# Machine Translation of Arabic Dialects

**Wael Salloum**

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in the Graduate School of Arts and Sciences

**COLUMBIA UNIVERSITY**

2018

# ABSTRACT

## Machine Translation of Arabic Dialects

**Wael Salloum**

This thesis discusses different approaches to machine translation (MT) from Dialectal Arabic (DA) to English. These approaches handle the varying stages of Arabic dialects in terms of types of available resources and amounts of training data. The overall theme of this work revolves around building dialectal resources and MT systems or enriching existing ones using the currently available resources (dialectal or standard) in order to quickly and cheaply scale to more dialects without the need to spend years and millions of dollars to create such resources for every dialect.

Unlike Modern Standard Arabic (MSA), DA-English parallel corpora is scarcely available for few dialects only. Dialects differ from each other and from MSA in orthography, morphology, phonology, and to some lesser degree syntax. This means that combining all available parallel data, from dialects and MSA, to train DA-to-English statistical machine translation (SMT) systems might not provide the desired results. Similarly, translating dialectal sentences with an SMT system trained on that dialect only is also challenging due to different factors that affect the sentence word choices against that of the SMT training data. Such factors include the level of dialectness (e.g., code switching to MSA versus dialectal training data), topic (sports versus politics), genre (tweets versus newspaper), script (Arabizi versus Arabic), and timespan of test against training. The work we present utilizes any available Arabic resource such as a preprocessing tool or a parallel corpus, whether MSA or DA, to improve DA-to-English translation and expand to more dialects and sub-dialects.

The majority of Arabic dialects have no parallel data to English or to any other foreign language. They also have no preprocessing tools such as normalizers, morphological analyzers, or tokenizers. For such dialects, we present an MSA-pivoting approach where DA sentences are translated to MSA first, then the MSA output is translated to English using the wealth of MSA-English parallel data. Since there is virtually no DA-MSA parallel data to train an SMT system, we build a rule-based DA-to-MSA MT system, ELISSA, that uses morpho-syntactic translation rules along with dialect identification and language modeling components. We also present a rule-based approach to quickly and cheaply build a dialectal morphological analyzer, ADAM, which provides ELISSA with dialectal word analyses.

Other Arabic dialects have a relatively small-sized DA-English parallel data amounting to a few million words on the DA side. Some of these dialects have dialect-dependent preprocessing tools that can be used to prepare the DA data for SMT systems. We present techniques to generate synthetic parallel data from the available DA-English and MSA-English data. We use this synthetic data to build *statistical* and *hybrid* versions of ELISSA as well as improve our rule-based ELISSA-based MSA-pivoting approach. We evaluate our best MSA-pivoting MT pipeline against three direct SMT baselines trained on these three parallel corpora: DA-English data only, MSA-English data only, and the combination of DA-English and MSA-English data. Furthermore, we leverage the use of these four MT systems (the three baselines along with our MSA-pivoting system) in two system combination approaches that benefit from their strengths while avoiding their weaknesses.

Finally, we propose an approach to model dialects from monolingual data and limited DA-English parallel data without the need for any language-dependent preprocessing tools. We learn DA preprocessing rules using word embedding and expectation maximization. We test this approach by building a morphological segmentation system and we evaluate its performance on MT against the state-of-the-art dialectal tokenization tool.

# Contents

# List of Figures

This page intentionally left blank.

# List of Tables

# Acknowledgements

After years of hard work and a long journey to complete my dissertation, I would like to express my gratitude to all the people who supported me and gave me guidance through out the years. First, I would like to express my gratitude to my advisor Nizar Habash. He has been a great advisor and mentor in life. His encouragement and continuous support is tremendous. Nizar is always keen to transfer all the knowledge he possess to his students and his door is always open to help with any issue. I would like also to thank the other members of the committee Kathleen McKeown, Owen Rambow, Michael Collins, and Smaranda Muresan for being part of this thesis and devoting time to review my dissertation.

I would like to thank all the people at the Center for Computational Learning Systems (CCLS) for being great colleagues and friends. It was a pleasure being part of the group and special thanks to Mona Diab, Owen Rambow, Ahmed El-Kholy, Heba ElFardy, Mohamed Altantawy, Ryan Roth, and Ramy Eskander. I spent great time with the students at CCLS and I feel lucky being among nice and smart people like Sarah Alkuhlani, Boyi Xie, Weiwei Guo, Vinod Kumar, Daniel Bauer, Apoorv Agarwal, Noura Farra and Mohammad Sadegh Rasooli. I hope we stay in contact and I wish them all the best in their life and career.

Finally, I would like to express my gratitude to all my family and friends. I have been lucky to receive an amazing support from my parents, my wife Linda, my sister Duha and brother Bassel. Their support and encouragement has been vital in my overcoming several hurdles in life.

This page intentionally left blank.

To my parents, my wife Linda, my daughter Emma,

my sister Duha, and my brother Bassel.

This page intentionally left blank.

# Chapter 1

# Introduction

A language can be described as a set of dialects, among which one "standard variety" has a special representative status.[1] The standard variety and the other dialects typically differ lexically and phonologically, but can also differ morphologically and syntactically. The type and degree of differences varies from language to another. Some dialects co-exist with the standard variety in a *diglossic* relationship (Ferguson, 1959) where the standard and the dialect occupy different roles, e.g., formal vs informal registers. Additionally, there are different degrees of dialect-switching that take place in such languages which puts sentences on a dialectness spectrum.

Non-standard dialects are the languages that people speak at home and in their communities. These colloquial spoken varieties have been confined to spoken form in the past; however, the emergence of online communities since the early 2000s has made these dialects ubiquitous in informal written genres such as social media. For any artificial intelligence (AI) system to draw insights from such genres, it needs to know how to process such informal dialects. Furthermore, while the recent advances in automatic speech recognition has passed the usability threshold for personal assistant software for many languages, these

[1]The line between "language" and dialects is often a political question; this is beautifully highlighted by the quip *A language is a dialect with an army and navy* often attributed to Linguist Max Weinreich.

products must consider the colloquial varieties the preferred form for dictation to be usable for diglossic languages.

Despite being ubiquitous and increasingly vital to AI applications' usability, most non-standard dialects are resource-poor compared to their standard variety. For statistical machine translation (SMT), which relies on the existence of parallel data, translating from non-standard dialects is a challenge. Common approaches to address this challenge include: pivoting on the standard variety, extending tools of the standard variety to cover dialects, noisy collection of dialectal training data, or simple pooling of resources for different dialects. In this thesis, we work on Arabic, is a prototypical diglossic language, and we present various approaches to deal with the limited resources available for its dialects. We tailor our solutions to the type and amount of resources available for dialects in terms of parallel data or preprocessing tools.

In this chapter, we give a short introduction to some machine translation (MT) techniques we use in our approaches. We also introduce Arabic and its dialects and the challenges they pose to natural language processing (NLP) in general and MT in particular. Finally, we present a summary of our contributions to the topic of machine translation of Arabic dialect.

## 1.1 Introduction to Machine Translation

A Machine Translation system (Hutchins, 1986; Koehn, 2009) takes content in one language as input and automatically produces a translation of that content in another language. Researchers have experimented with different types of content, like text and audio, and different levels of content granularity, like sentences, paragraphs, and documents. In this work we are only interested in text-based sentence-level MT.

**Rule-Based versus Statistical Machine Translation.** A Rule-Based (or, Knowledge-Based) Machine Translation (RBMT) system (Nirenburg, 1989) utilizes linguistic knowledge about the source and target languages in the form of rules that are executed to transform input content into its output equivalents. Statistical Machine Translation (SMT) (Brown et al., 1990), on the other hand, builds statistical models from a collection of data (usually in the form of sentence-aligned parallel corpus). It later uses those models to translate source language content to target language. While RBMT is thought of as a traditional approach to MT, it is still effective for languages with limited parallel data, also known as, low-resource language pairs. Additionally, new hybrid approaches have emerged to combine RBMT and SMT in situations where both fail to achieve a satisfactory level of accuracy and fluency.

**Statistical Machine Translation: Phrase-based versus Neural.** Phrase-based models to statistical machine translation (PBSMT) (Zens et al., 2002; Koehn et al., 2003; Och and Ney, 2004) give the state-of-the-art performance for most languages. The deep learning wave has recently reached the machine translation field. Many interesting network architectures have been proposed and have outperformed Phrase-Based SMT with large margins on language pairs like French-English and German-English. The caveat is that these models require enormous amounts of parallel data; e.g., in the case of French-English, they're trained on hundreds of millions of words. As of the time of writing this thesis, training NMT models on relatively small amounts of parallel data results in hallucinations. Our published research we present in this thesis was published before the advances in NMT. We have not used NMT in this work, mainly because our training data is relatively small and because our approaches do not depend necessarily on PBSMT; instead, they use PBSMT to extrinsically evaluate the effect of the preprocessing tools we create on machine translation quality. In this work, whenever we mention SMT we mean PBSMT.

**System Combination.** A Machine Translation System Combination approach combines the output of multiple MT system in order to achieve a better overall performance (Och et al., 2004; Rosti et al., 2007; Karakos et al., 2008; He et al., 2008). The combination could be achieved by selecting the best output hypothesis or by combining them on the word or phrase level using techniques such as confusion network or lattice decoding to list a few.

I will discuss more topics and techniques in machine translation relevant to my work in Chapter 2.

## 1.2 Introduction to Arabic and its Challenges for NLP

Arabic is a Central Semitic language that goes back to the Iron Age (Al-Jallad, 2017) and is now by far the most widely spoken Afro-Asiatic language. Contemporary Arabic is a collection of varieties that are spoken by as many as 422 million speakers, of which 290 million are native speakers, making Arabic the fifth most spoken language in the world in both native speakers and total speakers rankings.

### 1.2.1 Arabic as a Prototypical Diglossic Language

The sociolinguistic situation of Arabic provides a prime example of diglossia where these two distinct varieties of Arabic co-exist and are used in different social contexts. This diglossic situation facilitates code-switching in which a speaker switches back and forth between the two varieties, sometimes even within the same sentence. Despite not being the native language for any group of people, Modern Standard Arabic is widely taught at schools and universities and is used in most formal speech and in the writing of most books, newsletters, governmental documents, and other printed material. MSA is sometimes used in broadcasting, especially in the news genre. Unlike MSA, Arabic dialects are spoken as a first language and are used for nearly all everyday speaking situations, yet they do not have standard orthography.

### 1.2.2 Modern Standard Arabic Challenges

The Arabic language is quite challenging for natural language processing tasks. Arabic is a morphologically complex language which includes rich inflectional morphology, expressed both templatically and affixationally, and several classes of attachable clitics. For example, the Arabic word وسيكتبونها *w+s+y-ktb-wn+hA*[2] 'and they will write it' has two proclitics (+و *w+* 'and' and +س *s+* 'will'), one prefix -ي *y-* '3rd person', one suffix -ون *-wn* 'masculine plural' and one pronominal enclitic +ها *+hA* 'it/her'. Additionally, Arabic is written with optional diacritics that specify short vowels, consonantal doubling and the nunation morpheme. The absence of these diacritics together with the language's rich morphology lead to a high degree of ambiguity: e.g., the Buckwalter Arabic Morphological Analyzer (BAMA) produces an average of 12 analyses per word. Moreover, some Arabic letters are often spelled inconsistently which leads to an increase in both sparsity (multiple forms of the same word) and ambiguity (same form corresponding to multiple words), e.g., variants of Hamzated Alif, أ > or إ <, are often written without their Hamza (ء '): ا *A*; and the Alif-Maqsura (or dotless Ya) ى Y and the regular dotted Ya ي *y* are often used interchangeably in word final position (El Kholy and Habash, 2010). Arabic complex morphology and ambiguity are handled using tools for analysis, disambiguation and tokenization (Habash and Rambow, 2005; Diab et al., 2007).

### 1.2.3 Dialectal Arabic Challenges

Contemporary Arabic is a collection of varieties: MSA, which has a standard orthography and is used in formal settings, and DAs, which are commonly used informally and with increasing presence on the web, but which do not have standard orthographies. There are several DA varieties which vary primarily geographically, e.g., Levantine Arabic, Egyptian

[2]Arabic transliteration is in the Habash-Soudi-Buckwalter scheme (Habash et al., 2007).

Arabic, etc (Habash, 2010).

DAs differ from MSA phonologically, morphologically and to some lesser degree syntactically. The differences between MSA and DAs have often been compared to Latin and the Romance languages (Habash, 2006). The morphological differences are most noticeably expressed in the use of clitics and affixes that do not exist in MSA. For instance, the Levantine and Egyptian Arabic equivalent of the MSA example above is وحيكتبوها *w+H+y-ktb-w+hA* 'and they will write it'[3]. The optionality of vocalic diacritics helps hide some of the differences resulting from vowel changes; compare the diacritized forms: Levantine *wHayuktubuwhA*, Egyptian *waHayiktibuwhA* and MSA *wasayaktubuwnahA* (Salloum and Habash, 2011). It is important to note that Levantine and Egyptian differ a lot in phonology, but the orthographical choice of dropping short vowels (expressed as diacritics in Arabic script) bridges the gab between them. However, when writing Arabic in Latin script, known as Arabizi, which is an orthographical choice picked by many people mainly in social media discussions, chat and SMS genres, phonology is expressed by Latin vowels, which brings back the gap between dialects and sub-dialects.

All of the NLP challenges of MSA described above are shared by DA. However, the lack of standard orthographies for the dialects and their numerous varieties causes *spontaneous orthography*, which poses new challenges to NLP (Habash et al., 2012b). Additionally, DAs are rather impoverished in terms of available tools and resources compared to MSA; e.g., there is very little parallel DA-English corpora and almost no MSA-DA parallel corpora. The number and sophistication of morphological analysis and disambiguation tools in DA is very limited in comparison to MSA (Duh and Kirchhoff, 2005; Habash and Rambow, 2006; Abo Bakr et al., 2008; Habash et al., 2012a). MSA tools cannot be effectively used to handle DA: (Habash and Rambow, 2006) report that less than two-thirds of Levantine verbs can be analyzed using an MSA morphological analyzer; and (Habash

[3]Another spelling variation for Egyptian Arabic is to spell the word as وهيكتبوها *w+h+y-ktb-w+hA*.

et al., 2012a) report that 64% of Egyptian Arabic words are analyzable using an MSA analyzer.

### 1.2.4 Dialectness, Domain, Genre, and Timespan

In addition to the previous challenges, other aspects contribute to the challenges of Arabic NLP in general and MT in particular like the level of sentence dialectness, and the sentence domain and genre. Habash et al. (2008) defines five levels of sentence dialectness: 1) perfect MSA, 2) imperfect MSA, 3) Arabic with full dialect switching, 4) dialect with MSA incursions, and 5) pure dialect. These five levels create confusions to MT systems and increase errors in preprocessing tools like tokenizers. They also raise the question of whether or how to use the huge collection of MSA-English parallel corpora in training a DA-English SMT. If added on top of the limited DA-English data it could hurt the translation quality of some sentences while helping others based on their level of dialectness.

Similarly, the domain and the genre of the sentence will increase challenges for MT. The task of translating content of news, news wire, weblog, chat, SMS, emails, and speech transcripts will require more DA-English training data of the already limited parallel corpora. Add on top of that the timespan of the training data versus the dev/test sets. For example, consider a test set that uses recent terminology related to Arab Spring events, politicians, places, and recent phrases and terminology that are never mentioned in the older training data.

### 1.2.5 Overview and Challenges of Dialect-Foreign Parallel Data

Arabic dialects are in different states in terms of the amount of dialect-foreign parallel data. The Defense Advanced Research Projects Agency (DARPA), as part of its projects concerning with machine translation of Arabic and its dialects to English: Global Autonomous Language Exploitation (GALE) and Broad Operational Language Translation

(BOLT), has provided almost all of the DA-English parallel data available at the time of writing this thesis. The Egyptian-English language pair has the largest amount of parallel data ∼2.4MW (million words), followed by Levantine-English with ∼1.5MW, both provided by DARPA's BOLT. Other dialect-English pairs, like Iraqi-English, have smaller parallel corpora while the majority of dialects and subdialects have no parallel corpora whatsoever.

Modern Standard Arabic (MSA) has a wealth of MSA-English parallel data amounting to hundreds of millions of words. The majority of this data, however, is originated from the United Nations (UN) parallel corpus which is a very narrow genre that could hurt the quality of MT on other genres when combined with other, smaller, domain-specific parallel corpora. We have trained an SMT system on over two hundred million words of parallel corpora that include the UN corpus as part of NIST OpenMT Eval 2012 competition. When we tested this system in an MSA-pivoting approach to DA-to-English MT, it performed worse than a system trained on a subset of the corpora that excludes the UN corpus. Other sources for MSA-English data come from the news genre in general. While DARPA's GALE program provides about ∼49.5MW of MSA-English data mainly in the news domain, it is important to note that it comes from data collected before the year 2009 since this affects the translation quality of MSA sentences discussing later events such as the Egyptian revolution of 2011.

## 1.3 Contributions

In this section we present the research contributions of this dissertation along with the released tools which we developed as part of this work.

### 1.3.1 Thesis Contributions

In the work presented in this thesis, we are concerned with improving the quality of Dialectal Arabic to English machine translation. We propose approaches to handle different dialects based on their *resource availability* situation. By resources we mean **DA-English parallel data** and **DA-specific preprocessing tools** such as morphological analyzers and tokenizers. We do not consider labeled data such as treebanks since these resources are used to train preprocessing tools. We build tools and resources that use and extend the currently available resources to quickly and cheaply scale to more dialects and sub-dialects.

Figure 1.1 shows a layout of the different settings of Arabic dialects in terms of resource availability. The columns represent the unavailability or availability of DA preprocessing tools while the rows represent the unavailability or availability of DA-English parallel data. This layout results in four quadrants that display an overview of our contributions in the four DA settings. Figure 1.2 displays diagrams representing our baselines and approaches in the four quadrants. Figures 1.3, 1.4, 1.5, and 1.6 follow the same 4-quadrant layout and display the diagrams, baselines and approaches of each quadrant. This dissertation is divided into three parts discussing three quadrants out of the four. We don't discuss quadrant 2 in details since it can be achieved from quadrant 1 by creating DA tools.

In the first part, we propose solutions to handle **dialects with no resources** (Part-I). Figure 1.3 shows an overview of baselines and contributions in this part. The available resources in this case are parallel corpora and preprocessing tools for Arabic varieties other than this dialect. In our case study, we have MSA-English data and an MSA preprocessing tool, MADA (Habash and Rambow, 2005; Roth et al., 2008). The best baseline we can get is by preprocessing (normalizing, tokenizing) the MSA side of the parallel data with the available tools and training an SMT system on it.

- **ADAM and morphological tokenization.** The biggest challenge for translating these dialects with an MSA-to-English SMT system is the large number of out-of-

vocabulary (OOV) words. This is largely caused by dialectal morphemes attaching to words many of which come from MSA. A quick and cheap approach to handle OOVs of these dialects is to build a morphological segmentation or tokenization tool to break morphologically-complex words into simpler, more frequent, tokens. For this purpose, we propose ADAM which extends an existing morphological analyzer for Modern Standard Arabic to cover a few dialects. We show how tokenizing a dialectal input sentence with ADAM can improve its MT quality when translating with an MSA-to-English SMT system.

- **ELISSA and MSA-pivoting.** If we can translate dialectal words and phrases to their MSA equivalents, instead of just tokenizing them, perhaps the MSA-to-English SMT system can have a better chance translating them. There is virtually no DA-MSA parallel data to train an SMT system. Therefore, we propose a rule-based DA-to-MSA MT system called ELISSA. ELISSA identifies dialectal words and phrases that need to be translated, and uses ADAM in a morpho-syntactical analysis-transfer-generation approach to produce a lattice of MSA options. ELISSA, then, scores and decodes this lattice with an MSA language model. The output of ELISSA is then tokenized by MADA to be translated by the MSA-to-English SMT system. It is important to note that ELISSA can be used as in the context where a dialect has a DA preprocessing tools but no DA-English data. In that case we can replace ADAM inside ELISSA with the DA-specific analyzer.

In the second part (Part-II), which represents the third quadrant, we concern with **dialects that have parallel data as well as preprocessing tools**. Figure 1.4 and Figure 1.5 display our baselines and approaches for Part-II in the third quadrant. The DA-English parallel data and preprocessing tools allow for the creation of better baselines than the MSA-to-English SMT system. The questions are whether an MSA-pivoting approach can

still improve over a direct-translation SMT system, and whether adding the MSA-English data to the DA-English data can improve or hurt the SMT system's performance.

- **Synthetic parallel data generation.** Using the MSA-English parallel corpora and the DA-to-MSA MT system (ELISSA) we had from the first part, we implement two sentence-level pivoting techniques to generate synthetic MSA side for the DA-English data.

- **Statistical/hybrid ELISSA and improved MSA-pivoting.** We use this DA-$\text{MSA}_{synthetic}$-English parallel data to build statistical and hybrid versions of ELISSA as well as improve the MSA-pivoting approach.

- **System combination.** We compare the best MSA-pivoting system to three direct translation SMT systems: one trained on DA-English corpus only, one trained on MSA-English corpus only, and one trained on the two corpora combined. Instead of choosing one best system from the four, we present two system combination approaches to utilize these systems in a way the benefits from their strengths while avoiding their weaknesses.

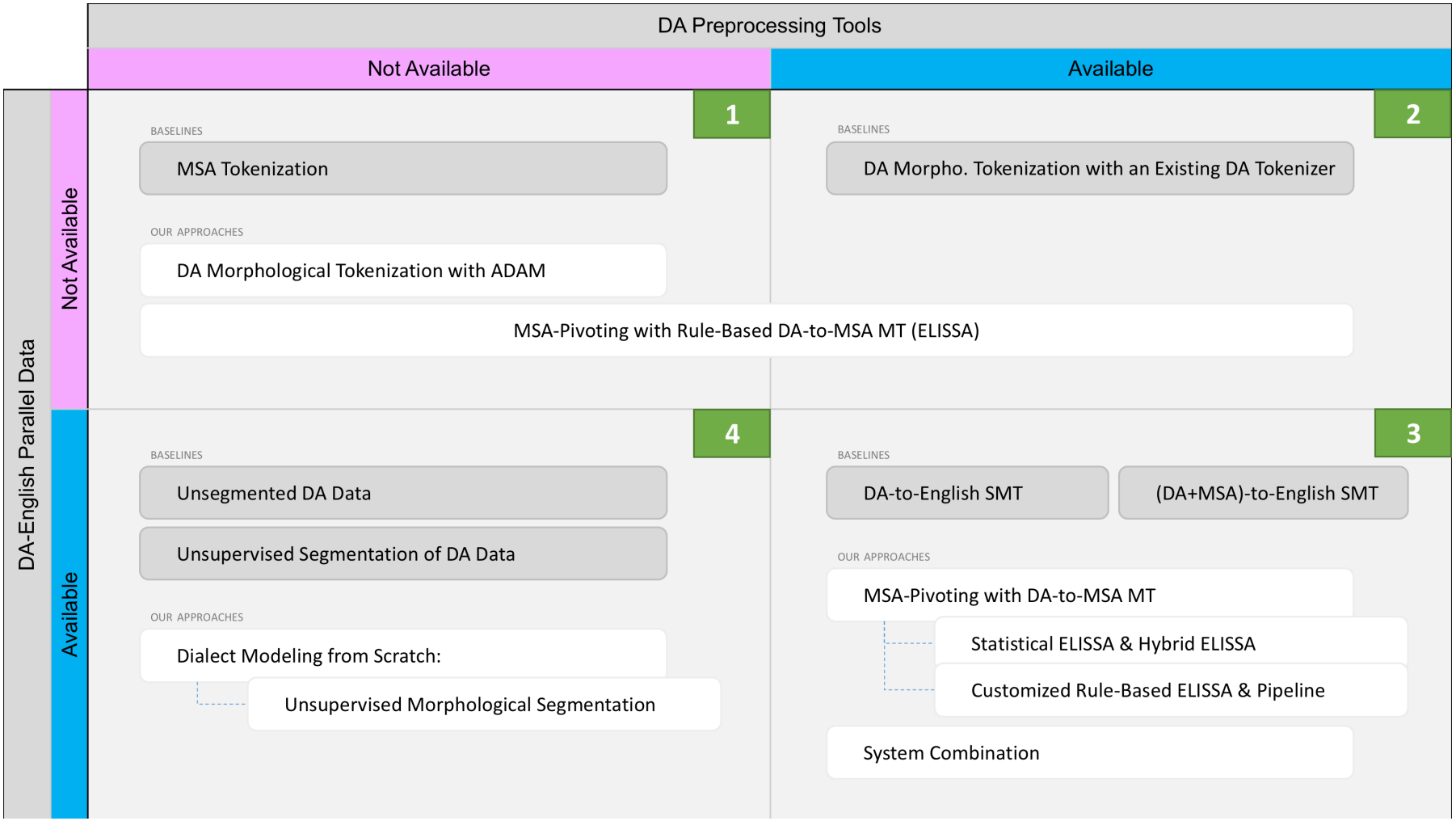



Figure 1.1: Our contributions: A view of our baselines and approaches. The columns represent the unavailability or availability of DA preprocessing tools while the rows represent the unavailability or availability of DA-English parallel data. The cells present our contributions in each setting.

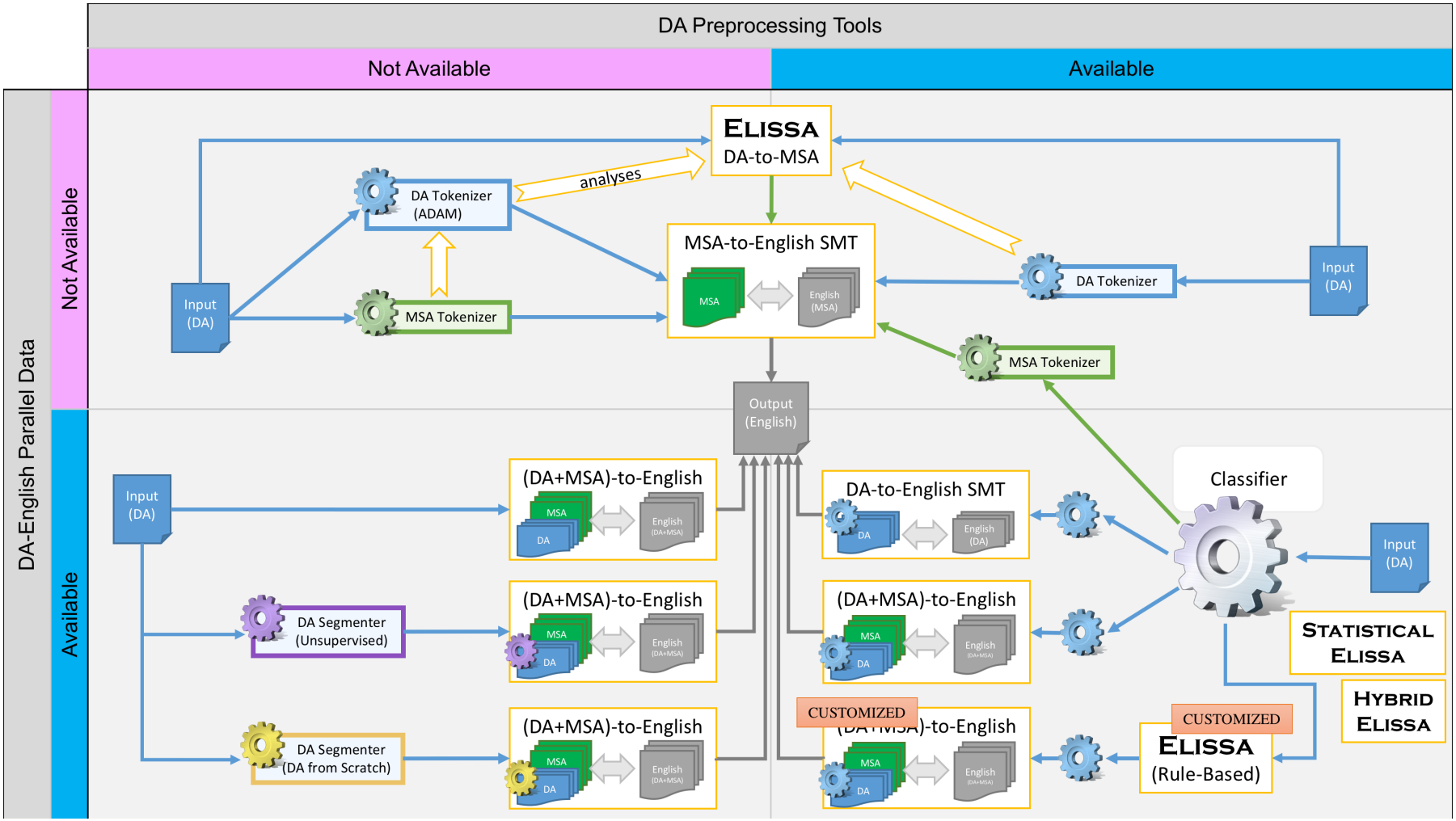



Figure 1.2: Our contributions: A diagram view of our baselines and approaches. The columns represent the unavailability or availability of DA preprocessing tools while the rows represent the unavailability or availability of DA-English parallel data. The cells present our contributions in each setting.

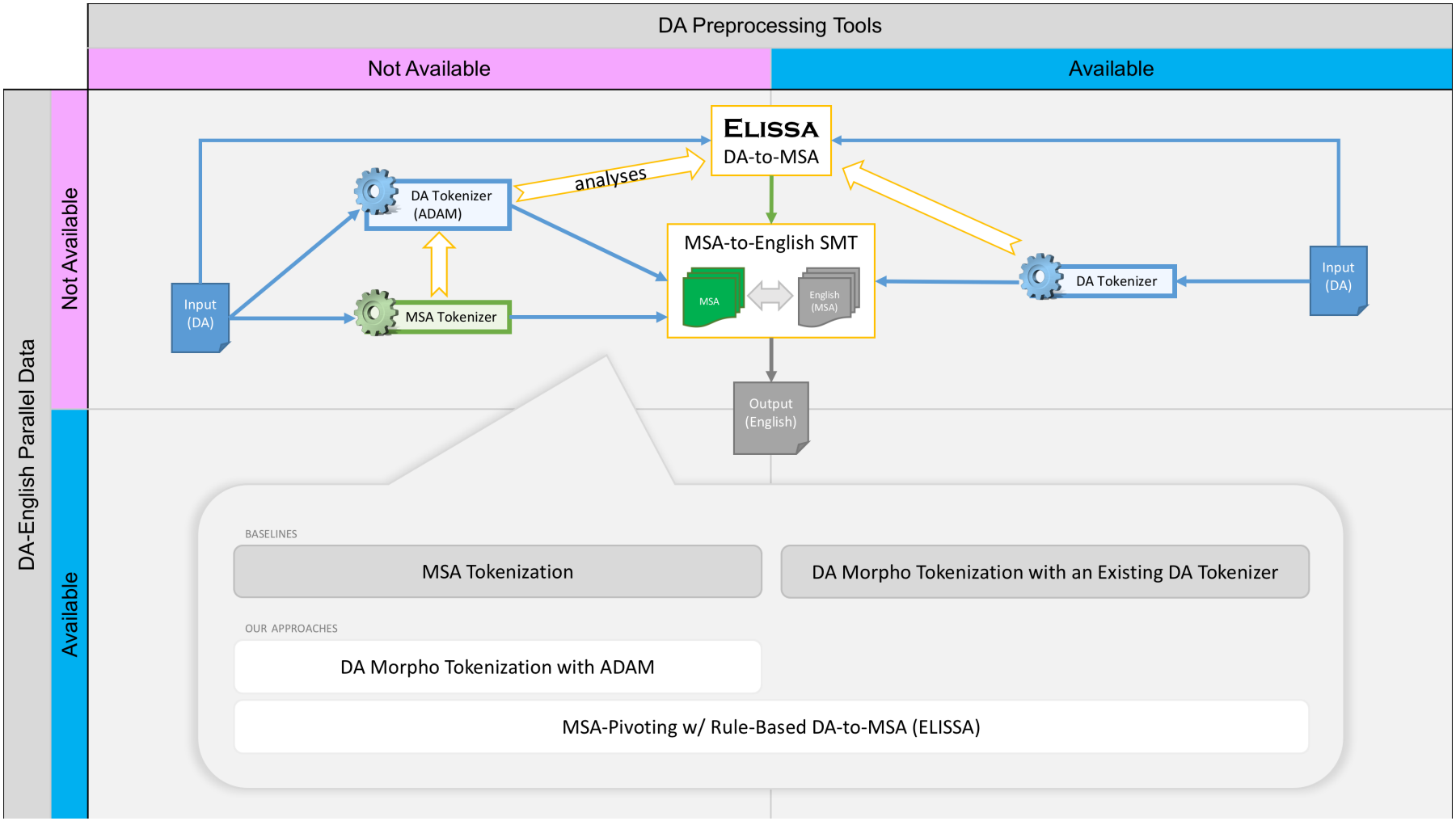



Figure 1.3: Our contributions: A view of our baselines and approaches for quadrant 1 and 2. The columns represent the unavailability or availability of DA preprocessing tools while the rows represent the unavailability or availability of DA-English parallel data. The cells present our contributions in each setting.

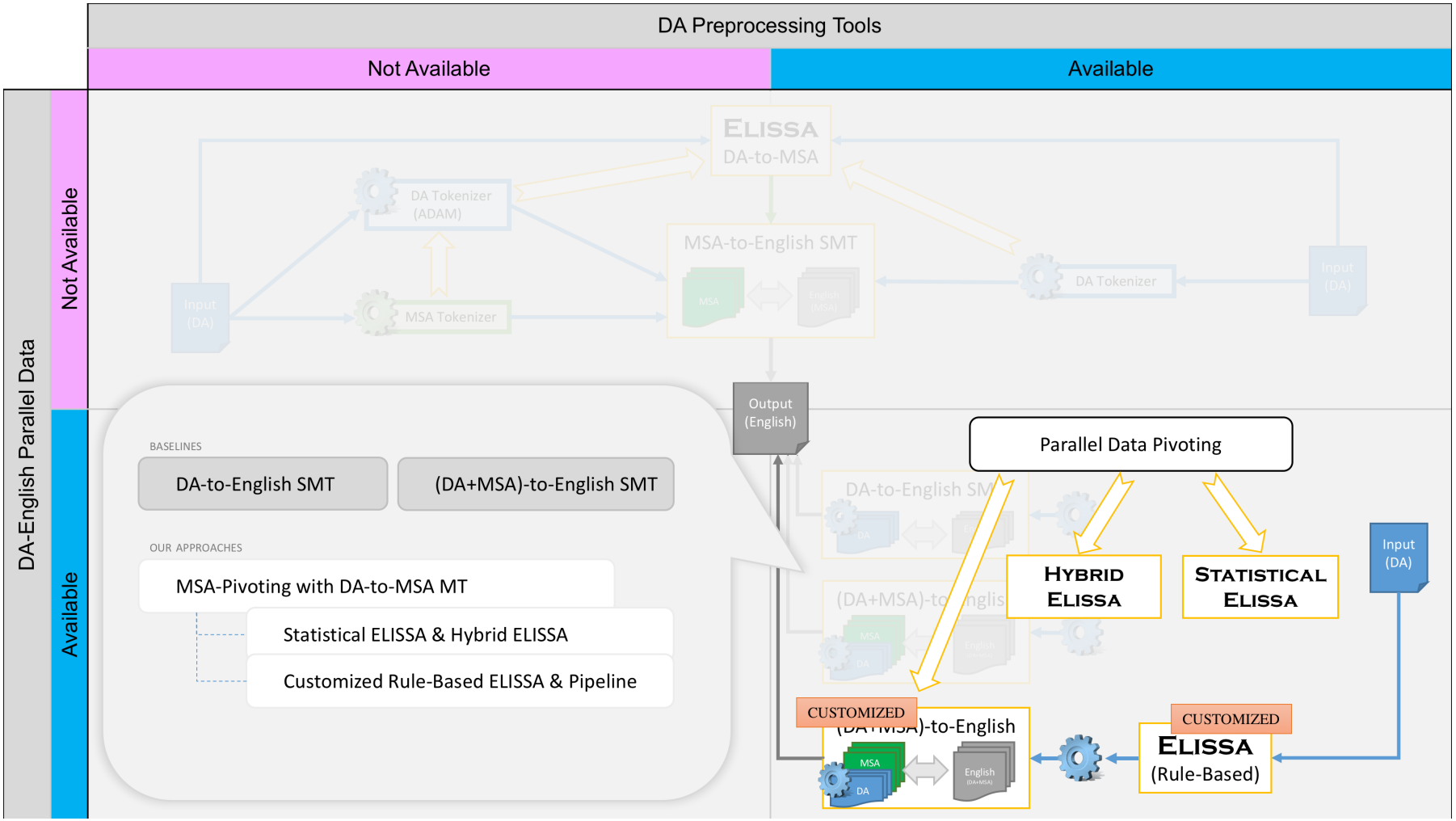



Figure 1.4: Our contributions: A view of our baselines and approaches for the parallel data pivoting of quadrant 3. The columns represent the unavailability or availability of DA preprocessing tools while the rows represent the unavailability or availability of DA-English parallel data. The cells present our contributions in each setting.

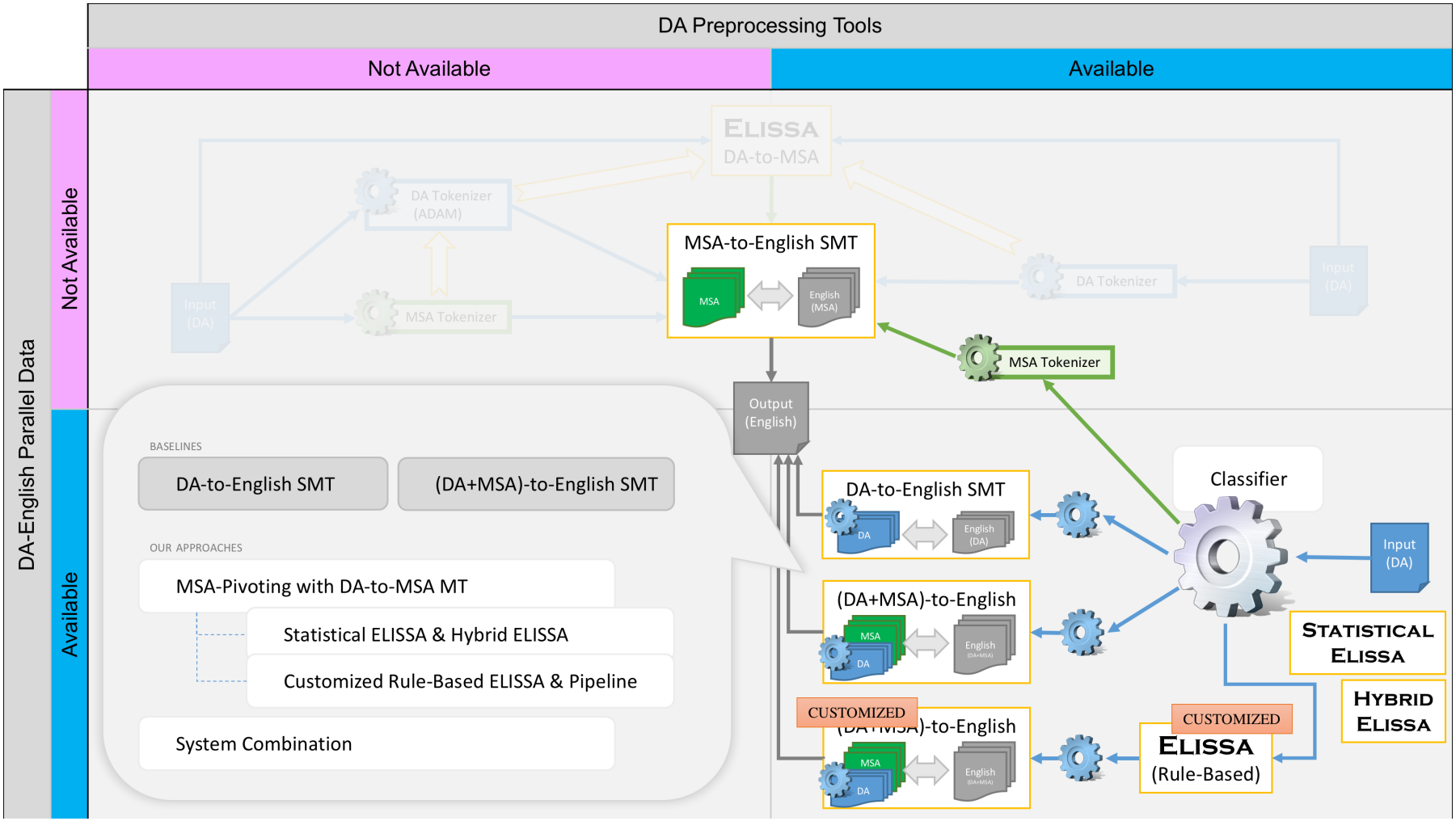



Figure 1.5: Our contributions: A view of our baselines and approaches for the quadrant 3 depicting our system combination approach. The columns represent the unavailability or availability of DA preprocessing tools while the rows represent the unavailability or availability of DA-English parallel data. The cells present our contributions in each setting.

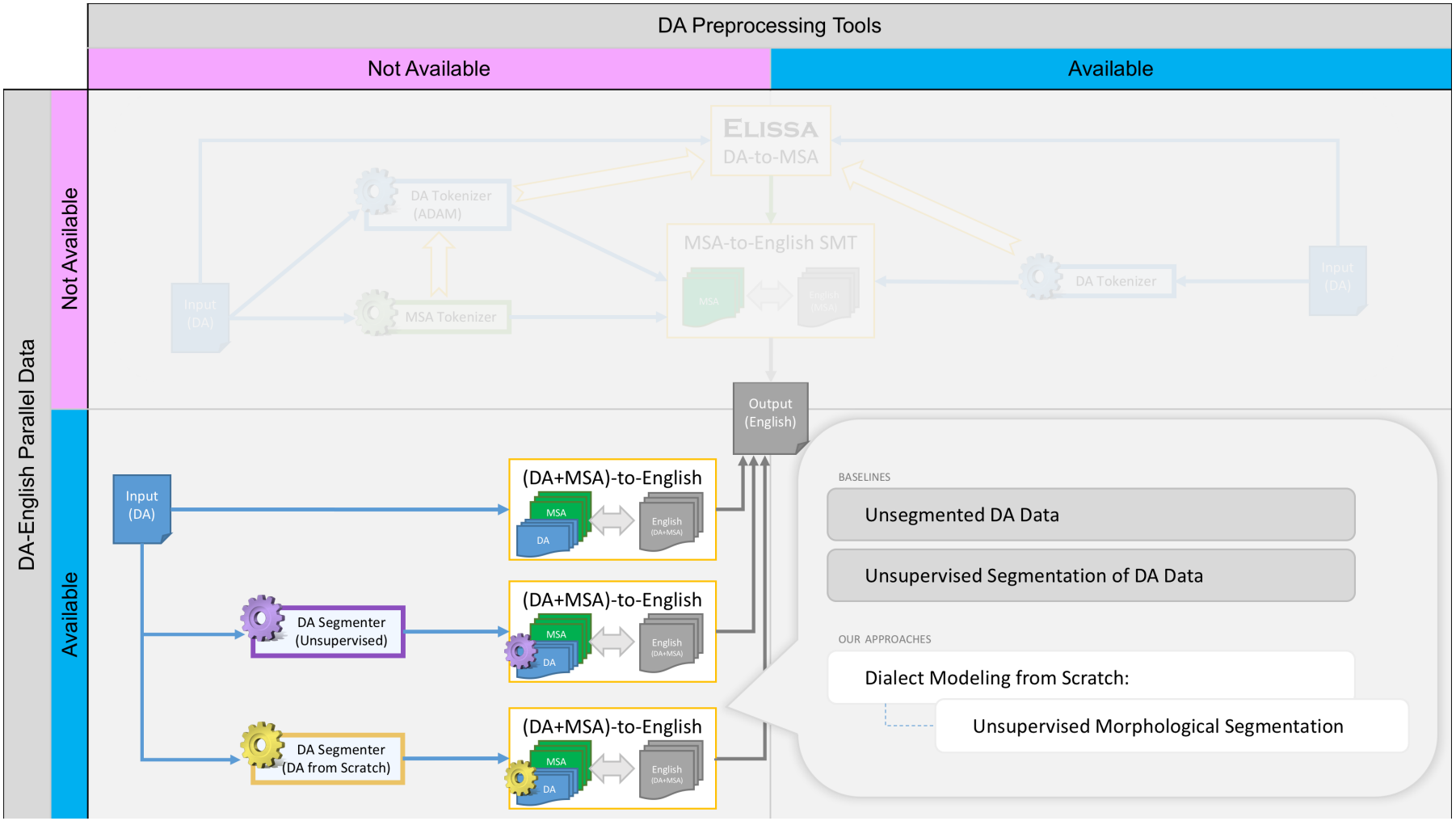



Figure 1.6: Our contributions: A view of our baselines and approaches for quadrant 4 where DA-English data is available but DA tools are not. The columns represent the unavailability or availability of DA preprocessing tools while the rows represent the unavailability or availability of DA-English parallel data. The cells present our contributions in each setting.

In the third and final part, we present an approach to scale to more dialects. This part concerns with **dialects with some parallel data and no processing tools**. In fact, our approach choses to ignore any existing dialect-specific preprocessing tools (including MSA ones) and tries, instead, to learn unified tools for all dialects. To do so, it relies heavily on an abundant resource: monolingual text, in addition to any available DA-English corpora. Figure 1.6 displays our baselines and approaches to Part-III in the fourth quadrant. We present a morphological segmentation system as an example of our approach. A system like this provides a huge boost to MT since it dramatically reduces the size of the vocabulary. Additionally, it maps OOV words to in-vocabulary (INV) words.

- **Learning morphological segmentation options from monolingual data.** A morphological segmentation system needs a tool that provides a list of segmentation options for an input word. We present an unsupervised learning approach to build such a tool from word embeddings learned from monolingual data. This tool provides morphological segmentation options weighted, out-of-context, using expectation maximization.

- **Morphological segmentation for MT purposes.** We use the tool above to label select words in the DA side of the DA-English parallel data with a segmentation option that best aligns to the translation on the English side. We train context-sensitive and context-insensitive supervised segmentation systems on this automatically labeled data. Usually, after training a supervised tokenization system on a human labeled treebank, researchers experiment with different tokenization schemes to find out which one performs better on MT. Our decision in using token alignments to the English side as a factor in deciding on the best segmentation choice while automatically labeling the data biases our system toward generating tokens that better align and translate to English words. This allows our segmenter to approach a better segmentation scheme tailored to the target language.

The contributions of this dissertation can be extended to a wider applications outside of Arabic Dialect NLP and machine translation. Some of the insights can be used for NLP or MT of other languages with dialects and diglossia. Moreover, some of the techniques presented can be used for different genres in the same language. Furthermore, some of our approaches for handling low resource languages can be extended to handle other low resource or morphologically-complex languages.

### 1.3.2 Released Tools

During the work on this thesis, we developed and release the following resources:

1. ADAM. An Analyzer of Dialectal Arabic Morphology. Available from Columbia University.

2. ELISSA. A Dialectal to Standard Arabic Machine Translation System. Available from the Linguistic Data Consortium.

3. A Modern Standard Arabic Closed-Class Word List. Available from Columbia University.

I also participated in the following resources:

1. Tharwa. A Large Scale Dialectal Arabic - Standard Arabic - English Lexicon. Available from Columbia University.

2. SPLIT. Smart Preprocessing (Quasi) Language Independent Tool. Available from Columbia University.

### 1.3.3 Note on Data Sets

Since most of this work was supported by two DARPA programs, GALE and BOLT, different data sets and parallel corpora became available at different points of those programs. As

a result, the systems evaluated in this thesis have been trained and/or evaluated on slightly different data sets. Rerunning all experiments with a unified data set is time consuming and should not change the conclusions of this work.

## 1.4 Thesis Outline

This thesis is structured as follows. In Chapter 2 we discuss the literature related to the our work. The main body of this thesis is divided into three parts as discussed in Section 1.3.

Part-I includes Chapter 3, where we present ADAM, our dialectal morphological analyzer, and Chapter 4, where we describe ELISSA and evaluate the MSA-pivoting appraoch. Part-II consists of two chapters: 5 and 6. In Chapter 5 we present techniques to generate synthetic parallel data and we discuss Statistical ELISSA and Hybrid ELISSA. We also evaluate different MSA-pivoting approaches after adding the new data and tools. In Chapter 6 we evaluate competing approaches to DA-to-English MT and we present two system combination approaches to unite them.

Part-III includes Chapter 7 which discusses our scalable approach to dialect modeling from limited DA-English parallel data and presents a morphological segmenter. We conclude and present future work directions in Chapter 8.

# Chapter 2

# Related Work

In this chapter, we discuss the literature on natural language processing (NLP) and machine translation of Arabic dialects and related topics.

## 2.1 Introduction to Machine Translation

A Machine Translation system (Hutchins, 1986; Koehn, 2009) takes content in one language as input and automatically produces a translation of that content in another language. Researchers have experimented with different types of content, like text and audio, and different levels of content granularity, like sentences, paragraphs, and documents. In this work we are only interested in text-based sentence-level MT.

### 2.1.1 Rule-Based versus Statistical Machine Translation

A Rule-Based (or, Knowledge-Based) Machine Translation (RBMT) system (Nirenburg, 1989) utilizes linguistic knowledge about the source and target languages in the form of rules that are executed to transform input content into its output equivalents. Statistical Machine Translation (SMT) (Brown et al., 1990), on the other hand, builds statistical models from a collection of data (usually in the form of sentence-aligned parallel corpus). It

later uses those models to translate source language content to target language. Phrase-based models to statistical machine translation (PBSMT) (Zens et al., 2002; Koehn et al., 2003; Och and Ney, 2004) give the state-of-the-art performance for most languages.

While RBMT is thought of as a traditional approach to MT, it is still effective for languages with limited parallel data, also known as, low-resource language pairs. Additionally, new hybrid approaches have emerged to combine RBMT and SMT in situations where both fail to achieve a satisfactory level of accuracy and fluency.

### 2.1.2 Neural Machine Translation (NMT)

The deep learning wave has recently reached the machine translation field. Many interesting network architectures have been proposed and have outperformed Phrase-Based SMT with large margins on language pairs like French-English and German-English. The caveat is that these models require enormous amounts of parallel data; e.g., in the case of French-English, they're trained on hundreds of millions of words. Creating that amount of professional translations for a language pair costs hundreds of millions of dollars. As of the time of writing this thesis, training NMT models on relatively small amounts of parallel data results in hallucinations.

Almahairi et al. (2016) presents one of the early research on NMT for the Arabic language. They compare Arabic-to-English and English-to-Arabic NMT models to their phrase-based SMT equivalents with different preprocessing techniques for the Arabic side such as normalization and morphological tokenization. It is also important to note that the focus of this work is on Modern Standard Arabic (MSA) and not on dialectal Arabic. Their systems were trained on 33 million tokens on the MSA side, which is considered relatively large in terms of MT training data. Their results show that phrase-based SMT models outperform NMT models in both directions on in-domain test sets. However, NMT models outperform PBSMT on an out-of-domain test set in the English-to-Arabic direction.

Their research also shows that neural MT models significantly benefit from morphological tokenization.

Our published research we present in this thesis was published before the advances in NMT. We have not used NMT in this work, mainly because our training data is relatively small and because our approaches do not depend necessary on PBSMT; instead, they use PBSMT to extrinsically evaluate the effect of the preprocessing tools we create on machine translation quality. In this work, whenever we mention SMT we mean PBSMT.

## 2.2 Introduction to Arabic and its Challenges for NLP

Arabic is a Central Semitic language that goes back to the Iron Age and is now by far the most widely spoken Afro-Asiatic language. Contemporary Arabic is a collection of varieties that are spoken by as many as 422 million speakers, of which 290 million are native speakers, making Arabic the fifth most spoken language in the world in both native speakers and total speakers rankings.

### 2.2.1 A History Review of Arabic and its Dialects

**Standard Arabic varieties.** Scholars distinguish between two standard varieties of Arabic: Classical Arabic and Modern Standard Arabic. Classical Arabic is the language used in the Quran. Its orthography underwent fundamental changes in the early Islamic era such as adding dots to distinguish letters and adding diacritics to express short vowels. In the early 19th century, Modern Standard Arabic (MSA) was developed from Classical Arabic to become the standardized and literary variety of Arabic and is now one of the six official languages of the United Nations.

**Dialectal Arabic varieties.** Dialectal Arabic, also known as Colloquial Arabic, refers to many regional dialects that evolved from Classical Arabic, sometimes independently from

each other. They are heavily influenced by the indigenous languages that existed before the Arab conquest of those regions and co-existed with Arabic thereafter. For example, Aramaic and Syriac influenced Levantine, Coptic influenced Egyptian, and Berber influenced Moroccan. Furthermore, due to the occupation of most of these regions by foreign countries, the dialects were influenced, to varying degrees, by foreign languages such as Turkish, French, English, Italian, and Spanish. These factors led to huge divisions between Arabic dialects to a degree where some varieties such as the Maghrebi dialects, for example, are unintelligible to a speaker of a Levantine dialect. Figure 2.1 shows different Arabic dialects and their geographical regions in the Arab world. Table 2.1 shows regional grouping of major Arabic dialects, although it is important to note that these major dialects may have sub-dialects.

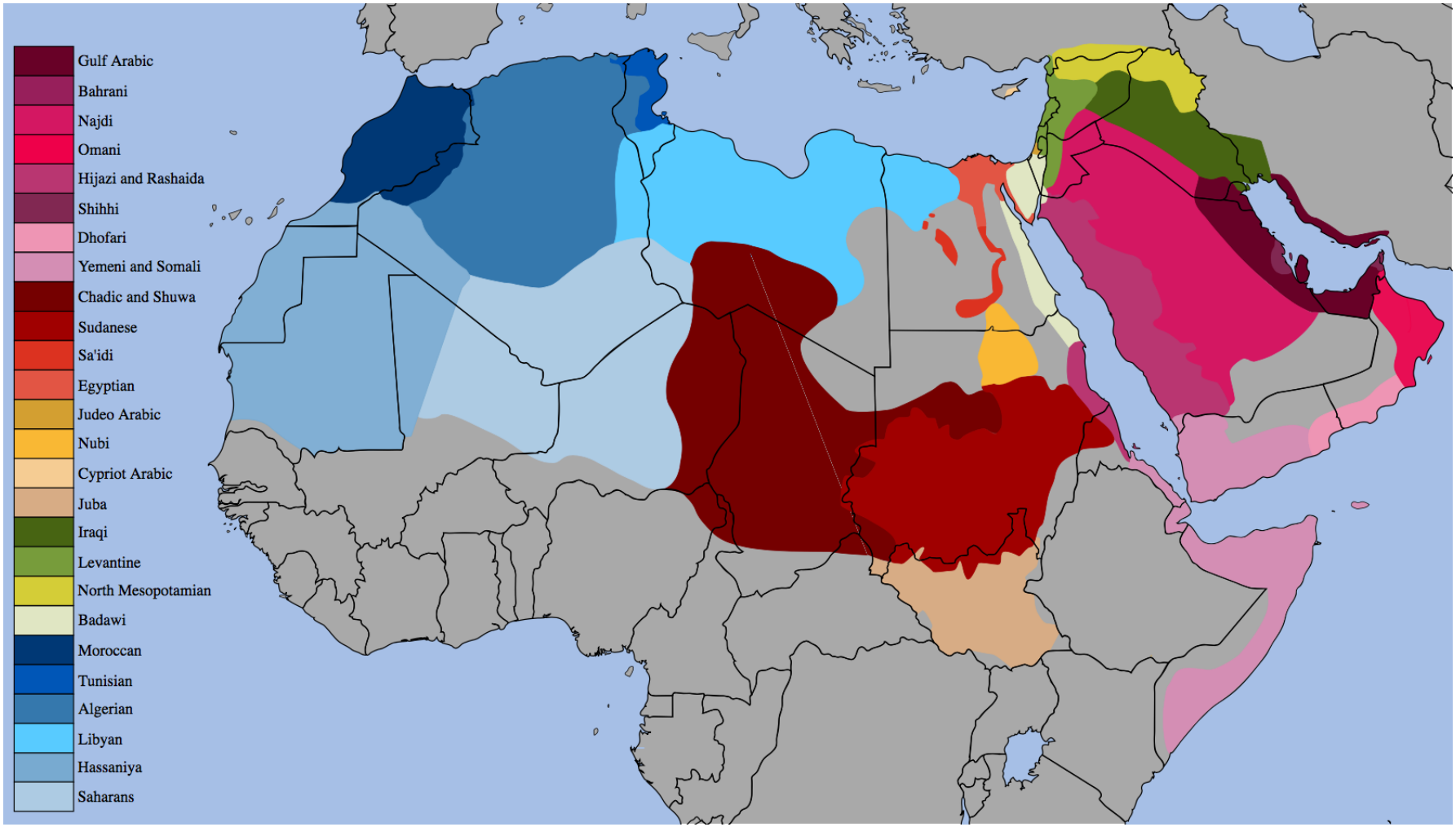



Figure 2.1: Arabic Dialects and their geographic areas in the Arab world. Source: Wikipedia.

### 2.2.2 Arabic as a Prototypical Diglossic Language

The sociolinguistic situation of Arabic provides a prime example of diglossia where these two distinct varieties of Arabic co-exist and are used in different social contexts. This diglossic situation facilitates code-switching in which a speaker switches back and forth between the two varieties, sometimes even within the same sentence. Despite not being the native language for any group of people, Modern Standard Arabic is widely taught at schools and universities and is used in most formal speech and in the writing of most books, newsletters, governmental documents, and other printed material. MSA is sometimes used in broadcasting, especially in the news genre. Unlike MSA, Arabic dialects are spoken as a first language and are used for nearly all everyday speaking situations, yet they do not have standard orthography.

### 2.2.3 Modern Standard Arabic Challenges

The Arabic language is quite challenging for natural language processing tasks. Arabic is a morphologically complex language which includes rich inflectional morphology, expressed both templatically and affixationally, and several classes of attachable clitics. For example, the Modern Standard Arabic (MSA) word وسيكتبونها *w+s+y-ktb-wn+hA*[1] 'and they will write it' has two proclitics (+و *w+* 'and' and +س *s+* 'will'), one dprefix -ي *y-* '3rd person, imperfective', one suffix -ون *-wn* 'masculine plural' and one pronominal enclitic +ها *+hA* 'it/her'. Additionally, Arabic is written with optional diacritics that specify short vowels, consonantal doubling and the nunation morpheme[2]. The absence of these diacritics together with the language's rich morphology lead to a high degree of ambiguity:

[1] Arabic transliteration is in the Habash-Soudi-Buckwalter scheme (Habash et al., 2007).

[2] In Arabic, and some other semitic languages, the nunation morpheme is one of three vowel diacritics that attaches to the end of a noun or adjective to indicate that the word ends in an alveolar nasal without the need to add the letter Nūn (ن 'n').

| **Regional Group** | **Dialect** |
|---|---|
| Levantine Region | Central Levantine Arabic (Central Syrian, Lebanese)<br>North Syrian Arabic (e.g., Aleppo and Tripoli dialects)<br>Cypriot Maronite Arabic<br>South Levantine Arabic (Jordanian, Palestinian)<br>Druz Arabic<br>Alawite Arabic<br>Levantine Bedawi Arabic |
| Egyptian Region | Egyptian Arabic (Cairo, Alexandria, and Port Said varieties)<br>Sa'idi Arabic |
| Sudanese Region | Sudanese Arabic<br>Chadian Arabic<br>Juba Arabic<br>Nubi Arabic |
| Mesopotamian Region | South (Gelet) Mesopotamian Arabic (Baghdadi Arabic, Euphrates (Furati) Arabic)<br>North (Qeltu) Mesopotamian Arabic (Mosul, Judeo-Iraqi) |
| Arabian Peninsula Region | Gulf Arabic (Omani, Dhofari, Shihhi, Kuwaiti)<br>Yemeni Arabic (Sanaani, Hadhrami, Tihamiyya, Ta'izzi-Adeni, Judeo-Yemeni)<br>Hejazi Arabic<br>Najdi Arabic<br>Bareqi Arabic<br>Baharna Arabic<br>Northwest Arabian Arabic (Eastern Egyptian Bedawi, South Levantine Bedawi, and North Levantine Bedawi) |
| Maghrebi Region | Moroccan Arabic<br>Tunisian Arabic<br>Algerian Arabic<br>Libyan Arabic<br>Hassaniya Arabic<br>Algerian Saharan Arabic<br>Maghrebi varieties of Judeo-Arabic (Judeo-Tripolitanian, Judeo-Moroccan, Judeo-Tunisian)<br>Western Egyptian Bedawi Arabic |
| Central Asian | Khorasani Arabic<br>Tajiki Arabic<br>Uzbeki Arabic |
| Western | Andalusian<br>Siculo-Arabic (Maltese, Sicilian) |

Table 2.1: Regional groups of Arabic and their dialects.

e.g., the Buckwalter Arabic Morphological Analyzer (BAMA) produces an average of 12 analyses per word. Moreover, some Arabic letters are often spelled inconsistently which leads to an increase in both sparsity (multiple forms of the same word) and ambiguity (same form corresponding to multiple words), e.g., variants of Hamzated Alif, أ> or إ<, are often written without their Hamza (ء '): ا *A*; and the Alif-Maqsura (or dotless Ya) ى Y and the regular dotted Ya ي *y* are often used interchangeably in word final position (El Kholy and Habash, 2010). Arabic complex morphology and ambiguity are handled using tools for analysis, disambiguation and tokenization (Habash and Rambow, 2005; Diab et al., 2007).

### 2.2.4 Dialectal Arabic Challenges

Contemporary Arabic is a collection of varieties: MSA, which has a standard orthography and is used in formal settings, and DAs, which are commonly used informally and with increasing presence on the web, but which do not have standard orthographies. There are several DA varieties which vary primarily geographically, e.g., Levantine Arabic, Egyptian Arabic, etc (Habash, 2010).

DAs differ from MSA phonologically, morphologically and to some lesser degree syntactically. The differences between MSA and DAs have often been compared to Latin and the Romance languages (Habash, 2006). The morphological differences are most noticeably expressed in the use of clitics and affixes that do not exist in MSA. For instance, the Levantine and Egyptian Arabic equivalent of the MSA example above is وحيكتبوها *w+H+y-ktb-w+hA* 'and they will write it'[3]. The optionality of vocalic diacritics helps hide some of the differences resulting from vowel changes; compare the diacritized forms: Levantine *wHayuktubuwhA*, Egyptian *waHayiktibuwhA* and MSA *wasayaktubuwnahA* (Salloum and Habash, 2011). It is important to note that Levantine and Egyptian differ a lot in phonology, but the orthographical choice of dropping short vowels (expressed as diacritics

[3]Another spelling variation for Egyptian Arabic is to spell the word as وهيكتبوها *w+h+y-ktb-w+hA*.

in Arabic script) bridges the phonological gab between them. However, when writing Arabic in Latin script, known as Arabizi, which is an orthographical choice picked by many people mainly in social media discussions, chat and SMS genres, phonology is expressed by Latin vowels, which brings back the gap between dialects and sub-dialects.

All of the NLP challenges of MSA described above are shared by DA. However, the lack of standard orthographies for the dialects and their numerous varieties causes *spontaneous orthography*, which poses new challenges to NLP (Habash et al., 2012b). Additionally, DAs are rather impoverished in terms of available tools and resources compared to MSA; e.g., there is very little parallel DA-English corpora and almost no MSA-DA parallel corpora. The number and sophistication of morphological analysis and disambiguation tools in DA are very limited in comparison to MSA (Duh and Kirchhoff, 2005; Habash and Rambow, 2006; Abo Bakr et al., 2008; Habash et al., 2012a). MSA tools cannot be effectively used to handle DA: (Habash and Rambow, 2006) report that less than two-thirds of Levantine verbs can be analyzed using an MSA morphological analyzer; and (Habash et al., 2012a) report that 64% of Egyptian Arabic words are analyzable using an MSA analyzer.

### 2.2.5 Dialectness, Domain, Genre, and Timespan

In addition to the previous challenges, other aspects contribute to the challenges of Arabic NLP in general and MT in particular like the level of sentence dialectness, and the sentence domain and genre. Habash et al. (2008) define five levels of sentence dialectness: 1) perfect MSA, 2) imperfect MSA, 3) Arabic with full dialect switching, 4) dialect with MSA incursions, and 5) pure dialect. These five levels create confusions to MT systems and increase errors in preprocessing tools like tokenizers. They also raise the question of whether or how to use the huge collection of MSA-English parallel corpora in training a DA-English SMT. If added on top of the limited DA-English data it could hurt the translation quality of

some sentences while helping others based on their level of dialectness.

Similarly, the domain and the genre of the sentence will increase challenges for MT. The task of translating content of news, news wire, weblog, chat, SMS, emails, and speech transcripts will require more DA-English training data of the already limited parallel corpora. Add on top of that the timespan of the training data versus the dev/test sets. For example, consider a test set that uses recent terminology related to Arab Spring events, politicians, places, and recent phrases and terminology that are never mentioned in the older training data.

### 2.2.6 Overview and Challenges of Dialect-Foreign Parallel Data

Arabic dialects are in different states in terms of the amount of dialect-foreign parallel data. The Defense Advanced Research Projects Agency (DARPA), as part of its projects concerning with machine translation of Arabic and its dialects to English: Global Autonomous Language Exploitation (GALE) and Broad Operational Language Translation (BOLT), has provided almost all of the DA-English parallel data available at the time of writing this thesis. The Egyptian-English language pair has the largest amount of parallel data $\sim$2.4MW (million words), followed by Levantine-English with $\sim$1.5MW, both provided by DARPA's BOLT. Other dialect-English pairs, like Iraqi-English, have smaller parallel corpora while the majority of dialects and subdialects have no parallel corpora whatsoever.

Modern Standard Arabic (MSA) has a wealth of MSA-English parallel data amounting to hundreds of millions of words. The majority of this data, however, is originated from the United Nations (UN) parallel corpus which is a very narrow genre that could hurt the quality of MT on other genres when combined with other, smaller, domain-specific parallel corpora. We have trained an SMT system on over two hundred million words of parallel corpora that include the UN corpus as part of NIST OpenMT Eval 2012 compe-

tition. When we tested this system in an MSA-pivoting approach to DA-to-English MT, it performed worse than a system trained on a subset of the corpora that excludes the UN corpus. Other sources for MSA-English data come from the news genre in general. While DARPA's GALE program provides about ∼49.5MW of MSA-English data mainly in the news domain, it is important to note that it comes from data collected before the year 2009 since this affects the translation quality of MSA sentences discussing later events such as the Egyptian revolution of 2011.

## 2.3 Dialectal Arabic Natural Language Processing

### 2.3.1 Extending Modern Standard Arabic Resources

Much work has been done in the context of MSA NLP (Habash, 2010). Specifically for Arabic-to-English SMT, the importance of tokenization using morphological analysis has been shown by many researchers (Lee, 2004; Zollmann et al., 2006; Habash and Sadat, 2006). For the majority of Arabic dialects, dialect-specific NLP resources are non-existent or in their early stages. Several researchers have explored the idea of exploiting existing MSA rich resources to build tools for DA NLP, e.g., Chiang et al. (2006) built syntactic parsers for DA trained on MSA treebanks. Such approaches typically expect the presence of tools/resources to relate DA words to their MSA variants or translations. Given that DA and MSA do not have much in terms of parallel corpora, rule-based methods to translate DA-to-MSA or other methods to collect word-pair lists have been explored. For example, Abo Bakr et al. (2008) introduced a hybrid approach to transfer a sentence from Egyptian Arabic into MSA. This hybrid system consisted of a statistical system for tokenizing and tagging, and a rule-based system for constructing diacritized MSA sentences. Moreover, Al-Sabbagh and Girju (2010) described an approach of mining the web to build a DA-to-MSA lexicon. In the context of DA-to-English SMT, Riesa and Yarowsky (2006) presented

a supervised algorithm for online morpheme segmentation on DA that cut the OOVs by half.

### 2.3.2 Dialectal Arabic Morphological Analysis

There has been a lot of work on Arabic morphological analysis with a focus on MSA (Beesley et al., 1989; Kiraz, 2000; Buckwalter, 2004; Al-Sughaiyer and Al-Kharashi, 2004; Attia, 2008; Graff et al., 2009; Altantawy et al., 2011; Attia et al., 2013). By comparison, only a few efforts have targeted DA morphology (Kilany et al., 2002; Habash and Rambow, 2006; Abo Bakr et al., 2008; Salloum and Habash, 2011; Mohamed et al., 2012; Habash et al., 2012a; Hamdi et al., 2013).

Efforts for Modeling dialectal Arabic morphology generally fall in two camps. First are solutions that focus on **extending MSA tools to cover DA phenomena**. For example, (Abo Bakr et al., 2008) and (Salloum and Habash, 2011) extended the BAMA/SAMA databases (Buckwalter, 2004; Graff et al., 2009) to accept DA prefixes and suffixes. Such efforts are interested in mapping DA text to some MSA-like form; as such they do not model DA linguistic phenomena. These solutions are fast and cheap to implement.

The second camp is interested in **modeling DA directly**. However, the attempts at doing so are lacking in coverage in one dimension or another. The earliest effort on Egyptian that we know of is the Egyptian Colloquial Arabic Lexicon (Kilany et al., 2002). This resource was the base for developing the CALIMA Egyptian morphological analyzer (Habash et al., 2012a; Habash et al., 2013). Another effort is the work by (Habash and Rambow, 2006) which focuses on modeling DAs together with MSA using a common multi-tier finite-state-machine framework. Mohamed et al. (2012) annotated a collection of Egyptian for morpheme boundaries and used this data to develop an Egyptian tokenizer. Eskander et al. (2013b) presented a method for automatically learning inflectional classes and associated lemmas from morphologically annotated corpora. Hamdi et al. (2013) takes

advantage of the closeness of MSA and its dialects to build a translation system from Tunisian Arabic verbs to MSA verbs. Eskander et al. (2016a) presents an approach to annotating words with a conventional orthography, a segmentation, a lemma and a set of features. They use these annotations to predict unseen morphological forms, which are used, along with the annotated forms, to create a morphological analyzer for a new dialect.

The second approach to modeling Arabic dialect morphology usually results in better quality morphological analyzers compared to the shallow techniques presented by the first camp. However, they are expensive and need a lot more resources and efforts. Furthermore, they are harder to extend to new dialects since they require annotated training data and/or hand-written rules for each new dialect.

The work we present in Chapter 3 is closer to the first camp. We present detailed evaluations of coverage and recall against two state-of-the art systems: SAMA for MSA and CALIMA for Egyptian Arabic. The work we present in Chapter 7 falls under the second camp in that it tries to model dialects directly from monolingual data and some parallel corpora.

**Morphological Tokenization for Machine Translation.** Reducing the size of the vocabulary by tokenization morphologically complex words proves to be very beneficial for any statistical NLP system in general, and MT in particular. Many researchers have explored ways to come up with a good tokenization scheme for Arabic when translating to English (Maamouri et al., 2004; Sadat and Habash, 2006). While SMT systems typically use one tokenization scheme for the whole Arabic text, Zalmout and Habash (2017) experimented with different tokenization schemes for different words in the same Arabic text. They evaluated their approach on SMT from Arabic to five foreign languages varying in their morphological complexity: English, French, Spanish, Russian and Chinese. Their work showed that these different target languages require different source language tokenization schemes. It also showed that combining different tokenization options while training the

SMT system improves the overall performance, and considering all tokenization options while decoding further enhances the performance. Our work in Chapter 7 is similar to Zalmout and Habash (2017)'s work in that the segmentation of a word is influenced by the target language (in our case English) and this can change if the target language changes. We differ from that work in that we do not use tokenization schemes or combine them; instead, we learn to segment words, and that segmentation is dependent on the word itself and on the context of that word.

### 2.3.3 Dialect Identification

For token level dialect identification, Biadsy et al. (2009) present a system that identifies dialectal words in speech and their dialect of origin through the acoustic signals. In Elfardy et al. (2013) the authors perform token-level dialect identification by casting the problem as a code-switching problem and treating MSA and Egyptian Dialectal Arabic as two different languages. For sentence level dialect identification, in Elfardy and Diab (2013), the same authors, use features from their token-level system to train a classifier that performs sentence-level Dialectal Arabic Identification. Zaidan and Callison-Burch (2011) crawl a large dataset of MSA-DA news' commentaries. The authors annotate part of the dataset for sentence-level dialectness on Amazon Mechanical Turk and employ a language modeling (LM) approach to solve the problem. (Akbacak et al., 2011) used dialect-specific and cross-dialectal phonotactic models that use Support Vector Machines and Language Models to classify four Arabic dialects: Levantine, Iraqi, Gulf and Egyptian.

## 2.4 Machine Translation of Dialects

Dialects present many challenges to MT due to their spontaneous, unstandardized nature and the scarcity of their resources. In this section we discuss different approaches to handle

dialects.

### 2.4.1 Machine Translation for Closely Related Languages.

Using closely related languages has been shown to improve MT quality when resources are limited. Hajič et al. (2000) argued that for very close languages, e.g., Czech and Slovak, it is possible to obtain a better translation quality by using simple methods such as morphological disambiguation, transfer-based MT and word-for-word MT. Zhang (1998) introduced a Cantonese-Mandarin MT that uses transformational grammar rules. In the context of Arabic dialect translation, Sawaf (2010) built a hybrid MT system that uses both statistical and rule-based approaches for DA-to-English MT. In his approach, DA is normalized into MSA using a dialectal morphological analyzer. In this work, we present a rule-based DA-MSA system to improve DA-to-English MT. Our approach used a DA morphological analyzer (ADAM) and a list of hand-written morphosyntactic transfer rules. This use of "resource-rich" related languages is a specific variant of the more general approach of using pivot/bridge languages (Utiyama and Isahara, 2007; Kumar et al., 2007). In the case of MSA and DA variants, it is plausible to consider the MSA variants of a DA phrase as monolingual paraphrases (Callison-Burch et al., 2006; Du et al., 2010). Also related is the work by Nakov and Ng (2011), who use morphological knowledge to generate paraphrases for a morphologically rich language, Malay, to extend the phrase table in a Malay-to-English SMT system.

### 2.4.2 DA-to-English Machine Translation

Two approaches have emerged to alleviate the problem of DA-English parallel data scarcity: using MSA as a bridge language (Sawaf, 2010; Salloum and Habash, 2011; Salloum and Habash, 2013; Sajjad et al., 2013), and using crowd sourcing to acquire parallel data (Zbib et al., 2012).

**Pivoting Approaches.**

Sawaf (2010) built a hybrid MT system that uses both statistical and rule-based approaches to translate both DA and MSA to English. In his approach, DA is normalized into MSA using a character-based normalizer, MSA and DA-specific morphological analyzers, and a class-based n-gram language model to classify words into 16 dialects (including MSA). These components produce a lattice annotated with probabilities and morphological features (POS, stem, gender, etc.), which is then n-best decoded with character-based and word-based, DA and MSA language models. The 1-best sentence is then translated to English with the hybrid MT system. He also showed an improvement of up to 1.6% BLEU by processing the SMT training data with his technique.

Sajjad et al. (2013) applied character-level transformation to reduce the gap between DA and MSA. This transformation was applied to Egyptian Arabic to produce EGY data that looks similar to MSA data. They reduced the number of OOV words and spelling variations and improved translation output.

**Cheaply Obtaining DA-English Parallel Data**

Zbib et al. (2012) demonstrated an approach to cheaply obtain DA-English data via Amazon's Mechanical Turk (MTurk). They create a DA-English parallel corpus of 1.5M words and used it along with a 150M MSA-English parallel corpus to create the training corpora of their SMT systems. They create a DA-English parallel corpus of 1.5M words and trained an SMT system on it. They built another SMT system from this corpus augmented with a 150M MSA-English parallel corpus to study the effect of the size of DA data and the noise MSA may cause. They found that the DA-English system outperforms the DA+MSA-English even though the ratio of DA data size to MSA data size is 1:100. They also used MTurk to translate their dialectal test set to MSA in order to compare to the MSA-pivoting approach. They showed that even though pivoting on MSA (produced by

Human translators in an oracle experiment) can reduce OOV rate to 0.98% from 2.27% for direct translation (without pivoting), it improves by 4.91% BLEU while direct translation improves by 6.81% BLEU over their 12.29% BLEU baseline (direct translation using the 150M MSA system). They concluded that simple vocabulary coverage is not sufficient and the domain mismatch is a more important problem.

Our research in Part-I falls under the first category – pivoting on MSA. In Part-II we present a combination of the two approaches.

## 2.5 Machine Translation System Combination

The most popular approach to MT system combination involves building confusion networks from the outputs of different MT systems and decoding them to generate new translations (Rosti et al., 2007; Karakos et al., 2008; He et al., 2008; Xu et al., 2011). Other researchers explored the idea of re-ranking the n-best output of MT systems using different types of syntactic models (Och et al., 2004; Hasan et al., 2006; Ma and McKeown, 2013). While most researchers use target language features in training their re-rankers, others considered source language features (Ma and McKeown, 2013).

Most MT system combination work uses MT systems employing different techniques to train on the same data. However, in the system combination work we present in this thesis (Chapter 6), we use the same MT algorithms for training, tuning, and testing, but we vary the training data, specifically in terms of the degree of source language dialectness. Our approach runs a classifier trained only on source language features to decide which system should translate each sentence in the test set, which means that each sentence goes through one MT system only.

## 2.6 Morphological Segmentation

In this section we present a brief review of the literature on supervised and unsupervised learning approaches to morphological segmentation.

### 2.6.1 Supervised Learning Approaches to Morphological Segmentation

Supervised learning techniques, like MADA, MADA-ARZ and AMIRA (Habash and Rambow, 2005; Habash et al., 2013; Diab et al., 2007; Pasha et al., 2014), have performed well on the task of morphological tokenization for Arabic machine translation. They require hand-crafted morphological analyzers, such as SAMA (Graff et al., 2009), or at least annotated data to train such analyzers, such as CALIMA (Habash et al., 2012c), in addition to treebanks to train tokenizers. This is expensive and time consuming; thus, hard to scale to different dialects.

### 2.6.2 Unsupervised Learning Approaches to Morphological Segmentation

Given the wealth of unlabeled monolingual text freely available on the Internet, many unsupervised learning algorithms (Creutz and Lagus, 2002; Stallard et al., 2012; Narasimhan et al., 2015) took advantage of it and achieved outstanding results, although not to a degree where they outperform supervised methods, at least on DA to the best of our knowledge. Traditional approaches to unsupervised morphological segmentation, such as MORFESSOR (Creutz and Lagus, 2002; Creutz and Lagus, 2007), use orthographic features of word segments (prefix, stem, and suffix). Eskander et al. (2016b) uses Adaptor Grammars for unsupervised learning of language-independent morphological segmentation.

Many researchers worked on integrating semantics in the learning of morphology

(Schone and Jurafsky, 2000; Narasimhan et al., 2015) especially with the advances in neural network based distributional semantics (Narasimhan et al., 2015; Wu et al., 2016). Wu et al. (2016) adopts a data-driven approach to learn a wordpiece model (WPM) which generates a deterministic segmentation for any character sequence. This model breaks words into pieces while inserting a special character that guarantees the unambiguous recovery of the original character sequence. These wordpieces provide a morphological model that is especially helpful in the case of out-of-vocabulary (OOV) or rare words.

In Part-III, we present an unsupervised learning approach to morphological segmentation. This model is driven by Arabic semantic, learned with distributional semantics models from large quantities of Arabic monolingual data, as well as English semantics, learned by pivoting on English words in an automatically-aligned Arabic-English parallel corpus.

This page intentionally left blank.

# Part I

# Translating Dialects with No Dialectal Resources

# Chapter 3

# Analyzer for Dialectal Arabic Morphology (ADAM)

In this chapter, we discuss a quick and cheap way to extend existing dialectal morphological analyzers to create analyzers for dialects that have no resources. We present an intrinsic evaluation of this analyzer and compare it to the base analyzer it extended. We also present an extrinsic evaluation in which an NT pipeline uses this analyzer to tokenize dialectal words to help produce better translations.

## 3.1 Introduction

Arabic dialects, or the local primarily spoken varieties of Arabic, have been receiving increasing attention in the field of natural language processing (NLP). An important challenge for work on these dialects is to create morphological analyzers, or tools that provide for a particular written word all of its possible analyses out of context. While Modern Standard Arabic (MSA) has many such resources (Graff et al., 2009; Smrž, 2007; Habash, 2007), Dialectal Arabic (DA) is quite impoverished (Habash et al., 2012a). Furthermore, MSA and the dialects are quite different morphologically: (Habash et al., 2012a) report

that only 64% of Egyptian Arabic words are analyzable using an MSA analyzer. So, using MSA resources for processing the dialects will have limited value. And, as for any language or dialect, developing good large scale coverage lexicons and analyzers can take a lot of time and effort.

In this chapter, we present ADAM (Analyzer for Dialectal Arabic Morphology). ADAM is a poor man's solution to developing a quick and dirty morphological analyzer for dialectal Arabic. ADAM can be used as is or can function as the first step in bootstrapping analyzers for Arabic dialects. It covers all part-of-speech (POS) tags just like any other morphological analyzer; however, since we use ADAM mainly to process text, we do not model phonological difference between Arabic dialects and we do not evaluate the difference in phonology. In this work, we apply ADAM extensions to MSA clitics to generate proclitics and enclitics for different Arabic dialects. This technique can also be applied to stems to generate dialectal stems; however, we do not do that in this work.

## 3.2 Motivation

ADAM is intended to be used on dialectal Arabic text to improve Machine Translation (MT) performance; thus, we focus on orthography as opposed to phonology. While consonants and long vowels are written in Arabic as actual letters, short vowels are optional diacritics over or under the letters. This leads to people ignoring short vowels in writing since the interpretation of the word can be inferred from the context. Even when people write short vowels, they are inconsistent and the short vowels might end up over or under the wrong letter due to visual difficulties. Research in MT, therefore, tends to drop short vowels completely and since ADAM is built to improve MT performance, we choose to drop short vowels from ADAM.

Morphemes of different Arabic dialects (at least the ones we are addressing in this work: Levantine, Egyptian, and Iraqi) usually share similar morpho-syntactic behavior

such as future particles, progressive particle, verb negation, pronouns, indirect object pronouns, and propositions. Furthermore, many morphemes are shared among these dialects especially when dropping short vowels. Therefore, modeling orthographic morphology of multiple dialects in one system seems reasonable. When querying ADAM, the user has the option to specify the dialect of the query word to exclude other dialects' readings.

In an analysis we did in Salloum and Habash (2011), we found that 26% of out-of-vocabulary (OOV) terms in dialectal corpora have MSA readings or are proper nouns. The rest, 74%, are dialectal words. We classify the dialectal words into two types: words that have MSA-like stems and dialectal affixational morphology (affixes/clitics) and those that have dialectal stem and possibly dialectal morphology. The former set accounts for almost half of all OOVs (49.7%) or almost two thirds of all dialectal OOVs. In this work, we only target dialectal affixational morphology cases as they are the largest class involving dialectal phenomena that do not require extension to stem lexica.

## 3.3 Approach

In this section, we describe our approach for developing ADAM.

### 3.3.1 Databases

ADAM is built on top of SAMA databases (Graff et al., 2009). The SAMA databases contain three tables of Arabic stems, complex prefixes and complex suffixes and three additional tables with constraints on matching them. We define a *complex prefix* as the full sequence of prefixes/proclitics that may appear at the beginning of a word. *Complex suffixes* are defined similarly. MSA, according to the SAMA databases, has 1,208 complex prefixes and 940 complex suffixes, which are made up of 49 simple prefixes and 177 simple suffixes, respectively. The number of combinations in prefixes is a lot bigger than in suffixes, which explains the different proportions of complex affixes to simple affixes.

ADAM follows the same database format of the ALMOR morphological analyzer/generator (Habash, 2007), which is the rule-based component of the MADA system for morphological analysis and disambiguation of Arabic (Habash and Rambow, 2005; Roth et al., 2008). As a result, ADAM outputs analyses as lemma and feature-value pairs including clitics. This makes it easier to replace ALMOR database with ADAM database in any MSA NLP system that uses ALMOR to extended to the dialects processed by ADAM. The model, however, has to be re-trained on dialectal data. For example, MADA can be extended to Levantine by plugging ADAM database in place of ALMOR database and training MADA on Levantine TreeBank.

### 3.3.2 SADA Rules

We extend the SAMA database through a set of rules that add Levantine, Egyptian, and Iraqi dialectal affixes and clitics to the database. We call this *Standard Arabic to Dialectal Arabic* mapping technique SADA.[1] To add a dialectal affix (or clitic), we first look for an existing MSA affix with the same morpho-syntactic behavior, and then write a rule (a regular expression) that captures all instances of this MSA affix (either by itself or within complex affixes) and replace them with the new dialectal affix. In addition to changing the surface form of the MSA affix, we change any feature in the retrieved database entry if needed such as Part-Of-Speech (POS), proclitics and enclitics, along with adding new features if needed such as 'dia' that gives the dialect of this new dialectal affix. Finally, the new updated database entries are added to the database while preserving the original entries to maintain analyzing MSA words.

[1] SADA, صدى, *SadY*, means 'echo' in Arabic.

### Scaling ADAM to more dialects

SADA rules were created by the author of this thesis who is a native speaker of Levantine Arabic with good knowledge about Egyptian and Iraqi. Writing the rules took around 70 hours of work and did not require any computer science knowledge. The task does not require a linguist either, any native speaker with basic understanding of morphology (especially POS) can write these rules. Therefore, using crowdsourcing, ADAM can be extended easily and cheaply to other dialects or sub-dialects compared to other approaches (such as MAGEAD and CALIMA) that may take months if not years to cover a new dialect. Moreover, since SADA rules can be applied to any ALMOR-like database, both MAGEAD and CALIMA can be extended by SADA to create a version of ADAM superior to these analyzers. We extend CALIMA with SADA and evaluate it in Section 3.4.

### Analysis of dialectal data

To come up with the list of rules, we started with a list of highly-frequent dialectal words we acquired from Raytheon BBN Technologies in 2010. The process of creating the word list started by extracting all the words that are in annotated non-MSA regions in the GALE transcribed audio data (about 2000 hours) and intersecting them with words in the GALE web data (Webtext). Normally, many of these words are MSA and had to be excluded automatically and manually to ended up with a list of 22,965 types (821,700 tokens) that are, for the most part, dialectal words. Each dialectal word occurred with different frequencies in the two corpora above. The maximum of the two frequencies was picked as the word frequency and the list was ordered according to this frequency. We annotated the top 1000 words in this list for dialect and POS to study the dialectal phenomena we are dealing with. We analyzed the morphology of these words to identify the frequent types of morphemes and their spelling variations along with the common morphemes and shared morpho-syntactic behavior among dialects. This analysis led the creation of the

first version of SADA rules. New rules were added later after getting more dialectal text to analyze.

**Classes of extensions**

We classify our extensions of SADA into two classes: dialectal affixes with comparable MSA equivalents and dialectal affixes that have no MSA equivalent. We discuss these classes by presenting two examples, one for each class.

For the first type, we consider the dialectal future prefix +ح *H*+ 'will' (and its orthographical variations: the Levantine +رح *rH*+ and the Egyptian +ه *h*+. This prefix has a similar behavior to the standard Arabic future particle +س *s*+. As such, an extension rule would create a copy of each occurrence of the MSA prefix and replace it with the dialectal prefix. SADA uses this rule to extend the SAMA database and adds the prefix `Ha/FUT_PART` and many other combinations involving it, e.g., `wa/PART+Ha/FUT_PART+ya/IV3MS`, `Ha/FUT_PART+na/IV1P`, etc.

For the second type, we consider the Levantine dialect demonstrative prefix +ه *h*+ 'this/these' that attaches to nouns on top of the determiner particle +ال *Al*+ 'the'. Since this particle has no equivalent in MSA, we have a rule that extends the determiner particle +ال *Al*+ 'the' to allow the new particle to attach to it. This is equivalent to having a new particle +هال *hAl*+ 'this/these the' that appears wherever the determiner particle is allowed to appear.

The rules (1,021 in total) introduce 16 new dialectal prefixes (plus spelling variants and combinations) and 235 dialectal suffixes (again, plus spelling variants and combinations). Table 3.1 presents a sample of the new proclitics/enclitics added by SADA.

As an example of ADAM output, consider the second set of rows in Figure 3.1, where a single analysis is shown.

| Prefix | Dialect | POS | Comments |
|---|---|---|---|
| b | L,E | PROG_PART | Simple present |
| mn | L | PROG_PART | Simple present (with n/IV1P) |
| d | I | PROG_PART | Simple present |
| Em, Eb | L | PROG_PART | Continuous tense |
| H | M | FUT_PART | Future particle |
| h | E | FUT_PART | Future particle |
| rH | L | FUT_PART | Future particle |
| mA, m | M | NEG_PART | Negation |
| t | L | JUS_PART | 'in order to' |
| hAl | L,I | DEM_DET_PART | 'this/these' the |
| E | L,I | PREP_PART | 'on/to/about' |
| EAl, El | M | PREP_DET_PART | 'on/to/about the' |
| yA | M | VOC_PART | Vocative particle |
| **Suffix** | **Dialect** | **POS** | **Comments** |
| l+[$pron_{PGN}$] | Dia. | PREP+VSUFF_IO:[$PGN$] | Indirect object, e.g., lw, lhA, etc. |
| $ | E,L | NEG_PART | Negation suffix |
| $ | I | PRON_2MS | Suffixing pronoun |
| j | I | PRON_2FS | Suffixing pronoun |
| ky | L | PRON_2FS | When preceded by a long vowel |
| yk | L | PRON_2FS | When preceded by a short vowel |
| ww | L | VSUFF_SUBJ:3P+VSUFF_DO:3MS | Suffix: subject is 3P, object is 3MS |

Table 3.1: An example list of dialectal affixes added by SADA. 'L' is for Levantine, 'E' for Egyptian, 'I' for Iraqi, and 'M' for multi-dialect. PNG is for Person-Number-Gender.

| Levantine Word | وماحيكتبلو *wmAHyktblw* | | | | | |
|---|---|---|---|---|---|---|
| English Equiv. | 'And he will not write to him' | | | | | |
| Analysis: | Proclitics | | | [ Lemma & Features ] | Enclitics | |
| Levantine: | w+ | mA+ | H+ | yktb | +l | +w |
| POS: | conj+ | neg+ | fut+ | [katab IV subj:3MS voice:act] | +prep | +pron$_{3MS}$ |
| English: | and+ | not+ | will+ | he writes | +to | +him |

Figure 3.1: An example illustrating the ADAM analysis output for a Levantine Arabic word.

## 3.4 Intrinsic Evaluation

In this section we evaluate ADAM against two state-of-the-art morphological analyzers: SAMA (v 3.1) (Graff et al., 2009) for MSA and CALIMA (v0.6) (Habash et al., 2012a) for Egyptian Arabic. We apply the SADA extensions to both SAMA and CALIMA to produce two ADAM versions: ADAM$_{sama}$ and ADAM$_{calima}$.

We compare the performance of the four analyzers on two metric: out-of-vocabulary (OOV) rate and in-context part-of-speech recall. We consider data collections from Levan-

tine and Egyptian Arabic. In this work we do not evaluate on Iraqi.

### 3.4.1 Evaluation of Coverage

| **Data Set** | | Levantine | | Egyptian | |
|---|---|---|---|---|---|
| **Word Count** | | *Type* | *Token* | *Type* | *Token* |
| | | 137,257 | 1,132,855 | 315,886 | 2,670,520 |
| **System** | **Metric** | *Type* | *Token* | *Type* | *Token* |
| SAMA | *OOV Rate* | 35.5% | 16.1% | 47.2% | 14.0% |
| ADAM$_{sama}$ | *OOV Rate* | 16.1% | 5.5% | 33.4% | 7.0% |
| CALIMA | *OOV Rate* | 20.4% | 6.9% | 34.4% | 7.2% |
| ADAM$_{calima}$ | *OOV Rate* | **15.6%** | **5.3%** | **32.3%** | **6.6%** |

Table 3.2: Coverage evaluation of the four morphological analyzers on the Levantine and Egyptian side of MT training data in terms of Types and Tokens OOV Rate.

We compare the performance of the four analyzers outlined above in terms of their OOV rate: the percentage of analyzable types or tokens out of all types or tokens, respectively. This metric does not guarantee the correctness of the analyses, just that an analysis is available. For tasks such as undiacritized tokenization, this may actually be sufficient in some cases.

For evaluation, we use the dialectal side of the *DA-English* parallel corpus.[2] This DA side contains ∼3.8M untokenized words of which ∼2.7M tokens (and ∼315K types) are in Egyptian Arabic and ∼1.1M tokens (and ∼137K types) are in Levantine Arabic.

Table 3.2 shows the performance of the four morphological analyzers on both Levantine and Egyptian data in terms of type and token OOV rates. ADAM$_{sama}$ and ADAM$_{calima}$ improve over the base analyzers they extend (SAMA and CALIMA, respectively). For SAMA, ADAM$_{sama}$ reduces the OOV rates by over 50% in types and 66% in tokens for Levantine. The respective values for Egyptian Arabic types and tokens are 29% and 50%. The performance of ADAM$_{sama}$ is quite competitive with CALIMA, a system that took years

[2]This part of the thesis assume that DA-English for the target dialects do not exist. We are using the DA side of this corpus for evaluation only.

and a lot of resources to develop. The OOV rates on Egyptian Arabic for ADAM$_{sama}$ and CALIMA are almost identical; but ADAM$_{sama}$ outperforms CALIMA on Levantine Arabic, which CALIMA was not designed for. Furthermore, ADAM$_{calima}$ improves over CALIMA although by a smaller percentage, suggesting that the ADAM approach can be useful even with well developed dialectal analyzers.

### 3.4.2 Evaluation of In-context Part-of-Speech Recall

We evaluate the four analyzers discussed above in terms of their in-context POS recall (IPOSR). IPOSR is defined as the percentage of time an analyzer produces an analysis with the correct POS in context among the set of analyses for a particular word. To compute IPOSR, we need manually annotated data sets: the Levantine Arabic TreeBank (LATB) (Maamouri et al., 2006) and the Egyptian Arabic (ARZ) TreeBank (Eskander et al., 2013a).[3] We report IPOSR in terms of types and tokens for Levantine and Egyptian on the four analyzers in Table 3.3

| **Data Set** | | **Levantine TB** | | **Egyptian TB** | |
|---|---|---|---|---|---|
| **Word Count** | | *Type** | *Token* | *Type** | *Token* |
| | | 4,201 | 19,925 | 65,064 | 309,386 |
| **System** | **Metric** | *Type** | *Token* | *Type** | *Token* |
| SAMA | *OOV Rate* | 17.1% | 9.8% | 20.3% | 8.4% |
| | *POS Recall* | 68.3% | 64.6% | 60.0% | 75.1% |
| ADAM$_{sama}$ | *OOV Rate* | 2.8% | 1.2% | 7.6% | 2.0% |
| | *POS Recall* | 86.7% | 79.7% | 75.5% | 91.4% |
| CALIMA | *OOV Rate* | 3.8% | 1.7% | 5.6% | 1.6% |
| | *POS Recall* | 86.0% | 80.2% | 85.4% | 94.7% |
| ADAM$_{calima}$ | *OOV Rate* | **2.5%** | **1.0%** | **5.2%** | **1.4%** |
| | *POS Recall* | **87.8%** | **80.7%** | **85.5%** | **94.7%** |

Table 3.3: Correctness evaluation of the four morphological analyzers on the Levantine and Egyptian TreeBanks in terms of Types and Tokens. Type* is the number of unique word-POS pairs in the treebank.

[3]This part of the thesis assume that tools and treebanks for the target dialects do not exist. We are using these DA treebanks for evaluation only.

We observe, first of all, that the OOV rates in the treebank data are much less than OOV rates in the data we used in the previous section on coverage evaluation. The reduction in OOV rate using the dialectal analyzers (beyond SAMA) is also more intense. This may be a result of the treebank data being generally cleaner and less noisy than the general corpus data we used. Next, we observe that SAMA has very low IPOSR rate that are consistent we previous research cited above. ADAM$_{sama}$ improves the overall IPOSR for both Levantine and Egyptian Arabic by about 27% and 23% relative for types and tokens, respectively. ADAM and CALIMA are almost tied in performance in Levantine Arabic; but CALIMA outperforms ADAM for Egyptian Arabic as expected. Finally, ADAM$_{calima}$ improves a bit more on CALIMA for Levantine Arabic, and make less of an impact for Egyptian Arabic. All of this suggests that the ADAM solution is quite competitive with state-of-the-art analyzers given the ease and speed in which it was created. ADAM can make a good bootstrapping method for annotation of dialectal data or for building more linguistically precise dialectal resources.

We should point out that this recall-oriented evaluation ignores possible differences in precision which are likely to result from the fact that the ADAM method tends to produce more analyses per word than the original analyzers it extends. In fact, in the case of Egyptian Arabic, ADAM$_{sama}$ produces 21.8 analyses per word as compared to SAMA's 13.9; and ADAM$_{calima}$ produces 31.4 analyses per word as opposed to CALIMA's 26.3. Without a full, careful and large-scale evaluation of the produced analyses, it is hard to quantify the degree of correctness or plausibility of the ADAM analyses.

## 3.5 Extrinsic Evaluation

In this section we evaluate the use of ADAM$_{sama}$ to tokenize dialectal sentences before translating them with an MSA-to-English SMT system. We do not evaluate ADAM$_{calima}$ since a DA-specific tool like CALIMA is not supposed to exist in this part of the thesis.

This part assumes that only MADA (which uses SAMA internally) is available and can be used to build a baseline MT system.

### 3.5.1 Experimental Setup

We use the open-source Moses toolkit (Koehn et al., 2007) to build a phrase-based SMT system trained on mostly MSA data (64M words on the Arabic side) obtained from several LDC corpora including very limited DA data. Our system uses a standard phrase-based architecture. The parallel corpus is word-aligned using GIZA++ (Och and Ney, 2003a). Phrase translations of up to 10 words are extracted in the Moses phrase table. The language model for our system is trained on the English side of the bitext augmented with English Gigaword (Graff and Cieri, 2003). We use a 5-gram language model with modified Kneser-Ney smoothing. Feature weights are tuned to maximize BLEU on the NIST MTEval 2006 test set using Minimum Error Rate Training (Och, 2003). The English data is tokenized using simple punctuation-based rules. The Arabic side is segmented according to the Arabic Treebank (ATB) tokenization scheme (Maamouri et al., 2004) using the MADA+TOKAN morphological analyzer and tokenizer v3.1 (Habash and Rambow, 2005; Roth et al., 2008). The Arabic text is also Alif/Ya normalized. MADA-produced Arabic lemmas are used for word alignment. Results are presented in terms of BLEU (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005). All evaluation results are case insensitive.

### 3.5.2 The Dev and Test Sets

Our devtest set consists of sentences containing at least one non-MSA segment (as annotated by LDC)[4] in the Dev10 audio development data under the DARPA GALE program. The data contains broadcast conversational (BC) segments (with three reference transla-

[4] http://www.ldc.upenn.edu/

tions), and broadcast news (BN) segments (with only one reference, replicated three times). The data set contained a mix of Arabic dialects: Iraqi, Levantine, Gulf, and Egyptian.

The particular nature of the devtest being transcripts of audio data adds some challenges to MT systems trained on primarily written data in news genre. For instance, each of the source and references in the devtest set contained over 2,600 *uh*-like speech effect words (uh/ah/oh/eh), while the baseline translation system we used only generated 395. This led to severe brevity penalty by the BLEU metric. As such, we removed all of these speech effect words in the source, references and our MT system output.

Another similar issue was the overwhelming presence of commas in the English reference compared to the Arabic source: each reference had about 14,200 commas, while the source had only 64 commas. Our MT system baseline predicted commas in less than half of the reference cases. Similarly we remove commas from the source, references, and MT output. We do this to all the systems we compare in this paper.

We split this devtest set into two sets: a development set (dev) and a blind test set (test), and we call them speech-dev and speech-test, respectively. The splitting is done randomly at the document level. The dev set has 1,496 sentences with 32,047 untokenized Arabic words. The test set has 1,568 sentences with 32,492 untokenized Arabic words.

### 3.5.3 Machine Translation Results

We present the results of our ADAM$_{sama}$-based MT pipeline against a baseline system. The baseline system is an SMT system trained on 64M words of MSA-English data[5]. The baseline system uses MADA to ATB-tokenize the training, tuning, and dev/test sets.

**The ADAM-based MT Approach.** For our system, we use the same SMT system as the baseline where the training and tuning sets were tokenized by MADA. However, we handle

[5]This data is provided by DARPA GALE and is mostly MSA with dialectal inclusions.

the evaluation sets differently. We first process an evaluation set sentence with MADA like in the baseline, then we send out-of-vocabulary (OOV) words to ADAM$_{sama}$ to obtain their analyses. These OOV words have no chance of getting translated otherwise, so they are a safe bet. To produce the tokens in a way consistent with the training data which is ATB-tokenized with MADA that uses an internal system called TOKAN for tokenization of analyses, we create a version of TOKAN, TOKAN$_D$, that can handle dialectal analysis. We send ADAM$_{sama}$ analyses to TOKAN$_D$ which produce the final tokens. Although this may create implausible output for many cases, it is sufficient for some, especially through the system's natural addressing of orthographic variations. Also, splitting dialectal clitics from an MSA (or MSA-like) stem is sometimes all it takes for that stem to have a chance to be translated.

**Results on the development set.** Table 3.4 shows the results of our system, ADAM$_{sama}$ tokenization, against the baseline in terms of BLEU and METEOR. The first column lists the two systems while the second column shows results on our development set: speech-dev. Our system improves over the baseline by 0.41% BLEU and 0.53% METEOR. All results are statistically significant against the baseline as measured using paired bootstrap resampling (Koehn, 2004b).

**Results on the blind test set.** The third column of Table 3.4 present results on our blind test set: speech-test. We achieve consistent results with the development set: 0.36% BLEU and 0.55% METEOR.

## 3.6 Conclusion and Future Work

In this chapter, we presented ADAM, an analyzer of dialectal Arabic morphology, that can be quickly and cheaply created by extending existing morphological analyzers for MSA or

| Test Sets | speech-dev | | | | speech-test | | | |
|---|---|---|---|---|---|---|---|---|
| | BLEU | Diff. | METEOR | Diff. | BLEU | Diff. | METEOR | Diff. |
| **Baseline** | 37.20 | 0.00 | 53.65 | 0.00 | 38.18 | 0.00 | 53.92 | 0.00 |
| **ADAM$_{sama}$ tokenization** | 37.61 | 0.41 | 54.18 | 0.53 | 38.54 | 0.36 | 54.47 | 0.55 |

Table 3.4: Results for the dev set (speech-dev) and the blind test set (speech-test) in terms of BLEU and METEOR. The 'Diff.' column shows result differences from the baseline. The rows of the table are the two MT systems: baseline (where text was tokenized by MADA) and ADAM tokenization (where input was tokenized by ADAM$_{sama}$).

other Arabic varieties. The simplicity of ADAM rules makes it easy to use crowdsourcing to scale ADAM to cover dialects and sub-dialects. We presented our approach to extending MSA clitics and affixes with dialectal ones although the ADAM technique can be used to extend stems as well. We did intrinsic and extrinsic evaluations of ADAM. The intrinsic evaluation showed ADAM performance against an MSA analyzer, SAMA, and a dialectal analyzer, CALIMA, in terms of coverage and in-context POS recall. Finally, we showed how using ADAM to tokenize dialectal OOV words can significantly improve the translation quality of an MSA-to-English SMT system. This means that ADAM can be a cheap option that can be implemented quickly for any Arabic dialect that has no dialectal tools or DA-English parallel data.

In the future, we plan to extend ADAM coverage of the current dialects and extend ADAM to cover new dialects. We expect this to not be too hard since the most dialectal phenomena are shared among Arabic dialects. We also plan to add dialectal stems in two ways:

1. **Copying and modifying MSA stems with SADA-like rules.** The mutations of many dialectal stems from MSA stems follow certain patterns than can be captured with SADA-like rules. For example, for a verb that belongs to a three-letter root with duplicate last letter (e.g., *Hbb* 'to love' and *rdd* 'to reply'), the stem that forms the verb with first person subject (e.g., in MSA, *>aHobabotu* 'I love' and *radadotu* 'I reply') is relaxed with a 'y' in Egyptian and Levantine (e.g, *Hab∼ayt* and *rad∼ayt*).

2. **Importing DA-MSA Lexicons.** DA-MSA dictionaries and lexicons, whether on the surface form level or the lemma level, can be selectively imported to ADAM database.

This page intentionally left blank.

# Chapter 4

# Pivoting with Rule-Based DA-to-MSA Machine Translation System (ELISSA)

## 4.1 Introduction

In Chapter 3 we discussed an approach to translating dialects with no DA-English parallel data and no dialectal preprocessing tools. The approach relies on a DA morphological analyzer, ADAM, that can be built quickly and cheaply, and uses it to tokenize OOV DA words. In this chapter we present another approach that uses this dialectal analyzer, but instead of just tokenizing, it translates DA words and phrases to MSA. Therefore, this approach pivots on MSA to translates dialects to English. For this purpose, we build a DA-to-MSA machine translation system, ELISSA, which executes morpho-syntactic translation rules on ADAM's analyses to generate an MSA lattice that is, then, decoded with a language model. ELISSA can use any DA morphological analyzer which means that this MSA-pivoting approach can be used for dialects with no DA-English data yet have morphological analyzers. In that scenario, building ADAM is not required.

## 4.2 Motivation

| | |
|---|---|
| **DA source** | بهالحالة هاي ما حيكتبولو عحيط الصفحه الشخصية تبعو ولا بدن ياه يبعتلن كومينتات لأنو ماخبرهون ايمتا رح يروح عالبلد.<br>*bhAlHAlp hAy mA Hyktbwlw EHyT AlSfHh Al$xSyp tbEw wlA bdn yAh ybEtln kwmyntAt l>nw mAxbrhwn AymtA rH yrwH EAlbld.* |
| **Human Reference** | In this case, they will not write on his profile wall and they do not want him to send them comments because he did not tell them when he will go to the country. |
| **Google Translate** | |
| Feb. 2013 | Bhalhalh Hi Hictpoulo Ahat Profile Tbau not hull Weah Abatln Comintat Anu Mabarhun Oamta welcomed calls them Aalbuld. |
| Jan. 2018 | In the case of Hae Ma Hiktpulo, the personal page of the personal page, they followed him, and they did not know what to do. |
| **Human**<br>**DA-to-MSA** | في هذه الحالة لن يكتبوا له على حائط صفحته الشخصية ولا يريدونه أن يرسل لهم تعليقات لأنه لم يخبرهم متى سيذهب إلى البلد.<br>*fy h*h AlHAlp ln yktbwA lh ElY HA}T SfHth Al$xSyp wlA yrydwnh >n yrsl lhm tElyqAt l>nh lm yxbrhm mtY sy*hb <lY Albld.* |
| **Google Translate** | |
| Feb. 2013 | In this case it would not write to him on the wall of his own and do not want to send their comments because he did not tell them when going to the country. |
| Jan. 2018 | In this case they will not write him on the wall of his personal page and do not want him to send them comments because he did not tell them when he would go to the country. |

Table 4.1: A motivating example for DA-to-English MT by pivoting (bridging) on MSA. The top half of the table displays a DA sentence, its human reference translation and the output of Google Translate. We present Google Translate output as of 2013 (when our paper that includes this example was published) and as of 2018 where this thesis was written. The bottom half of the table shows the result of human translation into MSA of the DA sentence before sending it to Google Translate.

Table 4.1 shows a motivating example of how pivoting on MSA can dramatically improve the translation quality of a statistical MT system that is trained on mostly MSA-to-English parallel corpora. In this example, we use Google Translate's online Arabic-English SMT system.We present Google Translate's output as of two different dates. The first date is February 21, 2013 when we tested the system for a paper that introduced this example (Salloum and Habash, 2013). The second data is January 2, 2018 before this thesis was submitted. We believe that the 2013 system was a phrase-based SMT system while the 2018 system might have included neural MT models. We constructed this example to showcase different dialectal orthographic and morpho-syntactic phenomena on the word and phrase

levels which we will discuss later in this chapter. As a result, this example can be classified as *pure dialect* on the levels of dialectness spectrum discussed in Chapter 1. This made this example particularly hard for Google Translate, although it was not our intention. We have seen Google Translate translates some dialectal words and fails on others before. So it could have some dialectal inclusions in its data but we do not know the amount or the dialects.

The table is divided into two parts. The top part shows a dialectal (Levantine) sentence, its reference translation to English, and its Google Translate translations. The 2013 Google Translate translation clearly struggles with most of the DA words, which were probably unseen in the training data (i.e., out-of-vocabulary – OOV) and were considered proper nouns (transliterated and capitalized).

The 2018 Google Translate translation is much more imaginative. This could be due to the use of neural MT (NMT) models. Outside of 'case' and 'personal page', the translation has nothing to do with the input sentence. Recent work on neural MT uses character based models which help with spelling errors and variations, and to a lesser degree, morphology; however, they have major drawbacks for dialectal Arabic due to spontaneous orthography and dropping of short vowels which causes most words to be one letter away from another, completely unrelated, word. Similarly, many neural MT models use autoencoders to compress the input sentence into a compact representation (attention mechanisms) which is then decoded with an autodecoder to generate the target sentence. While this network captures the semantics of the input and often generates syntactically sound sentences, it tends to hallucinate, especially with limited amounts of training data. The output that the 2018 Google Translate system provides for the DA source sentence suggests that it is probably using NMT models, especially that when we remove only the period from the end of the DA sentence above, the 2018 Google Translate produces this translation: 'In this case, what is the problem?'.

Although the output of the two versions of Google Translate differs widely, the conclusion stays the same. The lack of DA-English parallel corpora suggests pivoting on MSA can improve the translation quality. In the bottom part of the table, we show a human MSA translation of the DA sentence above and its Google translations. We see that the results are quite promising. The goal of ELISSA is to model this DA-MSA translation automatically. In Section 4.7.1, we revisit this example to discuss ELISSA's performance on it. We show its output and its corresponding Google translation in Table 4.3.

## 4.3 The ELISSA Approach

Since there is virtually no DA-MSA parallel data to train an SMT system, we resort to building a rule-based DA-to-MSA MT system, with some statistical components, we call ELISSA.[1] ELISSA relies on the existence of a DA morphological analyzer, a list of hand-written transfer rules, and DA-MSA dictionaries to create a mapping of DA to MSA words and phrases. The mapping is used to construct an MSA lattice of possible sentences which is then scored with a language model to rank and select the output MSA translations.

**Input and Output.** ELISSA supports untokenized (raw) input only. ELISSA supports three types of output: top-1 choice, an n-best list or a map file that maps source words/phrases to target phrases. The top-1 and n-best lists are determined using an untokenized MSA language model to rank the paths in the MSA translation output lattice. This variety of output types makes it easy to plug ELISSA with other systems and to use it as a DA preprocessing tool for other MSA systems, e.g., MADA (Habash and Rambow, 2005), AMIRA (Diab et al., 2007), or MADAMIRA (Pasha et al., 2014).

[1] In the following chapters, we refer to this version of ELISSA as Rule-Based ELISSA to distinguish it from Statistical ELISSA and Hybrid ELISSA that we build with the help of pivoting techniques that use the DA-English parallel data available in those chapters.

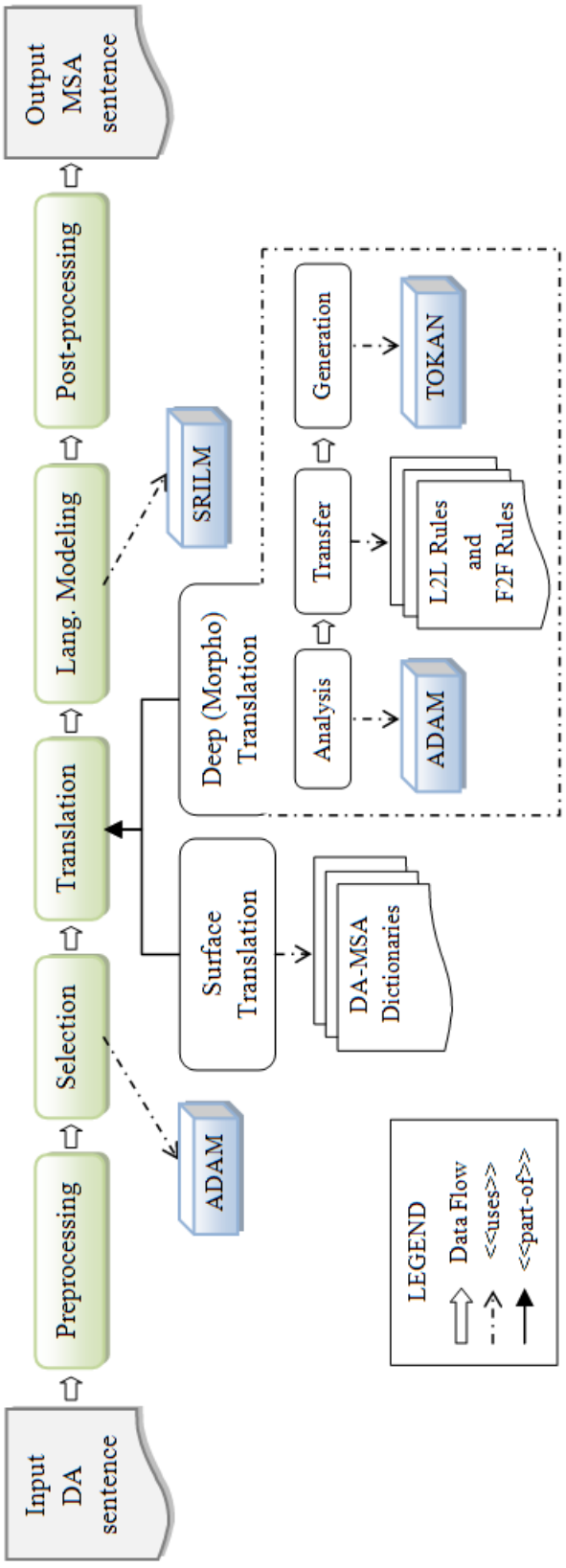



Figure 4.1: This diagram highlights the different steps inside ELISSA and some of its third-party dependencies. ADAM and TOKAN are packaged with ELISSA.

**Components.** ELISSA's approach consists of three major steps preceded by a *preprocessing and normalization* step, that prepares the input text to be handled (e.g., UTF-8 cleaning, Alif/Ya normalization, word-lengthening normalization), and followed by a *post-processing* step, that produces the output in the desired form (e.g., encoding choice). The three major steps are:

1. **Selection.** Identify the words and phrases to handle, e.g., dialectal or OOV words, or phrases with multi-word morpho-syntactic phenomena.

2. **Translation.** Provide MSA paraphrases of the selected words and phrases to form an MSA lattice.

3. **Language Modeling.** Pick the n-best fluent sentences from the generated MSA lattice after scoring with a language model.

In the following sections we will discus these components in details.

## 4.4 Selection

In the first step, ELISSA identifies which words or phrases to paraphrase and which words or phrases to leave as is. ELISSA provides different methods (techniques) for selection, and can be configured to use different subsets of them. In Section 4.8 we use the term "selection mode" to denote a subset of selection methods. Selection methods are classified into *Word-based selection* and *Phrase-based selection*.

### 4.4.1 Word-based selection

Methods of this type fall in the following categories:

a. User token-based selection: The user can mark specific words for selection using the tag '/DIA' (stands for 'dialect') after each word to select. This allows for the use of

dialectal identification systems, such as AIDA (Elfardy and Diab, 2012), to pre-select dialectal words.

b. User type-based selection: The user can specify a list of words to select from, e.g., OOVs. Also the user can provide a list of words and their frequencies and specify a cut-off threshold to prevent selecting a frequent word.

c. Morphology-based word selection: ELISSA uses ADAM (Salloum and Habash, 2011) to select dialectal words. The user can choose between selecting words that have DA analyses only (DIAONLY mode) or words with both DA and MSA analyses (DIAMSA mode).

d. Dictionary-based selection: ELISSA selects words based on their existence in the DA side of our DA-MSA dictionaries. This is similar to *User type-based selection* above, except that we use these dictionaries in the translation component.

e. All: ELISSA selects every word in an input sentence.

### 4.4.2 Phrase-based selection

| Rule Category | Selection Examples | Translation Examples |
|---|---|---|
| *Dialectal Idafa* | الجيش الوطني بتاعنا *Aljy$ AlwTny btAEnA*<br>'the-army the-national ours' | جيشنا الوطني *jy$nA AlwTny*<br>'our-army the-national' |
| *Verb + flipped direct and indirect objects* | حضرلها ياهن *HDrlhA yAhn*<br>'he-prepared-for-her them' | حضرهم لها *HDrhm lhA*<br>'he-prepared-them for-her' |
| *Special dialectal expressions* | بدو اياها *bdw AyAhA*<br>'his-desire her' | يريدها *yrydhA*<br>'he-desires-her' |
| *Negation + verb* | وما حيكتبولو *wmA Hyktbwlw*<br>'and-not they-will-write-to-him' | ولن يكتبوا له *wln yktbwA lh*<br>'and-will-not they-write to-him' |
| *Negation + agent noun* | فمش لاقية *fm$ lAqyp*<br>'so-not finding' | فلا تجد *flA tjd*<br>'so-not she-finds' |
| *Negation + closed-class words* | ما عدكم *mA Edkm*<br>'not with-you' | ليس لديكم *lys ldykm*<br>'not with-you' |

Table 4.2: Examples of some types of phrase-based selection and translation rules.

This selection type uses hand-written rules to identify dialectal multi-word constructions that are mappable to single or multi-word MSA constructions. The current count of these rules is 25. Table 4.2 presents some rule categories (first column) and related examples (second column).

In the current version of ELISSA, Phrase-based selection has precedence over word-based selection methods. We evaluate different settings for the selection step in Section 4.8.

## 4.5 Translation

In this step, ELISSA translates the selected words and phrases to their MSA equivalent paraphrases. The specific type of selection determines the type of the translation, e.g., phrase-based selected words are translated using phrase-based translation rules. The MSA paraphrases are then used to form an MSA lattice.

### 4.5.1 Word-based translation

This category has two types of translation techniques. The *surface translation* uses DA-to-MSA surface-to-surface (S2S) transfer rules (TRs) which depend on DA-MSA dictionaries. The *deep (morphological) translation* uses the classic rule-based machine translation flow: analysis, transfer and generation, which is similar to generic transfer-based MT (Dorr et al., 1999).

**Morphological Analysis.** In this step, we use a dialectal morphological analyzer, ADAM, which provides ELISSA with a set of analyses for each dialectal word in the form of lemma and features. These analyses will be processed in the next step, Transfer. ADAM only handles dialectal affixes and clitics, as opposed to dialectal stems. However, ADAM provides a backoff mode when it tries to guess the dialectal stem from the word and it provides a fake

| Dialect Word | وماحيكتبلو *wmAHyktblw* 'And he will not write for him' | | | | | |
|---|---|---|---|---|---|---|
| Analysis | Proclitics | | | [ Lemma & Features ] | Enclitics | |
| | w+<br>conj+<br>and+ | mA+<br>neg+<br>not+ | H+<br>fut+<br>will+ | yktb<br>[katab IV subj:3MS voice:act]<br>he writes | +l<br>+prep<br>+for | +w<br>+pron$_{3MS}$<br>+him |
| Transfer | Word 1 | | | Word 2 | Word 3 | |
| | Proclitics | [ L & F ] | | [ Lemma & Features ] | [ L & F ] | Enclitic |
| | conj+<br>and+ | [ lan ]<br>will not | | [katab IV subj:3MS voice:act]<br>he writes | [ li ]<br>for | +pron$_{3MS}$<br>+him |
| Generation | w+ | ln | | yktb | l | +h |
| MSA Phrase | ولن يكتب له *wln yktb lh* 'And he will not write for him' | | | | | |

Figure 4.2: An example illustrating the analysis-transfer-generation steps to translate a word with dialectal morphology into its MSA equivalent phrase. This is an extention to the example presented in Figure 3.1 and discussed in Chapter 3. '[ L & F ]' is an abbreviation of '[ Lemma & Features ]'

lemma with the analysis (the lemma is the stem appended with '_0'). This backoff mode is used in the next step to map a dialectal lemma to its MSA lemma translations.

**Morphosyntactic Transfer.** In the transfer step, we map ADAM's dialectal analyses to MSA analyses. This step is implemented using a set of morphosyntactic transfer rules (TRs) that operate on the lemma and feature representation produced by ADAM. These TRs can change clitics, features or lemma, and even split up the dialectal word into multiple MSA word analyses. Crucially the input and output of this step are both in the lemma and feature representation. A particular analysis may trigger more than one rule resulting in multiple paraphrases. This only adds to the fan-out which started with the original dialectal word having multiple analyses.

ELISSA uses two types of TRs: lemma-to-lemma (L2L) TRs and features-to-features (F2F) TRs. L2L TRs simply change the dialectal lemma to an MSA lemma. These rules are extracted from entries in the DA-to-MSA dictionaries where Buckwalter MSA lemmas can be extracted. The dialectal lemma is formatted to match ADAM's guesses (stem appended with '_0'). F2F transfer rules, on the other hand, are more complicated. As examples, two F2F TRs which lead to the transfer output shown in the third set of rows in Figure 4.2

(built on top of Figure 3.1 discussed in Chapter 3) can be described as follows (with the declarative form presented in Figure 4.3):

- If the dialectal analysis shows a negation proclitic and the verb is perfective, remove the negation proclitic from the verb and create a new word, the MSA negative-past particle لَمْ *lm*, to precede the current word and which inherits all proclitics preceding the negation proclitics. Change the verb's tense to imperfective and mood to jussive.

- If the dialectal analysis shows the dialectal indirect object enclitic, remove it from the word and create a new word to follow the current word, and modify the word with an enclitic pronoun that matches the features of the indirect object enclitic. The new word could be one of these two preposition: إلى *<lY* 'to' and +ل *l* 'to', resulting in two options in the final lattice. Alternatively, the rule should add a third option of dropping the preposition and the indirect object.

**Morphological Generation.** In this step, we generate Arabic words from all analyses produced by the previous steps. The generation is done using the general tokenizer/generator TOKAN (Habash, 2007) to produce the surface form words. Although TOKAN can accommodate generation in specific tokenizations, in the work we report here we generate only in untokenized form. Any subsequent tokenization is done in a post-processing step (see Figure 4.1 and the discussion in Section 4.3). The various generated forms are used to construct the map files and word lattices. The lattices are then input to the language modeling step presented next.

### 4.5.2 Phrase-based translation

Unlike the word-based translation techniques which map single DA words to single or multi-word MSA sequences, this technique uses hand-written multi-word transfer rules that

```
F2F-TR:  prc1:\S+_neg asp:p
  # A rule that is triggered wihen a perfective verb has any negation
  # particle. The '\S+' is a regular expression that matches any none
  # white space sequence.
{
   before ( # Insert a word before:
      insert ( lm )
        # Insert 'lm', an MSA word for negation in the past.
   ) # and:
   inside ( # Apply to the main word:
      update ( prc1:0 asp:i mod:j )
        # Clearing prc1 removes the negation particle from the verb.
        # 'asp:i mod:j' makes the verb imperfective and jussive.
   )
}

F2F-TR:  enc0:l(\S+)_prep(\S+)
  # E.g., if enc0:l3ms_prepdobj (the 'l' preposition with a 3rd person
  # masculine singular indirect object), copy the text captured by the
  # two (\S+) to $1 and $2 variables (in the same order).
{
   inside ( # Apply to the main word:
      update( enc0:0 )
        # Clearing enc0 removes the preposition and its pronoun.
   ) # and:
   after ( # Add words after:
      insert( l_prep | <lY | EPSILON )
        # Add these words as alternatives in the lattice.
        # EPSILON means do not add an arc in the lattice.
      update( enc0:$1_pron )
        # This update is applied to the inserted words: l_prep and <lY.
        # enc0:$1_pron uses $1 variable copied from the rule's header,
        # enc0:l(\S+)_prep(\S+); e.g., $1=3ms results in enc0:3ms_pron.
        # This update sets enc0 of both l_prep and <lY to $1_pron; e.g.,
        # enc0:3ms_pron will generate 'lh' and '<lyh', respectively.
   )
}
```

Figure 4.3: An example presenting two feature-to-freature transfer rules (F2F-TR). The rule can have one or more of these three sections: **before**, **inside**, and **after**. Each section can have one or more of these two functions: **insert** (to insert a new word in this section) and **update** (to update the word in this section). The '#' symbol is used for line comments.

| DA Phrase | وما راحولا *wmA rAHwlA* 'And they did not go to her' | | | | |
|---|---|---|---|---|---|
| Analysis | Word 1 | | Word 2 | | |
| | Proclitics | [Lemma & Features] | [Lemma & Features] | [Lemma & Features] | Enclitic |
| | w+<br>conj+<br>and+ | mA<br>[neg]<br>not | rAHw<br>[rAH PV subj:3MP]<br>they go | +l<br>+prep<br>+to | +A<br>+pron$_{3FS}$<br>+her |
| Transfer | Word 1 | | Word 2 | Word 3 | |
| | Proclitics | [Lemma & Features] | [Lemma & Features] | [Lemma & Features] | Enclitic |
| | conj+<br>and+ | [ lam ]<br>did not | [*ahab IV subj:3MP]<br>they go | [ <lY ]<br>to | +pron$_{3FS}$<br>+her |
| Generation | w+ | lm | y*hbwA | <ly | +hA |
| MSA Phrase | ولم يذهبوا إليها *wlm y*hbwA <lyhA* 'And they did not go to her' | | | | |

Figure 4.4: An example illustrating the analysis-transfer-generation steps to translate a dialectal multi-word phrase into its MSA equivalent phrase.

map multi-word DA constructions to single or multi-word MSA constructions. In the current system, there are 47 phrase-based transfer rules. Many of the word-based morphosyntactic transfer rules are re-used for phrase-based translation. Figure 4.4 shows an example of a phrase-based morphological translation of the two-word DA sequence وما راحولا *wmA rAHwlA* 'And they did not go to her'. If these two words were spelled as a single word, وماراحولا *wmArAHwlA*, we would still get the same result using the word-based translation technique only. Table 4.2 shows some rule categories along with selection and translation examples.

## 4.6 Language Modeling

The language model (LM) component uses the SRILM lattice-tool for weight assignment and n-best decoding (Stolcke, 2002). ELISSA comes with a default 5-gram LM file trained on ∼200M untokenized Arabic words of Arabic Gigaword (Parker et al., 2009). Users can specify their own LM file and/or interpolate it with our default LM. This is useful for adapting ELISSA's output to the Arabic side of the training data.

| | |
|---|---|
| **DA source** | (بهالحالة هاي)[1] (ما حيكتبولو)[2] عحيط[3] (الصفحه الشخصية تبعو)[4] ولا (بدن ياه)[5] يبعتلن[6] كومينتات[7] لأنو[8] ماخبرهون[9] امتا[10] (رح يروح)[11] عالبلد[12].<br>*(bhAlHAlp hAy)[1] (mA Hyktbwlw)[2] EHyT[3] (AlSfHh Al$xSyp tbEw)[4] wlA (bdn yAh)[5] ybEtln[6] kwmyntAt[7] l>nw[8] mAxbrhwn[9] AymtA[10] (rH yrwH)[11] EAlbld[12].* |
| **Human Reference** | In this case, they will not write on his profile wall and they do not want him to send them comments because he did not tell them when he will go to the country. |
| **Google Translate** | |
| Feb. 2013 | Bhalhalh Hi Hictpoulo Ahat Profile Tbau not hull Weah Abatln Comintat Anu Mabarhun Oamta welcomed calls them Aalbuld. |
| Jan. 2018 | In the case of Hae Ma Hiktpulo, the personal page of the personal page, they followed him, and they did not know what to do. |
| **ELISSA DA-to-MSA** | (في هذه الحالة)[1] (لن يكتبوا له)[2] (علي حائط)[3] (صفحته الشخصية)[4] ولا (يريدونه ان)[5] (يرسل اليهم)[6] تعليقات[7] لانه[8] (لم يخبرهم)[9] متي[10] سيذهب[11] (الي البلد)[12] .<br>*(fy h*h AlHAlp)[1] (ln yktbwA lh)[2] (Ely HA}T)[3] (SfHth Al$xSyp)[4] wlA (yrydwnh An)[5] (yrsl Alyhm)[6] tElyqAt[7] lAnh[8] (lm yxbrhm)[9] mty[10] sy*hb[11] (Aly Albld)[12].* |
| **Google Translate** | |
| Feb. 2013 | In this case it would not write to him on the wall of his own and do not want to send them comments that he did not tell them when going to the country. |
| Jan. 2018 | In this case they will not write him on the wall of his personal page and do not want him to send them comments because he did not tell them when he will go to the country. |

Table 4.3: Revisiting our motivating example, but with ELISSA-based DA-to-MSA middle step. ELISSA's output is Alif/Ya normalized. Parentheses are added for illustrative reasons to highlight how multi-word DA constructions are selected and translated. Superscript indexes link the selected words and phrases with their MSA translations.

## 4.7 Intrinsic Evaluation: DA-to-MSA Translation Quality

To evaluate ELISSA's MSA output, we first revisit our motivating example and then perform a manual error analysis on the dev set.

### 4.7.1 Revisiting our Motivating Example

We revisit our motivating example in Section 4.2 and show automatic MSA-pivoting through ELISSA. Table 4.3 is divided into two parts. The first part is copied from Table 4.1 for convenience. The second part shows ELISSA's output on the dialectal sentence and its Google Translate translations. Parentheses are added for illustrative reasons to highlight how multi-word DA constructions are selected and translated by ELISSA. Superscript

indexes link the selected words and phrases with their MSA translations. ELISSA MSA output is near perfect and it helps the 2018 Google Translate system to produce near perfect English.

### 4.7.2 Manual Error Analysis

Statistical machine translation systems are very robust; therefore, meddling with their input might result in undesired consequences. When building ELISSA we paid careful attention to selecting dialectal words and phrases to handle in a given DA sentence. We required two conditions:

1. **Words and phrases that *need* to be handled.** This happens when when ELISSA does not have enough confidence in the Arabic-to-English SMT system's ability to translate them. For example, selecting the very frequent dialectal word طب *Tb*, which is often used to start a sentences (as in the English use of 'well' (exclamation), 'but', or 'so') or just to take turn in or interrupt a conversation, will probably result in bad translations for two reasons: 1) the SMT systems probably has seen this word in various context and knows well how to translate it to English; and 2) the word shares the same surface form with the Arabic word for 'medicine' (the only difference is a short vowel which is not spelled), which could result in translating 'medicine' to 'so'.

2. **Words and phrase that ELISSA *knows how* to handle.** ELISSA selects words and phrases that the Translation component can translate. For example, if ELISSA selects a phrase with a dialectal phenomena that is not implemented in the Translation component, unrelated rules could fire on part or all of the phrase and generate wrong output.

For these reasons, we are interested in evaluating ELISSA's accuracy (precision) in se-

lecting and translating DA words and phrases. We do not evaluate recall since ELISSA intentionally ignores some DA words and phrases. We conducted a manual error analysis on the speech-dev set comparing ELISSA's input to its output using our best system settings from the experiments above. Out of 708 affected sentences, we randomly selected 300 sentences (42%). Out of the 482 handled tokens, 449 (93.15%) tokens have good MSA translations, and 33 (6.85%) tokens have wrong MSA translations. Most of the wrong translations are due to spelling errors, proper nouns, and weak input sentence fluency (especially due to speech effect). This analysis clearly validates ELISSA's MSA output. Of course, a correct MSA output can still be mistranslated by the MSA-to-English MT system if it is not in the vocabulary of the system's training data.

## 4.8 Extrinsic Evaluation: DA-English MT

In the following subsections, we evaluate our ELISSA-based MSA-pivoting approach.

### 4.8.1 The MSA-Pivoting Approach

**MSA Lattice Output as Input to Arabic-English SMT.** In Salloum and Habash (2011) we presented the first version of ELISSA used in an MSA-pivoting pipeline where the MSA lattice is passed to the MSA-to-English SMT system without decoding. Although ELISSA can produce lattices as output, we do not evaluate this approach here. For more information, please refer to Salloum and Habash (2011).

**MSA Top-1 Output as Input to Arabic-English SMT.** In all the experiments in this work, we run the DA sentence through ELISSA to generate a top-1 MSA translation, which we then tokenize through MADA before sending to the MSA-English SMT system. Our baseline is to not run ELISSA at all; instead, we send the DA sentence through MADA before applying the MSA-English MT system.

### 4.8.2 Experimental Setup

We discuss the data and the tools used for this evaluation.

#### MT Tools and Training Data

We use the open-source Moses toolkit (Koehn et al., 2007) to build a phrase-based SMT system trained on mostly MSA data (64M words on the Arabic side) obtained from several LDC corpora including some limited DA data. Our system uses a standard phrase-based architecture. The parallel corpus is word-aligned using GIZA++ (Och and Ney, 2003a). Phrase translations of up to 10 words are extracted in the Moses phrase table. The language model for our system is trained on the English side of the bitext augmented with English Gigaword (Graff and Cieri, 2003). We use a 5-gram language model with modified Kneser-Ney smoothing. Feature weights are tuned to maximize BLEU on the NIST MTEval 2006 test set using Minimum Error Rate Training (Och, 2003). This is only done on the baseline systems. The English data is tokenized using simple punctuation-based rules. The Arabic side is segmented according to the Arabic Treebank (ATB) tokenization scheme (Maamouri et al., 2004) using the MADA+TOKAN morphological analyzer and tokenizer v3.1 (Habash and Rambow, 2005; Roth et al., 2008). The Arabic text is also Alif/Ya normalized. MADA-produced Arabic lemmas are used for word alignment. Results are presented in terms of BLEU (Papineni et al., 2002). All evaluation results are case insensitive.

#### The Dev and Test Sets

We use the same development (dev) and test sets used in Chapter 3 which we call speech-dev and speech-test, respectively. We remind the reader that these two sets consist of speech transcriptions of multi-dialect (Iraqi, Levantine, Gulf, and Egyptian) broadcast conversational (BC) segments (with three reference translations), and broadcast news (BN)

segments (with only one reference, replicated three times).

We also evaluate on two web-crawled blind test sets: the Levantine test set presented in Zbib et al. (2012) (we will call it web-lev-test) and the Egyptian Dev-MT-v2 development data of the DARPA BOLT program (we will call it web-egy-test). The web-egy-test has two references while the web-lev-test has only one reference.

The speech-dev set has 1,496 sentences with 32,047 untokenized Arabic words. The speech-test set has 1,568 sentences with 32,492 untokenized Arabic words. The web-lev-test set has 2,728 sentences with 21,179 untokenized Arabic words. The web-egy-test set has 1,553 sentences with 21,495 untokenized Arabic words.

### ELISSA Settings

| **Test Set** | **speech-dev** | |
|---|---|---|
| | BLEU | Diff. |
| **Baseline** | 37.20 | 0.00 |
| **Select:** OOV | 37.75 | 0.55 |
| **Select:** ADAM | 37.88 | 0.68 |
| **Select:** OOV U ADAM | 37.89 | 0.69 |
| **Select:** DICT | 37.06 | -0.14 |
| **Select:** OOV U ADAM U DICT | 37.53 | 0.33 |
| **Select:** (OOV U ADAM) - (Freq >= 50) | 37.96 | 0.76 |
| **Select:** (OOV U ADAM U DICT) - (Freq >= 50) | 38.00 | 0.80 |
| **Select:** Phrase; (OOV U ADAM) | 37.99 | 0.79 |
| **Select:** Phrase; ((OOV U ADAM) - (Freq >= 50)) | 38.05 | 0.85 |
| **Select:** Phrase; ((OOV U ADAM U DICT) - (Freq >= 50)) | **38.10** | **0.90** |

Table 4.4: Results for the speech-dev set in terms of BLEU. The 'Diff.' column shows result differences from the baseline. The rows of the table are the different systems (baseline and ELISSA's experiments). The name of the system in ELISSA's experiments denotes the combination of selection method. In all ELISSA's experiments, all word-based translation methods are tried. Phrase-based translation methods are used when phrase-based selection is used (i.e., the last three rows). The best system is in bold.

We experimented with different method combinations in the selection and translation components in ELISSA. We use the term selection mode and translation mode to denote a certain combination of methods in selection or translation, respectively. We only present

the best selection mode variation experiments. Other selection modes were tried but they proved to be consistently lower than the rest. The 'F2F+L2L; S2S' word-based translation mode (using morphosyntactic transfer of features and lemmas along with surface form transfer) showed to be consistently better than other method combinations across all selection modes. In this work we only use 'F2F+L2L; S2S' word-based translation mode. Phrase-based translation mode is used when phrase-based selection mode is used.

To rank paraphrases in the generated MSA lattice, we combine two 5-gram untokenized Arabic language models: one is trained on Arabic Gigaword data and the other is trained the Arabic side of our SMT training data. The use of the latter LM gave frequent dialectal phrases a higher chance to appear in ELISSA's output; thus, making the output "more dialectal" but adapting it to our SMT input. Experiments showed that using both LMs is better than using each one alone.

### 4.8.3 Machine Translation Results

#### Results on the Development Set

Table 4.4 summarizes the experiments and results on the dev set. The rows of the table are the different systems (baseline and ELISSA's experiments). All differences in BLEU scores from the baseline are statistically significant above the 95% level. Statistical significance is computed using paired bootstrap re-sampling (Koehn, 2004a). The name of the system in ELISSA's experiments denotes the combination of selection method. ELISSA's experiments are grouped into three groups: simple selection, frequency-based selection, and phrase-based selection. Simple selection group consists of five systems: OOV, ADAM, OOV U ADAM, DICT, and OOV U ADAM U DICT. The OOV selection mode identifies the untokenized OOV words. In the ADAM selection mode, or the morphological selection mode, we use ADAM to identify dialectal words. Experiments showed that ADAM's DIAMSA mode (selecting words that have at least one dialectal analysis) is slightly better

than ADAM's DIAONLY mode (selecting words that have only dialectal analyses and no MSA ones). The OOV U ADAM selection mode is the union of the OOVs and ADAM selection modes. In DICT selection mode, we select dialectal words that exist in our DA-MSA dictionaries. The OOV U ADAM U DICT selection mode is the union of the OOVs, ADAM, and DICT selection modes. The results show that combining the output of OOV selection method and ADAM selection method is the best. DICT selection method hurts the performance of the system when used because dictionaries usually have frequent dialectal words that the SMT system already knows how to handle.

In the frequency-based selection group, we exclude from word selection all words with number of occurrences in the training data that is above a certain threshold. This threshold was determined empirically to be 50. The string '- (Freq >= 50)' means that all words with frequencies of 50 or more should not be selected. The results show that excluding frequent dialectal words improves the best simple selection system. It also shows that using DICT selection improves the best system if frequent words are excluded.

In the last system group, phrase+word-based selection, phrase-based selection is used to select phrases and add them on top of the best performers of the previous two groups. Phrase-based translation is also added to word-based translation. Results show that selecting and translating phrases improve the three best performers of word-based selection. The best performer, shown in the last raw, suggests using phrase-based selection and restricted word-based selection. The restriction is to include OOV words and selected low frequency words that have at least one dialectal analysis or appear in our dialectal dictionaries. Comparing the best performer to the OOV selection mode system shows that translating low frequency in-vocabulary dialectal words and phrases to their MSA paraphrases can improve the English translation.

**Results on the Blind Test Sets**

We run the system settings that performed best on the dev set along with the OOV selection mode system on the three blind test set. Results and their differences from the baseline are reported in Table 4.5. We see that OOV selection mode system always improves over the baseline for all test sets. Also, the best performer on the dev is the best performer for all test sets. The improvements of the best performer over the OOV selection mode system on all test sets confirm that translating low frequency in-vocabulary dialectal words and phrases to their MSA paraphrases can improve the English translation. Its improvements over the baseline for the three test sets are: 0.95% absolute BLEU (or 2.5% relative) for the speech-test, 1.41% absolute BLEU (or 15.4% relative) for the web-lev-test, and 0.61% absolute BLEU (or 3.2% relative) for the web-egy-test.

| **Test Set** | **speech-test** | | **web-lev-test** | | **web-egy-test** | |
|---|---|---|---|---|---|---|
| | BLEU | Diff. | BLEU | Diff. | BLEU | Diff. |
| **Baseline** | 38.18 | 0.00 | 9.13 | 0.00 | 18.98 | 0.00 |
| **Select:** OOV | 38.76 | 0.58 | 9.65 | 0.62 | 19.19 | 0.21 |
| **Select:** Phrase; ((OOV U ADAM U DICT) – (Freq >= 50)) | **39.13** | **0.95** | **10.54** | **1.41** | **19.59** | **0.61** |

Table 4.5: Results for the three blind test sets (table columns) in terms of BLEU. The 'Diff.' columns show result differences from the baselines. The rows of the table are the different systems (baselines and ELISSA's experiments). The best systems are in bold.

### 4.8.4 A Case Study

We next examine an example in some detail. Table 4.6 shows a dialectal sentence along with its ELISSA's translation, English references, the output of the baseline system and the output of our best system. The example shows a dialectal word هالمبلغ *hAlmblg* 'this-amount/sum', which is not translated by the baseline (although it appears in the training data, but quite infrequently such that all of its phrase table occurrences have restricted contexts, making it effectively an OOV). The dialectal proclitic هال+ *hAl+* 'this-'

comes sometimes in the dialectal construction: 'hAl+NOUN DEM' (as in this example: هالمبلغ هذا *hAlmblg h*A* 'this-amount/sum this'). ELISSA's selection component captures this multi-word expression and its translation component produces the following paraphrases: هذا المبلغ *h*A Almblg* 'this amount/sum' (*h*A* is used with masculine singular nouns), هذه المبلغ *h*h Almblg* 'this amount/sum' (*h*h* is used with feminine singular or irrational plural nouns), and هؤلاء المبلغ *h&lA' Almblg* 'these amount/sum' (*h&lA'* is used with rational plural nouns). ELISSA's language modeling component picks the first MSA paraphrase, which perfectly fits the context and satisfies the gender/number/rationality agreement (note that the word *Almblg* is an irrational masculine singular noun). For more on Arabic morpho-syntactic agreement patterns, see Alkuhlani and Habash (2011). Finally, the best system translation for the selected phrase is 'this sum'. We can see how both the accuracy and fluency of the sentence have improved.

| | |
|---|---|
| **DA sentence** | fmA mA AtSwr **hAlmblg h*A** yEny. |
| **ELISSA**'s output | fmA mA AtSwr **h*A Almblg** yEny. |
| **References** | I don't think **this amount** is I mean.<br>So I do not I do not think **this cost** I mean.<br>So I do not imagine **this sum** I mean |
| **Baseline** | So i don't think **hAlmblg this** means. |
| **Best system** | So i don't think **this sum** i mean. |

Table 4.6: An example of handling dialectal words/phrases using ELISSA and its effect on the accuracy and fluency of the English translation. Words of interest are bolded.

## 4.9 Conclusion and Future Work

We presented ELISSA, a tool for DA-to-MSA machine translation. ELISSA employs a rule-based MT approach that relies on morphological analysis, morphosyntactic transfer rules and dictionaries in addition to language models to produce MSA translations of dialectal sentences. A manual error analysis of translated selected words shows that our system

produces correct MSA translations over 93% of the time. This high accuracy is due to our careful selection of dialectal words and phrases to translate to MSA.

Using ELISSA to produce MSA versions of dialectal sentences as part of an MSA-pivoting DA-to-English MT solution, improves BLEU scores on three blind test sets by: 0.95% absolute BLEU (or 2.5% relative) for a speech multi-dialect (Iraqi, Levantine, Gulf, Egyptian) test set, 1.41% absolute BLEU (or 15.4% relative) for a web-crawled Levantine test set, and 0.61% absolute BLEU (or 3.2% relative) for a web-crawled Egyptian test set. This shows that the MSA-pivoting approach can provide a good solution when translating dialects with no DA-English parallel data, and that rule based approaches like ELISSA and ADAM can help when no preprocessing tools are available for those dialects.

In the future, we plan to extend ELISSA's coverage of phenomena in the handled dialects and to new dialects. We also plan to automatically learn additional rules from limited available DA-English data. Finally, we look forward to experimenting with ELISSA as a preprocessing system for a variety of dialect NLP applications similar to Chiang et al. (2006)'s work on dialect parsing, for example.

# Part II

# Translating Dialects with Dialectal Resources

# Chapter 5

# Pivoting with Statistical and Hybrid DA-to-MSA Machine Translation

## 5.1 Introduction

In Chapter 4, we presented an MSA-pivoting approach to DA-to-English MT where DA-English parallel data is not available. We showed how ELISSA, a rule-based DA-to-MSA MT system, can help an MSA-to-English SMT handle DA sentences by translating select DA words and phrases into their MSA equivalents.

In this chapter, we explore dialects with some parallel data. We present different techniques to utilize this DA-English corpus in order to answer the question of whether the MSA-pivoting approach to DA-to-English MT is still relevant when a good amount of DA-English parallel data is available. For that purpose, we leverage the use of a huge collection of MSA-English data and Rule-Based ELISSA which we explored before. We also present two new DA-to-MSA MT systems that can be built for dialects that have DA-English parallel corpus: Statistical ELISSA, a DA-to-MSA statistical MT system, and Hybrid ELISSA, a combination of Rule-Based and Statistical ELISSA. We present new combinations of MSA-pivoting systems and we evaluate the three DA-to-MSA MT systems based on the

quality of the English translations of their corresponding pivoting system.

## 5.2 Dialectal Data and Preprocessing Tools

In the previous part, we used the MADA+TOKAN morphological analyzer and tokenizer v3.1 (Roth et al., 2008) to segment the Arabic side of the training data according to the Arabic Treebank (ATB) tokenization scheme (Maamouri et al., 2004; Sadat and Habash, 2006). In this part, we have a dialectal preprocessing tool, MADA-ARZ (Habash et al., 2013), that performs normalizations and tokenization on Egyptian Arabic.

We use two parallel corpora. The first is a *DA-English* corpus of ∼5M tokenized words of Egyptian (∼3.5M) and Levantine (∼1.5M). This corpus is part of BOLT data. The ATB tokenization is performed with MADA-ARZ for both Egyptian and Levantine. The second is an *MSA-English* corpus of 57M tokenized words obtained from several LDC corpora (10 times the size of the DA-English data). This MSA-English corpus is a subset of the Arabic-English corpus we used in Chapter 3 and 4 that excludes datasets that may have dialectal data. Unlike the DA-English corpus, we ATB-tokenize the MSA side of this corpus with MADA+TOKAN.

The Arabic text is also Alif/Ya normalized. The English data is tokenized using simple punctuation-based rules.

## 5.3 Synthesizing Parallel Corpora

Statistical machine translation outperforms rule-based MT when trained on enough parallel data. DA-MSA parallel text, however, is scarce and not enough to train an effective SMT system. Therefore, we use the new DA-English parallel data to generate SMT parallel data using two sentence-level pivoting approaches: 1) using an English-to-MSA SMT system to translate the English side of the new data to MSA, and 2) using ELISSA to translate the DA

side to MSA. Both approaches result in a three-way DA-MSA-English parallel corpora. In the following subsections we discuss these two approaches. Diagram (a) in Figure 5.1 is an illustration of these two techniques.

#### Synthesizing DA-MSA data using English-to-MSA SMT

In this approach we try to automatically create DA-MSA training data starting from the existing DA-English parallel text ($\sim$5M tokenized words on the DA side). We use an English-to-MSA SMT system, trained on MSA-English parallel text (57M tokenized words on the MSA side), to translate every sentence on the English side of the DA-English parallel data to MSA. We call the resulted MSA data $\text{MSA}_t$ ('t' for translated) to distinguish it from naturally occurring MSA. This results in DA-English-$\text{MSA}_t$ sentence-aligned parallel data.

#### Synthesizing MSA-English data using ELISSA

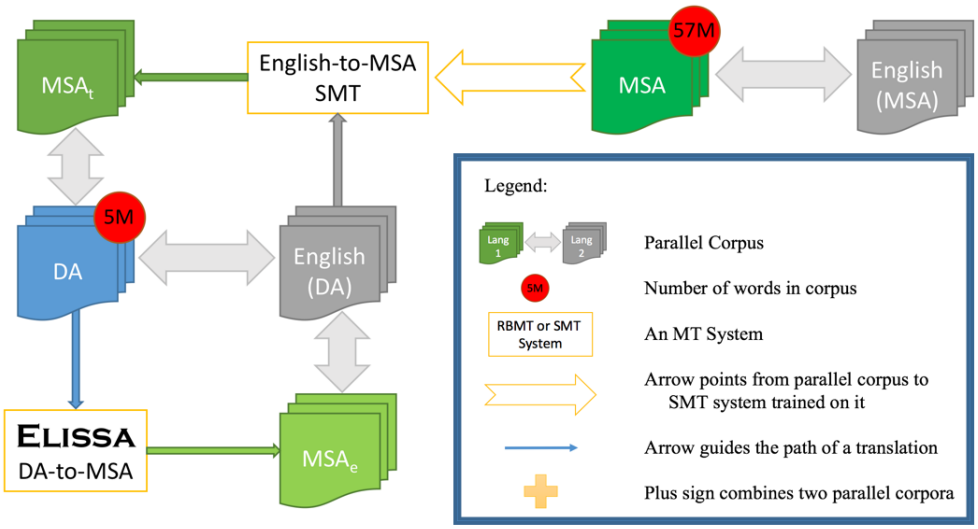
We run ELISSA on every sentence on the DA side of the DA-English parallel text to produce $\text{MSA}_e$ ('e' for ELISSA). This results in DA-English-$\text{MSA}_e$ sentence-aligned parallel data.

## 5.4 The MSA-Pivoting Approach

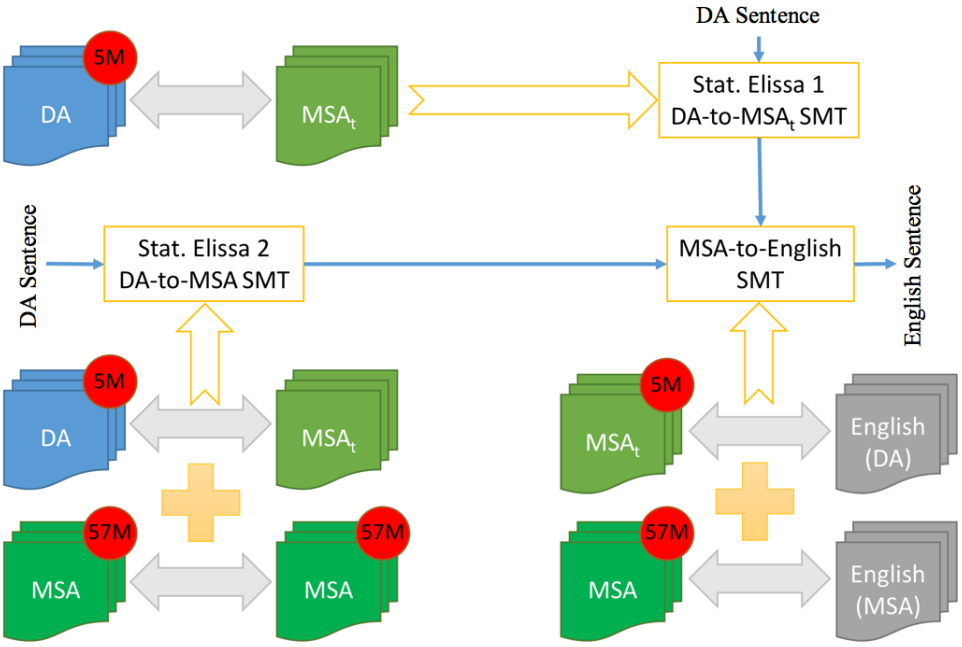
We use a cascading approach for pivoting that consists of two systems. The ***frontend system*** translates the dialectal input sentence to MSA. Our baseline frontend system passes the input as is to the backend system. The ***backend system*** takes the MSA output and translates it to English. We have three baseline system for our backend systems:

1. **MSA $\rightarrow$ Eng.** This system is trained on the MSA-English parallel corpus. This system is the closest to the system we trained in the previous part.

2. **DA $\rightarrow$ Eng.** This system is trained on the DA-English parallel corpus.

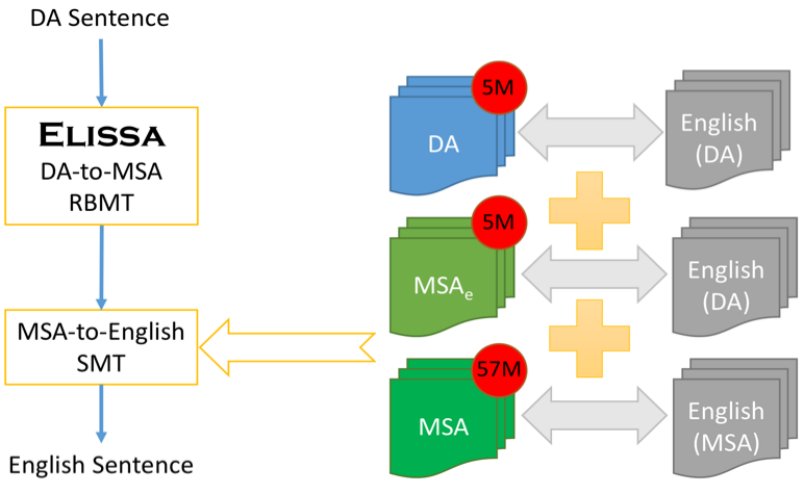



(a)

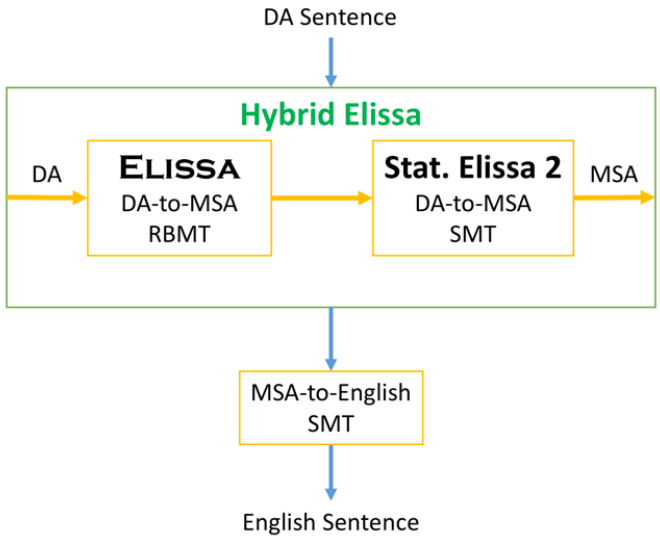



(b)

Figure 5.1: Synthetic parallel data generation and Statistical ELISSA. The legend is shown in a box above. Diagram (a) illusrates the translation process of the DA side of the parallel data to MSA$_t$ using an English-to-MSA SMT system trained on the 57M MSA-English parallel corpus, and the translation process of the English side of the DA-English parallel data to MSA$_e$ using Rule-Based Elissa. Diagram (b) illustrates the creation of the two SMT systems: Statistical ELISSA 1 and 2 using the generated MSA$_t$ data. Diagram (b) also shows the way these two systems will be used for MSA-pivoting.




(c)




(d)

Figure 5.2: The legend is shown in Figure 5.1. Diagram (c) presents a new MSA-pivoting approach using Rule-Based ELISSA followed by an MSA-to-English SMT trained on generated MSA$_e$ data along with the MSA-English and DA-English. Diagram (d) shows Hybrid ELISSA, a combination of Rule-Based ELISSA and Statistical ELISSA.

3. **DA+MSA → Eng.** This system is trained on the combination of the above two corpora.

### 5.4.1 Improving MSA-Pivoting with Rule-Based ELISSA

Our pivoting approach in Chapter 4 used an MSA-to-English SMT system as a backend system. The new DA-English data and the Egyptian preprocessing tool, MADA-ARZ, available in this part, provide harder baselines to rule-based ELISSA to beat. Therefore, we need to use the new available resources to ELISSA's advantage.

**Customizing the backend system for ELISSA's output.** We build a backend system that is familiar with ELISSA's output: We add the $MSA_e$-English part to the DA-English and MSA-English parallel data, as shown in Diagram c of Figure 5.2, and train an **$MSA_e$+DA+MSA→ English SMT** system on them. This pipeline is more robust than the one discussed in Chapter 4 because it tolerates some of ELISSA's errors since they will be repeated in the training data of the backend system. In this new pivoting approach, we use rule-based ELISSA as the frontend system and **$MSA_e$+DA +MSA→ English SMT** as the backend system.

**Customizing ELISSA to the backend system's training data.** This means that ELISSA's out-of-vocabulary list, low frequency list, and one of the two language models are built using the Arabic side of the training data of the backend system. The second language model is the default Arabic GigaWord LM packaged with ELISSA.

Since we are optimizing from the DA and MSA mix, we always tokenize ELISSA's output with MADA-ARZ.

### 5.4.2 MSA-Pivoting with Statistical DA-to-MSA MT

We use the synthetic DA-MSA$_t$ parallel text to train two DA-MSA SMT systems (Statistical ELISSA) that will be used as the frontend system in the MSA-pivoting DA-English MT system:

1. **Statistical ELISSA 1:** We train a **DA→MSA$_t$ SMT** on the DA-MSA$_t$ parallel data.

2. **Statistical ELISSA 2:** We add the 57M-word MSA of the MSA-English parallel text to both sides of DA-MSA$_t$ and we train a **DA+MSA→MSA$_t$+MSA SMT** system on it. The motivation behind this approach is that the alignment algorithm is going to assign a high probability for aligning an MSA word from the DA side to itself on the MSA side and, therefore, it will have an easier job aligning the remaining DA words to their MSA translations in a given sentence pair. Additionally, it allows this SMT system to produce MSA words that would be OOVs otherwise.

On the other side of the pivoting pipeline, we add the synthetic MSA$_t$-English parallel data to our MSA-English parallel data and train an **MSA+MSA$_t$→English SMT** system that will be used as the backend system in the MSA-pivoting DA-English MT system. This addition allows the backend system to be familiar with the output of Statistical ELISSA 1 and 2.

Diagram (b) of Figure 5.1 illustrates the creation of these three SMT systems.

### 5.4.3 MSA-Pivoting with Hybrid DA-to-MSA MT

Rule-based MT systems use linguistic knowledge to translate a source sentence, or phrase, to a target sentence, or phrase. On the other hand, statistical MT systems use statistical evidence to justify the translation. To put it in terms of precision and recall, we designed our RBMT system to attempt at translating words and phrases only when it's confident enough about the change. This high precision approach results in linguistically motivated

changes in a source sentence while keeping many other words and phrases as is, either because the confidence is low, or because there are no rules that match. In contrast, SMT systems attempt at translating every word and phrase and most of the time have something to backoff to; hence, high recall.

To make the best of the two worlds, we combine RBMT and SMT systems in one system: Hybrid ELISSA, which runs Rule-Based ELISSA on input sentences and passes them to Statistical ELISSA (we choose the **DA+MSA→MSA+MSA$_t$ SMT** system since it performs better).

## 5.5 Evaluation

In this section, we evaluate the performance of the three MSA-pivoting approaches against our baselines.

### 5.5.1 Experimental Setup

**MT tools and settings.** We use the open-source Moses toolkit (Koehn et al., 2007) to build four Arabic-English phrase-based statistical machine translation systems (SMT). Our systems use a standard phrase-based architecture. The parallel corpora are word-aligned using GIZA++ (Och and Ney, 2003a). The language model for our systems is trained on English Gigaword (Graff and Cieri, 2003). We use SRILM Toolkit (Stolcke, 2002) to build a 5-gram language model with modified Kneser-Ney smoothing. Feature weights are tuned to maximize BLEU on tuning sets using Minimum Error Rate Training (Och, 2003). Results are presented in terms of BLEU (Papineni et al., 2002). All evaluation results are case *insensitive*.

**Customization of rule-based ELISSA to backend systems.** We experimented with customizing rule-based (RB) ELISSA to the Arabic side of four backend systems and we ran

experiments will all combinations. We excluded customization to the Arabic side of the **MSA$_e$+DA+MSA $\rightarrow$ English** system because we do not want ELISSA to learn and repeat its mistakes (since MSA$_e$ is RB ELISSA's output). We found that the best performance is achieved when RB ELISSA is customized to the MSA side for our MSA-based backend systems (**MSA $\rightarrow$ English**, and to the combination of DA and MSA sides (DA+MSA) for the other three systems. Customizing RB ELISSA to DA only when translating with **DA $\rightarrow$ English** and customizing to MSA+MSA$_t$ when translating with **MSA+MSA$_t$ $\rightarrow$ English** gave lower results.

**The tuning and test set.** Since the only dialectal preprocessing tool available to us is MADA-ARZ which covers Egyptian Arabic, in this chapter we evaluate only on Egyptian. We use for our test set the Egyptian BOLT Dev V2 (EgyDevV2) which contains 1,553 sentences with two references.

We tune our MSA-based backend systems (**MSA $\rightarrow$ English** and **MSA$_t$+MSA $\rightarrow$ English**) on an MSA test set, NIST MTEval MT08, which contains 1,356 sentences with four references. We tune the other three backend systems on the Egyptian BOLT Dev V3 (EgyDevV3) which contains 1,547 sentences with two references.

To tune our SMT-based frontend systems, Statistical ELISSA 1 and 2, we synthesize a tuning set in the same way we synthesized their training data. We translate the English side of EgyDevV3 with the same English-to-MSA SMT system to produce its MSA$_t$ side, then we tune both Statistical ELISSA versions on this DA-MSA$_t$ tuning set.

### 5.5.2 Experiments

#### Evaluation of the Importance of Dialectal Tokenization

In this subsection, we discuss the effect of having a DA tokenization tool when DA-English is not available. We tokenize the test set EgyDevV2 with both MADA and MADA-ARZ and

| | **MSA → Eng.** | |
|---|---|---|
| **Tokenization Tool** | BLEU | METEOR |
| MADA | 9.14 | 17.33 |
| MADA-ARZ | 18.61 | 26.42 |

Table 5.1: Results comparing the performance of MADA-ARZ against MADA when used to tokenize the Egyptian test set (EgyDevV2) before passing it to the **MSA → English** system. This table shows the importance of dialectal tokenization when DA-English data is not available.

| | **MSA → Eng.** | | **DA → Eng.** | | **DA+MSA → Eng.** | | **$MSA_t$+MSA → Eng.** | | **$MSA_e$+DA+ MSA → Eng.** | |
|---|---|---|---|---|---|---|---|---|---|---|
| | BLE. | MET. | BLE. | MET. | BLE. | MET. | BLE. | MET. | BLE. | MET. |
| No frontend | 18.61 | 26.42 | 20.23 | 28.10 | 21.61 | 28.83 | – | – | – | – |
| Stat. ELISSA 1 | 7.84 | 19.75 | 8.59 | 19.92 | 9.37 | 20.25 | 7.86 | 20.23 | – | – |
| Stat. ELISSA 2 | 19.33 | 26.96 | 17.94 | 26.64 | 19.46 | 27.53 | 18.83 | 27.21 | – | – |
| RB ELISSA | 19.36 | 26.98 | 20.27 | 28.09 | 21.66 | 28.87 | 19.62 | 26.95 | **22.17** | **29.18** |
| Hybrid ELISSA | 19.43 | 26.97 | 17.80 | 26.60 | 19.45 | 27.54 | 19.85 | 26.93 | 19.80 | 27.73 |

Table 5.2: Results of the pivoting approaches. Rows show frontend systems, columns show backend systems, and cells show results in terms of BLEU (white columns, abbreviated as BLE.) and METEOR (gray columns, abbreviated as MET.).

we translate the tokenized set with the **MSA → English** system. Table 5.1 shows the results of the two tokenization approaches. We see that by just tokenizing a dialectal set with a DA-specific tokenization tool we get an improvement of 9.47% BLEU and 9.09% METEOR absolute. This is due to MADA-ARZ dialectal normalization and tokenization which reduces the number of out-of-vocabulary words. This shows the importance of dialectal tokenization and motivates the work we do in Part III where we scale to more dialects.

### Results on the Test Set

Table 5.2 shows the results of our pivoting approaches. Rows show frontend systems, columns show backend systems, and cells show results in terms of BLEU and METEOR.

The first section of the table shows a Direct Translation approach where no frontend system is used for preprocessing. The results are our baselines on the first three systems. It is important to note that the baselines are high because MADA-ARZ takes care of many

dialectal pheonomena and tokenization solves a huge chunk of the problem since it dramatically reduces the vocabulary size and increase frequency.

The second section of the table presents the results of translating the English side to $\text{MSA}_t$. As expected, Statistical ELISSA 1 produces very low results and adding the MSA data to Statistical ELISSA 2 training data has dramatically improved alignment; however, results are still lower than the baselines. This is caused by the errors resulting from using an English-to-MSA SMT system to translate the English side which then propagate to later steps.

The third section of the table presents the results of running rule-based ELISSA on input sentences and passing the output to backend systems. The best result comes from using rule-based ELISSA with the compatible **DA+MSA**$\rightarrow$**MSA+MSA**$_t$ SMT system. It outperforms the best direct translation system by 0.56% BLEU and 0.35% METEOR. These improvements are statistically significant above the 95% level. Statistical significance is computed using paired bootstrap re-sampling (Koehn, 2004a).

The last section of the table shows the results for Hybrid ELISSA which suggest that Statistical ELISSA is hurting rule-based ELISSA's performance.

## 5.6 Conclusion and Future Work

In this chapter, we show that MSA-pivoting approaches to DA-to-English MT can still help when the available parallel data for a dialect is relatively small compared to MSA. The key for the improvements we presented is to exploit the small DA-English data to create automatically generated parallel corpora on which SMT systems can be trained. We translated the DA side of the DA-English parallel data to MSA using ELISSA, and added that data to the (DA+MSA)-English training data on which an SMT system was trained. That SMT system, when combined with ELISSA for preprocessing, outperforms all other direct translation or pivoting approaches. The main reason for this improvement

is that the SMT system is now familiar with ELISSA's output and can correct systematic errors performed by ELISSA. This RB ELISSA-based MSA-pivoting system is used in the Chapter 6 as the best MSA-pivoting system. We presented two new versions of ELISSA that use the synthetic parallel data. However, both systems failed to improve the translation quality.

In this work we used sentence-level pivoting techniques to synthesize parallel data. In the future we plan to use different pivoting techniques such as phrase table pivoting to create DA-MSA SMT systems. We also plan to automatically learn ELISSA's morphological transfer rules from the output of these pivoting techniques.

# Chapter 6

# System Combination

## 6.1 Introduction

In the previous chapter we introduced different systems built using a relatively small amount of Dialectal Arabic (DA) to English parallel corpus. The chapter evaluates two direct translation approaches: a statistical machine translation (SMT) system trained on the 5MW DA-English data and an SMT trained on the combination of this data with a 57MW MSA-English data. The previous chapter also uses the DA-English parallel data to improve the MSA-pivoting approach and shows that it *slightly* outperforms the two direct translation systems.

Arabic dialects co-exist with MSA in a diglossic relationship where DA and MSA occupy different roles, e.g., formal vs informal registers. Additionally, there are different degrees of dialect-switching that take place. This motivates the hypothesis that automatically choosing one of these systems to translate a given sentence based on its dialectal nature could outperform each system separately. Given that dialectal sets might include full MSA sentences, we add the MSA-to-English SMT system to the three dialect translation systems mentioned above resulting in four baseline MT systems.

In Section 6.3, we describe these four systems and present *oracle* system combination

results to confirm the hypothesis. In Section 6.4, we present our approach which studies the use of sentence-level dialect identification together with various linguistic features in optimizing the selection of the four baseline systems on input sentences that includes a mix of dialects.

## 6.2 Related Work

The most popular approach to MT system combination involves building confusion networks from the outputs of different MT systems and decoding them to generate new translations (Rosti et al., 2007; Karakos et al., 2008; He et al., 2008; Xu et al., 2011). Other researchers explored the idea of re-ranking the n-best output of MT systems using different types of syntactic models (Och et al., 2004; Hasan et al., 2006; Ma and McKeown, 2013). While most researchers use target language features in training their re-rankers, others considered source language features (Ma and McKeown, 2013).

Most MT system combination work uses MT systems employing different techniques to train on the same data. However, in the system combination work we present in this thesis (Chapter 6), we use the same MT algorithms for training, tuning, and testing, but we vary the training data, specifically in terms of the degree of source language dialectness. Our approach runs a classifier trained only on source language features to decide which system should translate each sentence in the test set, which means that each sentence goes through one MT system only.

## 6.3 Baseline Experiments and Motivation

In this section, we present our MT experimental setup and the four baseline systems we built, and we evaluate their performance and the potential of their combination. In the next section we present and evaluate the system combination approach.

### 6.3.1 Experimental Settings

#### MT Tools and Settings

We use the open-source Moses toolkit (Koehn et al., 2007) to build four Arabic-English phrase-based statistical machine translation systems (SMT). Our systems use a standard phrase-based architecture. The parallel corpora are word-aligned using GIZA++ (Och and Ney, 2003a). The language model for our systems is trained on English Gigaword (Graff and Cieri, 2003). We use SRILM Toolkit (Stolcke, 2002) to build a 5-gram language model with modified Kneser-Ney smoothing. Feature weights are tuned to maximize BLEU on tuning sets using Minimum Error Rate Training (Och, 2003). Results are presented in terms of BLEU (Papineni et al., 2002). All evaluation results are case *insensitive*. The English data is tokenized using simple punctuation-based rules. The MSA portion of the Arabic side is segmented according to the Arabic Treebank (ATB) tokenization scheme (Maamouri et al., 2004; Sadat and Habash, 2006) using the MADA+TOKAN morphological analyzer and tokenizer v3.1 (Roth et al., 2008), while the DA portion is ATB-tokenized with MADA-ARZ (Habash et al., 2013). The Arabic text is also Alif/Ya normalized. For more details on processing Arabic, see (Habash, 2010).

#### MT Train/Tune/Test Data

We use two parallel corpora. The first is a *DA-English* corpus of ∼5M tokenized words of Egyptian (∼3.5M) and Levantine (∼1.5M). This corpus is part of BOLT data. The second is an *MSA-English* corpus of 57M tokenized words obtained from several LDC corpora (10 times the size of the DA-English data).

We work with nine standard MT test sets: three MSA sets from NIST MTEval with four references (MT06, MT08, and MT09), four Egyptian sets from LDC BOLT data with two references (EgyDevV1, EgyDevV2, EgyDevV3, and EgyTestV2), and two Levantine

sets from BBN (Zbib et al., 2012)[1] with one reference.

Table 6.1 presents details about the sets we used. The fifth column of the table shows the tasks in which these MT test sets are used: SMT systems tuning sets (SMT Tune), system combination classifiers' training data (SC Train), or the development and blind test sets (Dev/Test). We used MT08 and EgyDevV3 to tune SMT systems while we divided the remaining sets among classifier training data (5,562 sentences), dev (1,802 sentences) and blind test (1,804 sentences) sets to ensure each of these new sets has a variety of dialects and genres (weblog and newswire). Details on the classifier's training data are in Section 6.4. For MT Dev and Test Sets, we divide MT09 into two sets according to genre: MT09nw consisting of 586 newswire sentences, and MT09wb consisting of 727 Web Blog sentences. We use the first half of each of EgyTestV2, LevTest, MT09nw, and MT09wb to form our dev set (1,802 sentences) and the second half to form our blind test set (1,804 sentences).

| **Set Name** | **Dia.** | **Sents** | **Refs** | **Used for** |
|---|---|---|---|---|
| MTEval 2006 (MT06) | MSA | 1,664 | 4 | SC Train |
| MTEval 2008 (MT08) | MSA | 1,356 | 4 | SMT Tune |
| MTEval 2009 (MT09) | MSA | 1,313 | 4 | Dev/Test |
| BOLT Dev V1 (EgyDevV1) | Egy | 845 | 2 | SC Train |
| BOLT Dev V2 (EgyDevV2) | Egy | 1,553 | 2 | SC Train |
| BOLT Dev V3 (EgyDevV3) | Egy | 1,547 | 2 | SMT Tune |
| BOLT Test V2 (EgyTestV2) | Egy | 1,065 | 2 | Dev/Test |
| Levantine Dev (LevDev) | Lev | 1,500 | 1 | SC Train |
| Levantine Test (LevTest) | Lev | 1,228 | 1 | Dev/Test |

Table 6.1: MT test set details. The four columns correspond to set name with short name in parentheses, dialect (Egy for Egyptian and Lev for Levantine), number of sentences, number of references, and the task it was used in.

[1] The Levantine sets are originated from one set presented in Zbib et al. (2012). Since this set is the only Levantine set available to us we had to divide it into dev (the first 1,500 sentences) and test (the rest: 1,228 sentences)

### 6.3.2 Baseline MT Systems

#### MT Systems

We use four MT systems (discussed in more details in Chapter 5):

1. ***DA-Only*.** This system is trained on the DA-English data and tuned on EgyDevV3.

2. ***MSA-Only*.** This system is trained on the MSA-English data and tuned on MT08.

3. ***DA+MSA*.** This system is trained on the combination of both corpora (resulting in 62M tokenized[2] words on the Arabic side) and tuned on EgyDevV3.

4. ***MSA-Pivot*.** This is the best MSA-pivoting system presented in Chapter 5. It uses ELISSA (Salloum and Habash, 2013) followed by an Arabic-English SMT system which is trained on both corpora augmented with the DA-English where the DA side is preprocessed with ELISSA then tokenized with MADA-ARZ. The result is 67M tokenized words on the Arabic side. EgyDevV3 was similarly preprocessed with ELISSA and MADA-ARZ and used for tuning the system parameters. Test sets are similarly preprocessed before decoding with the SMT system.

#### Baseline MT System Results.

We report the results of our dev set on the four MT systems we built in Table 6.2. The *MSA-Pivot* system produces the best singleton result among all systems. All differences in BLEU scores between the four systems are statistically significant above the 95% level. Statistical significance is computed using paired bootstrap re-sampling (Koehn, 2004a).

---

[2]Since the *DA+MSA* system is intended for DA data and DA morphology, as far as tokenization is concerned, is more complex, we tokenized the training data with dialect awareness (DA with MADA-ARZ and MSA with MADA) since MADA-ARZ does a lot better than MADA on DA (Habash et al., 2013). Tuning and Test data, however, are tokenized by MADA-ARZ since we do not assume any knowledge of the dialect of a test sentence.

| System Name | Training Data (TD) | | | | BLEU |
|---|---|---|---|---|---|
| | DA-En | MSA-En | $MSA_e$-En | **TD Size** | |
| 1. *DA-Only* | 5M | | | 5M | 26.6 |
| 2. *MSA-Only* | | 57M | | 57M | 32.7 |
| 3. *DA+MSA* | 5M | 57M | | 62M | 33.6 |
| 4. *MSA-Pivot* | 5M | 57M | 5M | 67M | **33.9** |
| **Oracle System Selection** | | | | | **39.3** |

Table 6.2: Results from the baseline MT systems and their oracle system combination. The first part of the table shows MT results in terms of BLEU for our Dev set on our four baseline systems (each system training data is provided in the second column for convenience). $MSA_e$ (in the fourth column) is the DA part of the 5M word DA-English parallel data processed with the ELISSA. The second part of the table shows the oracle combination of the four baseline systems.

### 6.3.3 Oracle System Combination

We also report in Table 6.2 an oracle system combination where we pick, for each sentence, the English translation that yields the best BLEU score. This oracle indicates that the upper bound for improvement achievable from system combination is 5.4% BLEU. Excluding different systems from the combination lowered the overall score between 0.9% and 1.8%, suggesting the systems are indeed complementary.

## 6.4 Machine Translation System Combination

The approach we take in this work benefits from the techniques and conclusions of previous chapters and related work in that we build different MT systems using those techniques but instead of trying to find which one is the best on the whole set, we try to automatically decide which one is the best for a given sentence. Our hypothesis is that these systems complement each other in interesting ways where their combination could lead to better overall performance stipulating that our approach could benefit from the strengths while avoiding the weaknesses of each individual system.

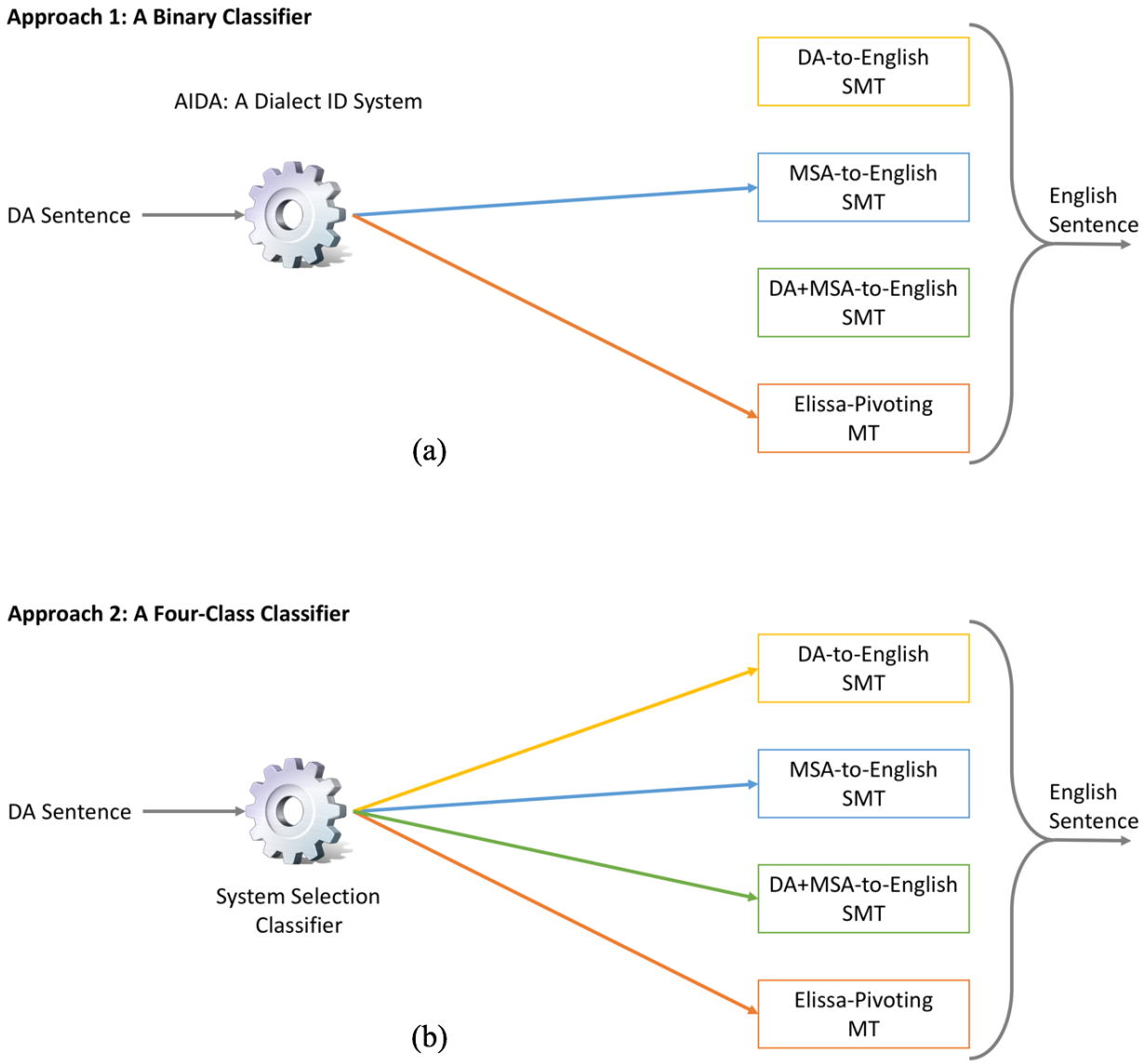



Figure 6.1: This diagram illustrates our two system combination approachs: (a) is our dialect ID binary classification appraoch which uses AIDA; and (b) is our feature-based four-class classification approach.

### 6.4.1 Dialect ID Binary Classification

For baseline system combination, we use the classification decision of Elfardy and Diab (2013)'s AIDA sentence-level dialect identification system to decide on the target MT system. Since the decision is binary (DA or MSA) and we have four MT systems, we considered all possible configurations and determined empirically that the best configuration is to select *MSA-Only* for the MSA tag and *MSA-Pivot* for the DA tag. We do not report other configuration results. Table 6.1, diagram (a), illustrates the use of AIDA as the binary classifier in our binary system combination approach.

### 6.4.2 Feature-based Four-Class Classification

For our main approach, we train a four-class classifier to predict the target MT system to select for each sentence using only source-language features. Table 6.1, diagram (b), shows the setup for such system.

We experimented with different classifiers in the Weka Data Mining Tool (Hall et al., 2009) for training and testing our system combination approach. The best performing classifier was Naive Bayes[3] (with Weka's default settings).

#### Training Data Class Labels

We run the 5,562 sentences of the classification training data through our four MT systems and produce sentence-level BLEU scores (with length penalty). We pick the name of the MT system with the highest BLEU score as the class label for that sentence. When there is a tie in BLEU scores, we pick the system label that yields better overall BLEU scores from the systems tied.

[3]Typically, in small training data settings (5,562 examples), generative models (Naive Bayes) outperform discriminative models (Ng and Jordan, 2002).

**Training Data Source-Language Features**

We use two sources of features extracted from untokenized sentences to train our four-class classifiers: *basic* and *extended features*.

***A. Basic Features***

These are the same set of features that were used by the dialect ID tool together with the class label generated by this tool.

*i. Token-Level Features.* These features rely on language models, MSA and Egyptian morphological analyzers and a Highly Dialectal Egyptian lexicon to decide whether each word is MSA, Egyptian, Both, or Out of Vocabulary.

*ii. Perplexity Features.* These are two features that measure the perplexity of a sentence against two language models: MSA and Egyptian.

*iii. Meta Features.* Features that do not directly relate to the dialectalness of words in the given sentence but rather estimate how informal the sentence is and include: percentage of tokens, punctuation, and Latin words, number of tokens, average word length, whether the sentence has any words that have word-lengthening effects or not, whether the sentence has any diacritized words or not, whether the sentence has emoticons or not, whether the sentence has consecutive repeated punctuation or not, whether the sentence has a question mark or not, and whether the sentence has an exclamation mark or not.

*iv. The Dialect-Class Feature.* We run the sentence through the Dialect ID binary classifier and we use the predicted class label (DA or MSA) as a feature in our system. Since the Dialect ID system was trained on a different data set, we think its decision may provide additional information to our classifiers.

***B. Extended Features***

We add features extracted from two sources.

*i. MSA-Pivoting Features.* ELISSA produces intermediate files used for diagnosis or debugging purposes. We exploit one file in which the system identifies (or, "selects") dialectal

words and phrases that need to be translated to MSA. We extract confidence indicating features. These features are: sentence length (in words), percentage of selected words and phrases, number of selected words, number of selected phrases, number of words morphologically selected as dialectal by a mainly Levantine morphological analyzer, number of words selected as dialectal by the ELISSA's DA-MSA lexicons, number of OOV words against the *MSA-Pivot* system training data, number of words in the sentences that appeared less than 5 times in the training data, number of words in the sentences that appeared between 5 and 10 times in the training data, number of words in the sentences that appeared between 10 and 15 times in the training data, number of words that have spelling errors and corrected by this tool (e.g., word-lengthening), number of punctuation marks, and number of words that are written in Latin script.

*ii. MT Training Data Source-Side LM Perplexity Features.* The second set of features uses perplexity against language models built from the source-side of the training data of each of the four baseline systems. These four features may tell the classifier which system is more suitable to translate a given sentence.

### 6.4.3 System Combination Evaluation

Finally, we present the results of our system combination approach on the Dev and Blind Test sets.

#### Development Set

The first part of Table 6.3 repeats the best baseline system and the four-system oracle combination from Table 6.2 for convenience. The third row shows the result of running our system combination baseline that uses the Dialect ID binary decision on the Dev set sentences to decide on the target MT system. It improves over the best single system baseline (*MSA-Pivot*) by a statistically significant 0.5% BLEU. Crucially, we should note

that this is a deterministic process.

| System | BLEU | Diff. |
|---|---|---|
| **Best Single MT System Baseline** | **33.9** | 0.0 |
| **Oracle** | *39.3* | *5.4* |
| **Dialect ID Binary Selection Baseline** | **34.4** | 0.5 |
| **Four-Class Classification** | | |
| Basic Features | 35.1 | 1.2 |
| Extended Features | 34.8 | 0.9 |
| Basic + Extended Features | **35.2** | 1.3 |

Table 6.3: Results of baselines and system selection systems on the Dev set in terms of BLEU. The best single MT system baseline is *MSA-Pivot*. The first column shows the system, the second shows BLEU, and the third shows the difference from the best baseline system. The first part of the table shows the results of our best baseline MT systems and the oracle combination repeated for convenience. It also shows the results of the Dialect ID binary classification baseline. The second part shows the results of the four-class classifiers we trained with the different feature vector sources.

The second part of Table 6.3 shows the results of our four-class Naive Bayes classifiers trained on the classification training data we created. The first column shows the source of sentence level features employed. As mentioned earlier, we use the Basic features alone, the Extended features alone, and then their combination. The classifier that uses both feature sources simultaneously as feature vectors is our best performer. It improves over our best baseline single MT system by 1.3% BLEU and over the Dialect ID Binary Classification system combination baseline by 0.8% BLEU. Improvements are statistically significant.

### Blind Test Set

Table 6.4 shows the results on our Blind Test set. The first part of the table shows the results of our four baseline MT systems. The systems have the same rank as on the Dev set and *MSA-Pivot* is also the best performer. The differences in BLEU are statistically significant. The second part shows the four-system oracle combination which shows a 5.5% BLEU upper bound on improvements. The third part shows the results of the Dialect ID Binary Classification which improves by 0.4% BLEU. The last row shows the four-class classifier

| System | BLEU | Diff. |
|---|---|---|
| *DA-Only* | 26.6 | |
| *MSA-Only* | 30.7 | |
| *DA+MSA* | 32.4 | |
| *MSA-Pivot* | **32.5** | |
| *Four-System Oracle Combination* | *38.0* | *5.5* |
| **Best Dialect ID Binary Classifier** | 32.9 | 0.4 |
| **Best Classifier: Basic + Extended Features** | **33.5** | **1.0** |

Table 6.4: Results of baselines and system selection systems on the Blind test set in terms of BLEU. Results in terms of BLEU on our Blind Test set. The first column shows the system, the second shows BLEU, and the third shows the difference from the best baseline system. The first part of the table shows the results of our baseline MT systems and the four-system oracle combination. The second part shows the Dialect ID binary classification technique's best performer results, and the results of the best four-class classifier we trained.

results which improves by 1.0% BLEU over the best single MT system baseline and by 0.6% BLEU over the Dialect ID Binary Classification. Results on the Blind Test set are consistent with the Dev set results.

| System | All | Dialect | MSA | Egyptian | Levantine | MSA NW | MSA WB |
|---|---|---|---|---|---|---|---|
| *DA-Only* | 26.6 | 19.3 | 33.2 | 20.5 | **16.0** | 34.4 | 32.0 |
| *MSA-Only* | 32.7 | 14.7 | **50.0** | 16.9 | 8.8 | **56.2** | **44.0** |
| *DA+MSA* | 33.6 | 19.4 | 46.3 | 21.1 | 13.8 | 51.1 | 41.9 |
| *MSA-Pivot* | 33.9 | **19.6** | 46.4 | **21.5** | 13.9 | 51.3 | 41.8 |
| *Four-System Oracle Combination* | *39.3* | *24.4* | *52.1* | *26.5* | *19.7* | *58.6* | *47.8* |
| **Best Four-Class Classifier** | **35.2** | **19.8** | **50.0** | **21.7** | 15.6 | **56.2** | 43.9 |

Table 6.5: Dialect and genre breakdown of performance on the Dev set for our best performing classifier against our four baselines and their oracle combination. Results are in terms of BLEU. Brevity Penalty component of BLEU is applied on the set level instead of the sentence level; therefore, the combination of results of two subsets of a set $x$ may not reflect the BLEU we get on $x$ as a whole set. Our classifier does not know of these subsets, it runs on the set as a whole; therefore, we repeat its results in the second column for convenience.

## 6.5 Discussion of Dev Set Subsets

We next consider the performance on different subsets of the Dev set: DA vs MSA as well as finer grained distinctions: Egyptian, Levantine, MSA for newswire (more formal) and

| | Egy. | Lev. | MSA-NW | MSA-WB |
|---|---|---|---|---|
| **Sample Size / Sub-Set Size** | 75/614 (14%) | 75/532 (12%) | 50/293 (17%) | 50/363 (14%) |
| **Classifier Selection** | **Egy.** | **Lev.** | **MSA-NW** | **MSA-WB** |
| Best MT system | 40% | 56% | 66% | 38% |
| 2nd-best MT system | 23% | 26% | 18% | 20% |
| 3rd-best MT system | 24% | 13% | 14% | 28% |
| Worst MT system | 13% | 5% | 2% | 14% |
| **Manual analysis of bad choices in Egy. and MSA-WB** | | | | |
| **Error Reason** | **Egy.** | **Error Reason** | | **MSA-WB** |
| Unfair BLEU | 40% | Highly dialectal | | 5% |
| MSA w/ recent terms | 19% | Code switching | | 5% |
| Blog/Forum MSA | 11% | Blog punctuation | | 33% |
| Code switching | 15% | Blog style writing | | 47% |
| Classif. bad choice | 15% | Classif. bad choice | | 10% |

Table 6.6: Error analysis of 250 sentence sample of the Dev set. The first part of the table shows the dialect and genre breakdown of the sample. The second part shows the percentages of each sub-sample being sent to the best MT system, the second best, the third best, or the worst. When the classifier selects the third our the fourth best MT system for a given sentence, we consider that a bad choice. We manually analyze the bad choices of our classifier on the hardest two sub-samples (Egyptian and MSA Weblog) and we identify the reasons behind these bad choices and report on them in the third part of the table.

MSA for weblogs (less formal). Table 6.5 summarizes the results in the Dev set (under column **All**) and provides the results on the various subsets of the Dev set. We remind the reader that our problem assumes that we do not know the dialect or genre of the sentences and that breakdown provided here is only part of analyzing the results. Similarly, all the oracle numbers provided (Row 6 in Table 6.5) are for reference only.

### 6.5.1 DA versus MSA Performance

The third and fourth columns in Table 6.5 show system performance on the DA and MSA subsets of the Dev set, respectively. The best single baseline MT system for DA is *MSA-Pivot* has a large room for improvement given the oracle upper bound (4.8% BLEU absolute). However, our best system combination approach improves over *MSA-Pivot* by a small margin of 0.2% BLEU absolute only, albeit a statistically significant improvement.

The MSA column oracle shows a smaller improvement of 2.1% BLEU absolute over the best single *MSA-Only* MT system. Furthermore, when translating MSA with our best system combination performer we get the same results as the best baseline MT system for MSA even though our system does not know the dialect of the sentences a priori.

If we consider the breakdown of the performance in our best overall (33.9% BLEU) single baseline MT system (*MSA-Pivot*), we observe that the performance on MSA is about 3.6% absolute BLEU points below our best results; this suggests that most of the system combination gain over the best single baseline is on MSA selection.

### 6.5.2 Analysis of Different Dialects

The last four columns of Table 6.5 show the detailed results on the different dialects and MSA genres in our data.

#### Genre performance analysis

The MSA portions are consistently best translated by *MSA-Only*. The results suggest that the weblog data is significantly harder to translate than the newswire (44% vs. 56.2% BLEU). This may be attributed to the train-test domain-mismatch where the MSA MT system *MSA-Only* training data is mostly newswire. The system combination yields the same results as the baseline best system for the MSA data within genre.

#### DA specific performance analysis

*DA-Only* is the best system to translate Levantine sentences which is similar to the findings by Zbib et al. (2012). However, this Levantine eval set is highly similar to the Levantine portion of the DA training data (BBN/LDC/Sakhr Arabic-Dialect/English Parallel Corpus) since both of them were collected from similar resources, filtered to be highly dialectal, and translated using the same technique (Amazon MTurk) (Zbib et al., 2012). This can

explain the large improvement in performance for the *DA-Only* system vis-a-vis all the other systems including the best performer, the combination system. The best baseline MT system to translate Egyptian is *MSA-Pivot* which gives a statistically significant 0.4% BLEU improvement over the second best system. Our best performer improves over the best single MT system by a statistically significant 0.2% BLEU. In general, we note that the performance on Egyptian is higher than on Levantine due to the bigger proportions of Egyptian training data compared to Levantine data for the single baseline MT systems and due to the fact that Egyptian sets have two references while Levantine sets have only one.

We can conclude from the above that it is hard to pick one MT system to translate Arabic sentences without knowing their dialect. However, if we know the dialect of an Arabic text, an MSA-trained MT system is sufficient to translate MSA sentences given the abundance of MSA parallel data. For dialectal sentences, it seems reasonable to build multiple systems that leverage different data settings with various complementarities while also leveraging explicit usage of automatic dialect identification system features to decide among them.

| | |
|---|---|
| **Source** | mA Srly w}t $wf hAlmslsl |
| **Reference** | i didn 't have time to watch that series |
| **MSA Trans** | lm ySr Aly wqt $wf h*A Almslsl |

| **MT System** | **Translation** | **Bleu** |
|---|---|---|
| *MSA-Only* | what Srly w}t hAlmslsl look | 2.4 |
| *DA-Only* | what happen to me when i see series | 6.4 |
| *DA+MSA* | what happened to me when i see series | 6.4 |
| *MSA-Pivot* | **did not** insist **time to look at this series** | **10.6** |

Table 6.7: System combination example in which our predictive system selects the right MT system. The first part shows a Levantine source sentence, its reference translation, and its MSA translation using the DA-MSA MT system. The second part shows the translations of our four MT systems and their sentence-level BLEU scores.

## 6.6 Error Analysis

We present a detailed error analysis on the different dialects and genres and we discuss the output of the different systems on an example sentence.

### 6.6.1 Manual Error Analysis

We performed manual error analysis on a Dev set sample of 250 sentences distributed among the different dialects and genres. The first part of Table 6.6 provides sample size and percentage to the sub-set size. The second part reports the percentage of our best performing system combination predictive system sending sentences of these sub-samples to the best, the second best, the third best, and the worst MT system. The percentages in each column sum to 100% of the sample of that column's dialect or genre. The Levantine and MSA News Wire sentences were easy to classify while Egyptian and MSA Weblog ones were harder. We did a detailed manual error analysis for the cases where the classifier failed to predict the best MT system. The sources of errors we found cover 89% of the cases. In 21% of the error cases, our classifier predicted a better translation than the one considered gold by BLEU due to BLEU bias, e.g., severe sentence-level length penalty due to an extra punctuation in a short sentence. Also, 3% of errors are due to bad references, e.g., a dialectal sentence in an MSA set that the human translators did not understand.

A group of error sources resulted from MSA sentences classified correctly as *MSA-Only*; however, one of the other three systems produced better translations for two reasons. First, since the MSA training data is from an older time span than the DA data, 10% of errors are due to MSA sentences that use recent terminology (e.g., Egyptian revolution 2011: places, politicians, etc.) that appear in the DA training data. Also, web writing styles in MSA sentences such as blog style (e.g., rhetorical questions), blog punctuation marks (e.g., "..", "???!!"), and formal MSA forum greetings resulted in 23%, 16%, and 6% of the cases, respectively.

Finally, in 10% of the cases our classifier is confused by a code-switched sentence, e.g., a dialectal proverb in an MSA sentence or a weak MSA literal translation of dialectal words and phrases. Some of these cases may be solved by adding more features to our classifier, e.g., blog style writing features, while others need a radical change to our technique such as word and phrase level dialect identification for MT system combination of code-switched sentences.

### 6.6.2 Example

Table 6.7 shows an interesting example in which our system combination classifier predicts the right system (*MSA-Pivot*). In this highly-Levantine sentence, the MSA system, as expected, produces three OOV words. The *DA-Only* and *DA+MSA* systems produce a literal translation of the first two words, drops an OOV word, and partially translate the last word. ELISSA confidently translates two words and a two-word phrase to MSA correctly. This confidence is translated into features used by our classifier which helped it predict the *MSA-Pivot* system.

## 6.7 Conclusion and Future Work

This chapter proves that the different MT approaches of MSA-pivoting and/or training data combinations for DA-to-English MT complement each other in interesting ways and that the combination of their selections could lead to better overall performance by benefiting from the strengths while avoiding the weaknesses of each individual system. This is possible due to the diglossic nature of the Arabic language.

We presented a sentence-level classification approach for MT system combination for diglossic languages. Our approach uses features on the source language to determine the best baseline MT system for translating a sentence. We get a 1.0% BLEU improvement over the best baseline single MT system.

In the future we plan to add more training data to see the effect on the accuracy of system combination. We plan to give different weights to different training examples based on the drop in BLEU score the example can cause if classified incorrectly. We also plan to explore confusion-network combination and re-ranking techniques based on target language features.

# Part III

# Scaling to More Dialects

# Chapter 7

# Unsupervised Morphological Segmentation for Machine Translation

## 7.1 Introduction

Resource-limited, morphologically-rich languages impose many challenges to Natural Language Processing (NLP) tasks since the highly inflected surface forms of these languages inflate the vocabulary size and, thus, increase sparsity in an already scarce data situation. Therefore, NLP in general and Machine Translation (MT) in particular can greatly benefit from unsupervised learning approaches to vocabulary reduction such as unsupervised morphological segmentation.

Dialectal Arabic (DA), the unspoken varieties of Arabic, is a case study of such languages due to its limited parallel and task-specific labeled data, and the large vocabulary caused by its rich inflectional morphology and unstandardized spontaneous orthography. Furthermore, the scarcity of DA parallel and labeled text is more pronounced when considering the large number of dialects and sub-dialects, the varying levels of dialectness and code switching, the diversity of domains and genres, and the timespan of the collected text. Hence, the need for unsupervised learning solutions to vocabulary reduction that use a

more sustainable and continuously fresh source of training data arises. One such source is the enormous amount of monolingual text available online that can be acquired on a daily basis across different dialects and in many genres and orthographic choices. Additionally, building or extending supervised NLP systems on the different dimensions mentioned above requires approaches to automatically creating labeled data for such tasks.

In this work we utilize huge collections of monolingual Arabic text along with limited DA-English parallel data to improve the quality of DA-to-English machine translation. We propose an unsupervised learning approach to morphological segmentation consisting of three successive systems. The first system uses word embeddings learned from huge amounts of monolingual Arabic text to extract and extend a list of possible segmentation rules for each vocabulary word and scores these rules with an Expectation Maximization (EM) algorithm. The second system uses the learned segmentation rules in another EM algorithm to label select DA words in DA-English parallel text with the best segmentation choice based on the English alignments of the word segments. Finally, the third system implements a supervised segmenter by training an Averaged Structured Perceptron (ASP) on the automatically labeled text. The three systems can be used independently for other purposes. We evaluate the performance of our segmenter intrinsically on a portion of the labeled text, and extrinsically on MT quality.

## 7.2 Related Work

In this section we review the literature on supervised and unsupervised learning approaches to morphological segmentation.

### 7.2.1 Supervised Learning Approaches to Morphological Segmentation

Supervised learning techniques, like MADA, MADA-ARZ and AMIRA (Habash and Rambow, 2005; Habash et al., 2013; Diab et al., 2007; Pasha et al., 2014), have performed well on the task of morphological tokenization for Arabic machine translation. They require hand-crafted morphological analyzers, such as SAMA (Graff et al., 2009), or at least annotated data to train such analyzers, such as CALIMA (Habash et al., 2012c), in addition to treebanks to train tokenizers. This is expensive and time consuming; thus, hard to scale to different dialects.

### 7.2.2 Unsupervised Learning Approaches to Morphological Segmentation

Given the wealth of unlabeled monolingual text freely available on the Internet, many unsupervised learning algorithms (Creutz and Lagus, 2002; Stallard et al., 2012; Narasimhan et al., 2015) took advantage of it and achieved outstanding results, although not to a degree where they outperform supervised methods, at least on DA to the best of our knowledge. Traditional approaches to unsupervised morphological segmentation, such as MORFESSOR (Creutz and Lagus, 2002; Creutz and Lagus, 2007), use orthographic features of word segments (prefix, stem, and suffix). However, many researchers worked on integrating semantics in the learning of morphology (Schone and Jurafsky, 2000; Narasimhan et al., 2015) especially with the advances in neural network based distributional semantics (Narasimhan et al., 2015).

In this work, we leverage the use of both approaches. We implement an unsupervised learning approach to automatically create training data which we use to train supervised algorithms for morphological segmentation. Our approach incorporates semantic informa-

tion from two sources, Arabic (through monolingual data) and English (through parallel data), along with linguistic features of the source word and its target segments to learn a morphological segmenter.

## 7.3 Approach

A typical supervised context-sensitive tokenization approach (Figure 7.1) depends on the existence of a morphological analyzer to provide a list of out-of-context analyses for each word in a sentence (Graff et al., 2009; Habash et al., 2012a). Using this analyzer, the system can turn an input sentence into a sausage lattice of analyses that can be decoded using a context-sensitive model trained on a manually annotated treebank (Habash and Rambow, 2005; Habash et al., 2013; Pasha et al., 2014). For machine translation purposes, the best ranking path in the lattice can then be tokenized into surface form tokens according to a tokenization scheme that was chosen to maximize alignment to the foreign language. Many researchers have explored ways to come up with a good tokenization scheme for Arabic when translating to English (Maamouri et al., 2004; Sadat and Habash, 2006). While SMT systems typically use one tokenization scheme for the whole Arabic text, Zalmout and Habash (2017) experimented with different tokenization schemes for different words in the same Arabic text. Their work showed that different target languages require different source language tokenization schemes. It also showed that combining different tokenization options while training the SMT system improves the overall performance, and considering all tokenization options while decoding further enhances the performance.

Inspired by the typical supervised tokenization approach discussed above, our approach to unsupervised morphological segmentation consists of two stages: first, we automatically create labeled data, and second, we train a supervised segmenter on it:

1. **Unsupervised labeling of segmentation examples.** To automatically label words with their desired segmentations we use both monolingual and parallel text. This stage involves two systems:

    a) A system that learns Arabic segmentation rules from monolingual text using distributional semantics (Section 7.4). This system is analogous to a morphological analyzer in that it produces *out-of-context* segmentation options.

    b) A system that labels Arabic words in the parallel text with their best segmentation rules using the English words to which the Arabic word is aligned (Section 7.5). This system is effectively incorporating English semantics in choosing the best *in-context* segmentation of an Arabic word in a sentence.

2. **Supervised segmentation.** Starting from the automatically labeled data created by the previous stage, we train a tagger that learns to score all possible segmentations for a given word in a sentence (Section 7.6).

One challenge to this approach is that the automatic labeling of words will introduce errors that will affect the quality of the supervised segmenter. To reduce the number of errors in the automatically labeled data we only label words when the system has a high confidence in its decision. This will result in many unlabeled words in a given sentence that raises another challenge to the supervised segmenter which we solve by modifying the training algorithm. The underlying assumption of this approach is that if the unsupervised labeling process does not cover all words in the vocabulary, the supervised segmenter will learn to generalized to the missed words and OOVs.

We evaluate this approach on two Arabic dialects: Egyptian and Levantine (the collection of Syrian, Lebanese, Jordanian, and Palestinian dialects). The following three sections discuss the three systems used in this approach and present the experimental setup, examples and discussion. The last of the three sections, Segmentation (Section 7.6), also present

the evaluation of the segmenter's accuracy on the automatically labeled data produced by the first stage.

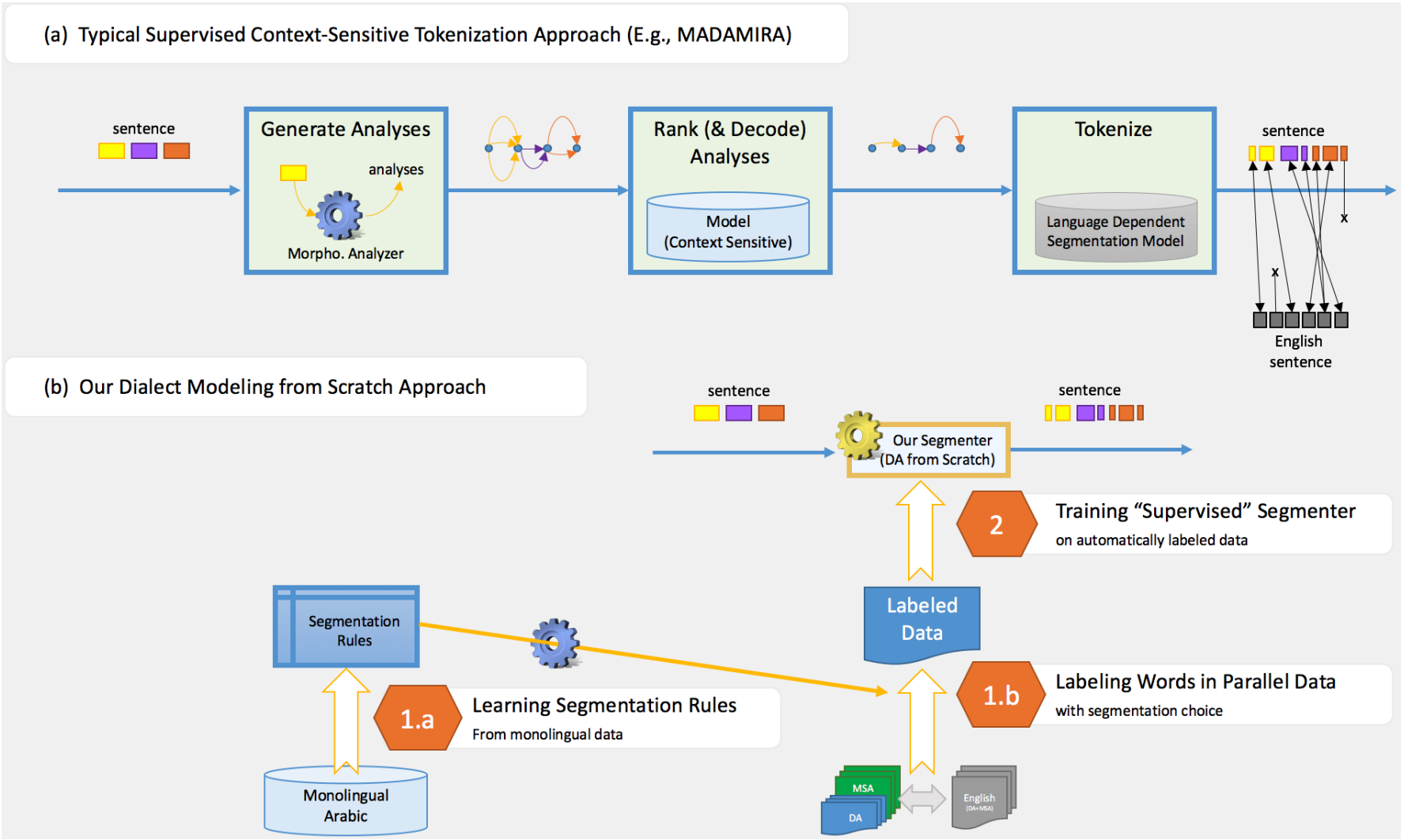



Figure 7.1: Our unsupervised segmenation approach (part (b)) in contrast to a typical supervised tokenization appraoch (part (a)).

## 7.4 Monolingual Identification of Segmentation Rules

In this section we discuss the approach we use to learn segmentation rules from monolingual data. The approach consists of three steps: word clustering, rule learning, and rule scoring.

### 7.4.1 Clustering based on Word Embeddings

We learn word vector representations (word embeddings) from huge amounts of monolingual Arabic untokenized text using Word2Vec (Mikolov et al., 2013). For every Arabic word $x$, we then compute the closest $N$ words using cosine distance (with cosine distance above threshold $D_a$). We consider these $N$ words to be $x$'s semantic cluster. Every cluster has a source word. It is important to mention that a word $x$ might appear in $y$'s cluster but not vice versa.

After a manual error analysis of the data, we picked a high cosine distance threshold $D_a = 0.35$ since we need only words that we have high confidence in their belonging to a cluster in order to produce high quality segmenation rules. This high threshold results in many words having small or even empty clusters:

1. The **small cluster** problem means that the cluster's main word will not have enough segmentation rules which means it might not have the final stem we hope to segment to (e.g., the word ماتجوزت *mAtjwzt* 'I-did-not-marry' might have ماتجوز *mAtjwz* 'did-not-marry' but not اتجوز *Atjwz* 'married', which is the desired stem). We attempt to solve this problem with a rule expansion algorithm discussed in the Section 7.4.2.

2. The **empty cluster** problem happens when the closest word to the cluster's main word is beyond the distance threshold. This means that we will not have any labeled example for this word; hence, it will be an OOV word for supervised segmenter. We design the segmenter so that it generalizes to unseen words.

Even with this high threshold, many words end up with **very large clusters** due to Word2Vec putting thousands of certain types of words very close in the vector space (e.g., proper names, words that appeared only few times in the training data). This adds noise to the rule scoring training data discussed later;. We solve this problem by deciding on a maximum cluster size $N = 200$.

### 7.4.2 Rule Extraction and Expansion

#### Rule Extraction

We extract segmentation rules from the clusters learned previously. For every word $x$, for every word $y$ in $x$'s cluster where $y$ is a substring of $x$ we generate a segmentation rule $x \rightarrow p$+ $y$ -$q$ where $y$ is the stem, $p$+ is the prefix (the substring of $x$ before $y$ starts), and -$q$ is the suffix (the substring of $x$ after $y$ ends). A rule might have an empty prefix or suffix denoted as P+ and -Q, respectively. If $y$ happens to appear at different indices inside $x$, we generate multiple rules (e.g., $x$ is 'hzhzt' and $y$ is 'hz' we produce 'hzhzt $\rightarrow$ P+ hz -hzt' and 'hzhzt $\rightarrow$ hz+ hz -t').

We define a function $dist(x \rightarrow y)$ as the cosine distance between words $x$ and $y$ if there is a rule from $x$ to $y$, equal to 1 if $x = y$, and 0 otherwise.

$$dist(x \rightarrow y) = \begin{cases} cosineDistance(x, y), & \text{if } \exists\ x \rightarrow p\text{+ } y\text{ -}q \\ 1, & \text{if } x = y \\ 0, & \text{otherwise} \end{cases} \tag{7.1}$$

#### Rule Expansion

Given the high cluster admission threshold, we consider expanding the rule set by adding further segmentations. We build an acyclic directed graph from all the extracted rules where words are nodes and rules are edges. Since a rule is always directed from a longer

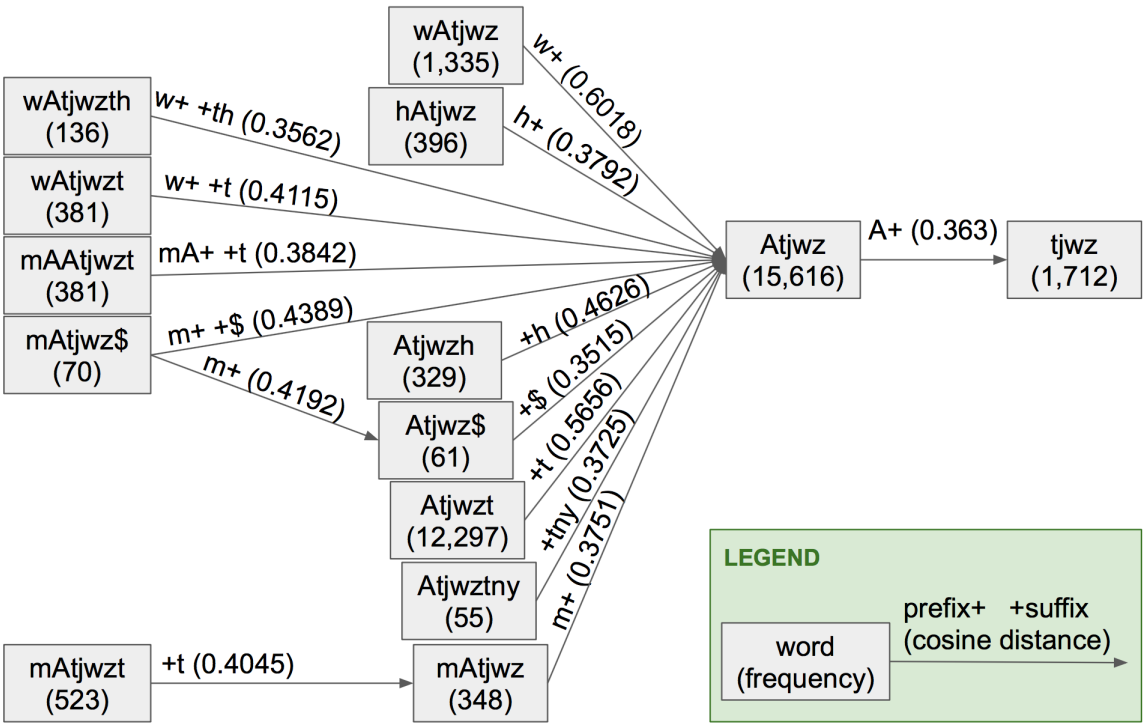



Figure 7.2: Example of a segmentation graph that leads to the word *Atjwz* 'I marry / he married'. The frequencies of the words are enclosed in paranthesis inside the nodes. The cosine distance and split affixes are on the arcs.

word to a shorter one, the graph will not have cycles and will not have very long paths. Figure 7.2 shows a partial segmentation graph built from some rules that lead to the word اتجوز *Atjwz* 'I marry / he married'.

We then expand the rules by generating a rule $x \rightarrow y$ for every node $x$ that has a path to node $y$ in the graph. To do so we recursively scan the graph starting from leaves (words that cannot be segmented) all the way back to the beginning of the graph, and add a rule $x \rightarrow y$ to the graph if there are two rules $x \rightarrow w$ and $w \rightarrow y$ where $w$ is a word. The recursive scan insures that we fully expand $w$ and add its outgoing edges to the graph before we expand $x$. It also allows us to compute a **confidence score** $conf(x \rightarrow y)$ that starts with the cosine distance between $x$ and $y$ and will increase the more paths we find between them:

$$conf(x \rightarrow y) = dist(x \rightarrow y) + \sum_{w} conf(x \rightarrow w) \times conf(w \rightarrow y) \tag{7.2}$$

**Projected Frequency**

The main reason for segmentation is to reduce the vocabulary size and thus increase word frequencies which improves the quality of any subsequent statistical system. However, chopping affixes off a word (as opposed to clitics) may affect the integrity of the word; for example it may slightly change the meaning of the word or may cause it to align to more general words (e.g., segmenting 'parking' to 'park -ing' in English may negatively affect alignment depending on the foreign language). Additionally, every time we make a segmentation decision we may introduce errors. Therefore, it is important to know at what point we do not need to segment a word anymore. To do so we consider word frequencies as part of scoring the rules because frequent words are likely to be aligned and translated correctly to their inflected English translations without the need for segmentations. For example, in Figure 7.2 we might not want to segment "Atjwz" to "tjwz" considering its high frequency and the number and frequencies of words that lead to it. We define a **projected frequency score** of a word as:

$$pf(y) = \sum_{x \in V} conf(x \rightarrow y) \times log(freq(x)) \tag{7.3}$$

where $x$ is any word in the vocabulary $V$ and $log(freq(x))$ is the logarithm of $x$'s frequency. We use logarithms to smooth the effect of frequent words on the final score to better represent $y$ as a hub for many words. Note that $conf(y \rightarrow y) = 1$ and thus we include $log(freq(y))$ in the score.

### Word-to-Stem Score

Given the projected frequency scores, we compute the ratio $pf(y)/pf(x)$ that represents the gain we get from segmenting $x$ to $y$. For example, if $x$ is frequent and has many words that lead to it, and $y$ gets most of its projected frequency from $x$, then the ratio will be small. But if $x$ is infrequent or not a hub and $y$ has a high $pf$ score through other sources, then the ratio will be high. Now, we can compute the final **word-to-stem score**, which we will be using next, as follows:

$$a2s(a \rightarrow s) = conf(a \rightarrow s) \times pf(s)/pf(a) \tag{7.4}$$

### Fully Reduced Words

All the rules extracted so far will segment a word to a shorter word with at least one affix. The automatic annotations need to have some examples where words do not get segmented in order for the segmenter to learn such cases. Therefore, we need to identify a list of words that cannot be segmented and thus produce rules that transform a word to itself with no affixes. Such rules will be of the form $x \rightarrow$ P+ $x$ -Q. The list does not have to be complete; it just needs to be of high confidence. To generate the list, we consider words that appear on the target side of rules but never on the source side. We then reduce the list to only frequent stems (with over 3000 occurrences) that have at least 3 words that can be segmented to them, which gives us enough confidence as we have seen them in various contexts and they have appeared in at least 3 clusters but yet do not have any substrings in their own clusters. These thresholds are determined empirically.

We compute the word-to-stem score for these fully reduced rules using this equation:

$$a2s(a \rightarrow a) = \sum_{x} conf(x \rightarrow a) \tag{7.5}$$

**Algorithm 1** Affix-stem joint probability esitmation.

*// Initialization*:

$v(p,s) \leftarrow \frac{\sum_{a \rightarrow psq} a2s(a \rightarrow s)}{\sum_{a \rightarrow p's'q} a2s(a \rightarrow s')}$ **for all** $p, s$

$u(q,s) \leftarrow \frac{\sum_{a \rightarrow psq} a2s(a \rightarrow s)}{\sum_{a \rightarrow ps'q'} a2s(a \rightarrow s')}$ **for all** $q, s$

*// Estimation*:

**for** $round := 1 \rightarrow MAX$ **do**

    *// Collect counts:*

    **for each** rule $a \rightarrow psq$ **do**

        $c_v(p,s) = \sum_{p'} \sum_{s'} v(p,s') \times v(p',s')$

        $c_u(q,s) = \sum_{q'} \sum_{s'} u(q,s') \times u(q',s')$

        $\delta \leftarrow a2s(a \rightarrow s) \times c_v(p,s) \times c_u(q,s)$

        $count_v(p,s)$ **+=** $\delta$

        $count_u(q,s)$ **+=** $\delta$

        $total$ **+=** $\delta$

    *// Estimate joint probabilities:*

    $v(p,s) \leftarrow count_v(p,s)/total$ **for all** $p, s$

    $u(q,s) \leftarrow count_u(q,s)/total$ **for all** $q, s$

*// Calculate rule scores*:

$score(a \rightarrow psq) = a2s(a \rightarrow s) \times v(p,s) \times u(q,s)$ **for all** rules $a \rightarrow psq$

### 7.4.3 Learning Rule Scores

Since a rule $a \rightarrow psq$ produces three segments: a stem $s$, a prefix $p$, and a suffix $q$, we define its score as the product of the word-to-stem score $a2s(a \rightarrow s)$, the joint probability of the prefix and the stem $v(p,s)$, and the joint probability of the suffix and the stem $u(q,s)$.

$$score(a \rightarrow psq) = a2s(a \rightarrow s) \times v(p,s) \times u(q,s) \tag{7.6}$$

#### Affix Correlation

To estimate the correlation between a prefix $p$ and a prefix $p'$, we iterate over all stems $s'$ and compute:

$$corr_{pref}(p',p) = \sum_{s'} v(p,s') \times v(p',s') \tag{7.7}$$

This score indicates the similarity in morpho-syntactic behavior of these two prefixes. For example, the Egyptian prefix +ه *h+* 'will' and the MSA prefix +وس *ws+* 'and will' attach to present tense verbs; therefore, we would expect them to share many of the stems in the rules they appear in, which leads to a high correlation score.[1]

We similarly define suffix correlation as:

$$corr_{suff}(q', q) = \sum_{s'} u(q, s') \times u(q', s') \tag{7.8}$$

**Affix-Stem Correlation**

Using these affix correlation scores, we can estimate the correlation between a prefix $p$ and a stem $s$ by iterating over all prefixes $p'$ that we have seen with $s$ in a rule and summing their correlation scores with $p$.

$$\begin{aligned} c_v(p, s) &= \sum_{p'} corr_{pref}(p', p) \\ &= \sum_{p'} \sum_{s'} v(p, s') \times v(p', s') \end{aligned} \tag{7.9}$$

We similarly define suffix-stem correlation as:

$$\begin{aligned} c_u(q, s) &= \sum_{q'} corr_{suff}(q', q) \\ &= \sum_{q'} \sum_{s'} u(q, s') \times u(q', s') \end{aligned} \tag{7.10}$$

[1]In practice, we iterate over the $N$ stems with the highest $v(p, s')$ values for a prefix because some prefixes, like +و *w+* 'and', attach to tens of thousands of stems and that unnecessarily slows the algorithm. We found $N = 500$ to be fast and provide good results.

**Affix-Stem Joint Probabilities**

Given affix-stem correlation scores, we can define the prefix-stem joint probability, $v(p, s)$, and the suffix-stem joint probability, $u(q, s)$, as follows:

$$\begin{aligned} v(p,s) &= \frac{c_v(p,s)}{\sum_{p',s'} c_v(p',s')} \\ u(q,s) &= \frac{c_u(q,s)}{\sum_{q',s'} c_u(q',s')} \end{aligned} \tag{7.11}$$

To estimate the parameters in these *recursive* equations we implement the expectation maximization (EM) algorithm shown in Algorithm 1. The initialization step uses only word-to-stem scores computed earlier; i.e., it is equivalent to the first round in the following EM loop with the exception that $\delta \leftarrow a2s(a \rightarrow s)$. We found that running the EM algorithm for fifty rounds provides good results.

### 7.4.4 Experiments

**Experimental Setup.**

We use two sets of monolingual Arabic: about 2 billion tokens from Arabic GigaWord Forth Edition which is mainly MSA, and about 400 million tokens of Egyptian text, resulting in about 2.4B tokens of monolingual Arabic text used to train Word2Vec (Mikolov et al., 2013) to build word vectors.

In this work, we did not have access to a sizable amount of Levantine text to add to the monolingual data. Access to Levantine text would help this task learn Levantine segmentation rules and thus hopefully improve the final system. We did not want to use the Levantine side of the parallel data to keep this system separate from the second system to avoid any resulting biases.

## 7.5 Alignment Guided Segmentation Choice

In the previous section we learned and scored segmentation rules for words out of context. In this section we use these rules and their scores to learn *in-context segmentations* of words guided by their English alignments. The premise of this approach is that if we find enough Arabic words where we are confident in their segmentation choices in-context given the English translation, then we can use those segmentation choices as labeled data to train a supervised segmenter.

### 7.5.1 Approach

Unsupervised learning of word alignments from a parallel corpus is pretty much established. A tool like Giza++ (Och and Ney, 2003b) can be run on the Arabic-English parallel data to obtain many-to-many word alignments. That means, each Arabic word aligns to multiple English words, and each English word aligns to multiple Arabic words. However, these algorithms look at the surface form without considering morphological inflections.

Our alignment algorithm concerns with aligning the internal structure of Arabic words (the rule segments) to their English translations. We start by running Giza++ on our Arabic-English parallel corpora to obtain initial, surface form alignments. Then, we consider one-to-many aligned pairs $\langle a_i, E_{a_i} \rangle$, where $a_i$ is an Arabic word at position $i$ and $E_{a_i} = (e_1, e_2, ..., e_{|E_{a_i}|})$ is the sequence of English words aligned to $a_i$ ordered by their position in the English sentence. Since the Arabic side of the parallel data is unsegmented, the plethora of inflected words will dramatically extend the vocabulary size and the Zipfian tail of infrequent words, which will negatively affect parameter estimation in Giza++ resulting in many inaccurate alignments. To reduce the effect of this problem on our algorithm, we expand the definition of $E_{a_i}$ to also include surrounding words of the words aligned to $a_i$ by Giza++. The order of the English words is preserved. We model dropping words from $E_{a_i}$ in our alignment model.

Given an aligned pair $\langle a_i, E_{a_i} \rangle$ where $a_i$ has a set of segmentation rules $R = \{r : r = a_i \rightarrow g_1 g_2 g_3\}$, we estimate an **alignment probability** for every rule $r$ based on aligning its segments to words in $E_{a_i}$. We, then, pick the rule with the highest probability, $r^*$, as the segmentation choice for $a_i$ in that context. It is important to note that the context here is determined by the English translation instead of the surrounding Arabic words. Note that the rule itself has a score derived from the Arabic context of the word through word embedding. Therefore, if we incorporate the rule score in the alignment probability model, we can combine Arabic semantics and English semantics in determining the segmentation choice in context.

### 7.5.2 The Alignment Model

In order to compute an alignment probability for every pair $(r, E_{a_i})$, we need to estimate how $r$'s segments translate to $E_{a_i}$'s tokens. To translate a source text sequence to a sequence in a target language, two main questions must be asked:

1. What target words/phrases should we produce?

2. Where should we place them?

#### Motivation: IBM Models

This subsection provides a quick introduction to IBM Models to give a general motivation to our proposed alignment model. Details that do not relate to our model are not discussed. For detailed discussion of IBM Models, refer to (Brown et al., 1993).

IBM Model 1 answers *only* the first question by introducing a **lexical translation model**, $t(e_i|f_j)$, that estimates how well a source token $e_i$ translates to a target token $f_j$. IBM Model 1 does not model word alignments explicitly which means once the target words are generated, they can be put in any order in the target sentence. To answer the

second question, IBM Model 2 adds an **absolute alignment model**, $a(i|j, m, n)$, that measures the probability of a target token at position $j$ in a target sentence of length $m$ to be aligned to a source word at position $i$ in a source sentence of length $n$. This independent modeling of translation and alignment makes the problem easier to solve.

IBM Model 3 takes the first question a step further by modeling fertility which allows source words to produce multiple target words or even get dropped from translation, and allows target words to be inserted without a source word generating them. Fertility gets handled by two separate models:

1. The **fertility model**, $y(n_{slots}|f)$, handles source word fertility by estimating the probability of a source word $f$ to produce zero or more slots to be filled with target words. If $n_{slots} = 0$, source word $f$ will be dropped from translation. If $n_{slots} > 0$, one or more target words will be generated.

2. **NULL insertion** models the introduction of new target words without a source translation.

While IBM Model 3 keeps the regular lexical translation model as is: $t(e_i|f_j)$, it reverses the direction of Model 2's absolute alignment model to become $d(j|i, m, n)$, which they call an **absolute distortion model**.

IBM Model 4 further improves Model 3 by introducing a **relative distortion model** which allows target words to move around based on surrounding words instead of the length of source and target sentences. Finally, IBM Model 5 fixes the deficiency in Model 3 and 4 that allows multiple target words to be placed in the same position.

### Our Alignment Probability Model

IBM Models were originally proposed as machine translation systems, but now they are widely used as part of word alignments as more advanced machine translation approaches were introduced. While our alignment model is inspired by IBM Models, we have no

intention to use it as an MT system; therefore, we are not bound to design an alignment model that generates fluent translations. In other words, the placement of English words in their *exact* positions (the second question) is not essential. Our model should measure how well a certain segmentation of an Arabic word $a_i$, produced by rule $r = a_i \rightarrow g_1 g_2 g_3$, aligns to English words in the target sequence $E_{a_i}$.

The English sequence could contain words that do not align to any segment of the source Arabic word. This is a result of erroneous alignments by Giza++ or due to our inclusion of surrounding words. To handle this, we model dropping words from $E_{a_i}$ by introducing a **NULL token** on the Arabic side (with index 4) that misaligned English words can align to. This makes the Arabic sequence of length 4, indexed: 1 for prefix, 2 for stem, 3 for suffix, and 4 for the NULL token. We use the variable $j$ to refer to this index. The English sequence can be of any length, denoted as $m$. As mentioned above, the original order of English words is preserved in the sequence $E_{a_i}$, but re-indexed from 1 to $m$. We use the variable $k$ to refer to this index.

**Definition: Alignment Vector.** An **alignment vector** is a vector of $m$ elements denoted as $L = (l_1, l_2, ..., l_m)$, where $l_k$ is position in the Arabic segment that $e_k$ aligns to. This allows multiple English words can align to the same token in the Arabic sequence; e.g., 'and' and 'will' can align to +وح *wH+* 'and will'. However, an English word cannot align to multiple Arabic tokens, which forces the English word to pick its best aligned Arabic token. We define $L_{(r,E_{a_i})}$ as the set of all possible alignment vectors that align $E_{a_i}$'s words to $r$'s segments and the NULL token.

**Definitions: Center and Direction.** We define the **center** of the English sequence as the English word that best aligns to the Arabic stem. We denote its index as $k_{stem}$. Given $k_{stem}$, we define a **direction vector** $D = \{d_k : d_k = \textbf{sgn}(k - k_{stem})\}$[2], where every English word

[2] **sgn** is the sign function.

$e_k$ has a direction $d_k$ relative to the center $e_{k_{stem}}$. This means that the center $e_{k_{stem}}$ has a direction of 0, words that appear *before* the center have a direction of –1, and words that appear *after* center have a direction of +1,

It is intuitive to assume that the direction of an English word relative to the center could have an impact on the decision of whether to align it to a prefix, a stem, a suffix, or even NULL to drop it from alignment as it might align to a previous or subsequent word in the Arabic sentence. To motivate this intuition let us observe *closed-class* and *open-class* English words and their relations to the types of Arabic segments based on their directions from the center.

In our approach to segmentation, an Arabic affix is split from the stem as one unit without further splitting its internal components which could contain pronouns, prepositions, or particles such as conjunction, negation, and future particles. These affixes tend to align to closed-class English words. For example, the Arabic definite article, +ال *Al*+ 'the', appears only in prefixes (e.g., in +وال *wAl*+ 'and the'); similarly, the English word 'the' appears only *before* the center when it aligns to a prefix. If 'the' appears *after* the center, it probably should be aligned to a subsequent Arabic word in the source sentence. Moreover, the Arabic conjunction particle, +و *w*+ 'and', appears in prefixes (e.g., in +وح *wH*+ 'and will') or as a separate word و *w*; therefore, when 'and' appears *before* or *at* the center it tends to align to a prefix or a stem, respectively. If 'and' appears *after* the center, it probably should be dropped. Furthermore, the English word 'to' could align to any token of the source sequence at any position in the target sequence; however, its direction relative to the center correlates with the position of the Arabic token it aligns to. Here are the four cases:

1. 'to' could align to a prefix containing the preposition[3] +ل *l*+ 'to' (as in this example attaching to a verb and a noun: "ليبعت لرفيقه" ***l****ybEt* ***l****rfyqh* '*to* send *to* his friend'). In such cases, the English word 'to' has a direction of –1.

[3]In Arabic linguistics, when *l*+ attaches to a verb, it's called a justification particle, not a preposition.

2. 'to' could align to a stem such as إلى *<lY* 'to', a separate preposition in Arabic. In these cases, 'to' has a direction of 0.

3. 'to' could align to a suffix containing the indirect object preposition لـ- *-l* 'to' (as in the suffix ولك- *-wlk* 'they, to you' in "يبعتولك" *ybEtwlk* 'they send to you'). In such cases, 'to' has a direction of +1.

4. 'to' could align to NULL if misaligned which could occur at any value for $d_k$.

Similar to closed-class words, open-class words tend to either align to the stem or to NULL. For example, there is no prefix or suffix that aligns to the word 'send'; therefore, if it appears on either side of the center, it probably belongs to a surrounding word of the current Arabic word $a_i$. This motivates the design of a probability distribution that capitalize on this correlation.

Our model answers the two questions introduced earlier with two separate probability distributions:

1. **Lexical Translation Model:** $t(e_k|g_{l_k})$**.** This model is identical to IBM Model 1. It estimates the probability of translating an Arabic segment to an English word. For example, $t(\text{'and'}|\text{'wH+'})$ represents the probability of producing the word 'and' from the prefix وح+ *wH+* 'and will'.

2. **Lexical Direction Model:** $z(l_k|e_k, d_k)$**.** This model estimates the probability of aligning an English word $e_k$ with direction $d_k$ to position $l_k$ in the Arabic sequence. For example, $z(1|\text{'and'}, -1)$ is the probability of the word 'and' aligning to a prefix knowing that it appeared *before* the center.

In this model, the *exact* position of the generated English word is not important; instead, the direction relative to center is what matters.

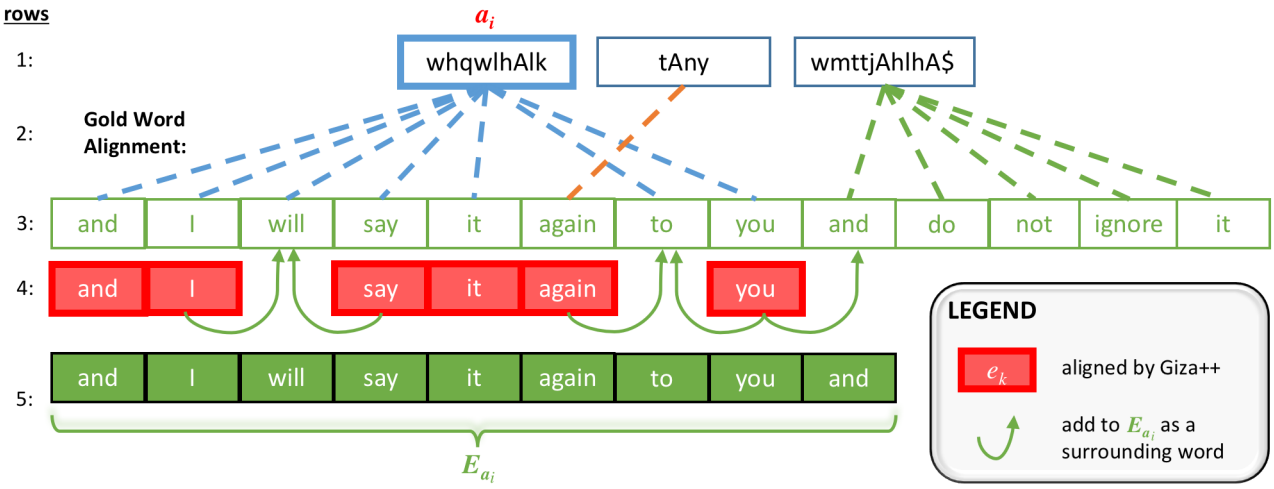



Figure 7.3: Example of sentence alignment that shows how we extract the English sequence $E_{a_i}$ that aligns to a source word $a_i$. The figure is organized as rows indexed from 1 to 5 as shown on the left margin. Row 1 and 3 show the source Arabic sentence and its English translation. Row 2 shows the perfect word-level alignment between the two sentences. Row 4 shows the automatic process of extracting $E_{a_i}$ by first adding words aligned by Giza++ (in red rectangles), and then adding surrounding words (identified by the green arrows). Row 5 shows the resulting $E_{a_i}$.

To compute $k_{stem}$ we find the English word with the highest $t \times z$ score as in the equation below.

$$k_{stem} = \arg\max_{k} t(e_k|g_2) \times z(2|e_k, 0)$$

This might seem like a circular dependency: $z$ depends on $d_k$ which is computed from $k_{stem}$ that depends on $z$. In other words, using the direction from the center while trying to find the center. In fact we do not need the direction from the center to compute $k_{stem}$. Instead, we set $d_k = 0$ in $z(2|e_k, 0)$, which, when multiplied with $t(e_k|g_2)$, basically asks the question: if word $e_k$ were to be selected as the center, how well will it align to position 2 (the stem) in the source sequence? This breaks the circular dependency.

### Example

Consider the Arabic sentence وهقولهالك تاني ومتتجاهلهاش *whqwlhAlk tAny wmttjAhlhA$* translated to English as 'And I will say it again to you and do not ignore it'. Figure 7.3

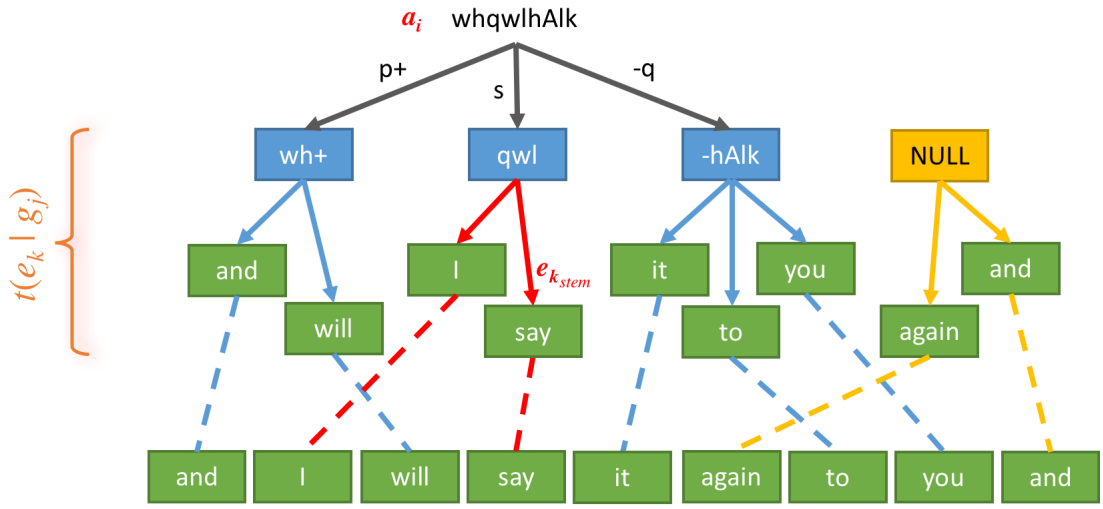
presents the two sentences in rows 1 and 3 (index is in the left margin), as well as their perfect word-level alignment (Row 2). For our example, we consider the first word, *whqwlhAlk*, as $a_i$ and we construct $E_{a_i}$ in Row 4 by first including words aligned by Giza++ (in red rectangles), and then adding surrounding words (identified by the green arrows). Row 5 shows the resulting $E_{a_i}$. Due to the infrequency of such highly-inflected words, Giza++ tends to make errors aligning them. In this case it erroneously aligns 'again' to $a_i$ and misses 'will' and 'to' which should have been aligned. Our inclusion of surrounding words results in adding the missed words, but also includes the trailing 'and' erroneously. This approach increases recall while compromising precision since it depends on the probabilistic model to maximize English alignment to $a_i$'s internal structure while dropping the misaligned English words.

Figure 7.4 shows the alignment of the Arabic word وهقولهالك *whqwlhAlk* from Figure 7.3 with its aligned English sequence $E_{a_i}$ = (and, I, will, say, it, again, to, you, and). This example shows how our model would score an **alignment vector** $L = (1, 2, 1, 2, 3, 4, 3, 3, 4)$ linking $E_{a_i}$ tokens one-to-many to the four Arabic tokens. $L$, shown in Part (b) of the Figure (index is in the left margin), is actually the gold alignment vector. Part (a) shows the **lexical translation model**, $t(e_k|g_{l_k})$, generating English words from Arabic tokens under alignment vector $L$. The English word 'say' is picked as **the center** over 'I' because $t(\text{say}|\text{qwl}) > t(\text{I}|\text{qwl})$. Part (b) shows how the **lexical direction model**, $z(l_k|e_k, d_k)$, predicts the position an English word $e_k$ with direction $d_k$ aligns to.

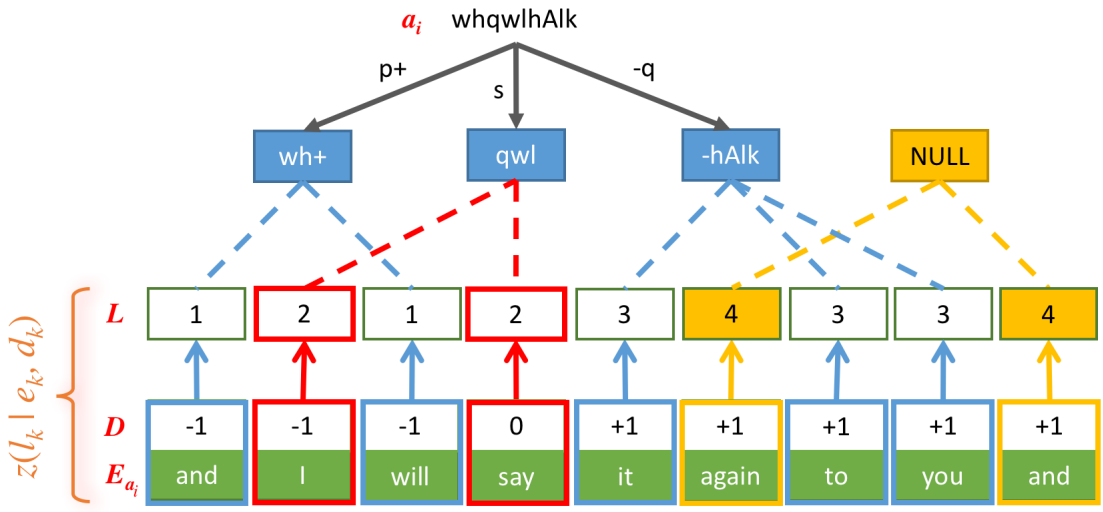
### Decoding with the Model: Finding the Best Segmentation Choice

The probability of an alignment vector $L$ that aligns the words of an English sequence, $E_{a_i}$, to the four Arabic tokens produced by a rule $r$ and the NULL token is denoted as $p_{align}(E_{a_i}, L|r)$ and is given by this equation:

$$p_{align}(E_{a_i}, L|r) = \prod_{k=1}^{m} t(e_k|g_{l_k}) \times z(l_k|e_k, d_k) \tag{7.12}$$




(a)




(b)

Figure 7.4: Example of alignment model parameters $t$ and $z$ for an Arabic word aligned to an English phrase.

To find the best segmentation choice for an Arabic word $a_i$ with a set of rules $R_{a_i}$, we pick the rule, $r^*$, that has the highest score when aligned to the English sequence $E_{a_i}$. To evaluate how well a rule $r$ aligns to $E_{a_i}$, we scan all possible alignment vectors generated from $r$ and $E_{a_i}$ to find the one with the highest probability $p_{align}(E_{a_i}, L|r)$. Therefore, the best segmentation choice for $a_i$ is generated by the rule $r^*$ that has the best alignment to $E_{a_i}$'s words among all other rules in $R_{a_i}$, as shown in the following equation.

$$r^* = \arg \max_{r \in R_{a_i}} \max_{L \in L_{(r, E_{a_i})}} p_{align}(E_{a_i}, L|r) \tag{7.13}$$

It is important to note that the rule score computed in Section 7.4, $score(r)$, is not used directly in these equations; however, it is used in estimating the model's parameters $t$ and $z$.

### 7.5.3 Parameter Estimation with Expectation Maximization

In this subsection we present our expectation maximization algorithm to estimate our model's parameters. First we start with initializing our parameters and then we explain the EM algorithm used for parameter estimation.

#### Initialization

Our initialization step starts by running Giza++ (Och and Ney, 2003b) on the untokenized Arabic-English parallel data to produce many-to-many word-level alignments. Using these alignments we estimate two word translation probability distributions for each Arabic word $a$ and English word $e$:

1. The **word translation probability**: $p_1(e|a) = count_{aligned}(a, e)/count(a)$.

2. The **reverse word translation probability**: $p_2(a|e) = count_{aligned}(a, e)/count(e)$.

Where $count_{aligned}(a, e)$ is the number of times we saw Arabic word $a$ aligned to English word $e$ in Giza++ output, while $count(x)$ is the frequency of word $x$ in the parallel corpora.

**Algorithm 2** Affix-stem joint probability esitmation.

1: *// Parameter initialization*:

$$t(e_k|g_h) = c(e_k, g_h)/\sum_{e'} c(e', g_h) \tag{7.14}$$

where $c(e_k, g_h) = p_1(e_k|g_h) + \sum_{\langle a_i, e_k\rangle} \sum_{\substack{a_i \to G \\ g_h \textbf{ in } G}} p_1(e_k|a_i) \times score(a_i \to G))$

$$z(l_k|e_k, d_k) = 1/4 \text{ // uniform} \tag{7.15}$$

2: *// Parameter estimation*:
3: **for** $round := 1 \to MAX$ **do**
4: **for each** aligned pair $\langle a_i, E_{a_i}\rangle$ **do**
5: $conf \leftarrow \sum_k p_1(e_k|a_i) \times p_2(a_i|e_k)$ *// Alignment Confidence:*
6: *// Collect counts:*
7: **for each** rule $r = a \to g_1 g_2 g_3$ **do**
8: $k_{stem} \leftarrow \arg\max_k t(e_k|g_2) \times z(2|e_k, 0)$
9: **if** $t(e_{k_{stem}}|g_2) \times z(2|e_{k_{stem}}, 0) >$ threshold **then**
10: *// Get direction vector:*
11: $D \leftarrow \{d_k = \text{sgn}(k - k_{stem})\}$
12: *// For each English word:*
13: **for** $k := 0 \to m$ **do**
14: *// For each alignment position:*
15: **for** $j := 0 \to 3$ **do**
16: $\delta_{tz} \leftarrow t(e_k|g_j)z(j|e_k, d_k)/\sum_h t(e_k|g_h)z(h|e_k, d_k)$
17: $\delta \leftarrow conf \times score(r) \times \delta_{tz}$
18: $count_t(e_k, g_j)$ **+=** $\delta$
19: $total_t(g_j)$ **+=** $\delta$
20: $count_z(j, e_k, d_k)$ **+=** $\delta$
21: $total_z(e_k, d_k)$ **+=** $\delta$
22: *// Estimate probabilities:*
23: $t(e_k|g_h) \leftarrow count_t(e_k, g_j)/total_t(g_j)$
24: $z(j|e_k, d_k) \leftarrow count_z(j, e_k, d_k)/total_z(e_k, d_k)$

The word translation probability distribution is used to initialize the $t(e_k|g_h)$ parameter of our model as shown in Algorithm 2, Equation 7.14. Since the stems are actual Arabic words in our definition, we will have $p_1(e_k|g_2)$ probabilities for stems; however, this is not possible for affixes. Therefore, we compute the count $c(e_k, g_h)$ by summing over all $\langle a_i, e_k \rangle$ pairs where $p_1(e_k|a_i) > 0$ and $a_i$ has one or more rules, $r = a_i \rightarrow G$, that generate the segment $g_h$, and computing the score $p_1(e_k|a_i) \times score(a_i \rightarrow G)$ that is then added to $p_1(e_k|g_h)$ if non-zero.

The $z(l_k|e_k, d_k)$ parameters are uniformly distributed as shown in Algorithm 2, Equation 7.15.

**Parameter Estimation**

Algorithm 2 presents an expectation maximization algorithm where every epoch iterates over every aligned Arabic-English pair $\langle a_i, E_{a_i} \rangle$ in the parallel text and computes counts from every possible segmentation of $a_i$ that aligns to $E_{a_i}$'s tokens. In Line 5, we compute the confidence in this aligned pair, $conf$, that will be used in computing $\delta$.

In Line 8, we compute $k_{stem}$ which gives us the English token, $e_{k_{stem}}$, with the highest alignment to the stem in the current segmentation of $a_i$. If the alignment score of this word, $t(e_{k_{stem}}|g_2) \times z(2|e_{k_{stem}}$ is lower than a threshold, we ignore the rule that produced this segmentation. The threshold can be manipulated to trade-off quality and number of labeled segmentation choices.

Once $k_{stem}$ is found, the direction vector, $D$, can be computed (Line 11). Then, every possible combination of $E_{a_i}$ tokens and $a_i$ segments are considered to compute $\delta_{tz}$, using the last iteration $t$ and $z$ parameters (Line 16), which is combined with the rule score and the aligned pair confidence score to compute $\delta$. $\delta$ is then used to compute the counts.

Finally, the model's probabilities are calculated (Lines 23-24) to be used in the next epoch.

### 7.5.4 Experiments

We use three parallel corpora obtained from several LDC corpora including GALE and BOLT data and preprocessed by separating punctuation marks and Alif/Yah normalization. The corpora are: Egyptian-English (Egy-En) corpus of ∼2.4M tokens, Levantine-English (Lev-En) corpus of ∼1.5M tokens, and MSA-English (MSA-En) Corpus of ∼49.5M tokens. The combined corpus, which amounts to ∼53.5M tokens, is word-aligned using GIZA++ (Och and Ney, 2003a) and used as training data to this step.

We trained the EM algorithm for 50 rounds on this data and we labeled 11.9 million words with acceptable confidence.

## 7.6 Segmentation

We train a supervised segmenter to learn how to chop off Arabic words in a given sentence. For every word in the sentence, the segmenter considers a list of rules to transform that word and scores them using the Averaged Structured Perceptron. The transformation rules are segmentation rules that produce a stem and affixes. This setup allows for more advanced transformation rules where the stem is not a partial string of the source word. Examples are spelling variation normalization, templatic morphology segmentation, and even infrequent-to-frequent word translation; although, new features should be introduced to capture those advanced transformations. The empty affixes "P+" and "-Q" are not generated in the segmented output. We train and evaluate our segmenter on train/test split sets of our automatically labeled data.

### 7.6.1 Challenges of Automatically Labeled Data

Two challenges for training and decoding arise from the nature of our automatically labeled data: frequent OOVs and unlabeled words in a training sentence.

**Frequent words as OOVs**

The general case in manually annotated training data for NLP tasks is that the data is selected randomly from the text expected to be handled by the NLP tool (sometimes with a bias towards more frequent cases in the target area such as domain, dialect, and genre). The frequent words in the vocabulary, as a result, will generally be labeled in the training data. This means that OOVs in a given test set are usually infrequent words and, thus, rare in those sets.

In our setup, we label words with their segmentation choice only when we have high confidence in our decision. This leaves our partially labeled training data with many unlabeled frequent words. These words are naturally frequent in a given test set, which means they will be OOVs for a system trained on our partially labeled training data.

This problem requires us, as we design the segmenter, to give special attention to its ability of learning how to generalize to unseen words, since those unseen words are now a frequent phenomena. To do so, we use the following strategy:

1. We introduce features that deliberately try *not* to memorize specific segmentations in order to allow them to generalize to OOVs.

2. We drop all segmentation rules learned in Section 7.4 since they do not extend to a large portion of the vocabulary. To generate a list of rules for a given word in a sentence, the decoder, now, considers all possible segmentations of that word that produce stems that have been seen in the parallel data. This will introduce new, unseen affixes, which is intended to generalize to unseen dialectal morphology.

3. Some of our best features use distance scores from the Arabic and English clusters of the word being processed (discussed later). Since these clusters were used in creating our labeled data, all labeled words have both clusters. This results in a generalizability issue as many other words may have one or no clusters. To solve

this issue, we deliberately drop either or both clusters for some labeled words in a random manner in order to allow other general features to be trained for such cases.

4. To evaluate our segmenter's ability to generalize to OOVs, we randomly drop some test set words from the training data to generate OOVs that we can evaluate on. This allows us to pick a system with high generalizability.

**Unlabeled words in a training sentence**

Unlike manually labeled data where all words in a training sentence are labeled, our data may have many unlabeled words in a given training sentence. This means that features of a rule cannot depend on the segmentation decision made for the previous word since we cannot guarantee knowing that decision during training. Therefore, the decoder cannot use the Viterbi algorithm; instead, it picks the segmentation rule with the highest score for every word independently. We do, however, use features that look at the possible segmentation rules of surrounding words which are inspired by the gender/number agreement of Arabic.

### 7.6.2 Features

Global Linear Models (GLMs) allow us to use million of features to score the different rules of a given word in context. A feature is 1 if a certain condition is satisfied by the rule and 0 otherwise. For example, the feature below fires up when a rule is trying to segment the prefix وه+ *wh+* 'and will' from a stem with 4 letters (which could be a present verb in Arabic).

$$\phi_{1000}(a \rightarrow psq) = \begin{cases} 1, & \text{if length of stem } s = 4 \\ & \text{and prefix } p = \text{ "wh+"} \\ 0, & otherwise \end{cases}$$

The perceptron learns the weights of those features which are added to the score of any rule that satisfy the feature's condition. As expected, the perceptron learns an above zero weight for the example feature above.

### Feature Types

A *feature type* is a general definition of a group of features that have the same condition structure but differ in the assigned values. For example, The feature presented above belongs to a feature type that looks at the prefix and the length of the stem produced by a given rule. We will use the term "*conditioned on*" to represent the feature type's condition that must be satisfied by the rule for the feature weight to be added to its score. For example, the feature type of the example above will be described as "conditioned on the prefix and the length of the stem", which means that this *feature type* will produce a feature for every prefix and every stem length.

Below we list the different categories of feature types we experimented with although most of them did not make it to the final system.

### Word and Segments Features

The features under this category look at the source word and the output segments to learn to memorize combinations. We define types conditioned on: (a) the rule's stem, (b) the prefix, (c) the suffix, (d) the source word and the stem, (e) the source word and the prefix, (f) the source word and the suffix, (g) the stem and the prefix, (h) the stem and the suffix, and (i) the prefix and the suffix. Only three types are used in the best system: (a), (g), and (h).

### Surrounding Words Features

These *context-aware* features look at the segments (stem, prefix, and suffix) and the surrounding words. We experimented with feature types conditioned on each segment and

each of the adjacent words (resulting in 6 feature types). We also experimented with feature types conditioned on each segment of the rule and affixes of the adjacent words (separate or combined) which is inspired by gender/number agreements and repeated definite article and conjunction particle. None of the types above helped improve the best system.

**Length Features**

These features look at a segment or its length and the length of the stem or the source word. The helpful feature types are conditioned on the prefix and the length of the stem, the suffix and the length of the stem, the length of the word and the length of the stem, the length of the word and the length of the prefix, the length of the word and the length of the suffix. All other combinations were not helpful.

**Frequency Features**

These features use frequencies and counts as a measure of confidence in the rule's segments. Since using frequencies will cause sparse features, we put frequencies in bins by taking the integer part of the logarithm of the frequency (we call it *int-log*).

For word and stem frequencies, we found that base 100 for the logarithm worked best and put these frequencies in four bins: 0 for frequencies bellow 100, 1 for 100 to 10,000, 2 for 10,000 to 1 million, and 3 for above a million. Two feature types use stem and word frequencies: one is conditioned on the *int-log*$_{100}$ of the stem frequency in the parallel data (to promote rules that produce frequent stems), and the other is conditioned on the *int-log*$_{100}$ of the difference between the stem frequency and the source word frequency (to restrict the segmentation of very frequent words);

We define two more feature types that look at the number of unique stems that have been seen with the prefix or suffix generated by the rule: The first type is the prefix-stem-count, which is conditioned on the *int-log*$_3$ of that number for prefixes. The second type, suffix-stem-count, is similar but for suffixes. The intuition is that affixes seen with a variety

of stems in our rules are more trustworthy than affixes seen with few stems. For example, a prefix like +والwA*l*+ 'and the' attaches to 3,196 unique stems and, thus, a rule that produces it is promoted by this prefix-stem-count feature which is conditioned on *int-log*$_3(3196) = 7$. Base 3 showed to be the best performer for these features.

All of these four feature types were helpful and used in the best system.

**Affixes' Features**

There are two groups of these features: one that looks at affixes of the current rule, and one that looks at affixes of all rules of the current word. In the first group we have a feature type conditioned on the prefix and another conditioned on the suffix of the current rule. Both types did not help. After manually analyzing the errors we found that while frequent affixes like +و *w*+ "and" should be tokenized, this decision cannot be made independently of the other rules as an alternative rule might produce +وال*wAl*+ "and the" which probably should be selected over *w*+.

This conclusion motivated the second group which includes: 1) a feature type conditioned on the prefix of the current rule and the competing prefixes of the other rules; and 2) a similar feature type but conditioned on suffixes instead. An example feature of the first type is conditioned on the current rule's prefix being *wAl*+ and the alternative rules' prefixes being *wA*+, *w*+, and the empty string (P+). The perceptron learned a high weight for this feature and a negative weight for the feature with the same alternative prefixes but the current prefix being *w*+. The two feature types of the second group improved the performance of the best system.

**Clusters' Features**

In Section 7.4.1 we built word clusters from word vectors learned from monolingual data. We call these clusters *Arabic clusters* since they are motivated by Arabic semantics. Additionally, we build word clusters by pivoting on English words in the alignment data. We call

these clusters *English clusters* since they are semantically motivated by English. In Section 7.5.3 we described how we estimate word translation probabilities $p_1(e|a)$ and reverse word translation probabilities $p_2(a|e)$ for each Arabic word $a$ and English word $e$ from the word alignments produced by running GIZA++ on the parallel text. The English pivoting algorithm uses those probability distributions to compute the distance between any two Arabic words $x$ and $y$ by pivoting on every English word as described by this equation[4]:

$$dist_e(x \rightarrow y) = \sum_e p_1(e|x)^{\lambda_1} \times p_2(x|e)^{\lambda_2} \times p_1(e|y)^{\lambda_3} \times p_2(y|e)^{\lambda_4} \tag{7.16}$$

For every Arabic word $x$ we build a word cluster that contains the closest 2000 Arabic words, $y$, where $dist_e(x \rightarrow y) > D_e$ , the threshold for English clusters.

Using the two distance measures of the two types of clusters, we use a feature type conditioned on these conditions:

1. The type of the cluster: Arabic or English.

2. The *int-log* value of the cluster's distance between the source word and the stem. As discussed earlier, the *int-log* function is used to smooth the range of distance values into a small number of bins.

3. The order of that distance among the distance values of all other stems generated by the alternative rules of the source word. To do so, we order all the stems generated by the rules of the source word in terms of their distance to the source word, and we condition on that order.

4. A flag that tells us whether we have a cluster for the source word or not. This flag allows us to distinguish between between two cases: 1) the source word has a cluster but the stem generated by the current rule does not belong to that cluster; in such

[4] For simplicity we set $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1$, although these weights can be tuned; e.g., by evaluating against the resulting clusters from clusters learned from monolingual data.

case the distance will be infinity but this flag will be 0; and 2) the source word does not have a cluster, which means the distance is also infinity but this flag is 1. This flag allows us not to penalize a stem for having an infinity distance when the source word does not have a cluster.

This feature type generates 104 features that help improve the performance of the system. We experimented with different combinations of features that look at the stem distance to surrounding words and to *potential* stems of surrounding words (generated by the rules) with a window of up to 5-words on each side, but none proved to be helpful. Although these contextual features are attempting to produce context-dependent segmentation based on different senses of the source words identified by the context around it, we found that the benefits of such segmentations are alleviated by the damage of the segmentation inconsistency they cause for the same word.

**Pattern Features**

Since Arabic morphology is templatic, we use features that look at the pattern of the stem along with the affixes produced by a rule. Since certain patterns can help indicate verbs or nouns we hypothesize that certain affixes will have high correlation with them; therefore, the features conditioned on them should have high weights. To keep our model generalizable to all Arabic dialects, we extract simple patterns from the stem by masking all letters except long vowels (ا *A*, و *w*, and ي *y*[5]) and Ta Marbuta (ة *p*) which appears only as the last letter of nouns or adjectives but not verbs.

We use two pattern feature types: one conditioned on the prefix and the stem pattern, and one conditioned on the suffix and the stem pattern. Both feature types improve the performance of the best system.

[5] Since the text is Alif/Yah normalized, it does not contain the other forms of Alif or Alif Maqsura (ى *Y*)

Examples of the first feature type are features that are conditioned on a prefix and the pattern _ـا_ *y_A_* (where '_' represents a masked letter), which matches some "hollow" verbs in present tense; e.g., ينام *ynAm* 'he sleeps'. Among those features are features conditioned on verb prefixes (e.g., +و بـ *wb+* "and he", +حـ *H+* "will", +وسـ *ws+* "and will", +ما *mA+* "not", and the empty prefix P+) which obtained positive weights as expected.

An example of the second feature type is the feature conditioned on the pattern *y_A_* and the suffix -كوا *-kwA* 'your$_{(PLURAL)}$' (as in رايكوا *rAykwA* 'your opinion'). This feature obtained a negative weight. A manual error analysis showed that, in many cases, the 'k' of '-kwA' is part of the word. For example, consider the word يشاركوا *y$ArkwA* 'they participate', where *y$Ark* 'participate' is a verb and '-wA' 'they' is a suffix. Since *y$Ar* is a valid word in Arabic meaning 'pointed to', a rule that produces *y$Ar -kwA* will be considered by the segmenter. The feature above will vote such a rule down. Another rule that produces *y$Ark -wA* will be voted up by another feature of the same feature type. That feature is conditioned on the pattern *y_A___* and the suffix *-wA*. And so this feature type can help the segmenter in finding a better segmentation.

Feature types conditioned on the pattern alone or the pattern with both affixes did not help improve the performance.

**Affix and Adjacent Letter Features**

We use two feature types: one that is conditioned on the prefix and the first letter of the stem, and one conditioned on the suffix and the last letter of the stem. Both features proved to be helpful and are included in the best system.

An example of the first type is a feature conditioned on the prefix +وا *wA+* 'and I' and the first letter of the stem being ل *l*. This feature has a negative weight as expected, which promotes alternative rules that produce the prefixes +والـ *wAl+* 'and the' and +و *w+* 'and'.

#### Affix Correlation Features

These features are based on the affix correlation $corr_{pref}(p', p)$ and $corr_{suff}(q', q)$ discussed in Section 7.4. Unlike before, we make use of the English clusters in addition to the Arabic clusters to learn affix correlations. The non-zero affix correlations are stored for all affix pairs in a lookup table to be used during decoding. During feature extraction for a rule $a \rightarrow psq$, we look at all prefixes $p'$ that have been seen with stem $s$ and compute $c_v(p, s) = \sum_{p'} corr_{pref}(p', p)$. We similarly compute $c_u(q, s) = \sum_{q'} corr_{suff}(q', q)$. We define two feature types: one conditioned on *int-log* of $c_v(p, s)$ and one conditioned on *int-log* of $c_u(q, s)$. Both types help improve performance.

### 7.6.3 Experiments and Evaluation

#### Experimental Setup

We implement our own averaged structured perceptron trainer and decoder, discussed in Section 7.6. We split the automatically labeled examples from the previous step into train ($\sim$9.9M labeled tokens) and dev and test sets (1M labeled tokens each).

#### Evaluation

We ran hundreds of experiments in a linguistically-motivated greedy approach to engineer good feature types for our segmenter. The systems learned from top performing feature type combinations were then evaluated by the machine translation experiments to pick the best segmenter.

We empirically determined the number of epochs to be 14. In development and test we automatically generate OOVs by randomly omitting Arabic and English clusters as discussed earlier. This makes the non-cluster features fully responsible for segmenting those words without the reliance on cluster features, which allows the segmenter to tune their weights and thus generalize to actual MT sets' OOVs.

| | dev | | | | test | | | |
|---|---|---|---|---|---|---|---|---|
| | # correct | / | # tokens | **accuracy** | # correct | / | # tokens | **accuracy** |
| All Tokens | 710,527 | / | 721,771 | 98.44% | 684,123 | / | 693,994 | 98.58% |
| **Breakdown by INVs and OOVs:** | | | | | | | | |
| INVs | 575,529 | / | 575,531 | ∼100.00% | 556,767 | / | 556,767 | 100.00% |
| OOVs | 134,998 | / | 146,240 | 92.31% | 127,356 | / | 137,227 | 92.81% |
| **OOV Categories:** | | | | | | | | |
| No Arabic Cluster | 18,111 | / | 19,247 | 94.10% | 14,304 | / | 16,441 | 87.00% |
| No English Cluster | 66,756 | / | 72,694 | 91.83% | 16,435 | / | 17,947 | 91.58% |
| Neither Cluster | 7,423 | / | 8,283 | 89.61% | 1,650 | / | 1,959 | 84.23% |
| Both Clusters | 58,690 | / | 62,582 | 93.78% | 98,689 | / | 104,798 | 94.17% |
| No-Segmentation | 5,641 | / | 7,284 | 77.44% | 7,379 | / | 9,441 | 78.16% |

Table 7.1: The segmentation decoder results in terms of accuracy (number of correct segementations / total number of tokens) on both the dev and blind test sets. The first section shows the results on all tokens, while the following sections break the tokens down into categories.

Table 7.1 presents the performance of the best segmenter system in terms of accuracy on both dev and test sets (the last two columns across all sections). The sections of the table represent the breakdown of tokens into categories for in depth evaluation. The accuracy scores for each of these categories were used to engineer our feature types to ensure that they generalize to frequent OOVs belonging to those categories. We also make sure, while automatically generating OOVs in dev/test sets, that we have enough tokens in each category to guarantee a representative evaluation.

Since the segmenter's job is to pick a segmentation rule out of a generated list of rules, we evaluate only on words with more than one rule (multi-choice) which constitute 721,771 tokens of the 1M-token dev set and 693,994 tokens of the 1M-token blind test set. The rest of the tokens have only one segmentation rule: no segmentation. The first section of Table 7.1 shows the accuracy of the segmenter on multi-choice tokens. In the second section, we break down the evaluation to INVs (in-vocabulary words) and OOVs.

Since we might not have Arabic or English clusters for many words in test sets, we define four categories representing the absence of either cluster, both, or neither. These categories are mutually exclusive. We also evaluate on OOVs that should not be segmented,

yet have multiple rules, to reduce our decoder's over-segmentation. The third section of Table 7.1 presents evaluation on multi-choice OOV categories. The results on the dev set carries on to the blind test set.

## 7.7 Evaluation on Machine Translation

### 7.7.1 MT Experimental Setup

#### MT Train/Tune/Test Data

We combine the training, dev, and test sets we used to train and evaluate our segmenter into one parallel corpus and we use it to train our MT systems. The MT tune, dev, and test sets, however, are selected from several standard MT test sets. We use three Egyptian sets from LDC BOLT data with two references (EgyDevV2, EgyDevV3, and EgyTestV2), and one Levantine set from BBN (Zbib et al., 2012) with one reference which we split into LevDev and LevTest. We use EgyDevV3 to tune our SMT systems. We use the remaining sets for development and test on both Egyptian and Levantine. For dev we use EgyDevV2 and LevDev and for test we use EgyTestV2 and LevTest. It is important to note that the segmenter has never seen these tune/dev/test sets. The segmenter was only trained on the MT training data.

#### MT Tools and Settings

We use the open-source Moses toolkit (Koehn et al., 2007) to build our Arabic-English phrase-based statistical machine translation systems (SMT). Our systems use a standard phrase-based architecture. The language model for our systems is trained on English Gigaword (Graff and Cieri, 2003). We use SRILM Toolkit (Stolcke, 2002) to build a 5-gram language model with modified Kneser-Ney smoothing. Feature weights are tuned to maximize BLEU on the tuning set using Minimum Error Rate Training (Och, 2003). Results

| | dev | | | | test | | | |
|---|---|---|---|---|---|---|---|---|
| | Egy | | Lev | | Egy | | Lev | |
| | **BLEU** | **MET.** | **BLEU** | **MET.** | **BLEU** | **MET.** | **BLEU** | **MET.** |
| $MT_{UNSEGMENTED}$ | 19.2 | 27.0 | 13.5 | 21.3 | 21.8 | 28.1 | 13.3 | 21.7 |
| $MT_{MORFESSOR}$ | 20.8 | 28.4 | 13.7 | 21.6 | 21.7 | 29.2 | 13.6 | 22.4 |
| $MT_{MADAMIRA\text{-}EGY}$ | **21.5** | 29.0 | 15.2 | 22.4 | **23.0** | **30.0** | 15.2 | 23.1 |
| **Our Systems:** | | | | | | | | |
| $MT_{CONTEXT\text{-}SENSITIVE}$ | 21.0 | 28.3 | 15.6 | 22.9 | 22.4 | 29.3 | 15.6 | 23.6 |
| $MT_{CONTEXT\text{-}INSENSITIVE}$ | 21.4 | **29.2** | **16.2** | **23.5** | **23.0** | 29.9 | **16.3** | **24.0** |
| **Our Best System's Inprovements:** | | | | | | | | |
| *Over* $MT_{UNSEGMENTED}$ | +2.2 | +2.2 | +2.7 | +2.2 | +1.2 | +1.8 | +3.0 | +2.3 |
| *Over* $MT_{MORFESSOR}$ | +0.6 | +0.8 | +2.5 | +1.9 | +1.3 | +0.7 | +2.7 | +1.6 |
| *Over* $MT_{MADAMIRA\text{-}EGY}$ | *-0.1* | +0.2 | +1.0 | +1.1 | 0.0 | *-0.1* | +1.1 | +0.9 |

Table 7.2: Evaluation in terms of BLEU and METEOR (abbreviated as MET.) of our two MT systems: S1 and S2 on a dev (first set of columns) and a blind test sets (second set of colunms). In the first section we present three baselines: $MT_{UNSEGMENTED}$, $MT_{MORFESSOR}$ and $MT_{MADAMIRA\text{-}EGY}$. In the second section we present our two MT systems: $MT_{CONTEXT\text{-}SENSITIVE}$ trained on text segmented by a segmentation system that uses context-sensitive features, and $MT_{CONTEXT\text{-}INSENSITIVE}$ trained on text segmented by a segmentation system that uses only context-insensitive features. The third section shows the differences between our best system results and those of the three baselines.

are presented in terms of BLEU (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005). All evaluation results are case *insensitive*. The English data is tokenized using simple punctuation-based rules. The Arabic text is also Alif/Ya normalized. For more details on processing Arabic, see (Habash, 2010).

### 7.7.2 MT Experiments

We use the same parallel data to train all of our MT systems and the same dev and test sets to evaluate. The only difference is the preprocessing of the Arabic side of training, dev, and test data. Table 7.2 shows MT experiment results in terms of BLEU and METEOR on dev (first set of columns) and blind test (second set of columns).

#### Baseline Systems

We build three baseline MT systems to compare our systems against. In the first baseline system we Alif/Yah normalize the Arabic side but we leave it unsegmented. We call

this baseline $MT_{UNSEGMENTED}$. The other two baseline systems are based on two previous research efforts representing two approaches to morphological segmentation. The first is a tool for language-independent, unsupervised learning of morphology: MORFESSOR (Creutz and Lagus, 2002) to segment the Arabic side, and the second is a dialect-specific tool that requires handcrafted resources and is trained on hand-labeled data: MADAMIRA-EGY, the version of MADAMIRA (Pasha et al., 2014) that handles Egyptian as well as MSA. To the best of our knowledge, MADAMIRA-EGY is the best system for morphological segmentation of dialectal Arabic. We use these two tools to preprocess Arabic and we name the resulting two MT systems after them: $MT_{MORFESSOR}$ and $MT_{MADAMIRA-EGY}$, respectively. All Arabic textual data (parallel and monolingual) were used to train MORFESSOR.

The first section of Table 7.2 presents results on these baselines. On the dev set, $MT_{MORFESSOR}$ performs significantly better than $MT_{UNSEGMENTED}$ on Egyptian (1.6% BLEU, 1.4% METEOR) and slightly better on Levantine (0.2% BLEU, 0.3% METEOR). This could be due to the limited Levantine text in MORFESSOR's segmentation training data compared to Egyptian and MSA. $MT_{MADAMIRA-EGY}$ outperforms the other baselines on both dialects on both metrics. An interesting case is $MT_{MADAMIRA-EGY}$ results on Levantine dev; it improves over $MT_{UNSEGMENTED}$ by 2.3% BLEU, 2.0% METEOR, and over $MT_{MORFESSOR}$ by 1.5% BLEU, 0.8% METEOR. $MT_{MADAMIRA-EGY}$'s good performance on Levantine can be explained by the fact that these two dialects share many of their dialectal affixes and clitics (e.g., +حـ *H+* 'will', +بـ *b+* 'simple present', ـلك *-lk* "to you") as well as lemmas. Moreover, most of the phonological differences between Levantine and Egyptian do not show up in the orthographic form since Arabic writers tend to drop short vowels and normalize some letters such as the three-dotted Jeem چ to single-dotted Jeem ج.This leads to many Levantine words looking identical to their Egyptian equivalents although they are pronounced differently.

### Our MT Systems

We present two MT systems to evaluate two of our segmentation models. The first model is trained using the best performing combination of context sensitive and insensitive features while the second model uses the best performing combination of context insensitive features only (presented in Table 7.1). We call the resulting MT systems: $\text{MT}_{\text{CONTEXT-SENSITIVE}}$ and $\text{MT}_{\text{CONTEXT-INSENSITIVE}}$, respectively. We present MT results on our systems in the second section of Table 7.2. $\text{MT}_{\text{CONTEXT-INSENSITIVE}}$ outperforms $\text{MT}_{\text{CONTEXT-SENSITIVE}}$ across dialects and metrics. Investigating the output of both systems shows that the inconsistencies generated by context-based segmentation outweighs the benefits of disambiguation, especially that phrase-based statistical machine translation is robust toward infrequent systematic segmentation errors across training, tuning, and test sets.

The third section of Table 7.2 reports the differences between our best system's results and those of the three baselines. $\text{MT}_{\text{CONTEXT-INSENSITIVE}}$ improves over both resource-free baselines ($\text{MT}_{\text{UNSEGMENTED}}$ and $\text{MT}_{\text{MORFESSOR}}$) across dev sets and metrics ranging from 2.2% BLEU on Egyptian and 2.7% BLEU on Levantine over $\text{MT}_{\text{UNSEGMENTED}}$ to 0.6% BLEU on Egyptian and 2.5% BLEU on Levantine over $\text{MT}_{\text{MORFESSOR}}$. These results demonstrate the usefulness of such approach where resources are unavailable.

When compared to $\text{MT}_{\text{MADAMIRA-EGY}}$, however, performance on Egyptian differs from Levantine. The results on Egyptian are inconclusive: in terms of BLEU, $\text{MT}_{\text{MADAMIRA-EGY}}$ leads by 0.1%; while in terms of METEOR, $\text{MT}_{\text{CONTEXT-INSENSITIVE}}$ leads by 0.2%. These results mean that our best MT system is on par with $\text{MT}_{\text{MADAMIRA-EGY}}$ on Egyptian; which we consider a good result since MADAMIRA-EGY has been optimized for years with a wealth of dialect and task specific resources. On Levantine, however, our system outperforms $\text{MT}_{\text{MADAMIRA-EGY}}$ by 1.0% BLEU, 1.1% METEOR. The results on blind test sets, presented in the second set of columns of Table 7.2 labeled "test", agree with the results on the dev sets and confirm their conclusions.

| | Reference | we see them a lot in the hamra and salehieh markets |
|---|---|---|
| **System** | **Processed Arabic** | **English Translation** |
| $MT_{UNSEGMENTED}$ | bn$wfhn ktyr bswq AlHmrA wAlSAlHyp | wAlSAlHyp red market ; we see them influenced a lot |
| $MT_{MORFESSOR}$ | b+ n$wfhn ktyr b+ swq AlHmrA w+ Al+ SAlHyp | to see them a lot in the market the red salihiyah , |
| $MT_{MADAMIRA\text{-}EGY}$ | bn$wfhn ktyr b+ swq AlHmrA wAlSAlHyp | we see a lot of hamra wAlSAlHyp market |
| $MT_{CONTEXT\text{-}INSENSITIVE}$ | bn+ $wf -hn ktyr b+ swq Al+ HmrA w+ AlSAlHyp | we see them a lot in souk al hamra and al-saleheyya |

Table 7.3: An example Arabic sentence translated by the three baselines and our best system.

### 7.7.3 Example and Discussion

Table 7.3 presents an example Levantine sentence translated by the three baselines and our best system. While each baseline has its own errors, our system produces a perfect translation although the reference does not match it word to word due to the several acceptable transliterations[6] of the mentioned proper names found in our MT training data. This results in penalties by BLEU and, in this example, METEOR; nevertheless, the translation is sound.

The example contains words with different characteristics that are handled differently and sometimes similarly by the four systems:

1. The word بنشوفهن *bn$wfhn* 'we see them' has rich Levantine morphology. Unlike $MT_{MORFESSOR}$ and $MT_{MADAMIRA\text{-}EGY}$ our system segments this word to three tokens that map directly to the three English words of the correct translation.

2. The word بسوق *bswq* 'in market' has MSA morphology and is segmented correctly by all systems (except $MT_{UNSEGMENTED}$) which results in correct translations ($MT_{CONTEXT\text{-}INSENSITIVE}$ translates *swq* to "Souk", a correct transliteration found in our MT training data since it is frequently part of a proper noun).

3. The word الحمرا *AlHmrA* 'Al Hamra' ('Al' is the definite article in Arabic) is a

[6] None of the systems discussed in this work has a transliteration component. All transliterations produced by these systems (e.g., the three different transliterations of *AlSAlHyp*) are found in our MT training data.

proper noun although the word literally means “red” which led to the mistakes by $MT_{UNSEGMENTED}$ and $MT_{MORFESSOR}$. Both $MT_{CONTEXT\text{-}INSENSITIVE}$ and $MT_{MADAMIRA\text{-}EGY}$ produce an acceptable transliteration.

4. The word والصالحية *wAlSAlHyp* ‘and Al Salehieh‘ is the name of the second market with the conjunction particle و *w* ‘and’ attached to it. Both $MT_{CONTEXT\text{-}INSENSITIVE}$ and $MT_{MORFESSOR}$ succeed in segmenting this word to produce an acceptable translation and transliteration, although $MT_{MORFESSOR}$ fails to produce ‘and’. This word show an advantage that our segmenter and MORFESSOR has over MADAMIRA-EGY. Since they learn their morphemes and stems from data, they can better handle morphologically inflected proper nouns and dialectal/infrequent lemmas that do not appear in MADAMIRA-EGY’s internal morphological analyzer database.

## 7.8 Conclusion and Future Directions

In this chapter, we presented an approach to cheaply scale to many dialects without the need for DA preprocessing tools. This approach attempts at learning an underlying Arabic preprocessing models for all Arabic varieties including MSA. The approach expects a small amount of DA-English parallel data along with a sizable amount of MSA-English data.

Our approach learns out-of-context preprocessing rules for dialectal Arabic from unlabeled monolingual data. We use an unsupervised approach on large quantities of unlabeled Arabic text to extract a list of out-of-context preprocessing rules with weights estimated with expectation maximization. We use these rules in another unsupervised learning approach to automatically label words in the dialectal side of a DA-English parallel corpus. In a given DA sentence, a word is labeled in-context with its best preprocessing rule which generates tokens that maximize alignment and translation to English words in the English translation of the corresponding sentence. This synthetic labeled corpus is used to train

a supervised segmenter with features designed to capture general orthographic, morphological, and morphosyntactic behavior in Arabic words in order to generalize to unseen words.

We evaluated our approach on morphological segmentation and showed significant improvements on Egyptian and Levantine compared to other unsupervised segmentation systems. We also showed that our system is on par with the state-of-the-art morphological tokenizer for Egyptian Arabic built with supervised learning approaches that require manually labeled data, a large budget, and years to build. This shows that our approach can cheaply and quickly scale to more dialects while still performing on par with the best supervised learning algorithm. Furthermore, our evaluation on Levantine Arabic showed an improvement of 3% over an unsegmented baseline, 2.7% over the unsupervised segmentation system, and 1.1% over the supervised tokenization system, in terms of BLEU. This is especially important given that our system was not trained on monolingual Levantine text, which means that Levantine preprocessing rules were not leaned; yet, our segmenter was able to generalize to Levantine.

In the future, we plan to evaluate our work on more dialects and subdialects where DA-English are not available. This is analogous to the ADAM tokenization approach, where in-vocabulary stems are given a chance to be translated by separating unseen dialectal affixes from them.

We also plan to apply our approach to tasks other than morphological segmentation. In this work we extracted rules from Arabic clusters where a stem is a substring from the source word. While this helped split affixes from stems, it did not attempt at modifying the stem or the affixes. The Arabic cluster for a given word contains words that are not substrings, yet they are very close to the source word. Here are two examples:

1. **Some words are a spelling variation of the source word.** We can use Levenstein distance to identify these words (with a threshold cost) and add stem-to-stem rules.

When these rules are combined with other rules in the segmentation graph, the rule expansion step will produce rules that perform orthographic normalization side by side with morphological segmentation. While Levenstein distance can be used to penalize the $a2s$ score, we might need to add a *character mutation* probability to the model to be estimated with EM. This probability can later be used as a feature in the segmenter to allow unseen to be normalized.

2. **Some words are a translation/synonym of the source word.** This could be a synonym in the same dialect or a translation to another dialect (including MSA). This motivates a replacement for the rule-based ELISSA approach where infrequent words are translated to frequent words as opposed to MSA equivalents. Like the previous item, stem-to-stem rules can be added and extended in the rule expansion step.

# Chapter 8

# Conclusion and Future Directions

In the research presented in this thesis, we worked on improving the quality of Dialectal Arabic to English machine translation. We categorize our dialect translation approaches into three categories based on the availability of resources (parallel data and preprocessing tools) for these dialects. The three categories are: dialects that have virtually no resources, dialects that have some DA-English data and preprocessing tools, and dialects with DA-English data and no preprocessing tools. We build tools and resources that use and extend the currently available resources to quickly and cheaply scale to more dialects and sub-dialects. Following is a summary of contributions followed by a discussion of future work directions.

## 8.1 Summary of Contributions and Conclusions

The first challenge we targeted in this thesis is translating dialects that have virtually no resources. We proposed an MSA-pivoting pipeline that uses a morphological analyzer and an DA-to-MSA MT system built with rule-based approaches.

- **ADAM and morphological tokenization.** The biggest challenge for translating these dialects with an MSA-to-English SMT system is the large number of out-of-

vocabulary (OOV) words. This is largely caused by dialectal morphemes attaching to words many of which come from MSA. A quick and cheap approach to handle OOVs of these dialects is to build a morphological segmentation or tokenization tool to break morphologically-complex words into simpler, more frequent, tokens. For this purpose, we presented ADAM, an analyzer of dialectal Arabic morphology, that can be quickly and cheaply created by extending existing morphological analyzers for MSA or other Arabic varieties. The simplicity of ADAM rules makes it easy to use crowdsourcing to scale ADAM to cover dialects and sub-dialects. We presented our approach to extending MSA clitics and affixes with dialectal ones although the ADAM technique can be used to extend stems as well. We showed how using ADAM to tokenize dialectal OOV words can improve the translation quality of an MSA-to-English SMT system by 0.35% BLEU.

- **ELISSA and MSA-pivoting.** Translating dialectal words and phrases to their MSA equivalents, instead of just tokenizing them, gives the MSA-to-English SMT system a better chance to translate them correctly. There is virtually no DA-MSA parallel data to train an SMT system. Therefore, we presented ELISSA, a tool for DA-to-MSA machine translation. ELISSA identifies dialectal word and phrases that need to be translated to MSA, and employs a rule-based MT approach that relies on morphological analysis, morphosyntactic transfer rules and dictionaries, in addition to language models to produce MSA translations of dialectal sentences. Using ELISSA to produce MSA versions of dialectal sentences as part of an MSA-pivoting DA-to-English MT solution, improves BLEU scores on three blind test sets by: 0.95% absolute BLEU (or 2.5% relative) for a speech multi-dialect (Iraqi, Levantine, Gulf, Egyptian) test set, 1.41% absolute BLEU (or 15.4% relative) for a web-crawled Levantine test set, and 0.61% absolute BLEU (or 3.2% relative) for a web-crawled Egyptian test set.

The second challenge we were concerned with is translating dialects that have parallel data as well as preprocessing tools which allows for the creation of a direct-translation DA-to-English SMT system. The questions are whether the MSA-pivoting approach is still relevant and whether using the MSA-English data can help.

- **Improving MSA-pivoting with DA-English data.** Using the DA-English data, we built three direct translation SMT systems: one trained on DA-English corpus only, one trained on MSA-English corpus only, and one trained on the two corpora combined. We showed that MSA-pivoting approaches to DA-to-English MT can still help when the available parallel data for a dialect is relatively small compared to MSA. The key for the improvements we presented is to exploit the small DA-English data to create automatically generated parallel corpora on which SMT systems can be trained. We translated the DA side of the DA-English parallel data to MSA using ELISSA, and added that data to the (DA+MSA)-English training data on which an SMT system was trained. That SMT system, when combined with ELISSA for preprocessing, outperforms the best direct translation approach by 0.56% BLEU. The main reason for this improvement is that the SMT system is now familiar with ELISSA's output and can correct systematic errors performed by ELISSA. We presented two new versions of ELISSA that use the synthetic parallel data: Statistical ELISSA and Hybrid ELISSA. However, both systems failed to improve the translation quality.

- **System combination.** Although our MSA-pivoting approach outperforms the three direct translation systems discussed above, we showed that combining these four systems can further improve the translation quality. We presented a sentence-level classification approach for machine translation system combination for diglossic languages. Our approach uses features on the source language to determine the best baseline MT system for translating a sentence. We get a 1.0% BLEU improvement

over the best baseline single MT system. This proves that these different DA-to-English MT approaches complement each other in interesting ways and that their combination could lead to better overall performance by benefiting from the strengths while avoiding the weaknesses of each individual system. This is possible due to the diglossic nature of the Arabic language.

Finally, we proposed an unsupervised approach to model dialects from scratch and build preprocessing tools from small amounts of parallel data to cheaply and quickly scale to many dialects and sub-dialects. While this approach can be used to translate dialects with DA-English data but no preprocessing tools, we deliberately ignore DA preprocessing tools, if they exists, to build a unified preprocessing model for dialects. To do so, our approach relies heavily on an abundant resource: monolingual text, in addition to any available DA-English corpora. We presented a morphological segmentation system as an example of our approach. A system like this provides a huge boost to MT since it dramatically reduces the size of the vocabulary.

- **Learning morphological segmentation options from monolingual data.** A morphological segmentation system needs a tool that provides a list of segmentation options for an input word. We presented an unsupervised learning approach to build such a tool from word embeddings learned from monolingual data. This tool provides morphological segmentation options weighted, out-of-context, using expectation maximization.

- **Morphological segmentation for MT purposes.** We use the tool above in another unsupervised learning approach to automatically label words in the dialectal side of a DA-English parallel corpus. In a given DA sentence, a word is labeled in-context with its best preprocessing rule which generates tokens that maximize alignment and translation to English words in the English translation of the corresponding sentence.

This synthetic labeled corpus is used to train a supervised segmenter with features designed to capture general orthographic, morphological, and morphosyntactic behavior in Arabic words in order to generalize to unseen words. We evaluated our approach on morphological segmentation and showed significant improvements over another *unsupervised* segmentation system ranging from 1.3% BLEU (on Egyptian) to 2.7% (on Levantine). We also showed that our system is on par with the state-of-the-art morphological tokenizer for Egyptian Arabic, MADAMIRA-EGY, built with *supervised* learning approaches that require manually labeled data, a large budget, and years to build. This shows that our approach can cheaply and quickly scale to more dialects while still performing on par with the best supervised learning algorithm. Our evaluation on Levantine Arabic against MADAMIRA-EGY showed an improvement of 1.1% BLEU. This is especially important given that our system was not trained on monolingual Levantine text, which means that Levantine preprocessing rules were not leaned; yet, our segmenter was able to generalize to Levantine.

## 8.2 Future Directions

In this section, we discuss the different research directions that we may explore in the future for each of the major contributions we discussed throughout this thesis.

### 8.2.1 Extending Existing Preprocessing Models

This direction builds on the currently available linguistic modeling of Arabic and its dialect to build lemma:feature-based morphological analyzers and morpho-syntactic translation systems.

**ADAM and morphological tokenization for MT.** Regarding rule-based morphological segmentation for dialects with no resources, we plan to extend ADAM coverage of current

dialects and new dialects by adding dialectal stems in two ways:

1. **Copying and modifying MSA stems with SADA-like rules.** The mutations of many dialectal stems from MSA stems follow certain patterns than can be captured with SADA-like rules. For example, for a verb that belongs to a three-letter root with duplicate last letter (e.g., *Hbb* 'to love' and *rdd* 'to reply'), the stem that forms the verb with first person subject (e.g., in MSA, *>aHobabotu* 'I love' and *radadotu* 'I reply') is relaxed with a 'y' in Egyptian and Levantine (e.g, *Hab∼ayt* and *rad∼ayt*).

2. **Importing DA-MSA Lexicons.** DA-MSA dictionaries and lexicons, whether on the surface form level or the lemma level, can be selectively imported to ADAM database.

**ELISSA and the pivoting approach.** In addition to extending ELISSA's coverage in the handled dialects and to new dialects with handcrafted rules, we plan to automatically learn rules from limited available DA-English data. In Chapter 5 we used sentence-level pivoting techniques to synthesize parallel data. We plan to use other pivoting techniques such as phrase table pivoting to create DA-to-MSA SMT systems. We can use these DA-MSA phrase tables to automatically learn ELISSA's morpho-syntactic transfer rules, as well as creating better Hybrid ELISSA models.

### 8.2.2 Modeling Dialect Preprocessing From Scratch

We plan to evaluate the work we discussed in Chapter 7 on more dialects and subdialects where DA-English are *not* available. This is analogous to the ADAM tokenization approach, where in-vocabulary stems are given a chance to be translated by separating unseen dialectal affixes from them.

We also plan to apply our approach to tasks other than morphological segmentation. While the rules we extracted from Arabic clusters where a stem is a substring from the source word helped split affixes from stems, it did not attempt at modifying the stem or the

affixes. The Arabic cluster for a given word contains words that are not substrings, yet they are very close to the source word. Here are two examples:

1. **Some words are a spelling variation of the source word.** We can use Levenstein distance to identify these words (with a threshold cost) and add stem-to-stem rules. When these rules are combined with other rules in the segmentation graph, the rule expansion step will produce rules that perform orthographic normalization side by side with morphological segmentation. While Levenstein distance can be used to penalize the $a2s$ score, we might need to add a *character mutation* probability to the model to be estimated with EM. This probability can later be used as a feature in the segmenter to allow unseen to be normalized.

2. **Some words are a translation/synonym of the source word.** The five levels of sentence dialectness discussed in Chapter 1 pose great challenges to machine translation. Although our system combination work alleviated this problem, the biggest error we found in our manual error analysis was code switching within the same sentence. Since word embedding builds the word's vector representation while predicting its context, the varying code switching across all five levels, when all monolingual data are combined, provides a great help for our embedding-based monolingual clustering. This is because the majority of code switching happens between the speaker's dialect and MSA, and as a result, the same MSA context surrounding words of different dialects brings these words' vectors together in the same vector space. This results in Arabic clusters with synonyms in the same dialect and translations across dialects (including MSA). This motivates a replacement for the rule-based ELISSA approach where infrequent words are translated to frequent words as opposed to MSA equivalents. Like the previous item, stem-to-stem rules can be added and extended in the rule expansion step.

### 8.2.3 System Combination of All Approaches

We plan on experimenting with different system combination architectures that combine all approaches discussed above and direct translation models. To better handle code switching within the same sentences, we plan to explore confusion-network combination and re-ranking techniques based on target language features along side source language features extracted from code switching points in the DA sentence.

# References


Hitham Abo Bakr, Khaled Shaalan, and Ibrahim Ziedan. 2008. A Hybrid Approach for Converting Written Egyptian Colloquial Dialect into Diacritized Arabic. In *The 6th International Conference on Informatics and Systems, INFOS2008*. Cairo University.

Murat Akbacak, Dimitra Vergyri, Andreas Stolcke, Nicolas Scheffer, and Arindam Mandal. 2011. Effective arabic dialect classification using diverse phonotactic models. In *INTERSPEECH*, volume 11, pages 737–740.

Ahmad Al-Jallad. 2017. The earliest stages of arabic and its linguistic classification. In *The Routledge Handbook of Arabic Linguistics*, pages 315–331. Routledge.

Rania Al-Sabbagh and Roxana Girju. 2010. Mining the Web for the Induction of a Dialectical Arabic Lexicon. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *LREC*. European Language Resources Association.

Imad Al-Sughaiyer and Ibrahim Al-Kharashi. 2004. Arabic Morphological Analysis Techniques: A Comprehensive Survey. *Journal of the American Society for Information Science and Technology*, 55(3):189–213.

Sarah Alkuhlani and Nizar Habash. 2011. A Corpus for Modeling Morpho-Syntactic Agreement in Arabic: Gender, Number and Rationality. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL'11)*, Portland, Oregon, USA.

Amjad Almahairi, Kyunghyun Cho, Nizar Habash, and Aaron C. Courville. 2016. First result on arabic neural machine translation. *CoRR*, abs/1606.02680.

Mohamed Altantawy, Nizar Habash, and Owen Rambow. 2011. Fast Yet Rich Morphological Analysis. In *Proceedings of the 9th International Workshop on Finite-State Methods and Natural Language Processing (FSMNLP 2011)*, Blois, France.

Mohammed Attia, Pavel Pecina, Antonio Toral, and Josef van Genabith. 2013. A corpus-based finite-state morphological toolkit for contemporary arabic. *Journal of Logic and Computation*, page exs070.

Mohammed Attia. 2008. *Handling Arabic Morphological and Syntactic Ambiguity within the LFG Framework with a View to Machine Translation.* Ph.D. thesis, The University of Manchester, Manchester, UK.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan.

Kenneth Beesley, Tim Buckwalter, and Stuart Newton. 1989. Two-Level Finite-State Analysis of Arabic Morphology. In *Proceedings of the Seminar on Bilingual Computing in Arabic and English*, page n.p.

Fadi Biadsy, Julia Hirschberg, and Nizar Habash. 2009. Spoken arabic dialect identification using phonotactic modeling. In *Proceedings of the Workshop on Computational Approaches to Semitic Languages at the meeting of the European Association for Computational Linguistics (EACL), Athens, Greece.*

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. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19:263–312.

Tim Buckwalter. 2004. Buckwalter Arabic Morphological Analyzer Version 2.0. LDC catalog number LDC2004L02, ISBN 1-58563-324-0.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 17–24.

David Chiang, Mona Diab, Nizar Habash, Owen Rambow, and Safiullah Shareef. 2006. Parsing Arabic Dialects. In *Proceedings of the European Chapter of ACL (EACL)*.

Mathias Creutz and Krista Lagus. 2002. Unsupervised discovery of morphemes. In *ACL'02 Workshop on Morphological and Phonological Learning*. ACL, July.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(1).

Mona Diab, Kadri Hacioglu, and Daniel Jurafsky. 2007. Automated Methods for Processing Arabic Text: From Tokenization to Base Phrase Chunking. In Antal van den Bosch and Abdelhadi Soudi, editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Kluwer/Springer.

Bonnie J. Dorr, Pamela W. Jordan, and John W. Benoit. 1999. A Survey of Current Research in Machine Translation. In M. Zelkowitz, editor, *Advances in Computers, Vol. 49*, pages 1–68. Academic Press, London.

Jinhua Du, Jie Jiang, and Andy Way. 2010. Facilitating translation using source language paraphrase lattices. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP'10, pages 420–429, Cambridge, Massachusetts.

Kevin Duh and Katrin Kirchhoff. 2005. POS tagging of dialectal Arabic: a minimally supervised approach. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, Semitic '05, pages 55–62, Ann Arbor, Michigan.

Ahmed El Kholy and Nizar Habash. 2010. Techniques for Arabic Morphological Detokenization and Orthographic Denormalization. In *Proceedings of the seventh International Conference on Language Resources and Evaluation (LREC)*, Valletta, Malta.

Heba Elfardy and Mona Diab. 2012. Token level identification of linguistic code switching. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING), IIT Mumbai, India.*

Heba Elfardy and Mona Diab. 2013. Sentence level dialect identification in arabic. In *ACL*, Sofia, Bulgaria, August.

Heba Elfardy, Mohamed Al-Badrashiny, and Mona Diab. 2013. Code switch point detection in arabic. In *Proceedings of the 18th International Conference on Application of Natural Language to Information Systems (NLDB2013)*, MediaCity, UK, June.

Ramy Eskander, Nizar Habash, Ann Bies, Seth Kulick, and Mohamed Maamouri. 2013a. Automatic correction and extension of morphological annotations. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 1–10, Sofia, Bulgaria, August. Association for Computational Linguistics.

Ramy Eskander, Nizar Habash, and Owen Rambow. 2013b. Automatic extraction of morphological lexicons from morphologically annotated corpora. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1032–1043, Seattle, Washington, USA, October. Association for Computational Linguistics.

Ramy Eskander, Nizar Habash, Owen Rambow, and Arfath Pasha. 2016a. Creating resources for dialectal arabic from a single annotation: A case study on egyptian and levantine. In *Coling: The 26th International Conference on Computational Linguistics*.

Ramy Eskander, Owen Rambow, and Tianchun Yang. 2016b. Extending the use of adaptor grammars for unsupervised morphological segmentation of unseen languages. In *Coling: The 26th International Conference on Computational Linguistics*.

Ferguson. 1959. *Diglossia. Word 15. 325340.*

David Graff and Christopher Cieri. 2003. English Gigaword, LDC Catalog No.: LDC2003T05. Linguistic Data Consortium, University of Pennsylvania.

David Graff, Mohamed Maamouri, Basma Bouziri, Sondos Krouna, Seth Kulick, and Tim Buckwalter. 2009. Standard Arabic Morphological Analyzer (SAMA) Version 3.1. Linguistic Data Consortium LDC2009E73.

Nizar Habash and Owen Rambow. 2005. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 573–580, Ann Arbor, Michigan.

Nizar Habash and Owen Rambow. 2006. MAGEAD: A Morphological Analyzer and Generator for the Arabic Dialects. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 681–688, Sydney, Australia.

Nizar Habash and Fatiha Sadat. 2006. Arabic Preprocessing Schemes for Statistical Machine Translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 49–52, New York City, USA.

Nizar Habash, Abdelhadi Soudi, and Tim Buckwalter. 2007. On Arabic Transliteration. In A. van den Bosch and A. Soudi, editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Springer.

N. Habash, O. Rambow, M. Diab, and R. Kanjawi-Faraj. 2008. Guidelines for Annotation of Arabic Dialectness. In *Proceedings of the LREC Workshop on HLT & NLP within the Arabic world*.

N. Habash, R. Eskander, and A. Hawwari. 2012a. A Morphological Analyzer for Egyptian Arabic. In *NAACL-HLT 2012 Workshop on Computational Morphology and Phonology (SIGMORPHON2012)*, pages 1–9.

Nizar Habash, Mona Diab, and Owen Rabmow. 2012b. Conventional Orthography for Dialectal Arabic. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Istanbul.

Nizar Habash, Ramy Eskander, and Abdelati Hawwari. 2012c. A Morphological Analyzer for Egyptian Arabic. In *Proceedings of the Twelfth Meeting of the Special Interest Group on Computational Morphology and Phonology*, pages 1–9, Montréal, Canada.

Nizar Habash, Ryan Roth, Owen Rambow, Ramy Eskander, and Nadi Tomeh. 2013. Morphological Analysis and Disambiguation for Dialectal Arabic. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Atlanta, GA.

Nizar Habash. 2006. On Arabic and its Dialects. *Multilingual Magazine*, 17(81).

Nizar Habash. 2007. Arabic Morphological Representations for Machine Translation. In A. van den Bosch and A. Soudi, editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Springer.

Nizar Habash. 2010. *Introduction to Arabic Natural Language Processing*. Morgan & Claypool Publishers.

Jan Hajič, Jan Hric, and Vladislav Kubon. 2000. Machine Translation of Very Close Languages. In *Proceedings of the 6th Applied Natural Language Processing Conference (ANLP'2000)*, pages 7–12, Seattle.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18.

Ahmed Hamdi, Rahma Boujelbane, Nizar Habash, Alexis Nasr, et al. 2013. The effects of factorizing root and pattern mapping in bidirectional tunisian-standard arabic machine translation. *MT Summit 2013*.

S. Hasan, O. Bender, and H. Ney. 2006. Reranking translation hypotheses using structural properties. EACL'06 Workshop on Learning Structured Information in Natural Language Applications.

Xiaodong He, Mei Yang, Jianfeng Gao, Patrick Nguyen, and Robert Moore. 2008. Indirect-hmm-based hypothesis alignment for combining outputs from machine translation systems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 98–107. Association for Computational Linguistics.

William John Hutchins. 1986. *Machine translation: past, present, future*. Ellis Horwood Chichester.

Damianos Karakos, Jason Eisner, Sanjeev Khudanpur, and Markus Dreyer. 2008. Machine translation system combination using itg-based alignments. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 81–84. Association for Computational Linguistics.

H. Kilany, H. Gadalla, H. Arram, A. Yacoub, A. El-Habashi, and C. McLemore. 2002. Egyptian Colloquial Arabic Lexicon. LDC catalog number LDC99L22.

George Anton Kiraz. 2000. Multitiered nonlinear morphology using multitape finite automata: a case study on Syriac and Arabic. *Computational Linguistics*, 26(1):77–105, March.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical Phrase-based Translation. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference (HLT/NAACL)*, pages 127–133, Edmonton, Canada.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Christopher Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Christopher Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic.

Philipp Koehn. 2004a. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July. Association for Computational Linguistics.

Philipp Koehn. 2004b. Statistical significance tests formachine translation evaluation. In *Proceedings of the Empirical Methods in Natural Language Processing Conference (EMNLP'04)*, Barcelona, Spain.

Philipp Koehn. 2009. *Statistical machine translation*. Cambridge University Press.

Shankar Kumar, Franz J. Och, and Wolfgang Macherey. 2007. Improving word alignment with bridge languages. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 42–50, Prague, Czech Republic.

Young-Suk Lee. 2004. Morphological Analysis for Statistical Machine Translation. In *Proceedings of the 5th Meeting of the North American Chapter of the Association for Computational Linguistics/Human Language Technologies Conference (HLT-NAACL04)*, pages 57–60, Boston, MA.

Wei-Yun Ma and Kathleen McKeown. 2013. Using a supertagged dependency language model to select a good translation in system combination. In *Proceedings of NAACL-HLT*, pages 433–438.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 2004. The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus. In *NEMLAR Conference on Arabic Language Resources and Tools*, pages 102–109, Cairo, Egypt.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, Mona Diab, Nizar Habash, Owen Rambow, and Dalila Tabessi. 2006. Developing and Using a Pilot Dialectal Arabic Treebank. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation, LREC'06*, Genoa, Italy.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*.

Emad Mohamed, Behrang Mohit, and Kemal Oflazer. 2012. Annotating and Learning Morphological Segmentation of Egyptian Colloquial Arabic. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Istanbul.

Preslav Nakov and Hwee Tou Ng. 2011. Translating from Morphologically Complex Languages: A Paraphrase-Based Approach. In *Proceedings of the Meeting of the Association for Computational Linguistics (ACL'2011)*, Portland, Oregon, USA.

Karthik Narasimhan, Regina Barzilay, and Tommi Jaakkola. 2015. An unsupervised method for uncovering morphological chains. *Transactions of the Association for Computational Linguistics (TACL)*.

Andrew Y Ng and Michael I Jordan. 2002. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes.

Sergei Nirenburg. 1989. Knowledge-based machine translation. *Machine Translation*, 4(1):5–24.

F. J. Och and H. Ney. 2003a. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och and Hermann Ney. 2003b. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–52.

Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Comput. Linguist.*, 30(4):417–449.

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Franz Josef Och. 2004. A smorgasbord of features for statistical machine translation. In *Meeting of the North American chapter of the Association for Computational Linguistics.*

Franz Josef Och. 2003. Minimum Error Rate Training for Statistical Machine Translation. In *Proceedings of the 41st Annual Conference of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, PA.

Robert Parker, David Graff, Ke Chen, Junbo Kong, and Kazuaki Maeda. 2009. Arabic Gigaword Fourth Edition. LDC catalog number No. LDC2009T30, ISBN 1-58563-532-4.

Arfath Pasha, Mohamed Al-Badrashiny, Mona T Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of arabic. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*.

Jason Riesa and David Yarowsky. 2006. Minimally Supervised Morphological Segmentation with Applications to Machine Translation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (AMTA06)*, pages 185–192, Cambridge,MA.

Antti-Veikko I Rosti, Spyridon Matsoukas, and Richard Schwartz. 2007. Improved word-level system combination for machine translation. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 312.

Ryan Roth, Owen Rambow, Nizar Habash, Mona Diab, and Cynthia Rudin. 2008. Arabic Morphological Tagging, Diacritization, and Lemmatization Using Lexeme Models and Feature Ranking. In *Proceedings of ACL-08: HLT, Short Papers*, pages 117–120, Columbus, Ohio.

Fatiha Sadat and Nizar Habash. 2006. Combination of Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the 21st International Conference on*

*Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 1–8, Sydney, Australia, July. Association for Computational Linguistics.

Hassan Sajjad, Kareem Darwish, and Yonatan Belinkov. 2013. Translating dialectal arabic to english. The 51st Annual Meeting of the Association for Computational Linguistics - Short Papers (ACL Short Papers 2013), Sofia, Bulgaria.

Wael Salloum and Nizar Habash. 2011. 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*, pages 10–21, Edinburgh, Scotland.

Wael Salloum and Nizar Habash. 2013. Dialectal Arabic to English Machine Translation: Pivoting through Modern Standard Arabic. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Atlanta, GA.

Hassan Sawaf. 2010. Arabic dialect handling in hybrid machine translation. In *Proceedings of the Conference of the Association for Machine Translation in the Americas (AMTA)*, Denver, Colorado.

Patrick Schone and Daniel Jurafsky. 2000. Knowledge-free induction of morphology using latent semantic analysis. In *Proceedings of CoNLL-2000 and LLL-2000*, pages 67–72, Lisbon, Portugal.

Otakar Smrž. 2007. ElixirFM — Implementation of Functional Arabic Morphology. In *Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources*, pages 1–8, Prague, Czech Republic, June. ACL.

David Stallard, Jacob Devlin, Michael Kayser, Yoong Keok Lee, and Regina Barzilay. 2012. Unsupervised morphology rivals supervised morphology for arabic mt. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 322–327. Association for Computational Linguistics.

Andreas Stolcke. 2002. SRILM an Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing*.

Masao Utiyama and Hitoshi Isahara. 2007. A comparison of pivot methods for phrase-based statistical machine translation. In *HLT-NAACL*, pages 484–491.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Daguang Xu, Yuan Cao, and Damianos Karakos. 2011. Description of the jhu system combination scheme for wmt 2011. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 171–176. Association for Computational Linguistics.

Omar F Zaidan and Chris Callison-Burch. 2011. The arabic online commentary dataset: an annotated dataset of informal arabic with high dialectal content. In *Proceedings of ACL*, pages 37–41.

Nasser Zalmout and Nizar Habash. 2017. Optimizing tokenization choice for machine translation across multiple target languages. *The Prague Bulletin of Mathematical Linguistics*, 108(1):257–269.

Rabih Zbib, Erika Malchiodi, Jacob Devlin, David Stallard, Spyros Matsoukas, Richard Schwartz, John Makhoul, Omar F. Zaidan, and Chris Callison-Burch. 2012. 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*, pages 49–59, Montréal, Canada, June. Association for Computational Linguistics.

R. Zens, F. J. Och, and H. Ney. 2002. Phrase-based statistical machine translation. In M. Jarke, J. Koehler, and G. Lakemeyer, editors, *KI - 2002: Advances in Artificial Intelligence. 25. Annual German Conference on AI*. Springer Verlag.

Xiaoheng Zhang. 1998. Dialect MT: a case study between Cantonese and Mandarin. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, ACL '98, pages 1460–1464, Montreal, Canada.

Andreas Zollmann, Ashish Venugopal, and Stephan Vogel. 2006. 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, New York City, USA.