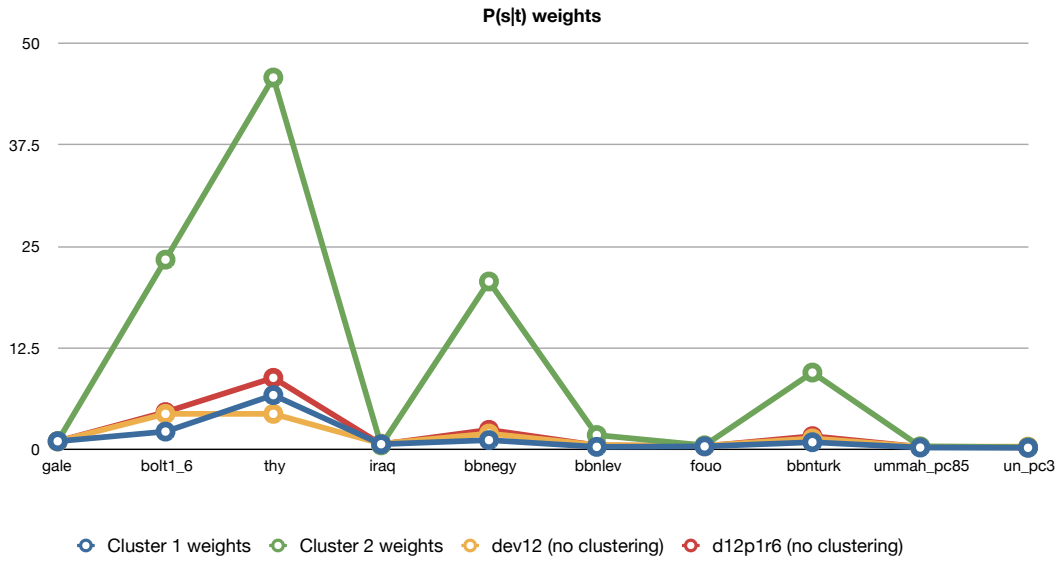

Figure 3.9: Comparing the four automatically assigned feature $p(s|t)$ weights of each cluster of d12+p1r6 tune set and when no clustering is used (for both tune sets d12 and d12+p1r6).

This means that cluster 2 is closer to EGY_DF than cluster 1 and so it gets

larger weights on EGY_DF bitext corpora. if we compare cluster 1 weights to the weights of our instance weighting on d12 in the second row in Table 3.15 (i.e. this is the same result presented in Table 3.13), we observed that cluster 1 got slightly similar weights which means some how that cluster 1 represents data that are mix of both MSA_NW_WB and EGY_DF as the case of d12 tune set. Looking at TB2 scores of SYS1_C2, we observed also that the clustering technique benefits d10 score (i.e. MSA_NW_WB) compared to the adapted baseline system adapted_d12, and because of the larger weights assigned to cluster 2 on EGY_DF corpora, it did not lose much on d12 and p1r6 (i.e. EGY_DF sets).

In general, we concluded that because Egyptian dialect is a mixture of MSA and additional dialectal words and dialectal structure, the clustering method helps clustering them and assigns different weights to each cluster. It is important to point out to the fact that clusters are not necessary representatives of EGY_DF and MSA_NW_WB with different degrees. It is difficult to label these unsupervised clusters especially for large number of clusters because they could be representatives of other features with different degrees (e.g. different degrees of EGY_DF and MSA_NW_WB, styles, genres, other dialects.. etc.)

In the context of BOLT project, we did not integrate this technique into our main delivered systems because the implementation was based on Moses server which does not support generating the n-best list which we need to apply CSLM re-scoring. Since CSLM re-scoring gives higher gain compared to the multi-domain system, we preferred using it in BOLT.

3.7.5 Adaptation using lightly supervised training

We used the method proposed by Schwenk [2008b], by applying lightly supervised training of the translation model to adapt the system to the EGY_DF genre as shown in Figure 3.10. In this technique, we are using automatic translation of large amount of in-domain monolingual text (i.e. Egyptian dialect in our case) to improve and adapt the baseline SMT system for in-domain translation task. This is done basically by adding portion of this large amount of new bitext, which consists of the source sentences and their automatic translation, to our SMT system

training data. This technique can be named unsupervised or lightly supervised training depends on the existence or the absence of the in-domain (i.e. discussion forum) training corpora in the language model of the system used for translation. We used lightly supervised name since our language model training data includes some in-domain monolingual data (i.e. forum3 in Table 3.11).
In order to improve the quality of the automatic translation we applied the following techniques:

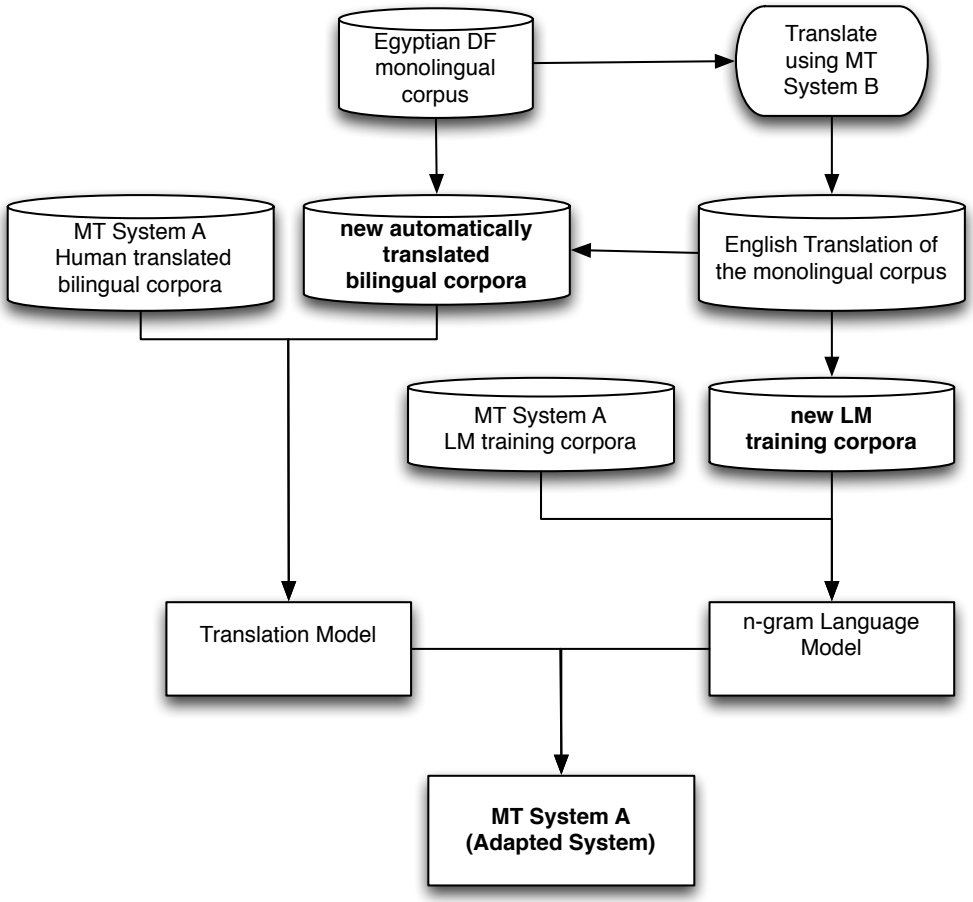

Figure 3.10: The lightly supervised training adaptation

- The SMT system that we used for automatic translation was built using instance weighting explained in Section 3.7.3 and the fill-up method proposed

by Bisazza et al. [2011]. The fill-up method is used to enrich our translation model with additional unknown phrases/vocabulary (i.e. unseen in the training data) from another large translation model that was built using additional large amount of out-of-domain bitext (i.e. formal MSA UN/News parallel corpora in our case). This helped decreasing the number of OOV words in our monolingual automatic translation task. The fill-up preserves all the entries and scores coming from the first model, and adds entries from the other models only if it is new. Moreover, a binary feature is added for each additional table to denote the provenance of an entry. These binary features work as scaling factors that can be tuned directly by MERT along with other features' weights.

- CSLM re-scoring [Schwenk, 2010] had been applied on 1000-best automatic translation list to re-rank them as it is proven to give better ranking and hence better TB2 score.

The portion that we added to our SMT training data was selected based on the bilingual data selection explained in Section 3.7.2 in order to score the new bitext and sort them according to sentence pairs which are more relevant to our domain (i.e. EGY_DF). In order to determine the best amount of data we can use from this new artificial bitext, we used empirical method by trying different amounts of them and study the impact on the system TB2 score. Table 3.16 lists the size of the new automatic bilingual corpora and the portions that we selected and added to LIUM BOLT EGY_DFsystem.

LDC ID	Size Ar/En tokens	Best selected amount Ar/En tokens
ldc2012e16d4	34m/38m	10.1m/11.7m
ldc2012e16d1	158.4m/176.3m	29.8m/34.3m
ldc2012e04	56.5m/63m	30.1m/34.5m
TOTAL	248.8m/277.3m	70.1m/80.5m

Table 3.16: The sizes of automatically translated monolingual Egyptian dialect corpora and the selected portion from it as used in BOLT project.

For Egyptian dialect, experimental results shown in Table 3.17 demonstrate the effectiveness of lightly supervised training, with a gain up to 0.2 over the

System	bitext/lightly sup. tokens	d10 (MSA_NW_WB)	p1r6 (EGY_DF)
Baseline1 (instance weighting+fillup)	16M/0	5.31	16.22
+lightly sup. bitext	16M/80M	5.42	16.02
+CSLM	16M/80M	**3.92**	**15.51**

Table 3.17: TB2 results of experiments using lightly supervised training to adapt the SMT system.

baseline system on p1r6 (i.e. EGY_DF). The gains increases to 0.71 after applying CSLM re-scoring. As expected, adaptation of the system using lightly supervised method degraded the translation quality of MSA_NW_WB which we accepted since our focus was on improving EGY_DF translation performance.

3.8 Operation sequence model

Durrani et al. [2011] presented a novel machine translation model which uses a linear sequence of operations to model the translation. This sequence includes both translation and reordering operations. The key ideas of this model are (i) new reordering operations that provide better restriction on the position that a word or phrase can move to. It also supports both long and short distance reordering, and (ii) a more flexible joint sequence model for the translation and reordering probabilities compared to the standard phrase-based MT. Durrani et al. [2011] reported that a statistically significant improvements have been achieved on BLEU for German-to-English and Spanish-to-English tasks, and comparable results for a French-to-English task.

The new generative model treats the translation process as a linear sequence of operations. The source and the target sentence are generated in parallel by these operations. The operations are: generation of a sequence of source and target words, gaps insertion as explicit target positions for reordering operations, and forward and backward jump operations which do the actual reordering. An n-gram model of the operations is used to estimate the probability of a sequence of

operations. This means that the reordering operation may depend on preceding operations like generation and vice versa. This is because the operations are coupled in single generative story. This allows a natural consistent reordering operation that can deal with long distance re-ordering as well as local re-ordering operations.

The experimental results of using OSM in our BOLT EGY_DF system are shown in Table 3.18 with improvement between 0.42 and 1 on TB2 metric.

System	d12 tune	d10 dev	d12 dev	p1r6 dev
Baseline	15.88	5.19	17.40	15.87
+OSM	**15.46**	**4.14**	**16.75**	**15.23**

Table 3.18: TB2 *scores for experiments of using OSM to improve LIUM SMT system.*

3.9 Word sense disambiguation technique

Several researches were conducted on incorporating word sense disambiguation (WSD) in SMT. Carpuat and Wu [2007] found that incorporating the predictions of a WSD system within a typical phrase-based SMT model consistently improves translation quality across all three different IWSLT Chinese-English test sets, as well as producing statistically significant improvements on the larger NIST Chinese-English MT task. They consistently integrate WSD models both during training, where sense definitions and sense-annotated data are automatically extracted from the word-aligned parallel corpora from SMT training, and during testing, where the phrasal WSD probabilities are used by the SMT system just like all the other lexical choice features. They extracted the context features from state-of-the-art WSD models. The evaluation is conducted on the actual translation task, rather than intermediate tasks such as word alignment. In my work, no sense-annotated data is used and the senses are not extracted from the word alignment but from the pre-trained phrase table. Also Chan et al. [2007], integrates a state-of-the-art WSD system into a state-of-the-art hierarchical phrase-based MT system, Hiero. They show that integrating a WSD system improves the performance of a state-of-the-art statistical MT system. In this

work, I also focus on using word sense disambiguation to improve SMT performance but using context vectors modeling. I focus on how to improve an SMT system for Egyptian by applying word sense disambiguation techniques on ambiguous words using their context. The goal is to help the SMT system to decrease the number of wrong translations and by these means, improve its performance. There are huge amounts of mono-lingual data available for Arabic, Egyptian, English and many other languages compared to the size of the available bilingual corpora. The idea is to utilize these data instead of bilingual data which are sparse resources. We used context vector representations of words to capture the word similarity as well as other syntactic and semantic regularities in the language.

Recently, Mikolov et al. [2013b] introduce Continuous Bag-of-Words (CBOW) models, an efficient method for learning high-quality vector representations of words from large amounts of unstructured text data. They propose an architecture that is similar to the feed-forward continuous space language model, where the non-linear hidden layer is removed and the projection layer is shared for all words (not just the projection matrix); thus, all words get projected into the same position (their vectors are averaged). CBOW uses continuous distributed representation of the context. The model architecture is shown in Figure 3.11. Note that the weight matrix between the input and the projection layer is shared for all word positions in the same way as in the CSLM.

One of the characteristic of the context vector representation (a.k.a word embeddings) is that similar words are likely to have similar context vectors representation. I utilized this idea in my research to disambiguate the word sense of ambiguous words by finding all possible distinct translations of ambiguous words by calculating the cosine similarity between these various translation and merge the senses who have high cosine similarity scores.

In the next section, I will explain the main idea of utilizing these context vector representations to achieve the following:

- Measure the similarity between words.

- Detect the words which are not ambiguous.
- Extract various senses for each ambiguous word.
- Merge similar senses for each ambiguous word.
- Utilize the new proposed sense table in SMT to assign a sense tag for some words in order to improve the SMT translation performance.

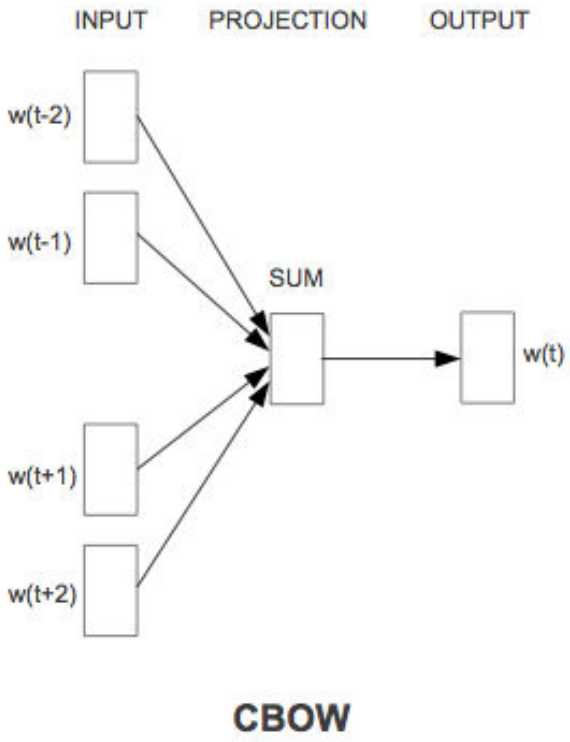

Figure 3.11: Graphical representation of the CBOW model. In the CBOW model, the distributed representations of context (or surrounding words) are combined to predict the word in the middle. source:[Mikolov et al., 2013b]

3.9.0.1 Word senses extraction algorithm

As shown in Figure 3.12, the following is the algorithm I proposed for word senses extraction:

1. Train two context vector models using huge amounts of mono-lingual data in source and target languages. I trained on BOLT Information Retrieval monolingual data (i.e. most of them are EGY_DF) which is about $\sim$ 448 millions tokens. For English, I used the pre-trained vector model released by

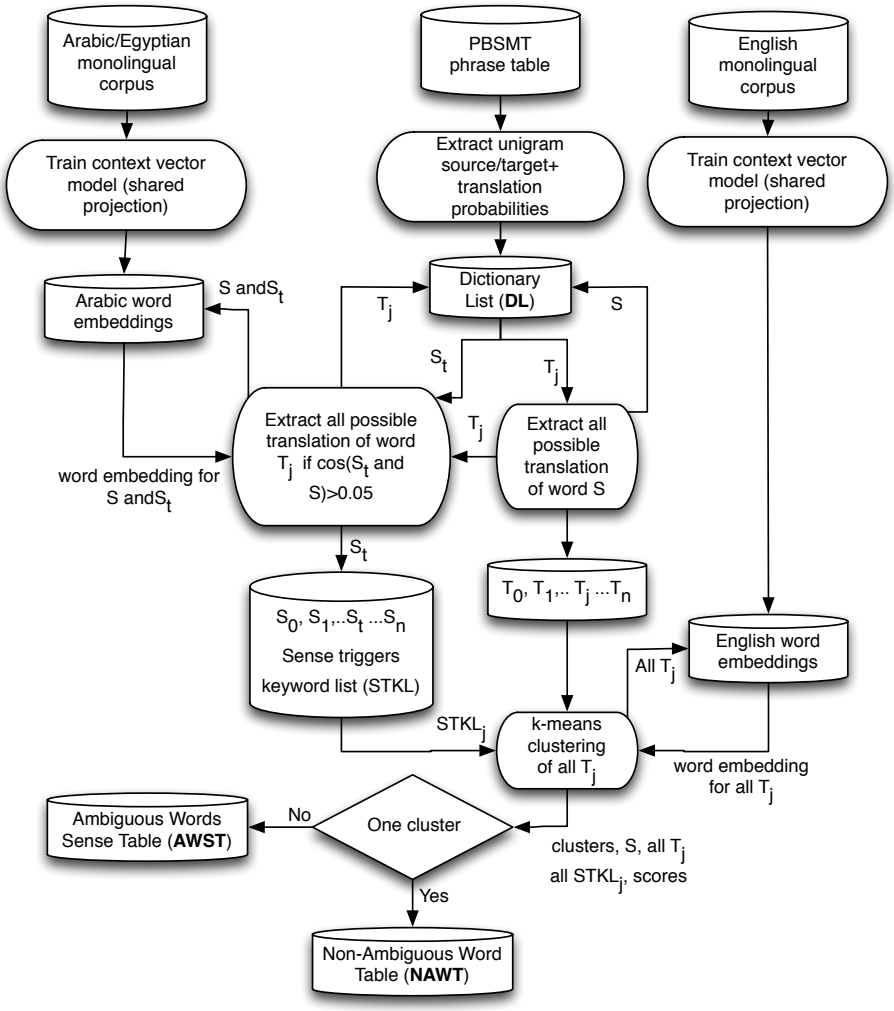

Figure 3.12: Word senses extraction algorithm

Mikolov et al. [2013a] which was trained on part of Google's News dataset (about 100 billion words). I used the word2vec tool to train the models.

2. Extract a Dictionary List (DL) using all unigram source words and their

unigram target translation from a trained translation model (i.e. phrase table) after discarding entries with joint counts less than 5, punctuations unigram and stop words.

3. Find all possible target translations T_j, $j = 0 \ldots n$ of each source word S_i in the extracted DL.

4. Calculate the cosine similarity between the context vector representation of the all target translations T_j and each other, k-means algorithm is used to cluster T_j based on cosine similarity measure.

5. Find all possible source translations S_t, $t = 0 \ldots m$ of each target word T_j using the extracted DL. Assign the target words T_j as default sense IDs for all S_t. We will call it *Sense Trigger Keywords List* ($STKL_{S_i}$) of the source word S_i.

6. Calculate the cosine similarity between the context vector representation of the found possible source translation S_t, $t = 0 \ldots m$ in ($STKL_{S_i}$) and the context vector representation of S_i and discard S_t with cosine similarity less than specific threshold (I used 0.05).

7. If S_t is identical with the source word S_i; we check if the target translation T_t is a transliteration of S_i using a transliteration mining algorithm. If it is a transliteration, then we assign a *NAME* sense ID to S_t instead of T_t which we were using as default sense ID.

8. Output all S_i words that has one sense ID for all S_t in ($STKL_{S_i}$) as non-ambiguous words in the Non-Ambiguous Word Table (NAWT), and the rest of words as ambiguous words with their sense IDs in the Ambiguous Words Sense Table (AWST).

Table 3.19 shows a sample from the NAWT table with words that the algorithm detected to be non-ambiguous words. Table 3.20 gives a sample from the AWST table with words that the algorithm detected as ambiguous words with possible sense IDs.

S_i	T_j	Sense $Keyword_j$
وارسو	Warsaw	وارسو
ليفني	Livni	ليفني
ساركوزي	Sarkozy	ساركوزي
امبارح	yesterday	الامس
امبارح	yesterday	البارحة
امبارح	yesterday	يوم
امبارح	yesterday	امس
اخضر	green	الاخضر
اخضر	green	الخضراء
اخضر	green	اخضر

Table 3.19: *Sample from the NAWT with words that our algorithm detected as non-ambiguous words*

3.9.0.2 Word sense disambiguation algorithm

One way to integrate my work into SMT is to assign a sense ID for each word in the SMT training, tune and test sets. Only ambiguous words will be tagged with the detected Sense ID. As shown in Figure 3.13, the Sense ID tag for the source word S_i is assigned based on the following algorithm:

1. Check if the word S_i is in AWST table of ambiguous words, if not, then do not tag the word and go to the next word.

2. Extract the words neighbor of S_i based on the used window in our context vector space model and calculate the context vector representation V_i by sum and normalize these neighbor words context vectors.

3. Get all senses of S_i from AWST table with the associated $STKL_{S_i}$ for each sense.

4. Calculate the cosine similarity between the neighbor context vector representation V_i and each S_t in $STKL_{S_i}$.

S_i	T_j/Sense ID	Meaning	Sense $Keyword_j$
مرسي	thanks	thanks	مرسي
	thanks	merci	مرسي
	thanks	thanks	الحمد
	thanks	thanks	الحمد
	thanks	thank	احمد
	mercy	mercy	مرسي
	mercy	mercy	يرحم
	NAME	morsi	مرسي
	NAME	mursi	مرسي
	NAME	morsy	مرسي
	port	marsa	مرسي
	port	mersa	مرسي
	anchorage	anchorage	مرسي
	port	portsaid	بورسعيد
	port	port	المرفا
	port	port	مرفا
	port	port	بورت
	port	port	ميناء
	port	ports	الموانئ

Table 3.20: *Sample from the AWST with words that our algorithm detected as ambiguous words; each sense has a sense ID and a sense keyword.*

5. If S_t has the highest cosine similarity with V_i, assign the sense ID associated with it to S_i.

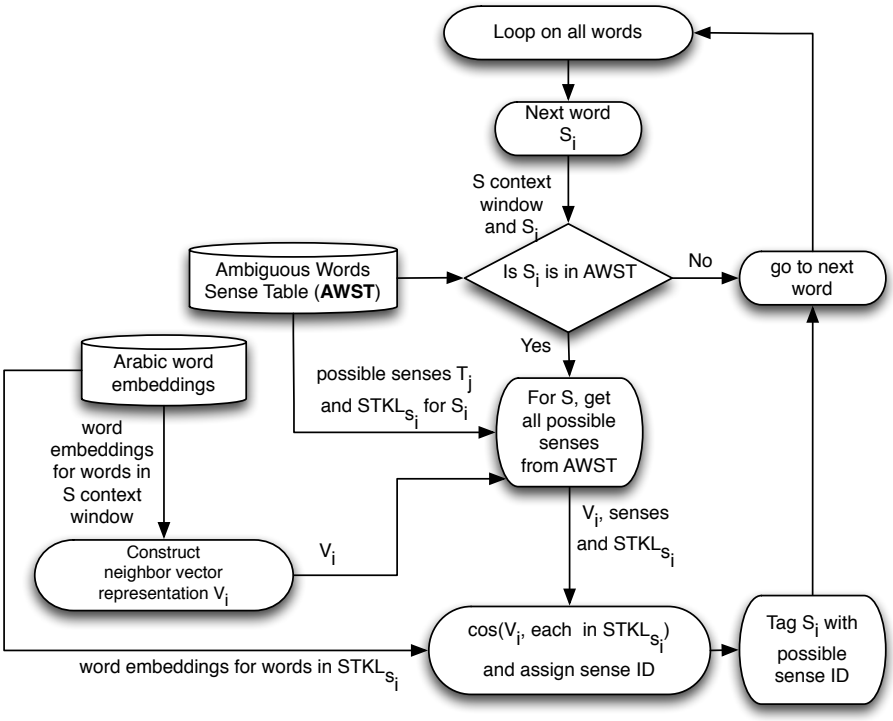

Figure 3.13: Using WSD algorithm based on word embeddings to detect sense IDs for the words in SMT corpora

Table 3.21 shows an example of a tagged sentence using the proposed technique. I used this approach to improve SMT system but it did not improve the final scores. However, by looking into the translation output, some translations are better compared to the baseline system translation as shown in Table 3.22.

3.10 CSLM rescoring

LIUM is developing and using since several years the continuous space language models (CSLM) toolkit. This toolkit was used in BOLT project beside other projects. The theoretical background of CSLM is presented in this Section 2.3.3.2.

Source Word	Meaning	Selected SENSE TAG
نشوف	see	see
مين	who	whose
ل#	has	-
ه+	-	-
مصلحة	interest	interest
ان	that	-
الانتخابات	election	-
ما#	will not	-
تتم	perform	done
ش+	-	-

Table 3.21: *Example of sense tagging of the SMT training data using our approach, ambiguous words are tagged with sense tag, while non-ambiguous words are not tagged*

We trained and adapted CSLM for each genre in BOLT. The main differences between these models are the training data. We used monolingual data selection methods explained in Section 3.7.1. We also used data resampling feature in CSLM toolkit. The resampling coefficients are determined by training individual back-off LM for each corpus in the training corpora, then interpolate them to get the LMs interpolation coefficients which are used as CSLM data resampling. We run CSLM training and re-scoring on 3D graphic cards from Nvidia in order to take advantage of their high computational power. The results of re-scoring with CSLM are shown in Table 3.23. I also worked on improving CSLM models by using additional auxiliary data. This work is presented separately in chapter 5 of this thesis.

3.11 Conclusion

This chapter describes BOLT program and its objectives, scope, data resources and constrains. It covers the activities and the different techniques that we used during BOLT project in different phases. The techniques developed to improve

description	sentence
source	كلام جميل اوي يا فتكات .
tagged source	words$(كلام) beautiful$(جميل) very$(اوي) يا NAME$(فتكات) .
reference	a very beautiful words , fatakat .
tagged translation	a very beautiful words , fatakat .
baseline translation	a very nice words , fatakat .
source	ديمقراطية ب # خطوط حمرا ؟
tagged source	NAME$(ديمقراطية) # ب lines$(خطوط) NAME$(حمرا) ?
reference	democracy with red lines ?
tagged translation	democracy with red lines ?
baseline translation	democracy red lines ?
source	مرسي شكرا ردود +كم يا قماميـر .
tagged source	thanks$(مرسي) thanks$(شكرا) replies$(ردود) قماميـر يا +كم .
reference	merci , thank you for your responses , girls ,
tagged translation	thank you , thank you for your responses , قماميـر .
baseline translation	thank you , thank you your responses , قماميـر .
source	ف # العدل اساس الحكم .
tagged source	ف# fair$(العدل) basis$(اساس) power$(الحكم) .
reference	justice is the basis of ruling .
tagged translation	so justice is the basis of ruling .
baseline translation	so justice is the basis of the power .
source	زغرتي يا ام حسان احنا نازلين الميدان .
tagged source	NAME$(الميدان) keep$(نازلين) here$(احنا) NAME$(حسان) يا زغرتي .
reference	ululate um hassan , we are going to the square .
tagged translation	زغرتي um hassan , we are going to the square .
baseline translation	زغرتي um hassan , we are going down to the square .

Table 3.22: *Example of improved translation by using the sense tagged SMT training data*

CSLM rescoring	d10	d12	p1r6	smschat	cts
No	3.00	15.66	15.24	18.04	16.70
Yes	**1.85**	**14.97**	**14.30**	**16.08**	**15.87**

Table 3.23: *Result of all genres development sets before and after re-scoring with CSLM. Each genre is rescored with a genre adapted CSLM. (scores in* TB2*)*

the translation quality of Arabic/Egyptian into English MT system, but in most cases they can be adapted to other languages with small effort. It also presents the results of LIUM Systems in the three international evaluations of the BOLT project.

Several general and Arabic related techniques have been implemented. One of these techniques is adapting our SMT systems to the Egyptian dialect, since the available training corpora, in the context of BOLT project, contains modern standard Arabic, and several dialects (i.e. Egyptian, Levantine and Iraqi). We improved the system performance by using domain adaptations techniques and treating different dialects as different domains. We used several adaptation techniques to adapt our system on the Egyptian dialect and/or the required system genre. The first technique is using instance weighting of translation models to improve the translation quality by giving more weights to Egyptian than modern standard Arabic or other Arabic dialects. Since our training corpora have various genres (i.e. NEWS, WEB, UN, DF, SMSCHAT and CTS), we adapt our systems by using data selection techniques. Two techniques were used, the first one is used to select the relevant sentences from monolingual corpora to improve and adapt the language models, while the second one is used to select the most relevant sentences from the bilingual corpora to improve the TMs. We also applied another method for the adaptation of SMT systems to Egyptian using the so-called "lightly supervised" training. In this technique, we are using automatic translation of large amount of in-domain monolingual text (i.e. Egyptian dialect in our case) to improve and adapt the baseline SMT system for in-domain translation task. This is done basically by adding portion of this large amount of new bitext, which consists of the source sentences and their automatic translation, to our SMT system training data.

Since Arabic is a morphologically rich language, the selection of the suitable morphological segmentation options is one of the important preprocessing steps in MT research. There are many morphological schemes that can be used to segment the Arabic words. I evaluated various Arabic segmentation schemes from full word form to fully segmented form to explore the effect on the system performance and translation quality.

In order to address ambiguous Arabic/Egyptian words translation errors, I worked on applying word sense disambiguation technique on them using their context. I integrated this technique into a phrase-based SMT system in order to improve the system performance in translating ambiguous words. Another challenge in MT is the dealing with the out-of-vocabulary words. I have performed research on several methods to decrease the out-of-vocabulary rate including proper noun transliteration.

Finally, many languages contain specific entities (like e.g. dates and numbers) which require special treatment, and the Arabic language is one of them. In this work, we addressed the problem of translation of these entities. In this context, since there is no integrated method to enable the correct translation of numbers and dates, I developed a method to detect numbers, dates and other entities and then transform them, if needed, from the source language format to the target language format.

Chapter 4

Semi-supervised Transliteration Mining from Parallel and Comparable Corpora

4.1 MT and transliteration

One of the challenges in MT research is dealing with the OOV words. One way to decrease the OOV rate is by transliterating proper nouns (or names). In this chapter, I will focus on dealing with the challenge of transliteration of Arabic proper nouns into English. Transliteration is the process of writing a word (mainly proper nouns) from one language in the alphabet of another language. This requires mapping the pronunciation of the word from the original language to the closest possible pronunciation in the target language. Both the word and its transliteration are called a Transliteration Pair (TP).

Since I am using a statistical-based approach throughout this thesis, I will need data to train the system. In this case, the training data should be a bilingual list of TPs in Arabic and English. Since we do not have this training data available, we have to deal with the automatic extraction of these TPs from the available corpora. In this work, I deal with two types of corpora, a bilingual corpora and a comparable corpora. A comparable corpus is a pair of corpora

in two different languages, which come from the same domain. The automatic extraction of TPs from parallel or comparable corpora is called Transliteration Mining (TMI).

Recently, TMI has gained considerable attention from the research community. There are several methods to perform TMI: supervised, unsupervised and semi-supervised. In this chapter we will focus on a semi-supervised method with both parallel corpora and comparable corpora. The reasons that we consider the proposed method a semi-supervised one are as follows. The first reason is that the initial TPs list should be obtained manually or it can be generated using a supervised rules based Arabic-English transliteration. The second reason is that the method uses a rule-based normalization step which is written by human specifically focusing on the similarity and difference of pronunciation of Arabic and English language pair.

In this chapter, I present my work on performing TMI, getting the TPs, building a transliteration system and evaluating it. Even though the main goal of this work is to improve SMT performance by transliterating proper nouns OOVs (POOVs), however I was not able to evaluate this work in the context of SMT. One of the reasons of not integrating our transliteration system in our BOLT systems is that the percentage of POOVs is very small (vary from 1-4%). This means that the expected gain from transliterating POOVs is very small taking into account the following: transliteration of other OOVs types because of name entity recognition errors of the NER tool, wrong transliteration since the accuracy reported in this chapter does not exceed 50%, scoring MT output using BLEU or TB2 after removing OOVs gives better scores. For these reasons, the transliteration systems we developed are evaluated independently on the name transliteration task using de-facto standard metrics. I was not able to compare my results to other research in international tasks like Name Entity Workshop (NEWS) [Min Zhang, 2012] since no similar workshop was held since the last one in 2012.

The chapter is organized as follows: the next section is the related work, Section 4.3 presents the challenges of Arabic transliteration, followed by a description of the TMI using parallel corpora. This technique is extend to comparable cor-

pora in Section 4.5. The partitioning method for improving backward and forward transliteration is presented in Section 4.6, finally the chapter concludes with a discussion of the perspectives of this work in Section 4.7.

4.2 Related work

The related work includes TMI and transliteration research. For TMI, several methods are possible to perform it, supervised, unsupervised and semi-supervised. Also, some TMI researches focus on parallel corpora and others on comparable corpora. Holmes et al. [2004] use a variant of the so-called SOUNDEX methods and n-grams to improve precision and recall of name matching in the context of transliterated Arabic name search. Originally, SOUNDEX was developed by Russell [1918]. This is an algorithm used for indexing names by sound as pronounced in English. The SOUNDEX code for a name consists of a letter followed by three numerical digits: the letter is the first letter of the name, and the digits encode the remaining consonants. Similar sounding consonants share the same digit. For example, the labial consonants B, F, P, and V are each encoded as the number 1. The method proposed by Holmes et al. [2004] reduce the orthographical variations by 30% using SOUNDEX. They improved precision slightly but they observed a decrease in recall. Darwish [2010] presents two methods for improving TMI: phonetic conflation of letters and iterative training of a transliteration model. The first method is an improved SOUNDEX phonetic algorithm. They propose SOUNDEX like conflation scheme to improve the recall and F-measure. Also an iterative training method was presented that improves the recall but decreases the precision.

Kuo et al. [2006] present an adaptive learning framework for Phonetic Similarity Modeling (PSM) that supports the automatic construction of transliteration lexicons. PSM measures the phonetic similarity between source and target words pairs. In a bitext snippet, when an source language word EW is spotted, the method searches for the word's possible target transliteration CW in its neighborhood. EW can be a single word or a phrase of multiple source language words. In their work, they initialize the learning algorithm with minimum machine transliteration knowledge, then it starts acquiring more transliteration

knowledge iteratively, from the Web. They study an active learning and an unsupervised learning strategy, respectively, which minimize human supervision in terms of data labeling. They report that unsupervised learning is an effective way for rapid PSM adaptation while active learning is the most effective in achieving high performance. Another TMI method relies on a Bayesian technique proposed by Fukunishi et al. [2011]. This method simultaneously co-segments and force-aligns the bilingual segments through rewards the re-use of features already in the model. The main assumption is that transliteration pairs can be derived by using bilingual sequence pairs already learned by the model, or by introducing a very short unobserved pair into the derivation. They assume that incorrect pairs are likely to have large contiguous segments that are costly to force-align with the model. The transliteration classifier is trained on features derived from the alignments of the candidate pairs as well as other heuristic features. They report results which indicate that transliteration mining of English-Japanese using this method should be possible at high levels of precision and recall.

Kashani et al. [2006] presented and evaluated a transliteration system by combining two different techniques and taking the best of each. They introduced a three phase algorithm which is based on a Hidden Markov Model approach, but also leverages information available in on-line databases. The algorithm achieved an accuracy approaching 80%. One encountered problem was the lack of training data, resulting in less accurate performance for some cases.

El-Kahky et al. [2011] adapt graph reinforcement to work with large training sets. They introduce a parametrized exponential penalty to the formulation of graph reinforcement which led to improvement in precision. They report that TMI quality using comparable corpora is impacted by the presence of phonically similar words in comparable text, so they extracted the related segments that have high translation overlap and used them for TMI, which leads to higher precision for the suggested TMI methods. An automatic language pair independent method for transliteration mining using parallel corpora is proposed by Sajjad et al. [2012]. They model transliteration mining as interpolation of transliteration and non-transliteration sub-models. Two methods, unsupervised and semi-

supervised were presented with the results that show that semi-supervised method is out-performing the unsupervised one.

For transliteration research, Al-Onaizan and Knight [2002] use two algorithms based on sound and spelling mappings using finite state machines to perform the transliteration of Arabic names. They report that the transliteration model can be trained on relatively small list of names which is easier to obtain than the average amount of data needed for training phonetic based models. Jiampojamarn et al. [2009] present DirecTL, a language independent approach to transliteration. DirecTL is based on an online discriminative sequence prediction model that employes EM-based many-to-many unsupervised alignment between target and source. Sajjad et al. [2011] use a joint source channel model on the aligned orthographic transliteration units of the automatically extracted TPs. They compare the results with three online transliteration systems and report better results.

Recently, Durrani et al. [2014] propose three methods for integrating an unsupervised transliteration model into the Moses SMT toolkit [Koehn et al., 2007c]. They extract a transliteration corpus from the parallel data and build a transliteration model from it which is used to translate OOVs or named-entities. They propose to induce a transliteration model from parallel data and use it to translate OOV words. The approach is unsupervised and language independent. By integrating this method in SMT, they observed improvements from 0.23-0.75 BLEU points across 7 language pairs. They compared the extracted transliteration corpora with the gold standard one and reported that their corpora provide better rule coverage.

4.3 The challenges of Arabic transliteration

There are several challenges related to Arabic translation as listed in Section 2.4. One of the challenges is how to perform transliteration of Arabic POOVs in order to decrease the number of OOVs in the translation output. This is a challenge because there are some Arabic letters which have no phonically equivalent letters in English (e.g. ض and ط), and also some English letters do not have phonically equivalent letters in Arabic (e.g. v). Another challenge is the missing of short vowels (i.e. diacritics) in the Arabic text, while they should be mapped to existing

letters in English text during the transliteration process. Additionally, some Arabic letters can be mapped to any letter from a group of phonically close English letters (e.g. ب to p or b), and some Arabic letters can be mapped to a sequence of English letters (e.g. خ to 'kh'). There is also a tokenization challenge, since unlike English, sometimes, the Arabic name is concatenated to one clitic (e.g. preposition ب or conjunction و) or both together (e.g. وب), which requires an advanced detection and segmentation for these clitics before performing the transliteration.

The proposed TMI algorithm is based on the following pronunciation (and hence transliteration) observations in the English language:

1. In most cases, we can sort the letter's impact on transliteration from low to high as follows:
   - Phonetically similar vowels have low impact.
   - Phonetically dissimilar vowels have medium impact.
   - Consonants letters have significant impact.
2. Double vowels producing a long vowel sound have more impact on the pronunciation of the English word.
3. A sequence of two or more different vowels, has a special pronunciation which has more impact on the pronunciation of the English word.
4. A vowel at the initial position or at the final position in the word has significant impact on the pronunciation (e.g. the names: Adham, Samy).

4.3.1 English normalization and three level similarity scores for TMI

We developed three normalization functions which can be used to normalize the word transliterated from Arabic and the English word with the goal to make the two words phonetically comparable. These normalized forms are used to calculate the similarity between the transliterated word and the English word. Three levels of similarity are used. The first level calculates the similarity of all vowels and consonants. The second level calculates the similarity of long vowels and vowel letters at the beginning and the end position of the words as well as consonant letters. The third level calculates the similarity of consonant letters only. The details of each normalization function are presented in the following:

(1) $Norm_{similar}$ normalization function: Normalize the transliteration of the Arabic as well as the English word. The objective of the normalization is folding English letters with similar phonetic to one symbol. In $Norm_{similar}$, all letters are converted to lower case, phonetically equivalent consonants and vowels (i.e. these English letters which are mapped to the same Arabic letter) are folded to one letter (e.g. p and b are normalized to b, v and f are normalized to f, i and e are normalized to e), double consonants are replaced by one letter (since they are mapped to double letter which is actually written as one Arabic letter with Shadda above it), and finally a hyphen "-" is inserted after the initial letters "al" which is the transliteration of the usually concatenated Arabic article " ال "- if it is not already followed by it.

(2) $Norm_{vowels}$ normalization function: Using $Norm_{similar}$ output, double vowels are replaced by one similar upper-case letter (i.e. ee is replaced by E), non-initial and non-final vowels are removed only if not followed by a vowel or not preceded by a vowel.

(3) $Norm_{consonants}$ normalization function: Using $Norm_{vowels}$, hyphen - and vowels are removed.

Hence, for each Arabic word A and English word E, if A_t is the transliteration of A into English, we can calculate the following three similarity scores

$$TLS_i = \frac{Levenshtein(Norm_i(A_t), Norm_i(E))}{|Norm_i(E)|} \tag{4.1}$$

with i in $similar, vowelsandconsonants$. We use the well-known Levenshtein distance at character level.

4.4 TMI using parallel corpora

In this section, we will introduce a corpus based computational method to extract TPs from a parallel corpus. In order to evaluate the extracted pairs, we trained a letter based statistical transliteration system on TPs and evaluate the system performance.

4.4.1 TMI algorithm for parallel corpora

The algorithm as shown in Figure 4.1 is designed to compare two aligned words and detect the words which are transliterations of each other, with respect to the observations in Section 4.3.

The algorithm proceeds in 7 steps:

(1) First, the parallel corpus is tagged using a part-of-speech (POS) tagger. We used Stanford POS tagger [Toutanova et al., 2003] for English and Mada/Tokan [Nizar Habash and Roth, 2009] for Arabic POS tagging.

(2) Then, we align the tagged bitexts using Giza++ [Och and Ney, 2003a]. Using the source/target alignment file, we remove all aligned word pairs with POS tags other than noun (NN) or proper noun (PNN) tags and we remove all English words starting with lower-case letters. Words which have the lowest alignment scores are removed (about 5% from the total number of aligned word pairs).

(3) After that the POS tags are removed from Arabic and English words since they are not needed any more.

(4) Then, the Arabic word is transliterated into the English word A_t using a rule based transliteration system or a previously trained statistical character based transliteration system.

(5) The transliteration of the Arabic word A_t as well as the English word are normalized to $Norm_{similar}$, $Norm_{vowels}$ and $Norm_{consonants}$ as explained in Section 4.3.1. The objective of the normalization is folding English letters with similar phonetic to the same letter or symbol.

(6) For each aligned Arabic transliterated word A_t and English word E, their normalized forms are used to calculate the three levels of similarity scores which are stored in a Transliteration Table (TT).

(7) TPs are extracted from the TT by applying a threshold on the three levels similarity scores. We selected the thresholds using an empirical method described in Section 4.4.3.2.

4.4.2 Transliteration system for TMI evaluation

The transliteration system is built using the Moses toolkit. We train a letter-based SMT system on the list of TPs extracted using our TMI algorithm explained in Section 4.4.1. The distortion limit is set to 0 to disable any reordering. Since the length of most names will not exceed 20 letters, we set the maximum phrase length to 20, however the system can still learn from and translate names which exceeds this limit. The transliteration system should be able to learn a transliteration model using the alignment of the letters using Giza++, and hence be able to generate the possible transliterations of a name written in the source language script into a name written in the target language script. This research focuses on the following points:

- Use the TMI algorithm to extract a list of TPs that we can use to build a transliteration system.
- Acquire a list of target language names to train the letter-based language model which is needed to improve the LM of the letter-based SMT system.
- Study the impact of the segment length on the transliteration quality. In this context, two systems are trained to evaluate the segmentation for the word letters. We compared two segmentation scheme:
  - Simple segmentation of the word by separating individual letters.
  - Advanced segmentation of the word that segments the word into a group of 1-2 letters based on predefined phonetic units which combine two English letters, based on their position in the word, in one substring (e.g. 'kh', 'kn', 'wh', 'sh' and 'ck').
- The impact of using different tuning metrics, we compared the following metrics: TER, BLEU, TB2.

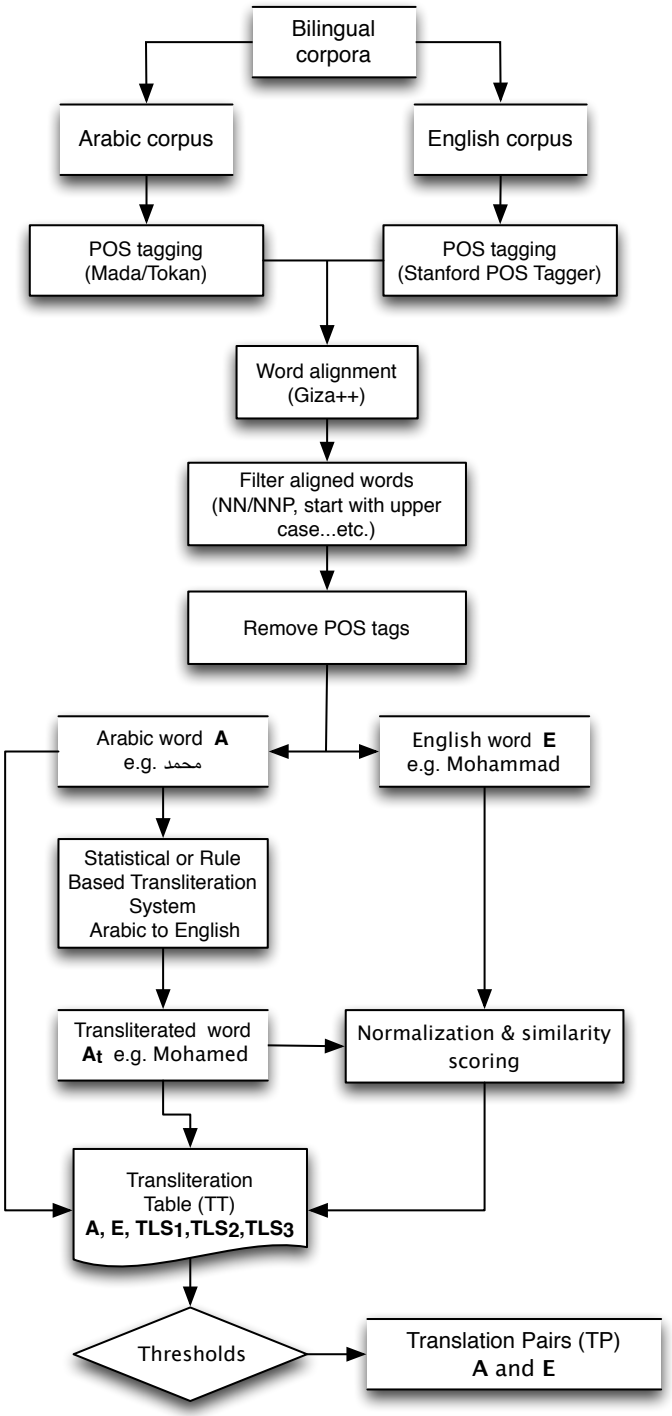

Figure 4.1: Extracting TPs from parallel corpora

- Evaluate the performance of the TMI algorithm by using TPs to build a transliteration system. The transliteration system performance is correlated with the quality of the extracted TPs, and hence the TMI performance.

4.4.3 Experiments and evaluation

The objectives of developing our transliteration system is to evaluate the quality of our TMI algorithm and perform research on improving the transliteration quality especially for unseen names in the training data. We evaluated the proposed TMI algorithm using an Arabic/English parallel corpus which contains about 3.8 million Arabic words and 4.4 million English words. The evaluation of the TMI algorithm is performed by training a statistical system on the extracted TPs and evaluating the quality of the transliteration output.

The extracted TPs are divided into three parts: a training data set which varies in size in function of the selected thresholds of the 3-levels (from $9k$ to $10.5k$), tune and test sets ($\sim 1k$ for each). All occurrences of words in the tune or test set were removed from the training set.

In order to evaluate the quality of our transliteration system, we used the de-facto standard metrics and evaluation tools from the Name Entity Workshop (NEWS) [Min Zhang, 2012]: ACC, mean F-Score, MRR, and MAP_{ref}.

The following notation is further assumed:
N: total number of names (source words) in the test set.
n_i: number of reference transliterations for i-th name in the test set.
$r_{i,j}$:j-th reference transliteration for i-th name in the test set
K_i: Number of candidate transliterations produced by a transliteration system.
$c_{i,k}$: k-th candidate transliteration (system output) for i-th name in the test set ($1 \leq k \leq 10$)

Here is a short description of each metric:

- **Word Accuracy in top-1 (ACC)**, it measures the correctness of the first transliteration candidate in each candidate transliteration list generated by

a transliteration system. The following equation is used to calculate it:

$$ACC = \frac{1}{N}\sum_{i=1}^{N}\begin{cases} 1 \text{ if } \exists\ r_{i,j} : r_{i,j} = c_{i,1} \\ 0 \text{ otherwise} \end{cases} \tag{4.2}$$

- **F-Score**. The mean F-score measures the difference, on average, between the first transliteration hypothesis and its closest reference. if the first transliteration hypothesis matches one of the references, the F-score will equal to 1. It equals 0, if there are no common letters between the first transliteration hypothesis and any of the references. The F-score is a function of Precision and Recall. The length of the Longest Common Subsequence(LCS) between a candidate and a reference are used to calculate the Precision and Recall. LCS is calculated using the following equation:

$$LCS(c, r) = \frac{1}{2}(|c| + |r| - ED(c, r)) \tag{4.3}$$

where ED is the edit distance and $|x|$ is the length of x. If the best matching reference is given by

$$r_{i,m} = argmin_j(ED(c_{i,1}, r_{i,j})) \tag{4.4}$$

The recall R_i, the precision P_i and the F-score for i-th word are calculated as follows:

$$R_i = \frac{LCS(c_{i,1}, r_{i,m})}{|r_{i,m}|} \tag{4.5}$$

$$P_i = \frac{LCS(c_{i,1}, r_{i,m})}{|c_{i,1}|} \tag{4.6}$$

$$F_i = 2\frac{R_i \times P_i}{R_i + P_i} \tag{4.7}$$

- **Mean Reciprocal Rank (MRR)** is measured for any right answer produced by the system, among the candidates. 1/MRR gives the average rank of the correct transliteration. An MRR close to 1 implies that the correct answer is mostly produced close to the top of the n-best lists. It is calculated as follows:

$$RR_i = \begin{cases} min_j \frac{1}{j} if \exists\ r_{i,j}, c_{i,k} : r_{i,j} = c_{i,k} \\ 0 \text{ otherwise} \end{cases} \tag{4.8}$$

$$MRR = \frac{1}{N} \sum_{i=1}^{N} RR_i \tag{4.9}$$

- MAP_{ref} measures the precision in the n-best transliteration output for the i-th source name, for which reference transliterations are available. If all of the references are produced, then the MAP is 1. If $num(i, k)$ is the number of correct transliteration hypotheses for the i-th source name in k-best list, the MAP_{ref} is calculated using the following equation:

$$MAP_{ref} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{n_i} \left(\sum_{k=1}^{n_i} num(i, k) \right) \tag{4.10}$$

4.4.3.1 Acquiring LM resources

We used two resources to get lists of English names to train letter based LMs. The first language model, LM1, is trained on a list of proper names extracted from the English Gigaword corpus (using only Xinhua, Agence France Presse and New York Times parts). The extraction is done using the Stanford Named Entity Recognizer (NER) [Finkel et al., 2005]. The second language model, LM2, is trained on the English part of the extracted TPs from our bilingual corpora. The Table 4.1 below compares the performance of two systems which are using LM1 and LM2. These results show that the system LM2 gives better accuracy score but lower mean F-score. Since in the context of MT, the accuracy is more important, so we decided to use the second language model (LM2) in the rest of our experiments.

System	ACC	Mean F-Score	MRR	MAP_{ref}
SYS430(LM1)	0.43750	0.88160	0.54787	0.43750
SYS430(LM2)	0.44159	0.87860	0.54862	0.44160

Table 4.1: *The result comparison on tune set between two systems using different language models (LM1 vs. LM2)*

4.4.3.2 Selection of the thresholds

Several systems were trained to determine the best thresholds to be used in our experiments. The experiments show that the best thresholds for the 3 scores on the tune set are (TLS_1, TLS_2, TLS_2)=(0.5, 0.4, 0). The thresholds are highly dependent on the normalization functions $Norm_{similar}$, $Norm_{vowels}$ and $Norm_{consonants}$, so changing the normalization functions will require a re-selection of the three thresholds. The results for different thresholdsare given in Table 4.2.

System	TLS_1	TLS_2	TLS_3	ACC	Mean F-Score	MRR	MAP_{ref}
SYS420(LM2) TPs=9167	0.4	0.2	0	0.43545	0.87940	0.54188	0.43545
SYS430(LM2) TPs=9070	0.4	0.3	0	0.44159	0.87860	0.54862	0.44160
SYS540(LM2) TPs=10529	0.5	0.4	0	0.44774	0.88226	0.55012	0.44774
SYS542(LM2) TPs=12698	0.5	0.4	0.2	0.43647	0.88042	0.54220	0.43647

Table 4.2: *Tuning set results using different systems trained on different TPs using different thresholds (tuning on* TB2*)*

4.4.3.3 Tuning metric selection

We used the MERT tool for weight optimization [Och, 2003][Bertoldi et al., 2009]. We faced several problems in optimizing the log linear features weights. The first problem is that there are four evaluation metrics as presented in Section 4.4.3,

it would be difficult to perform the optimization on all of them together. In the same time, the MERT toolkit in Moses is limited to specific set of standard SMT evaluation metrics (e.g. BLEU, TER,... etc.) and it does not support using external metrics. We decided to evaluate a three of well known SMT evaluation metrics namely BLEU, TER and TB2 to select the one that gives the best transliteration scores. Table 4.3 shows that TB2 gives better results than using BLEU or TER alone.

System	MERT metric	ACC	Mean F-Score	MRR	MAP_{ref}
SYS430(LM2)	BLEU	0.43648	0.87662	0.54322	0.43647
	TER	0.43545	0.87638	0.54263	0.43545
	TB2	0.44159	0.87860	0.54862	0.44160

Table 4.3: *The results on tune set when use various tuning metrics*

4.4.3.4 Segmentation techniques

We used two segmentation techniques, the first technique simply segments the NE into characters, the second one is an advanced segmentation that groups together letters that form one phonetic sound in one segment (e.g. ph, ch, sh, etc). Table 4.4 shows the results of both segmentation techniques. One can see that the second technique helps the letters alignment between source and target and hence significantly improves the transliteration output. The results presented in the current and next sections are different than the results in previous sections because we re-applied the TMI algorithm again using our best transliteration system (i.e. SYS540). This results in obtaining better TPs. The transliteration system trained on these TPs (i.e. SYS540-2) is improved compared to the previous one (i.e. SYS540). This is the explanation of the increase in accuracy and other scores in Table 4.4.

4.4.3.5 Results

Using three levels similarity scores thresholds=(0.5, 0.4, 0) as explained in Section 4.4.3.2, the total number of extracted TPs is 10529. Table 5.1 shows some statistics on the extracted TPs.

System	Segmentation	ACC	Mean F-Score	MRR	MAP_{ref}
SYS540(LM2)	One letter	0.44774	0.88226	0.55012	0.44774
SYS540-2(LM2)		0.47951	0.89248	0.59226	0.47951
	1 and 2 letters	0.50000	0.89589	0.61178	0.5000

Table 4.4: *Results on the tune set using one letter segmentation vs. advanced segmentation*

Data	Number of Words	Extracted TPs %
Bitext-Arabic	3.8m	0.27 %
Bitext-English	4.4m	0.24 %
List of aligned words	1.25m	0.84 %
List of aligned NN	161k	6.5 %

Table 4.5: *Statistics on the extracted TPs*

Set	ACC	Mean F-Score	MRR	MAP_{ref}
tune	0.50000	0.89589	0.61178	0.5000
test	0.46162	0.88412	0.58221	0.4616

Table 4.6: *tune and test sets scores*

Finally, we list in Table 4.6 the results of best system on the tune and test set. Both tune and test sets have not seen before in the training data.

4.5 TMI using comparable corpora

In this section, we will introduce a corpus based computational method to extract transliteration pairs from comparable corpora. In order to evaluate the extracted pairs, we trained a letter based statistical transliteration system on them and evaluate the system performance which is correlated with the TMI quality.

4.5.1 TMI algorithm for comparable corpora

Since it is easier to collect and find monolingual text than parallel text, it would be useful if we can perform TMI using comparable corpora of monolingual text for any pair of languages. Figure 4.2 shows an overview of the TMI algorithm for

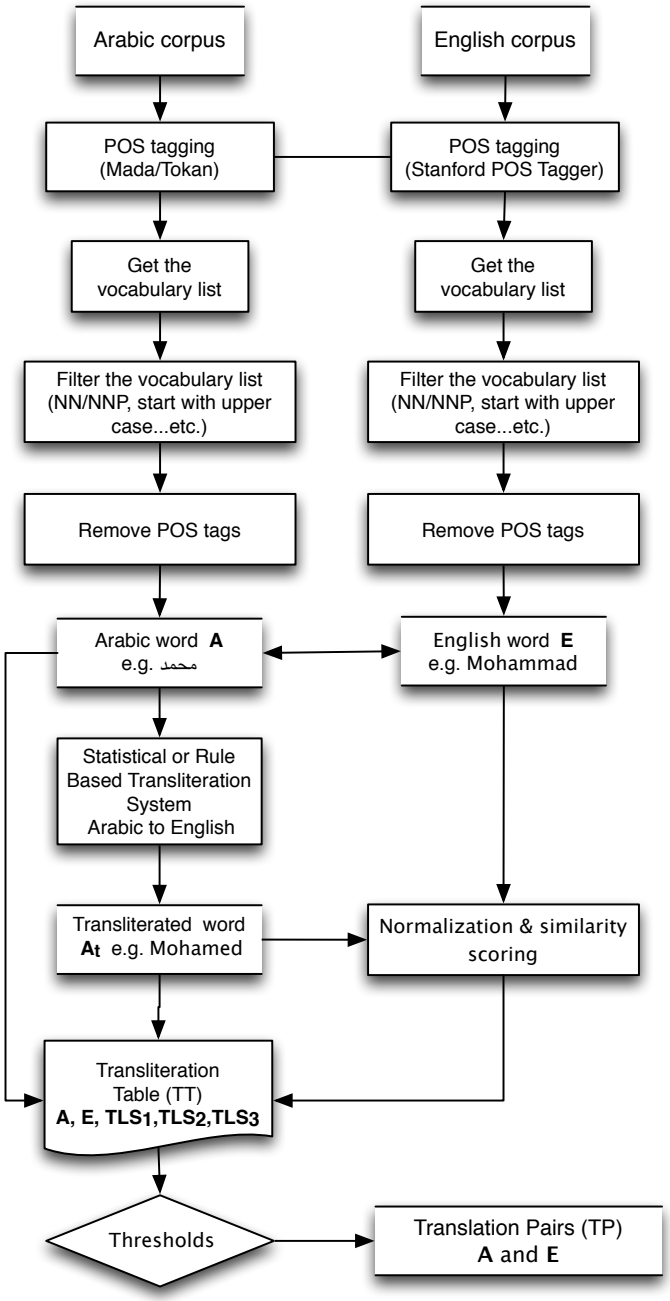

Figure 4.2: Extracting TPs from comparable corpora

comparable corpora. The algorithm is designed to remove the non-nouns words in order to minimize the number of words in each monolingual text. The next step, is detecting the words which are transliteration of each other, with respect to the observations listed in Section 4.3. We score the words similarity using three levels similarity scores to generated the transliteration table (TT). TPs are extracted from the TT using three thresholds on the three levels of similarity scores. The following steps explain the TMI algorithm:

(1) Each monolingual corpus is tagged using a POS tagger. We used Stanford POS tagger [Toutanova et al., 2003] for English and Mada/Tokan [Nizar Habash and Roth, 2009] for Arabic POS tagging.

(2) All words with POS tags other than noun (NN) or proper noun (PNN) tags and all English words starting with lower-case letters are removed (only for target corpora).

(3) The remaining words are un-tagged (i.e. removing the POS tags from source text and target text).

(4) Two unique word lists are derived (LIST_SRC and LIST_TRG) from both source and target texts.

(5) The source word list (LIST_SRC) is transliterated into target language (LIST_SRC_TRANS) using rule based transliteration system (or previously created statistical based transliteration system).

(6) The transliteration of source word list is normalized as well as the English word list to $Norm_{similar}$, $Norm_{vowels}$ and $Norm_{consonants}$ as explained in Section 4.3.1. The objective of the normalization is folding English letters with similar or close phonetic to same letter or symbol.

(7) The normalized values are used, for each transliterated word in the source language list WORD_AR_TRANS and target language word WORD_EN, and the 3-similarity scores are calculated between them. All scores are stored in the transliteration table (TT).

(8) The TPs are extracted from the TT by applying a selected thresholds on the three levels similarity scores.

4.5.2 Experiments and evaluation

4.5.2.1 Purpose and data sets

We evaluated the proposed TMI algorithm by applying it on the Arabic Gigaword corpus (about 270.3 million Arabic words using only XIN, AFP and NYT parts) and the English Gigaword corpus (roughly 1.47 billion English words using only XIN, AFP and NYT parts).

The extracted TPs are used as training data. We used the same tune set and test set extracted from parallel corpus as mentioned in Section 4.4.3.

As before, all occurrences of words in the tune set or test set were removed from the training data.

We selected the thresholds using empirical method. Several systems were trained to evaluate the best thresholds to be used in our experiments. Only two thresholds are compared, other thresholds are discarded because they almost give the same TPs. The experiments show that the best thresholds for 3-scores on tune set are (TLS_1, TLS_2, TLS_2)=(0.4, 0.3, 0) since they give slightly better mean F-Score and MRR. The scores of the tune set with different thresholds are mentioned in Table 4.7. Table 4.7 lists the systems with the TLS scores' thresholds used to select data to train each one.

System	TLS_1	TLS_2	TLS_3	ACC	Mean F-Score	MRR	MAP_{ref}
GSYS420 TPs=1.63M	0.4	0.2	0	0.30021	0.83973	0.40807	0.30021
GSYS430 TPs=1.96M	0.4	0.3	0	0.30021	0.84001	0.40817	0.30021

Table 4.7: *Tuning set results with different thresholds*

4.5.2.2 Results

Using three levels similarity scores thresholds are (0.4, 0.3, 0) as explained in Section 4.5.2.1, the total number of extracted TPs is 1.96 millions. Table 4.8 shows TPs percentage with respect to the comparable corpora (in both languages) total number of words and the total number of words with NNP/NN POS tag. In Table 4.9, we list the transliteration system results using the evaluation metrics

mentioned in Section 4.4.3. We are reporting the scores for both tune set and test set. Both tune set and test set has not seen before in the training data.

Data	Number of Words	Extracted TPs %
Arabic Gigaword	270.3m	0.73%
Arabic Gigaword NN	18.7m	10.48%
English Gigaword	1.47b	0.13%
English Gigaword NN	8.1m	24.20%

Table 4.8: *Extracted TPs rate*

Set	ACC	Mean F-Score	MRR	MAP_{ref}
tune	0.30021	0.84001	0.40817	0.30021
test	0.27329	0.83345	0.39788	0.27329

Table 4.9: *tune and test set scores*

4.6 Improving backward and forward transliteration by partitioning training data

There are two types of transliteration, forward and backward. In forward transliteration, the names are transliterated from their original language to another language (e.g. the native Arabic name "محمد" is transliterated to "Mohamed" in English). In backward transliteration, the transliterated names are transliterated back to the original names in their native language (e.g. "بوش" will be transliterated back to "Bush"). This section discusses these two types of transliteration in order to improve the transliteration performance.

4.6.1 Related work

Kang and Choi [2000] presented a very effective bi-directional automatic English/Korean transliteration and back-transliteration methodology. The used method consists of character alignment and decision tree learning. They wanted to induce the transliteration rules for the English alphabet and the back-transliteration rules for the Korean alphabet. They also developed a highly accurate character alignment algorithm, which is able to align two words in a desirable constrained way across languages. The alignment method is partially language independent. The only language dependent part is the alignment evaluation metrics that may also be easily constructed without much effort. Qin and Chen [2011] propose a forward-backward transliteration system between English and Chinese for the shared task of NEWS 2011. Combined recognizers based on Conditional Random Fields (CRF) are applied to transliterate between source and target languages. Huge amounts of features and long training time are the motivations for decomposing the task into several recognizers. To prepare the training data, they

performed segmentation and alignment in terms of not only syllables and single Chinese characters, but also phoneme strings and corresponding character strings. When transliterating from English into Chinese, their combined system achieved accuracy in top-1 equal to 0.312, compared with the best performance in NEWS 2011, which was 0.348. For backward transliteration, their system achieved top-1 accuracy of 0.167, which is better than others in NEWS 2011. Hamdy Mubarak and AlMasry [2009] introduced a complete system for correction, diacritization, and transliteration of names from Arabic to English with an accuracy of 0.89 on blind test-data. Their system uses bilingual training data, along with morphological analysis (using Sakhr's Morphological Analyzer), some heuristic rules and observations to achieve these results in combination with traditional statistical language processing and machine learning algorithms.

4.6.2 Partitioning technique

In order to study the impact of partitioning Arabic-English transliteration training data, tune set and test set. We propose to partition each dataset into three parts:

- Transliterated names which are originally Arabic (called forward transliteration)
- Transliterated names which are originally English (called backward transliteration)
- A third partition for names which are shared or difficult to categorize in the other two partitions.

The partitioning technique uses two language specific features: the first feature is the source or target language phonetics which are missing in the other language. This is motivated by the following two facts:

- It is difficult to transliterate these names without transliterating the phonetic variants to the closest possible phonetic variant in the target language. For example the Arabic letter ض has no equivalent in English, hence it is mapped to the English letter "D" which is the closest possible substitution.

The Arabic letter ض can not be a transliteration of any English phonetic unit and hence the name is more likely from Arabic origin.

- It is difficult to transliterate to a target language phonetic that has no mapping back to source language phonetic. For example the English letter "X" has no equivalent in Arabic. Hence it can not be a transliteration of Arabic phonetics unless the name origin is English.

The second feature we used is the common letter patterns (or sequences) of names in each language. For example, in Arabic, if a name contains the letter sequence عبد, then its origin is certainly Arabic.

4.6.2.1 Partitioning rules

We obtained the list of TPs using transliteration mining technique detailed in Section 4.4 using parallel Arabic-English corpus. We divided this list into three sets, training data, tune set and test set. The partitioning of the data is done for all transliteration pair in each set as following:

- A weight is assigned for each letter or sequence of letters that is more frequent in source or target language. For Arabic and English, some examples of these letter sequences are presented in Table 4.10 with suggested weights. In these experiments, the weight is set manually based on the closeness of the letter phonetic to other phonetic in the other language.

letters	Sad ص	Daad ض	Eien ع	Al ال	Abo أبو	Abd عبد	p/P	x/X
weight	1	1.5	2	2	2	2	1.5	2

Table 4.10: *Arabic and English specific letter(s) or pattern and their proposed weights*

- For each transliteration pair two scores, S and T, are calculated to measure whether the name origin is Arabic or English.

- A final score is calculated as following:
  $S_{total} = S_{source}$ - S_{target}.

- The transliteration pair is partitioned as follows:
  - Arabic partition if S_{total} >0
  - English partition if S_{total} <0
  - non-determined partition if $S_{total} = 0$.

In the following sections, we will present the experiments and results for forward and backward transliteration.

4.6.3 Experiments and Evaluation

4.6.3.1 Apply partitioning technique

Table 4.11 shows a sample of names and its transliteration from Arabic partition, while Table 4.12 shows a sample of names and its transliteration from English partition. The results of partitioning the training data of $\sim 20.3k$ transliteration pairs, the tune and test sets are shown in Table 4.13. We will evaluate the impact of partitioning technique by using the partitioned sets to train several statistical based transliteration systems compared to a baseline system that is using all data sets before partitioning them.

Arabic	English transliteration
عبادي	Abadi
عبدالمقصود	AbdelMaqsoud
عبيدات	Abidat
احمد	Ahmed
البوناصر	Albunasser
الشهرستاني	AlShahristani
الزغزغي	AlZaghzaghi
المحلاوي	AlMehlawi

Table 4.11: *Examples from forward transliteration partition*

Arabic	English transliteration
البرتفيل	Albertville
المودوفار	Almodovar
براكس	Brax
تشايكوفسكي	Chikovski
ديكسون	Dixon
دونوفان	Donovan
هوبمان	Hopman
كسباريان	Kasparian

Table 4.12: *Examples from backward transliteration partition*

Partition ID	No of TPs training data	# of TPs in tuneset	# of TPs in testset
ALL	20345	1000	1000
AR	7575	195	219
EN	2038	91	100
ND	10732	714	681

Table 4.13: *Number of TPs in each partition vs. the total size*

4.6.3.2 Experiments

We used the partitioned training data to train three transliteration systems with the same tools and setup as in the previous experiments detailed in Section 4.4.2. Additionally, we used the method described in Section 3.7.3 which performs instance weighting of translation models, based on sufficient statistics. It separately optimizes four features weights in the Moses translation model through perplexity optimization. [Sennrich, 2012] independently performs perplexity minimization for the following features of the standard Moses SMT translation model: the phrase translation probabilities $p(\overline{s}|\overline{t})$ and $p(\overline{t}|\overline{s})$, and the lexical weights $lex(\overline{s}|\overline{t})$ and $lex(\overline{t}|\overline{s})$. Also we used the Operation Sequence Model as proposed in [Durrani et al., 2013] and implemented in the Moses toolkit. In these experiments we trained three systems which are adapted for forward, backward and non-determined transliteration. The description of the training data and tune sets used for each system is presented in Table 4.14.

We tuned the transliteration systems using MERT as shown in Table 4.14 using the training sets, tune sets presented in Table 4.13. Table 4.15 shows the results on the tune sets of the three adapted transliteration systems using partitioning of training data technique. The transliteration system adapted on the names originally Arabic achieved 0.4497 accuracy compared to 0.3757 for the baseline system, representing a gain of about 0.074. A significant improvement also reported on *the mean F-score*, MRR and MAP_{ref} metrics.

System	Training Data	Corpus Weighting Set	MERT tune set
Baseline(AR/EN/ND)	ALL	-	AR+EN+ND
SysAR	ALL	AR	AR
SysEN	ALL	EN+ND	EN
SysND	ALL	AR+ND	AR+ND

Table 4.14: *The adapted transliteration systems and the used: training corpora, corpus weighting set (if used) and* MERT *tune set*

System	tune set	ACC	Mean F-Score	MRR	MAP_{ref}
Baseline AR	AR	0.3757	0.8843	0.4840	0.3717
Partition AR	AR	0.4497	0.9066	0.5531	0.4471
Baseline EN	EN	0.3240	0.8553	0.4563	0.3237
Partition EN	EN	0.3222	0.8645	0.4377	0.3222
Baseline ND	ND	0.3476	0.8663	0.4711	0.3473
Partition ND	ND	0.3533	0.8664	0.4699	0.3527

Table 4.15: *Tune sets scores on baseline vs. partitioned systems*

4.6.3.3 Results

We compared our results on the test set to our baseline system that is not using the partitioning technique. The transliteration system for names originally Arabic achieved 0.3704 accuracy compared to 0.3333 for the baseline system, representing a gain of about 0.0371. Some improvements are also reported on *mean* $F-score$, MRR and MAP_{ref} metrics. Also, the transliteration system for names originally English got improvements of about 0.0206 on accuracy as shown in Table 4.16. We also notice significant improvements for the other metrics.

System	test set	ACC	Mean F-Score	MRR	MAP_{ref}
Baseline AR	AR	0.3333	0.8683	0.4428	0.3345
Partition AR	AR	0.3704	0.8759	0.4750	0.3693
Baseline EN	EN	0.3609	0.8537	0.4458	0.3583
Partition EN	EN	0.3815	0.8663	0.4615	0.3815
Baseline ND	ND	0.3649	0.8565	0.4872	0.3649
Partition ND	ND	0.3634	0.8580	0.4762	0.3634

Table 4.16: *Test sets scores on baseline vs. partitioned systems*

4.7 Conclusion

In this chapter we introduced a new semi-supervised transliteration mining method for parallel and comparable corpora. The method is mainly based on new suggested three level scores to extract the transliteration pairs. The transliteration system trained on the transliteration pairs extracted from the parallel corpus achieved an accuracy of 0.46 and a mean F-score of 0.88 on the test set. We also applied our transliteration mining approach on two Arabic and English comparable corpora. The system trained on transliteration pairs extracted from them achieved an accuracy of 0.27 and a mean F-score of 0.83. This shows that the proposed semi-supervised transliteration mining algorithm is effective.

In a second set of experiments, we build separate systems for forward and backward transliteration. The detection of the transliteration direction is fully

automatic, for training, tune and test corpora. By these means, we were able to significantly improve the transliteration performance when the origin was detected (about 50% of the training and 30% of the tune and test data). As expected, there is no notable change in the performance when the transliteration direction can not be automatically detected and a generic system is used.

The method presented in this chapter is specific to Arabic and English, but the framework can be used for other language pairs after replacing the language specific modules and rules.

Chapter 5

CSLM improvement

5.1 Introduction

It is well known that the language model on the target sentence plays an important role to achieve high quality statistical machine translation. We have already applied several adaptation techniques to the language models developed by the SMT group of LIUM, e.g. data selection, interpolation, etc. (read more details in Section 3.7).

In this chapter, I will present improvements of the CSLM which I have developed during the last period of my PhD. The idea is to provide additional information at the input of the neural network. In extension of similar work for recurrent NN LM by Mikolov and Zweig [2012], we will name these additional inputs "auxiliary information". I used different type of auxiliary features including line length, text genre, line context vector representation, ... etc. By these means, better domain and context specific LM estimations can be obtained. I will report the results using perplexity as well as when these improved CSLMs are integrated into an SMT system. This is performed by re-scoring the n-best list and adding an additional feature function.

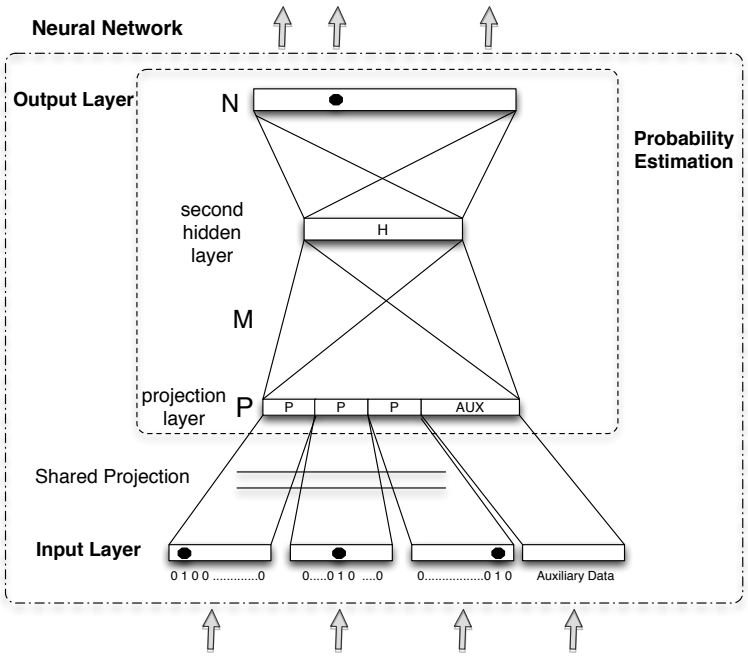

Figure 5.1: Adding additional auxiliary feature input to the CSLM

5.2 Modified architecture of the CSLM

The basic architecture of a CSLM with auxiliary data is shown in Figure 5.1. The example in the figure shows only one additional auxiliary feature vector. This architecture would allow different auxiliary information for each n-gram, but since our goal is to model the topic or long-term context, we made the choice to keep the auxiliary data constant for all n-grams of one sentence. Therefore, the auxiliary data is loaded once for each sentence. If more than one auxiliary feature is desired, the dimension of the auxiliary feature vector will be equal to the sum of the individual feature dimensions. In this case the auxiliary feature vector will be the concatenation of two or more feature vectors. This architecture also allows us to use sentence-level features as well as document (or corpus) level features by using the same auxiliary vector for all lines in the document (or corpus).
The functionality of auxiliary features has been integrated in the open-source CSLM toolkit [1] [Schwenk, 2010]

[1] Available for download from https://github.com/hschwenk/cslm-toolkit

5.3 Related work

Although, I focus on improving CSLM in this work, some related research focus on improving the standard n-gram language models by integrating more context or semantic knowledge. Kuhn and De Mori [1990] proposed to calculate the probabilities which correspond to the relative proportion of the last N words. They present a combined LM that interpolates a general trigram LM and another LM that they called a cache-based LM which is trained on the last N words. The relative interpolation weights assigned to each component are based on the POS of each word. The cache component assigns higher probability to recently encountered words. In my work, the context is represented as a continuous space vector. It can be one line or the whole history back to the beginning of the document. In the latter case more weight is given to recent lines.

Bellegarda [2000] proposed a method to use more global constraints to improve LM since local constraints are already captured by the n-gram model. They use latent semantic analysis (LSA) which automatically discovers the semantic relationships between words and documents in a given corpus. In their approach, words and documents are mapped into a continuous semantic vector space, in which clustering techniques are used. This allows the characterization of parallel layers of semantic knowledge in the space, with variable granularity. The resulting LMs complement the conventional n-gram LMs. They suggested to use hybrid n-gram+LSA models to benefit from the advantages of several smoothing techniques.

In a similar work, Coccaro and Jurafsky [1998] integrated semantic knowledge into an n-gram LM using LSA and a word similarity algorithm. Since LSA is a bad predictor of frequent words, they used a geometric instead of a linear combination based on a per-word confidence metric. In my work, instead of using LSA , I use the line context vector representations which is calculated using the word embeddings of the words in this line. The word embeddings are the projections learned during CSLM training. We were motivated by what was reported recently by [Baroni et al., 2014] that using the context predictive models (i.e. word embedding) outperform classic count-vector-based distributional semantic approaches.

Other works, like the work of [Iyer and Ostendorf, 1999] focused on developing a sentence-level mixture language model that takes advantage of the topic constraints in a sentence or article. They proposed topic-dependent dynamic cache adaptation techniques in the framework of mixture models. An automatic clustering algorithm was used to classify text with two levels of mixture models for smoothing. In my work a predefined genre is assigned to different corpora, which is used as additional input to the neural network. However, it is also possible to use topics instead of genres and to assign the topic dynamically by using similar automatic clustering algorithm like the one used by Iyer and Ostendorf [1999].

Khudanpur and Wu [2000] proposed an LM that combines collocational dependencies with the syntactic structure and the topic of the sentence. They integrate these dependencies using a maximum entropy technique. They report a substantial improvement in perplexity and in the accuracy of a speech recognition task. In my work, instead of using the topic, I used the genre of the sentence. Since I am using auxiliary features on the sentence level, it could be envisioned to extend this to syntactic features.

Mikolov and Zweig [2012] focus on improving the performance of recurrent neural network language models (RNNLMs) by using a topic-conditioned RNNLM. They used a contextual real-value input vector in association with each input word. This vector is used to convey contextual information about the sentence being modeled. They use Latent Dirichlet Allocation (LDA) to get a compact vector-space representation of a long span context which they conventionally interpreted as a topic representation. They argue that their approach has the key advantage of avoiding the data fragmentation associated with building multiple topic models on different data subsets. The main differences with my work are, that I used a feed-forward neural network and context vector representation instead of LDA. Also, I evaluated the impact of using various types of auxiliary feature as explained in Section 5.4.

5.4 Auxiliary features

In this work, I experimented with two types of auxiliary features: the **first** one provides a feature of the current line itself (e.g. the number of words or genre) which allows us to train feature-conditioned continuous space language models. Some of these features are motivated by research in the machine translation quality estimation literature. The **second** type of auxiliary feature aims at providing a larger context. Table 5.1 summarizes the auxiliary features of these two types that we have experimented with.

One of the basic auxiliary feature I used is **LineLen** or the line length, expressed in number of words. I used an 1-of-n encoding to generate this feature vector. The ith value in the vector is set to 1 if the line length is equal to i, and zeros otherwise. I considered a maximum line length of $n = 200$, so if the line length exceeded 200 words, I use $n = 200$. In my experiments this 1-of-n encoding is projected into a continuous space like for the words.

Auxiliary feature	Description
LineLen	number of tokens in the line
Genre	The text genre (MSA_NW_WB, EGY_DF, EGY_SMS_CHAT, EGY_CTS or MSA_FORMAL)
CurrLine	sum of the word embeddings of the **current** line
PrecLine	sum of the word embeddings of the **preceding** line
LineHCurrLines	weighted sum of the **current** and **h preceding** lines' sum of the word embeddings
AllPrecCurrWords	weighted sum of the word embeddings of the **current** and **all preceding lines' words**
AllPrecWords	weighted sum of the word embeddings of **all preceding lines' words**
AllPrecLines	weighted sum of **all preceding** lines' sum of the word embeddings

Table 5.1: *Different types of auxiliary features used in our experiments*

The **Genre** consists of a binary vector with dimension equal to the number of genres we have. As for LineLen, we used a 1-of-n encoding. In our training data, we have 5 genres as shown in the second row in Table 5.1.

For the context vector representation auxiliary features, we used various ways to compose them. One of the composition is **CurrLine** $\hat{\alpha}_l$ of a line l. This will be the normalized sum of the word embeddings e_w of all tokens $w \in l$ computed as follows:

$$\hat{\alpha}_l = \frac{\sum_{w \in l} e_w}{||\sum_{w \in l} e_w||} \tag{5.1}$$

Similarly, **PrecLine** auxiliary feature $\hat{\beta}_l$ is calculated as follows:

$$\hat{\beta}_l = \frac{\sum_{w \in l-1} e_w}{||\sum_{w \in l-1} e_w||} \tag{5.2}$$

For **PrecHCurrLines**, we calculate the weighted sum of the context vector representation of the current line $\hat{\alpha}_l$ and the preceding H lines. The farther the line is in the past, the lower the weight is. The vector of a line l is calculated as follows:

$$\hat{\eta}_{l,H} = \frac{\sum_{i=l-H}^{l} \hat{\alpha}_i \lambda^{l-i}}{||\sum_{i=l-H}^{l} \hat{\alpha}_i \lambda^{l-i}||} \tag{5.3}$$

In our experiments we used different values of H=10, 30, 50 and λ=0.95.

The differences between **AllPrecLines** and **PrecHCurrLines** is that the first one does not include the current line context vector representation in the calculation of its vector and that it uses all preceding lines not just the H preceding lines. The equation used to calculate the feature vector of AllPrecLines of a line l is as follows:

$$\hat{\omega}_l = \frac{\sum_{i=1}^{l-1} \hat{\alpha}_i \lambda^{l-i}}{||\sum_{i=1}^{l-1} \hat{\alpha}_i \lambda^{l-i}||} \tag{5.4}$$

For the first line, we used the context vector representation of itself (i.e. $\hat{\omega}_1$= $\hat{\alpha}_1$). In our experiments, we used several weights: $\lambda = 0.85, 0.95, 0.98$.

For **AllPrecCurrWords**, the line context vector representation $\hat{\sigma}_l$ is calculated using all preceding words with a weight λ that gives more weight to the near

history **words** and lower weight to the far history **words**. The equation used to calculate the feature vector of AllPrecCurrWords of a line l is the following:

$$\hat{\sigma}_l = \frac{\sum_{i=1}^{W'-1} e_{w_i} \lambda^{W'-i}}{||\sum_{i=1}^{W'-1} e_{w_i} \lambda^{W'-i}||} \tag{5.5}$$

where W' is the number of words in the current and all preceding lines. In our work we experiment with the following weights: $\lambda = 0.75, 0.85, 0.95$.

AllPrecWords is calculated in a similar way as **AllPrecCurrWords**, but excluding the words of the current line. The equation used to calculate the feature vector of AllPrecWords of a line l as follows:

$$\hat{\delta}_l = \frac{\sum_{i=1}^{\dot{W}-1} e_{w_i} \lambda^{\dot{W}-i}}{||\sum_{i=1}^{\dot{W}-1} e_{w_i} \lambda^{\dot{W}-i}||} \tag{5.6}$$

where $\dot{W}$ is the number of words in all preceding lines.

5.5 Evaluation on Penn Treebank

I first evaluated my work on the English Penn Treebank (PTB) corpus [Marcus et al., 1993]. This is a very small corpus (< 1 million words training data), but it has the advantage that many comparable results are published. I limited my evaluation on PTB to use only the preceding line auxiliary feature (i.e. PrecLine). The features **LineLen** and **CurrLine** can not be used when using perplexity to evaluate an LM since they provide information on the future. However, it is valid and useful to apply them in an n-best list re-scoring framework, as discussed later in the following sections.

The perplexity values on PTB for several configurations are shown in Table 5.2. I experiment with different learning rate scales for the first layer of the neural network as shown in the third column in Table 5.2. This means that the first layer learning rate is scaled by this value which means that the network learns the weights faster than other layers weights and possibly learns better projection weights. **Copy** means that no weights are learned and the auxiliary feature vector is copied to the next layer directly.

In CSLM1, using auxiliary features and unified learning rate scale decreased the perplexity slightly. The same happen when I replaced **Copy** by a **sequence of double hyperbolic tangent** in CSLM3, and when I increased the learning rate scale to 2 in CSLM4, comparing to Baseline2. Changing the learning rate scale to 3 in CSLM5, again, decreased the perplexity by 7.5 on dev and 7.2 on test vs. Baseline2. So the perplexity of CSLM5 compared to Baseline1 decreased by 7.6% on dev and 7.5% on test.

System	**Auxiliary layer**	**First layer learning rate scale**	**DevSet** PPL	**TestSet** PPL
Baseline1 (No Aux)	-	1	133.19	127.66
Baseline2 (No Aux)	-	2	130.48	125.28
CSLM1	Copy	1	128.26	123.45
CSLM2	Copy	2	124.80	120.32
CSLM3	Seq. of two tanh	1	127.15	121.93
CSLM4	Seq. of two tanh	2	124.22	118.57
CSLM5	Seq. of two tanh	3	**122.98**	**118.08**

Table 5.2: *Perplexity on Penn Treebank using the PrecLine auxiliary feature.*

To understand these results, I compared systems with the same setup except for one variable. Comparing Baseline1 and Baseline2 shows the impact of increasing the learning rate scale from unified to 2. Also comparing CSLM1 and CSLM2 gives us the impact related to the increase of learning rate scale for word embeddings only since the **Copy** layer used for auxiliary feature does not have any weights. Also comparing CSLM1 and CSLM3, gives us the impact of using **sequence of double hyperbolic tangent** layer for auxiliary data instead of **Copy**. I observed that this allows the network to deeply learn from the auxiliary data. These three comparisons accumulated a perplexity decrease of 7.28 on dev and 7.03 on test. We concluded that using auxiliary feature decreases the perplexity with different meta-configuration and topology by around 7.5% on dev and test.

5.6 SMT experimental results

I evaluate the performance of our improved CSLMs which use auxiliary features in the context of SMT. This is done by using them to re-score the n-best list provided by an SMT system. A new CSLM score is added to the n-best list for each hypothesis and the coefficients of all feature functions are optimized. In the following subsections, we describe our baseline system and the rescoring results with some discussions.

5.6.1 SMT system baseline

We used BOLT project phase 3 system for EGY_SMSCHAT genre as our baseline. The description of the system is detailed in Chapter 3. We applied the CSLMs with different auxiliary features and reported the results as described in the following section.

5.6.2 Re-scoring n-best list results

CSLM models with various auxiliary features were trained using CSLM toolkit on three English corpora (total of 7.91m words) which are the target side of the bilingual corpora shown in Table 5.3. Also we described in the same table the used dev and test sets (we used Dev and Test to reference these sets in the following sections (i.e. smschat tune is called Dev and smschat dev is called Test).

The results obtained by re-scoring the n-best list created by the baseline system are summarized in Table 5.4. The table contains the best result for each auxiliary feature. Detailed results can be found in Tables 5.5 and 5.6. Since the test set BLEU scores of both **SMT Baseline** and **CSLM Baseline** without auxiliary data are the same, we decided to use SMT Baseline as the Baseline for the result analysis.

These results were obtained with the best meta-parameters (i.e. H and λ). In Table 5.4, we described the CSLM model, auxiliary feature dimension, auxiliary feature projection dimension along with the BLEU scores on dev and test. We used **projection** layer for **LineLen** auxiliary feature, **Copy** layer for **Genre**

type	data set	# English tokens	genre
train	gale	5.01	MSA
	bolt	2.05m	FORUM (Egyptian)
	smschat	845k	SMS/CHAT
	Total	7.9m	-
Dev	smschat-tune	25.6k	SMS/CHAT
Test	smschat-dev	24.6k	SMS/CHAT

Table 5.3: *Training corpora and dev set used to train and tune the CSLM models*

System	Auxiliary input		Dev	Test
	Aux dim/proj.	layer		
SMT baseline (No CSLM)	-	-	27.35	25.72
CSLM Baseline (No AuxData)	-	-	28.04	25.67
LineLen	1/200	Projection 200x320	28.65	26.14
Genre	5/-	Copy 5x5	**28.90**	**26.32**
CurrLine	320/-	Sequence of two tanh 320x320	28.29	26.09
PrecLine	320/-	Sequence of two tanh 320x320	28.67	**26.33**
LineHCurrLines λ=0.95, h=50	320/-	Sequence of two tanh 320x320	**28.92**	26.26
AllPrecCurrWords λ=0.75	320/-	Sequence of two tanh 320x320	28.52	25.86
AllPrecWords λ=0.95	320/-	Sequence of two tanh 320x320	**28.77**	**26.82**
AllPrecLines λ=0.98	320/-	Sequence of two tanh 320x320	28.63	**26.52**

Table 5.4: BLEU *scores of re-scoring the n-best list using different auxiliary data.*

auxiliary feature, **sequence of double hyperbolic tangent** layer for the rest of auxiliary features. All experiments are trained with 24-gram context size.

Looking at Table 5.4, we observed a good improvement using *LineLen* auxiliary feature, but *Genre* has relatively better gain on both dev and test. This means that *Genre* is better discriminative auxiliary feature. We observed that *PrecLine* provides better performance due to better context information compared to *CurrLine*. We also observed that CSLMs with auxiliary features which contain the current line (i.e. *AllPrecCurrWords*, *PrecHCurrLines*) generally have lower BLEU scores than CSLMs with auxiliary features which do not contain the current line. We concluded that using current line is not so useful for re-scoring n-best list because instead of predicting the next word, the CSLM would rather learn to find the next word from the input auxiliary feature making undesirable cycle in the model.

PrecLine has +0.6 BLEU gain on test. If one preceding line is useful, two or more preceding lines would be more useful (possibly weighted). We can verify this assumption by looking at *AllPrecLines* result, which uses auxiliary feature that does not contain the current line (i.e. both *AllPrecCurrWords*, *PrecHCurrLines* contain the current line). The results of *AllPrecLines* is 26.52 on test which is the second best BLEU score in Table 5.4, which confirms that our assumption is correct.

Looking at the additional results of *AllPrecLines* with different λ(s) in Table 5.5, we observed that larger λ weight improved the BLEU score on both dev and test sets. The best BLEU scores are obtained using *AllPrecWords* CSLM. The only difference between *AllPrecLines* and *AllPrecWords* is that the second one is weighted sum of words' embeddings, while the first one is the weighted sum of lines' embeddings. It means that *AllPrecWords* auxiliary feature includes better and consistent context information. One possible reason for this is that for *AllPrecLines* auxiliary feature vector, each line has a different length, and hence the *weight* on each line controls the contribution of a variable number of words. This clearly is less stable than using the weighted sum of individual words em-

System	λ	Dev	Test
SMT baseline	-	27.35	25.72
CSLM Baseline	-	28.04	25.67
CurrLine	-	28.29	26.09
PrecLine	-	28.67	26.33
AllPrecLines	0.85	28.06	25.52
AllPrecLines	0.95	28.59	26.42
AllPrecLines	0.98	**28.63**	**26.52**
AllPrecWords	0.75	28.37	26.36
AllPrecWords	0.85	28.74	26.49
AllPrecWords	0.95	**28.77**	**26.82**
AllPrecCurrWords	0.75	**28.52**	**25.86**
AllPrecCurrWords	0.85	28.23	25.59
AllPrecCurrWords	0.95	28.21	25.64

Table 5.5: BLEU *scores of re-scoring n-best list using AllPrecLines, AllPrecWords and AllPrecCurrWords auxiliary features with various weights. Auxiliary layer is a sequence of two tanh 320x320.*

beddings and hence the auxiliary feature vector will be independent of individual lines lengths. In Table 5.5, we noticed the same relation between λ and the BLEU scores as we discussed for *AllPrecWords* auxiliary feature.

Looking at the results of *AllPrecCurrWords* auxiliary feature in Table 5.5, we observed that the results are inconsistent on test, λ=0.75 gives better scores than λ=0.85, but also, λ=0.95 gives better scores than λ=0.85. We concluded that including word embeddings of both current line and preceding lines in the same auxiliary feature gives inconsistent results.

For the results of *PrecHCurrLines* in Table 5.6, generally, we observed that including more preceding lines does not give better scores on test (we used maximum 50 preceding lines in these experiments), even with H=50, the scores are not better than just one preceding line **PrecLine**. We concluded that the reason is that this auxiliary feature includes the current line embeddings which cause inconsistent results on dev and almost no improvement on test.

5.7 Conclusion

In this chapter I introduced a novel method to improve the continuous space language model using auxiliary features. I used different features which some of them are motivated by the important features in machine translation quality estimation literature. The suggested auxiliary features include text genre, line length and various types of context vector representations.

I reported perplexity improvement around 7.5% on dev and test using the English Penn Treebank dataset. I also reported an improvement on a translation task up to 1.4 Bleu on dev and 1.1 on test by re-scoring n-best list of a strong baseline phrase-based SMT system. Also, the results show that the weighted sum of the word embeddings is more stable and outperforms the line level weighted sum of embeddings. These results need to be validated on other tasks with different language pairs, genres and data sets.

In future work, I would like to try using combined features and explore syntactic features. Also I would like to experiment with additional features like source language features and study their impact on the CSLM performance.

System	**H**	**Dev**	**Test**
SMT baseline	-	27.35	25.72
CSLM Baseline	-	28.04	25.67
CurrLine	-	28.29	26.09
PrecLine	-	28.67	**26.33**
PrecHCurrLines	10	28.70	26.21
PrecHCurrLines	30	28.28	**26.26**
PrecHCurrLines	50	**28.92**	**26.26**

Table 5.6: Bleu *scores using PrecHCurrLines auxiliary feature with number of preceding lines H and* $\lambda = 0.95$*. Auxiliary layer is a sequence of two tanh 320x320.*

Chapter 6

Conclusions and prospects

In this dissertation, we reported the work done in the context of BOLT program covering the activities and the different techniques that we used during this project in different phases. These techniques have been developed to improve the translation quality of Arabic/Egyptian into English. We also presented the results of LIUM Systems in the three international evaluations of the BOLT project.

Our work contributes to several research areas in machine translation by proposing new methods, algorithms and frameworks in the following areas: transliteration mining, transliteration, domain adaptation, word sense disambiguation and the continuous space language modeling. Experiments are done to evaluate the proposed techniques and the results and analysis are reported.

We worked on several general and Arabic related techniques. One of these techniques is adapting our SMT systems to the Egyptian dialect. Since the available training corpora, in the context of BOLT project, contain modern standard Arabic, and several dialects (i.e. Egyptian, Levantine and Iraqi). We improved the system performance by using domain adaptations techniques treating different dialects as different domains. We applied five adaptation techniques to adapt our system on the Egyptian dialect as well as the required system genre. The first technique is using instance weighting of translation models to improve the translation quality by giving more weights to Egyptian than modern standard Arabic and other Arabic dialects. The second method is based on using multi-domain

approach proposed in [Sennrich et al., 2013]. We presented an architecture that delays the computation of translation model features until decoding time, allowing dynamic instance weighting using optimized weights. We also used a method for unsupervised adaptation with development and test data from multiple domains (i.e. MSA and Egyptian dialect in our case). We reported a significant improvement which shows the effectiveness of multi-domain approach. Since our training corpora have various genres (i.e. News, Web, United Nations, discussion forums, SMS/Chat and conversational telephone speech transcription), we used two data selection techniques to adapt our systems on different genres. The first one [Moore and Lewis, 2010] consists of selecting the relevant sentences from other out of domain monolingual corpora to improve and adapt the language models, while the second one [Axelrod et al., 2011] is selecting the most relevant parallel sentences from out of domain bilingual corpora to improve and adapt the translation models. We also applied a fifth method for the adaptation of our systems to Egyptian using the so-called "lightly supervised" training. In this technique, we are using automatic translation of large amount of in-domain monolingual text (i.e. Egyptian dialect in our case) to improve and adapt the baseline system for in-domain translation task. This is done basically by adding portion of this large amount of new bitext to our SMT system training data.

In order to address the translation errors of ambiguous Arabic and Egyptian words, I proposed a novel word sense disambiguation technique that uses ambiguous word context. The technique makes use of the word vector space models (i.e. word embeddings) to find the correct senses of ambiguous words using their context. I used this technique in a phrase-based SMT system in order to improve the system performance related to ambiguous words.

In another work, I have performed research on several methods to decrease the number of out-of-vocabulary words by transliterating proper nouns. I presented my work [Aransa et al., 2012] of training a letter-based statistical system on the list of transliteration pairs obtained using transliteration mining. I contributed a new method for semi-supervised transliteration mining using parallel and comparable corpora. The results shows that the proposed semi-supervised

transliteration mining algorithm is effective.

In transliteration work, I proposed adapting the transliteration system on forward or backward transliteration by partitioning the training data and using instance weighting techniques. The partitioning is done automatically using supervised method. I applied this proposed technique by building two separate systems for forward and backward transliteration from Arabic into English. The detection of the transliteration direction is fully automatic. I showed a significant improvement in the transliteration performance when the origin was detected (about 50% of the training and 30% of the tune and test data). As expected, there is no notable change in the performance when the transliteration direction cannot be automatically detected and in this case an unadapted system can be used.

Finally, I contributed a novel architecture to improve the continuous space language models using auxiliary features. I used different auxiliary features motivated by the important features in the quality estimation literature. The suggested auxiliary features include text genre, length of the line, line context vector representation calculated using different ways. Experiments are done on the English Penn Treebank data. I reported a perplexity improvement of 7.5% on development and test sets. Additionally, I reported the results of re-scoring n-best list of our phrase-based MT system with a gain up to 1.1 on BLEU metric.

6.1 Prospects

There are several prospects of the work on CSLM improvement using auxiliary features, the following enhancements are interesting: When CSLM is trained on a target side of a bilingual corpora, it would be interesting to study the use of additional auxiliary features extracted from the source language side of the bitext. This could be the line length, the topic, the genre and the context vector representation of the line or other source side features. Additionally, in my work a predefined genre ID is assigned to different corpora, which is used as auxiliary feature input to the neural network. It is possible to use topics instead of genres

and to assign the topic ID dynamically by using similar automatic clustering algorithm like the one used in [Iyer and Ostendorf, 1999].

The same auxiliary features we used in feed-forward neural network, can also be used with recurrent neural network language models (RNNLMs) architecture like the one used in [Mikolov and Zweig, 2012]. This work can conclude with a comparable results that recommend one architecture over the other when auxiliary features are used. Another interesting idea, is that we used auxiliary features in continuous space language modeling, it would be interesting to study the impact of using them in various continuous space translation models [Bahdanau et al., 2014; Schwenk, 2012; Sutskever et al., 2014].

From the work done in using word sense disambiguation to improve phrase-based systems, we proposed a word sense disambiguation technique and used it to tag ambiguous words with their sense IDs. It would be interesting to use and evaluate different integration approaches in phrase-based systems. Also, to use systems combination technique to combine output of the baseline system with the output of the system that uses our word sense disambiguation technique, which will benefit from good translations in both systems. Instead of applying our approach on all ambiguous words in the corpora, it would be good to just try to tag the problematic ambiguous words only. These words could be detected by translating training data or other bilingual corpus and using the translation errors as indicators for the problematic ambiguous words.

Finally, for multi-domain work, we used an unsupervised method to cluster the sentences of the development set. We obtained a bitext for each cluster, which are used to optimize the model weights. At decoding time for test set, we need to assign a cluster and its associated optimized weight vector to each sentence. One possible extension for this work is to optimize the log-linear feature weights by MERT and then use both the associated optimized weight vector and the new log-linear features weights to translate each sentence.

Appendix A

Publications

- Walid Aransa, Holger Schwenk, Loïc Barrault, LIUM, University of Le Mans. Semi-supervised transliteration mining from parallel and comparable corpora. Proceedings IWSLT 2012, 2012 [Aransa et al., 2012].
- Rico Sennrich, Holger Schwenk, and Walid Aransa. A multi-domain translation model framework for statistical machine translation. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 832-840,Sofia, Bulgaria, August 2013. Association for Computational Linguistics [Sennrich et al., 2013].
- Walid Aransa, Semi-supervised transliteration mining from parallel Corpora. journe des doctorants de l ecole doctorale STIM (JDOC13), Nantes, France [Aransa, 2013].
- Walid Aransa, Holger Schwenk, Loïc Barrault, LIUM, University of Le Mans. Improving CSLM using auxiliary features, IWSLT 2015, 3-4 December 2015, Da Nang, Vietnam.

Appendix B

MADA/TOKAN schemes aliases

Arabic Segmentation Scheme	Description
MADA-ATB	Tokenizes all clitics except for the definite article, normalizes alefs/yaa, uses + as clitic markers, and replaces (and) characters. Only one WORD form.
MADA-D1	Tokenizes question and conjunction clitics only; uses + as a clitic marker, normalizes alefs/yaas, and replaces (and) characters. Only one WORD form.
MADA-D2	Same as D1, but also tokenizes PART clitics
MADA-D3	Same as D2, but also tokenizes all articles and enclitics (basically all clitics are tokenized)
MADA-S1	Tokenizes only the CONJ, PART, DART and PRON clitics; uses + clitic markers, normalizes alefs/yaas, and replaces (and) characters. Only one WORD form.
MADA-S2	Same as S1, except that it explictly groups the CONJ, PART and DART proclitics; there is no whitespace between the grouped clitics, but the proclitic marker + is still present to distinguish them.
MADA-ATB+POS	Same as ATB, but adds a second form the PATB POS tag. The middle-dot character is used as a form separator by default.
MADA-OLD-ATB	A tokenization that was previously used in the PATB. Only explicitly tokenizes f+, w+, b+, k+, l+, and enclitics. Uses + as clitic markers, normalizes alefs/yaas, and replaces (and) characters.
MADA-ATB4MT	A large scheme consisting of 6 forms (also referred to as a "6-tier" scheme). Form 0 is a WORD form that tokenizes all clitics except the definite article, uses + as a clitic marker, and replaces (and); Form 1 is the same, but it also normalizes alefs/yaas; Form 2 is a LEXEME form, using + clitic markers and removing diacritics; Forms 3, 4, and 5 are the CATiB, Penn ATB and Buckwalter POS tags, respectively.
MADA-D34MT	Another large 6-form (6-tier) scheme. Effectively the same as ATB4MT, except that it tokenizes all clitics.
MADA-DIAC	A single form consisting of the original word (the surface form), stripped of diacritics, with no tokenization.

Table B.1: TOKAN_SCHEME Aliases (source: Mada+Tokan Manual)

Bibliography

Günter Neumann Rabih Zbib Abdelhadi Soudi, Ali Farghaly. *Challenges for Arabic Machine Translation*. John Benjamins Publishing, 2012. 42

Hassan Al-Haj and Alon Lavie. The impact of arabic morphological segmentation on broad-coverage english-to-arabic statistical machine translation. *Machine translation*, 26(1-2):3–24, 2012. 61

Yaser Al-Onaizan and Kevin Knight. Machine transliteration of names in arabic text. In *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*, SEMITIC '02, pages 1–13. Association for Computational Linguistics, 2002. doi: 10.3115/1118637.1118642. 103

Walid Aransa. Semi-supervised transliteration mining from parallel corpora. In *13e journée des doctorants de l'ED STIM*, Nantes, France, November 2013. 145

Walid Aransa, Holger Schwenk, and Loic Barrault. Semi-supervised transliteration mining from parallel and comparable corpora. In *Proceedings IWSLT 2012*, Hong-Kong, December 2012. 142, 145

D.J. Arnold, Lorna Balkan, Siety Meijer, R.Lee Humphreys, and Louisa Sadler. *Machine Translation: an Introductory Guide*. Blackwells-NCC, London, 1993. 9, 39, 45

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics. 67, 142

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014. URL http://arxiv.org/abs/1409.0473. 11, 144

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 238–247, 2014. 129

Jerome R Bellegarda. Exploiting latent semantic information in statistical language modeling. *Proceedings of the IEEE*, 88(8):1279–1296, 2000. 129

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March 2003. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=944919.944966. 31

Nicola Bertoldi, Barry Haddow, and Jean-Baptiste Fouet. Improved minimum error rate training in moses. *Prague Bull. Math. Linguistics*, pages 7–16, 2009. 57, 112

Arianna Bisazza, Nick Ruiz, and Marcello Federico. Fill-up versus interpolation methods for phrase-based smt adaptation. In *In Proceedings of IWSLT (2011)*, 2011. 85

P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, R. Mercer, and P. Roossin. A statistical approach to language translation. In *Proceedings of the 12th Conference on Computational Linguistics - Volume 1*, COLING '88, pages 71–76, Stroudsburg, PA, USA, 1988. Association for Computational Linguistics. ISBN 963 8431 56 3. doi: 10.3115/991635.991651. URL http://dx.doi.org/10.3115/991635.991651. 11

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. A statistical approach to machine translation. *Computationnal Linguistics*, 16 (2):79–85, 1990. ISSN 0891-2017. 15, 17, 18, 19

Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995. 71

Marine Carpuat and Dekai Wu. Improving statistical machine translation using word sense disambiguation. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*, pages 61–72. ACL, 2007. URL http://www.aclweb.org/anthology/D07-1007. 87

Marine Carpuat, Yuval Marton, and Nizar Habash. 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, 2010. URL http://aclweb.org/anthology/P10-2033. 43

Yee Seng Chan, Yee Seng Chan, and Hwee Tou" Ng. Word sense disambiguation improves statistical machine translation. *IN 45TH ANNUAL MEETING OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS (ACL-07*, pages 33–40, 2007. doi: 10.1.1.133.1853. 87

Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL-1996)*, pages 310–318, 1996. 29

David Chiang. Hope and fear for discriminative training of statistical translation models. *J. Mach. Learn. Res.*, 13:1159–1187, April 2012. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=2188385.2343684. 38

Noah Coccaro and Daniel Jurafsky. Towards better integration of semantic predictors in statistical language modeling. In *ICSLP*. Citeseer, 1998. 129

Kareem Darwish. Transliteration mining with phonetic conflation and iterative training. In *Proceedings of the 2010 Named Entities Workshop*, NEWS '10,

pages 53–56. Association for Computational Linguistics, 2010. ISBN 978-1-932432-78-7. 101

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977. 19

George Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, HLT '02, pages 138–145, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc. 37

Nadir Durrani, Helmut Schmid, and Alexander Fraser. A joint sequence translation model with integrated reordering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1045–1054, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-932432-87-9. URL http://dl.acm.org/citation.cfm?id=2002472.2002604. 86

Nadir Durrani, Alexander M. Fraser, Helmut Schmid, Hieu Hoang, and Philipp Koehn. *Can Markov Models Over Minimal Translation Units Help Phrase-Based SMT?*, pages 399–405. 2013. 123

Nadir Durrani, Hieu Hoang, Philipp Koehn, and Hassan Sajjad. Integrating an unsupervised transliteration model into statistical machine translation. *EACL 2014*, page 148, 2014. 103

Chris Dyer, Jonathan Weese, Hendra Setiawan, Adam Lopez, Ferhan Ture, Vladimir Eidelman, Juri Ganitkevitch, Phil Blunsom, and Philip Resnik. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the ACL 2010 System Demonstrations*, pages 7–12. Association for Computational Linguistics, 2010. 34

Ali El-Kahky, Kareem Darwish, Ahmed Saad Aldein, Mohamed Abd El-Wahab, Ahmed Hefny, and Waleed Ammar. Improved transliteration mining using graph reinforcement. In *Proceedings of the Conference on Empirical Methods*

in Natural Language Processing, EMNLP '11, pages 1384–1393. Association for Computational Linguistics, 2011. ISBN 978-1-937284-11-4. 102

Ahmed El Kholy and Nizar Habash. Techniques for arabic morphological detokenization and orthographic denormalization. In *LREC 2010 Workshop on Language Resources and Human Language Technology for Semitic Languages*, pages 45–51, 2010. 61

Ahmed El Kholy and Nizar Habash. Translate, predict or generate: Modeling rich morphology in statistical machine translation. In *Proc. of EAMT*, volume 12, 2012. 61

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370. Association for Computational Linguistics, 2005. doi: 10.3115/1219840.1219885. 111

Markus Freitag, Matthias Huck, and Hermann Ney. Jane: Open source machine translation system combination. In *Proc. of the Conf. of the European Chapter of the Assoc. for Computational Linguistics (EACL), Gothenburg, Sweden*, pages 29–32, 2014. 34

Takaaki Fukunishi, Andrew Finch, Seiichi Yamamoto, and Eiichiro Sumita. Using features from a bilingual alignment model in transliteration mining. In *Proceedings of the 3rd Named Entities Workshop (NEWS 2011)*, pages 49–57, Chiang Mai, Thailand, November 2011. Asian Federation of Natural Language Processing. 102

Qin Gao and Stephan Vogel. Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, SETQA-NLP '08, pages 49–57, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. 19

Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. Fast decoding and optimal decoding for machine translation. In

Proceedings of the 39th Annual Meeting on Association for Computational Linguistics, ACL '01, pages 228–235, Stroudsburg, PA, USA, 2001. Association for Computational Linguistics. doi: 10.3115/1073012.1073042. URL http://dx.doi.org/10.3115/1073012.1073042. 34

Nizar Habash. 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, 2008. 61

Nizar Habash and Owen Rambow. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 573–580. Association for Computational Linguistics, 2005. 57

Nizar Habash, Ryan Roth, Owen Rambow, Ramy Eskander, and Nadi Tomeh. Morphological analysis and disambiguation for dialectal arabic. In *HLT-NAACL*, pages 426–432, 2013. 57

Nizar Y Habash. Introduction to arabic natural language processing. *Synthesis Lectures on Human Language Technologies*, 3(1):1–187, 2010. 61

Mohamed AlSharqawy Hamdy Mubarak and Esraa AlMasry. Diacritization and transliteration of proper nouns from arabic to english. In Khalid Choukri and Bente Maegaard, editors, *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, Cairo, Egypt, 2009. The MEDAR Consortium. 120

Eva Hasler, Barry Haddow, and Philipp Koehn. Margin infused relaxed algorithm for moses. *Prague Bulletin of Mathematical Linguistics*, 96:69–78, 2011. doi: 10.2478/v10108-011-0012-3. 38

Kenneth Heafield. KenLM: faster and smaller language model queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, United Kingdom, July 2011. URL http://kheafield.com/professional/avenue/kenlm.pdf. 57

David Holmes, Samsum Kashfi, and Syed Uzair Aqeel. Transliterated arabic name search. In M. H. Hamza, editor, *Communications, Internet, and Information Technology*, pages 267–273. IASTED/ACTA Press, 2004. 101

Mark Hopkins and Jonathan May. Tuning as ranking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1352–1362, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-937284-11-4. URL http://dl.acm.org/citation.cfm?id=2145432.2145575. 38

Abraham Ittycheriah, Abraham Ittycheriah, and Salim" Roukos. Direct translation model 2. *IN HLT-NAACL 2007: MAIN CONFERENCE*, pages 57–64, 2007. doi: 10.1.1.169.7438. 38

Rukmini M Iyer and Mari Ostendorf. Modeling long distance dependence in language: Topic mixtures versus dynamic cache models. *Speech and Audio Processing, IEEE Transactions on*, 7(1):30–39, 1999. 130, 144

Sittichai Jiampojamarn, Aditya Bhargava, Qing Dou, Kenneth Dwyer, and Grzegorz Kondrak. Directl: a language-independent approach to transliteration. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, NEWS '09, pages 28–31. Association for Computational Linguistics, 2009. ISBN 978-1-932432-57-2. 103

Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition (Prentice Hall Series in Artificial Intelligence)*. Prentice Hall, 1 edition, 2000. 13, 39, 41, 44

Byung-Ju Kang and Key-Sun Choi. Automatic transliteration and back-transliteration by decision tree learning. In *Proceedings of the Second International Conference on Language Resources and Evaluation, LREC 2000, 31 May - June 2, 2000, Athens, Greece*. European Language Resources Association, 2000. ISBN 2-9517408-6-7. URL http://www.lrec-conf.org/proceedings/lrec2000/pdf/227.pdf. 119

Mehdi M. Kashani, Mehdi M. Kashani, Fred Popowich, and Anoop" Sarkar. Automatic transliteration of proper nouns from arabic to english. the challenge of arabic for nlp/mt. pages 76–84, 2006. doi: 10.1.1.113.5953. 102

Slava M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE TRANSACTIONS ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING*, 35:400–401, 1987. doi: 10.1.1.129.7219. 30

Sanjeev Khudanpur and Jun Wu. Maximum entropy techniques for exploiting syntactic, semantic and collocational dependencies in language modeling. *Computer Speech & Language*, 14(4):355–372, 2000. 130

Kevin Knight. Decoding complexity in word-replacement translation models. *Comput. Linguist.*, 25(4):607–615, December 1999. ISSN 0891-2017. URL http://dl.acm.org/citation.cfm?id=973226.973232. 34

Philipp Koehn. *Statistical Machine Translation*. Cambridge University Press, New York, NY, USA, 1st edition, 2010. xii, 24

Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. 23

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Meeting of the Association for Computational Linguistics*, pages 177–180, 2007a. 57

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: open source toolkit for statistical machine translation. In

Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, ACL '07, pages 177–180, Stroudsburg, PA, USA, 2007b. Association for Computational Linguistics. URL http://portal.acm.org/citation.cfm?id=1557769.1557821. 34

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180. Association for Computational Linguistics, 2007c. 103

Roland Kuhn and Renato De Mori. A cache-based natural language model for speech recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(6):570–583, 1990. 129

Jin-Shea Kuo, Haizhou Li, and Ying-Kuei Yang. Learning transliteration lexicons from the web. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 1129–1136. Association for Computational Linguistics, 2006. doi: 10.3115/1220175.1220317. 101

Lori Lamel, Jean-Luc Gauvain, Viet Bac Le, Ilya Oparin, and Sha Meng. Improved models for mandarin speech-to-text transcription. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 4660–4663. IEEE, 2011. 34

Alon Lavie and Abhaya Agarwal. Meteor: an automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 228–231, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1626355.1626389. 38

Hai Son Le, Ilya Oparin, Abdelkhalek Messaoudi, Alexandre Allauzen, Jean-Luc

Gauvain, and François Yvon. Large vocabulary soul neural network language models. In *INTERSPEECH*, pages 1469–1472, 2011. 34

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993. 133

Tomas Mikolov and Geoffrey Zweig. Context dependent recurrent neural network language model. In *2012 IEEE Spoken Language Technology Workshop (SLT), Miami, FL, USA, December 2-5, 2012*, pages 234–239. IEEE, 2012. ISBN 978-1-4673-5125-6. doi: 10.1109/SLT.2012.6424228. URL http://dx.doi.org/10.1109/SLT.2012.6424228. 127, 130, 144

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a. 90

Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013b. URL http://arxiv.org/abs/1301.3781. xiii, 88, 89

A Kumaran Ming Liu Min Zhang, Haizhou Li, editor. *Report of NEWS 2012 Machine Transliteration Shared Task*, volume pages 10–20, Jeju, Republic of Korea, July 2012. Association for Computational Linguistics. 100, 109

Robert C. Moore and William Lewis. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 220–224, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1858842.1858883. 66, 142

Owen Rambow Nizar Habash and Ryan Roth. Mada+tokan: A toolkit for arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In Khalid Choukri and Bente Maegaard, editors, *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, Cairo, Egypt, April 2009. The MEDAR Consortium. ISBN 2-9517408-5-9. 106, 116

Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 160–167, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/1075096.1075117. URL http://dx.doi.org/10.3115/1075096.1075117. 27, 38, 57, 112

Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29(1):19–51, March 2003a. ISSN 0891-2017. doi: 10.1162/089120103321337421. 106

Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignement models. *Computational Linguistics*, 29(1):19–51, 2003b. 19

Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29:19–51, March 2003c. 57

Franz Josef Och, Christoph Tillmann, Hermann Ney, and Lehrstuhl Fiir Informatik. Improved alignment models for statistical machine translation. In *University of Maryland, College Park, MD*, pages 20–28, 1999. 37

Franz Josef Och, Nicola Ueffing, and Hermann Ney. An efficient a* search algorithm for statistical machine translation. In *In Data-Driven Machine Translation Workshop*, pages 55–62, 2001. 34

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. 37, 55

Junho Park, Xunying Liu, Mark JF Gales, and Philip C Woodland. Improved neural network based language modelling and adaptation. In *INTERSPEECH*, pages 1041–1044, 2010. 34

Ying Qin and GuoHua Chen. *Proceedings of the 3rd Named Entities Workshop (NEWS 2011)*, chapter Forward-backward Machine Transliteration between

English and Chinese Based on Combined CRFs, pages 82–85. Asian Federation of Natural Language Processing, 2011. URL http://aclweb.org/anthology/W11-3212. 119

Anthony Rousseau. Xenc: An open-source tool for data selection in natural language processing. *The Prague Bulletin of Mathematical Linguistics*, 100: 73–82, 2013. 57

Robert Russell. Specifications of letters. US patent number 1,261,167, 1918. 101

Fatiha Sadat and Nizar Habash. Combination of arabic preprocessing schemes for statistical machine translation. 2006. 61

Hassan Sajjad, Alexander Fraser, and Helmut Schmid. An algorithm for unsupervised transliteration mining with an application to word alignment. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 430–439, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-932432-87-9. 103

Hassan Sajjad, Alexander Fraser, and Helmut Schmid. A statistical model for unsupervised and semi-supervised transliteration mining. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 469–477. Association for Computational Linguistics, July 2012. 102

Wael Salloum and Nizar Habash. Dialectal to standard arabic paraphrasing to improve arabic-english statistical machine translation. In *Proceedings of the First Workshop on Algorithms and Resources for Modelling of Dialects and Language Varieties*, DIALECTS '11, pages 10–21, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-937284-17-6. URL http://dl.acm.org/citation.cfm?id=2140533.2140535. 61

Wael Salloum and Nizar Habash. Dialectal arabic to english machine translation: Pivoting through modern standard arabic. In *HLT-NAACL*, pages 348–358, 2013. 61

Holger Schwenk. Efficient training of large neural networks for language modeling. In *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, volume 4, pages 3059–3064. IEEE, 2004. 33

Holger Schwenk. Continuous Space Language Models. 21(3):492–518, 2007. ISSN 0885-2308. doi: http://dx.doi.org/10.1016/j.csl.2006.09.003. 34, 57

Holger Schwenk. Investigations on large- scale lightly-supervised training for statistical machine translation. In *IWSLT*, pages 182–189, 2008a. 34

Holger Schwenk. Investigations on large scale lightly-supervised training for statistical machine translation. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, pages 182–189, 2008b. 83

Holger Schwenk. Continuous space language models for statistical machine translation. In *The Prague Bulletin of Mathematical Linguistics, (93):137–146.*, 2010. 34, 57, 85, 128

Holger Schwenk. Continuous space translation models for phrase-based statistical machine translation. In Martin Kay and Christian Boitet, editors, *COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Posters, 8-15 December 2012, Mumbai, India*, pages 1071–1080. Indian Institute of Technology Bombay, 2012. URL `http://aclweb.org/anthology/C/C12/C12-2104.pdf`. 57, 144

Holger Schwenk, Holger Schwenk, and Jean-luc" Gauvain. Connectionist language modeling for large vocabulary continuous speech recognition. *IN INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING*, pages 765–768, 2002. doi: 10.1.1.18.3191. 34

Holger Schwenk, Daniel Déchelotte, and Jean-Luc Gauvain. Continuous space language models for statistical machine translation. In *Proceedings of the COLING/ACL Conference*, pages 723–730, Morristown, NJ, USA, 2006. Association for Computational Linguistics. 34

Rico Sennrich. Perplexity minimization for translation model domain adaptation in statistical machine translation. In *Proceedings of the 13th Conference of the*

European Chapter of the Association for Computational Linguistics, pages 539–549, Avignon, France, April 2012. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/E12-1055. 71, 78, 123

Rico Sennrich, Holger Schwenk, and Walid Aransa. A multi-domain translation model framework for statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 832–840, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P13-1082. 75, 142, 145

Christophe Servan and Holger Schwenk. Optimising multiple metrics with mert. *The Prague Bulletin of Mathematical Linguistics*, 96:109–117, 2011. 55

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 223–231, 2006. 37, 55

Matthew Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. Fluency, adequacy, or HTER? Exploring different human judgments with a tunable MT metric. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, StatMT '09, pages 259–268, Athens, Greece, March 2009. Association for Computational Linguistics. 38

Harold Somers. Review article: Example-based machine translation. *Machine Translation*, 14(2):113–157, June 1999. ISSN 0922-6567. doi: 10.1023/A:1008109312730. URL http://dx.doi.org/10.1023/A:1008109312730. 15

Andreas Stolcke. Srilm - an extensible language modeling toolkit. *In Proceesings of the 7th International Conference on Spoken Language Processing (ICSLP 2002*, pages 901–904, 2002. doi: 10.1.1.157.2429. 57

Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014. 11, 144

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180. Association for Computational Linguistics, 2003. doi: 10.3115/1073445.1073478. 106, 116

Bernard Vauquois. A survey of formal grammars and algorithms for recognition and transformation in mechanical translation. In *IFIP Congress (2)*, pages 1114–1122, 1968. xii, 12

Ye-Yi Wang and Alex Waibel. Modeling with structures in statistical machine translation. In *Proceedings of the 17th international conference on Computational linguistics - Volume 2*, COLING '98, pages 1357–1363, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics. doi: 10.3115/980432.980790. URL http://dx.doi.org/10.3115/980432.980790. 34

Andy Way and Jinhua Du. The impact of source-side syntactic reordering on hierarchical phrase-based smt. 2010. 43

Rabih Zbib, Erika Malchiodi, Jacob Devlin, David Stallard, Spyros Matsoukas, Richard Schwartz, John Makhoul, Omar F. Zaidan, and Chris Callison-Burch. Machine translation of arabic dialects. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 49–59, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. ISBN 978-1-937284-20-6. URL http://dl.acm.org/citation.cfm?id=2382029.2382037. 61

Andreas Zollmann, Ashish Venugopal, and Stephan Vogel. Bridging the inflection morphology gap for arabic statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 201–204. Association for Computational Linguistics, 2006. 61