The Raymond and Beverly Sackler Faculty of Exact Sciences

The Blavatnik School of Computer Science

Deriving Paraphrases for Highly Inflected Languages, with a Focus on Machine Translation

Thesis submitted for the degree of Doctor of Philosophy

by

Kfir Bar

This work was carried out under the supervision of

Professor Nachum Dershowitz

Submitted to the Senate of Tel Aviv University

October 2013

Abstract

Paraphrasing is the act of generating an alternate sequence of words that conveys the same meaning. In this work, we explore the potential of using paraphrases to improve a corpus-based translation system, designed to translate a morphologically rich language into English. We focus on Arabic, a highly inflected language, whose words are generated by a comprehensive derivational and inflectional morphological system.

We describe an automatic data-driven paraphrasing procedure for Arabic, starting with two limited case studies. Our procedure utilizes comparable documents, that is, distinct documents covering the same topic, for learning the characteristics of potential paraphrases. A co-training approach is taken, with two classifiers, one designed to model the contexts surrounding occurrences of paraphrases, and the other trained to identify significant features of the words within paraphrases. In particular, we use morpho-syntactic features calculated for both classifiers, which proved to be effective for this task.

We employ a simplified version of our paraphrasing procedure to support a corpus-based translation system by deriving paraphrases for fragments of the input Arabic text. The experimental results were found to be encouraging. Our best system shows an increase of 1.73 in BLEU.

Simultaneously, we initiated a study of Arabic multiword expressions, a pervasive semantic apparatus in a natural language. In collaboration with another group, we created a repository of Arabic expressions, annotated by their category and enriched with linguistic information. A classifier is created for identifying expressions in running Arabic text, focusing on non-compositionality. We provide and discuss some results of our experiments.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1 Introduction

In this work, we explore two semantic phenomena of a highly inflected language with a special focus on improving automatic translation systems. In particular, we investigate automated ways to learn semantic equivalents, or paraphrases, that is, two fragments of text that have the same meaning in some contexts, and ways to learn multiword expressions, that is, sequences of words that co-occur statistically more than a chance, and whose semantics may include more than the compositional meaning of the individual words. Most of our experiments have been conducted on Arabic, a morphologically rich language and a member of the Semitic language family. Among other interesting characteristics, Semitic languages are based on complicated derivational as well as inflectional morphologies. Furthermore, the lack of short vowels, in writing, increases the level of ambiguity of a written Arabic word. Choosing to work on Arabic was natural, since Arabic, as opposed to other Semitic languages, has been widely investigated over the last several years, resulting in a relatively large pool of useful resources.

In this introductory chapter, we provide some background and motivation for our work, on which we elaborate in subsequent chapters. We begin by extracting Arabic single-word noun paraphrases, that is, synonyms, from a repository of words combined with WordNet 2.0 (Miller, 1995; Fellbaum, 1998), taking a direct approach. We continue with finding synonymous verbs in comparable documents, that is, different articles covering the same story. Thirdly, we infer Arabic multiword paraphrases from comparable documents, employing a machine-learning algorithm, based on the “co-training” technique (Blum and Mitchell, 1998). In the next stage, we use our paraphrasing methods to improve an automatic corpus-based Arabic-to-English translation system. In Chapter 6, we summarize our initial study on Arabic multiword expressions, joint work with colleagues at the Center for Computational Learning Systems (CCLS) at Columbia University. Figure 1-1 illustrates the components of this work and their interrelationships.

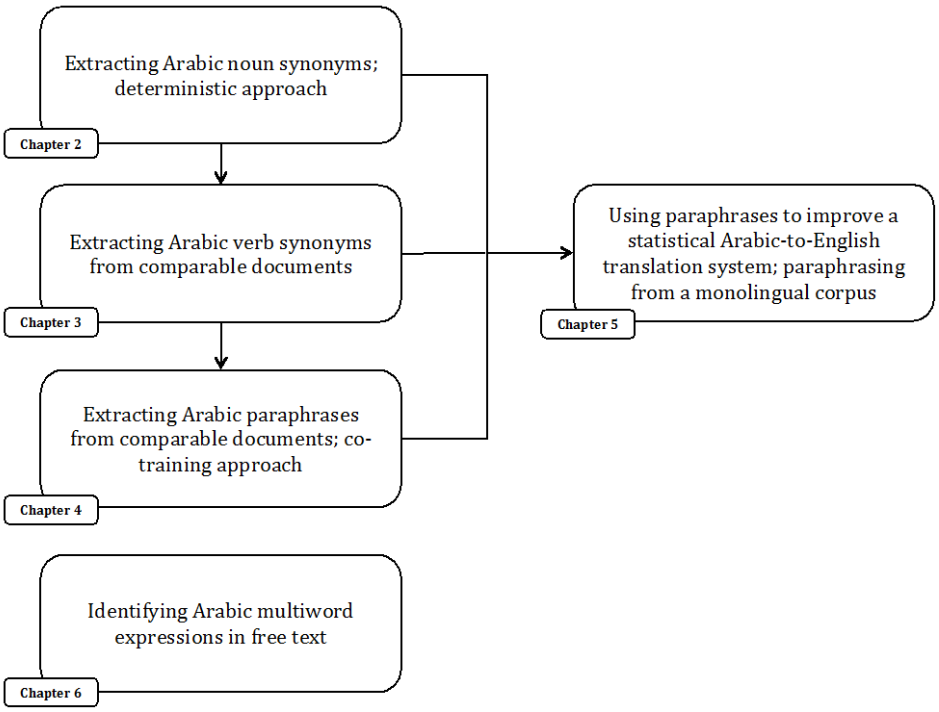


FIGURE 1-1 - Thesis structure.

The following is a summary of our relevant publications with their corresponding chapters:

- **Chapter 2:**

  Kfir Bar and Nachum Dershowitz. 2010. Using Synonyms for Arabic-to-English Example-Based Translation. In *Proceedings of the Association for Machine Translation in the Americas 9th Biennial Conference*, Denver, CO.

- **Chapter 3:**

  Kfir Bar and Nachum Dershowitz. 2012. Using Semantic Equivalents for Arabic-to-English Example-Based Translation". In *Challenges for Arabic Machine Translation. Edited by Abdelhadi Soudi, Ali Farghaly, Günter Neumann and Rabih Zbib, John Benjamins Publishing Company*, pages 49-72.

- **Chapter 4:**

  Kfir Bar and Nachum Dershowitz. 2012. Deriving Paraphrases for Highly Inflected Languages from Comparable Documents. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, pages 185-200, Mumbai, India.

- **Chapter 5:**

  Kfir Bar and Nachum Dershowitz. **Submitted (Oct 2013)**. Inferring Paraphrases for a Highly Inflected Language From a Monolingual Corpus.

- **Chapter 6:**

  Abdelati Hawwari Kfir Bar, and Mona Diab. 2012. Building an Arabic Multiword Expressions Repository. In *Proceedings of the Association for Computational Linguistics (ACL), Joint Workshop on Statistical Parsing and Semantic Processing of Morphologically Rich Languages*, pages 24-29, Jeju Island, Republic of Korea.

  Kfir Bar, Mona Diab and Abdelati Hawwari. Arabic Multiword Expressions. 2013. **To appear in** *Language, Culture, Computation: Studies in Honor of Yaacov Choueka*, *volume II, Nachum Dershowitz and Ephraim Nissan (eds.), Lecture Notes in Computer Science*, *vol. 8001, Springer-Verlag*. Berlin.

The following are the main contributions of this dissertation:

- A novel method for learning paraphrases from comparable documents and monolingual documents.
- The first work to deal with paraphrasing in Arabic, a highly inflected language.
- A method for identifying multiword expressions in running Arabic text, for the first time.
- Using Arabic paraphrases in statistical machine translation as word lattices, exploiting various weighting conditions and comparing the contributions of various levels of paraphrases, tuned by Minimum Error Rate Training (MERT).

We proceed as follows: In Section 1.1 we provide some background regarding the concept of paraphrases in general; in Section 1.2 we explore the notion of morphologically rich languages, followed by Section 1.3, which focuses specifically on Arabic. Section 1.4 is about the foundations of machine translation, and in Section 1.5 we give some background regarding multiword expressions. We present an overview on the co-training technique in Section 1.6, and conclude with Section 1.7, which is a comprehensive literature review.

1.1 Paraphrases

Paraphrases, or semantic equivalents, are pairs of sequences of words, both in the same language, that have the same meaning in at least some contexts. We usually distinguish between two levels of paraphrases: (1) *phrase* (sub-sentential) level refers to two text segments of varying size, and includes single words, better known as "synonyms"; and (2) *sentence* level, composed of two complete sentences. Figure 1-2 shows examples of both levels. In the first example, the phrases, *I spilled the beans* and *I exposed my secret*, have the same meaning in this context, hence are considered paraphrases. The second example is composed of two full sentences with the same meaning. Since the sense of a text fragment is determined only when its context is given, paraphrases are sometimes referred to as "dynamic translations".

Example 1 (phrase level):

(a) **I spilled the beans** and told Jacky I loved her

(b) **I exposed my secret** about my personal life

Example 2 (sentence level):

(a) **Beijing's policy toward Taiwan remains unchanged**

(b) **China did not change its policy toward Taiwan**

FIGURE 1-2 – Examples for phrase and sentence level paraphrases. The paraphrases are in boldface.

It is common to distinguish between several types of paraphrases. *Structural* paraphrases use comparable syntactic structures to express the same meaning. For instance, there is passive voice vs. active voice, as in *She ate the apple* vs. *The apple was eaten by her*; or possessive forms using *of* vs. using *'s*, as in *Jane's book* vs. *The book of Jane*.

Another type involves lexical differences, such as synonymous words expressing the same meaning. We usually refer to these as *lexical* paraphrases. For example, *My **horse** galloped away* vs. *My **mount** galloped away*. Similarly, *semantic* paraphrases involve phrasal differences; for example, *I don't have enough money to buy this yacht* and *I can't afford this yacht* are semantic paraphrases. A special case of semantic paraphrases is *idiomatic* paraphrases, referring to idiomatic expressions paired with compositional ways for expressing the same meaning. By way of example, the phrases (Figure 1-2) *I*

spilled the beans paired with *I exposed the secret*, are idiomatic paraphrases. Here, *I exposed the secret* is a compositional way to express the same meaning as the idiomatic expression *I spilled the beans*. Idiomatic expressions are part of a larger phenomenon, referred to as "multiword expressions", or just "expressions" for short. An *expression* is a multiword unit or a collocation of words that occur together statistically more often than by chance. Identifying idiomatic expressions is an important capability for many applications, especially machine translation, where expressions may be translated word-for-word incorrectly. In Chapter 6, we explore the topic of identifying Arabic multiword expressions in free text.

Another type of paraphrase is known as *referential* paraphrases. This typically refers to a concept that has the same meaning of the original phrase. For example, *Tuesday* vs. *The day before Wednesday*, or *Barack Obama* vs. *The current president of the US*. The last example requires a context that posits the current year, as described below. In this thesis we focus mainly on lexical and semantic paraphrases.

Paraphrases are in fact only one of several semantic relations that can hold between two phrases with their contexts; it can be seen as a special case of textual entailment (Dagan and Glickman, 2004), where each unit entails the other. For textual entailment, we require that a target textual assertion, known as the *hypothesis*, can be inferred from a given text. Figure 1-3 is an example of textual entailment. As implied, our focus is on pairs for which entailment holds in both directions, unlike in this example.

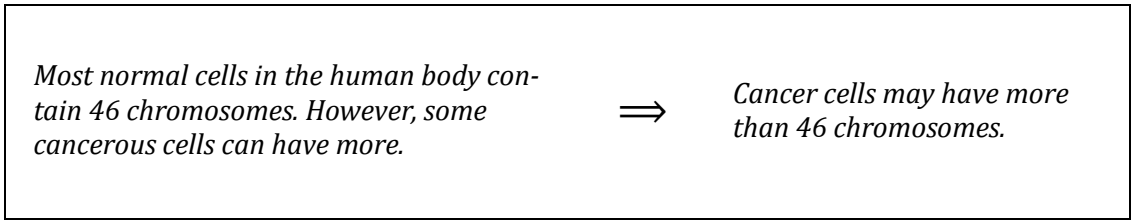


FIGURE 1-3 – An example of textual entailment.

We now provide some definitions to formalise what we have just described, borrowing notation from logic. Let *F*, *G* be textual units of any size, parsed for syntax and tagged for sense (phrases, including single words, may have multiple senses, a phenomenon that is usually referred to as *polysemy*). For simplicity, the context that is necessary for tagging the senses as well as the tagging method, are not relevant for the following definitions.

Textual entailment We write

$$C \vDash F \Rightarrow G$$

to mean that F subsumes G within the context C. The context C may be textual or may involve external knowledge. We say that *F textually entails G in context C*. Then, any "positive" occurrence of F in a sentence may be replaced by G provided C holds.

Equivalence Similarly, we write

$$C \vDash F \Leftrightarrow G$$

and say that *F and G are textually equivalent in context C*. In this case, any occurrences of F and G are interchangeable in sentences in which C holds. Alternatively, we say that *F and G are paraphrases in context C*, and as a special case, if F and G are composed of single words, we may say that they are *synonyms* in context C. Additionally, we write

$$\vDash F \Leftrightarrow G$$

when F entails G unconditionally (in all contexts).

For example, the following are possible assertions:

(a) $\vDash horse \Rightarrow quadruped$

Here, *horse* refers to the specific sense of the animal, as in *My white horse galloped away*, which entails *My white quadruped galloped away*. This substitution would not work in a "negative" position: *It was not my horse that rode away* does not necessary entail *It was not my quadruped that rode away*.

(b) $\vDash horse \Leftrightarrow mount$

as in *My white mount galloped away*. Here, the words *horse* and *mount* refer to the sense that makes them synonyms, hence are interchangeable in either positive or negative occurrences.

(c) $\vDash gallop \Rightarrow ride$

as in *My white horse rode away*. Here, *gallop* and *ride* are two verbs.

(d) $\vDash gallop \Leftrightarrow sprint$

as in *My white horse sprinted away*. Provided with those senses, the verbs *gallop* and *sprint* are synonyms.

(e) $\vDash My\ white\ horse\ galloped\ away\ gracefully\ this\ morning \Rightarrow$
$My\ horse\ galloped\ away$

Here, the entailment is formed by removing some of the phrasal modifiers (*white*, *gracefully this morning*).

(f) {year=1863} ⊨ $Abe\ Lincoln \Rightarrow living\ Republican$

Here, the entailment is correct only in the provided context. We also have

{Abraham Lincoln} ⊨ $Abraham \Rightarrow Abe$;

{year=1863} ⊨ $Abraham\ Lincoln \Rightarrow US\ President$.

(g) {focus=Jane} ⊨ $Jane\ rode\ her\ horse \Rightarrow She\ rode\ her\ horse$

Here, the context refers to anaphora resolution.

(h) ⊨ $Booth\ shot\ Lincoln \Rightarrow Lincoln\ was\ shot\ by\ Booth$

A change of voice is independent of context.

Paraphrase generation, or *paraphrasing*, is an important capability for many natural-language processing applications, including machine translation, as a possible work-around for the problem of limited coverage inherent in a corpus-based translation approach (Callison-Burch et al., 2006; Marton et el., 2009). Other applications include automatic evaluation of summaries (Zhao et al., 2008) and question answering (Duboue and Chu-Carroll, 2006; Riezler et al., 2007).

As an addendum, it is worth mentioning that some linguists are skeptical about the whole concept of paraphrases. Chafe (1971) argued that pairs of phrases such as those that we consider in this work to be paraphrases may have similar meanings, but cannot be considered identical. In his main argument, he questioned the notion that two sentences have the same meaning just because they possess the same truth-value. By way of example, he referred to the claim that a passive sentence means the same thing as its corresponding active phrasing. He suggested that saying *Oculists eye blondes* is not like saying *Blondes are eyed by oculists*. The first sentence focuses on something that oculists do, whereas the second focuses on something that happens to blondes. As another example, he claims that saying *The old lady kicked the bucket* is not comparable to saying *The old lady died*. The idiomatic expression *kick the bucket* has a different connotation than *die*. For example, he wrote, a sensitive English speaker would say that *Bertrand Russell died*, but not *Bertrand Russell kicked the bucket*.

Our main purpose in this work is to identify paraphrases that have an equivalent meaning to fragments of text in need of translation. We are aware that doing so may sometimes modify the overall atmosphere as expressed by the author of the original text.

In this work we explore several approaches for paraphrasing, focusing on Arabic, and use them to improve a corpus-based Arabic-to-English translation system.

As explained below, we mainly work on the lemma level, given that Arabic is a morphologically rich language. For that reason, we include among paraphrases other pairs of phrases that express the same meaning, regardless of their inflection for number, gender, and person. Although it may look wrong to say that the Arabic phrase *ktb AlmElm*,[1] "the teacher [*masculine*] wrote", is a paraphrase of *ktbt AlmElmp*, "the teacher [*feminine*] wrote", the English translation of both is identical. We are after improving machine translation by considering different wordings of fragments of the input text, one of which may result in a translation faithful to the original meaning. However, our generous definition of paraphrase takes this even further. Although English has a shallow morphology compared to Arabic, sometimes the different inflections of a pair of Arabic paraphrases may percolate to their English translations: *ktb AlmElm*, "the teacher [*masculine*] wrote", and *ktbt AlmElmAt*, "the teacher**s** [*feminine*] wrote", for example. Obviously, the English translations are not identical, but they are sufficiently similar for the purpose. Such cases can still be useful for machine translation, especially when the translation system cannot find a translation for one of the phrases.

1.2 Morphologically Rich Languages

Morphologically rich languages are known for their highly productive morphological processes that may produce a very large number of word forms, given one basic form combined with additional *morphemes*. A morpheme is the smallest grammatical unit in a language, which can be either a word by itself, also known as a *free morpheme*, or part of a word, also known as a *bound morpheme*, as in +*ing* in the verb *walking*.

Theoretically, following Eifring and Theil (2005), the complexity of the morphological system of a language may be measured as demonstrated in Figure 1-4. Analytic languages like Chinese, simply use the basic word forms in a sentence without any modifications. Such languages usually provide an extensive system of functional words for expressing various grammatical situations. Synthetic languages, on the other hand, allow one to change the basic form of a word to reflect some grammatical and functional roles. English is considered a light synthetic language, and in fact would be probably located slightly to the left of the "synthetic" node. Arabic, Hebrew, and some other Semitic languages are considered highly synthetic, introducing a complicated system of *inflectional morphology*. With inflectional morphology, words get inflected to express

[1] We use the Buckwalter transliteration for rendering Arabic script in ASCII (Buckwalter, 2002).

their grammatical properties while continuing to represent the same word. For instance, *walking* in the sentence *he is walking*, is an inflected form of the verb *walk*. However, *painting*, in the sentence *this painting is beautiful*, does not represent an inflection of the verb *paint*, but rather a word by itself. In fact, the verb *paint* and the noun *painting* are two distinct related words, such that *painting* is considered a *derivative* of the word *paint*. The machinery that enables such derivatives is often called *derivational morphology*.

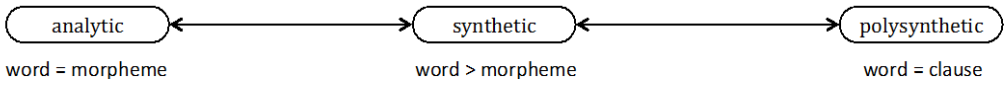


FIGURE 1-4 – Morphology complexity measurement line.

Another important method for word modification is compounding. With compounding, a word may be generated from two independent words. The meaning of such a compound word may be either a linear composition of the individual word meanings, like the English word *keyboard*, or the compound may creation a new meaning, like the English word *horseplay*. Germanic as well as Finnic languages are known for their extensive compounding technique.

In the extreme right end of the line in Figure 1-4, we find the polysynthetic languages that use a very complicated morphological system, which may even form words representing an entire clause. Most of the polysynthetic languages are members of the Eskimo and American Indian language families.

Synthetic and polysynthetic languages are also measured for their agglutination. On one hand there are the *agglutinative* languages, like Turkish, where morphemes express one and only one meaning. It means that among other properties, with agglutinative languages, any morpheme expresses only one meaning, and any set of morphemes are not merged when they are combined to form a word. In other words, each morpheme has its own clear boundaries. On the other hand, *inflective* languages are not bound by this rule, either by allowing morphemes to be merged when applied together or by having one morpheme that express more than one meaning. For example, in the Russian word домов (*domóv*), the *-óv* ending indicates both plurality and the genitive case; in the Arabic verb *yakotubuA*, “they will write”, the *+ubuA* ending indicates plurality in

present tense and the subjunctive mood. Inflective languages are also called *fusional* languages. Arabic, as other Semitic languages, is an inflective language.

In this sense, the class of morphologically rich languages includes any language that is located on the range starting somewhere after the "analytic" node and before the "synthetic" node, ending in the "polysynthetic" node. Obviously this includes agglutinative languages, such as Turkish and other Turkic languages, Korean, Basque, and Japanese; and inflective languages, such as Arabic, Hebrew, Greek, and many Indo-European languages. It is believed that this class of languages probably requires deeper methodologies for dealing with the complexity of the morphology, rather than simply treating every inflected surface form as an individual word. In other words, when processing a morphologically rich language, data sparseness becomes even more noticeable a problem.

1.3 Arabic

According to Ethnologue,[2] Arabic is one of the top five popular languages in the world, with 223 million native speakers. This number includes native speakers of the standard language as well as its many colloquial versions.

In this work we will only deal with Modern Standard Arabic (MSA), also known as literary Arabic (in Arabic: اللغة العربية الفصحى), which is widely used in formal settings. In fact, Arabic has a number of colloquial versions, which are different from each other in vocabulary, grammar, morphology, and pronunciation. Although the general assumption is that the colloquial versions were generated from the same Arabic language, sometimes they are so different such that it is not a rare situation to see a Moroccan Arabic speaker discoursing with a Syrian Arabic speaker in French. The two main problems with processing colloquial Arabic are: (1) the lack of available resources; and (2) the lack of writing standards. Both problems occur due to the fact that the majority of the formal publications in Arabic-speaking countries are written in MSA. Nevertheless, in some countries one can find various types of publications written in the local Arabic version. In fact, probably most of the modern Arabic literature is written in what is called *Middle* Arabic. The term Middle Arabic refers to a modified version of the standard language, highly affected by the local spoken dialect and culture. In this sense, Middle Arabic is a set of languages, each affected by its own local version.

[2] http://www.ethnologue.com

Rosenbaum (2000) defined the term *fushaammiyya* (in Arabic, *fusha* = "standard Arabic" and *ammiyya* = "colloquial Arabic") for Mixed Arabic, referring to the alternating writing style of Egyptian prose. As opposed to the middle language, with Mixed Arabic, people are switching back and forth between the standard language and the local colloquial version.

Most computational works on Arabic, including this thesis, focus on MSA. The Columbia Arabic Dialect Modeling (CADIM)[3] group, a part of the Center for Computational Learning Systems (CCLS) in Columbia University, is working on various aspects of the colloquial angle of Arabic.

It is worth mentioning here the Classical Arabic language, which is considered the ancestor of MSA. Classical Arabic was mostly used between the 4th and the 9th centuries. Being the language of the Quran, Classical Arabic is usually related with religious settings. On one hand, MSA remains quite similar in many aspects, including its morphology and syntax. The vocabulary of MSA, on the other hand, has changed over time, differentiating the modern language from the classical one.

Arabic vowels are divided into long and short ones, where short vowels, represented by diacritic marks, are usually omitted in writing. Therefore, written Arabic words are considerably more ambiguous than English words. In this sense, reading Arabic text is similar to reading the following unvocalized English text: *ths is nt an Arbc sntnc*.

As implied in the previous section, Arabic introduces both inflectional and derivational morphology. Being a key member of the Semitic language family, Arabic is highly inflected; Arabic words are derived from a root and a pattern (template), combined with prefixes and suffixes. The root consists of 3 or 4 consonants and the pattern is a sequence of consonants and variables for root letters. Using the same root with different patterns may yield words with different meanings. Words are then inflected for person, number and gender; proclitics and enclitics are added to indicate definiteness, conjunction, various prepositions, and possessive forms. For instance, the combination of the root *k.t.b* and the pattern *XaYaZa* (here, *X*, *Y,* and *Z* are variables) results in the verb *kataba*, "he wrote". One can inflect this word for different grammatical roles; for example, *katabotu*, "I wrote", is the inflected form of *kataba*, representing the 1st person of the perfect verb form. Combining the same root with the pattern *XaAYaZ* (note the additional *A*), results in a derivational verb كاتب (*kaAtaba*), "he corresponded with".

[3] http://www.ccls.columbia.edu/project/cadim-columbia-arabic-dialect-modeling

Let us now refine some known definitions in the context of this work:

Surface form The original form of the word, as occurring in the text.

Affix A morpheme that is attached to a word to modify its functional meaning. Arabic words carry prefixes, suffixes, and circumfixes, which are used for inflections as well as derivations.

Clitic In Arabic, clitics are divided to proclitics—including conjunctions, various prepositions, and definiteness—and pronoun enclitics, representing possessive forms in nouns and direct object in verbs.

Lemma The lemma represents a collection of inflected word forms that have the same meaning. In fact, a lemma does not represent a specific word form, but only a handle to the collection of words. Nevertheless, following the definitions of the Standard Arabic Morphological Analyser (SAMA) (Maamouri et al., 2010), a verb lemma is usually represented by the single, 3rd person, perfective version of the verb, concatenated with a number to represent the specific sense of that verb (important in cases of *polysemy*, that is, a lemma that have more than one meaning). For example, *kataba_1* is the lemma of the set of all inflected forms of the verb *kataba*, "he wrote", including both perfective and imperfective forms, and all inflections, with or without additional pro- and enclitics. We have found in most of our experiments that focus on the semantics that our algorithms benefit from working on the lemma level, being that the lemma represents a group of words that carry the same meaning

Stem The stem is the shortest morpheme that can be reached by removing clitics and affixations. For instance, the stem of ***katab***otu, "I wrote", is *katab*, retrieved by removing the suffix *otu*. Similarly, the stem of the verb *wasaya**kotub**uwna*, "and they will write", is *kotub*, removing the prefix *wasaya* and the suffix *uwna*. Note that the stems *katab* and *kotub* do not represent a specific word form, but only the shortest morpheme after removing clitics and affixations. In fact, you may notice that any verbs that can be generated from the stem *kotub* (e.g., *ya**kotub**u*, "he will write", *ta**kotub**u*, "she will write") as well as *katab* (e.g., ***katab**at*, "she wrote", ***katab**ona*, "we wrote") are all represented by the lemma *kataba_1*, from the previous example. The stem *katab* represents all the perfective forms (used to indicate past tense) and the stem *kotub* represents all the imperfective forms (used to indicate present and future tenses). Essentially, a verb is represented by two stems: perfective and imperfective forms, which are both related to the same lemma. Figure 1-5 illustrates this definition.

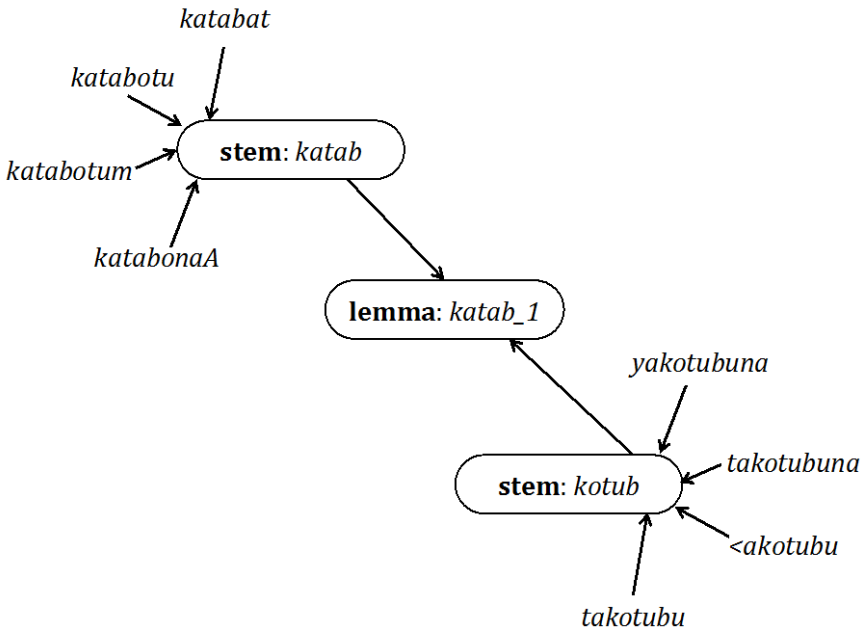


FIGURE 1-5 – An illustration of the stem-lemma relation.

A similar structure applies to nouns. Most Arabic nouns get inflected for plurality and duality using regular suffixes, also known as *sound* plurals. Arabic nouns are typically assigned a gender. Therefore, generating a sound plural form is done by adding the suffix *uwn* for a masculine noun, as in ***muslim****uwn*, "Muslims (*male*)", and the suffix *aAt* for a feminine noun, as in ***muslim****aAt*, "Muslims (*female*)". However, there are a relatively large number of irregular nouns the plural form of which is not generated using this rule; such forms are called *broken plurals*. For example, the plural form of *TAlb*, "student", is either *TlAb* or *Tlbp*. As in the verb case, broken plurals usually represent different stems, sharing the same lemma with singular forms.

Compound nouns are formed using what is called a *construct state*, or *Idafa* in Arabic, where a noun is combined with another one, typically to indicate a possessive relation, as in *baAbu {lmanozali*, "the door of the house". The first noun is the "head" or the "thing possessed" (in Arabic – *muDaAf*, literally "attached"), and the second noun is the "dependent" or the "possessor" (in Arabic – *muDaAf >ilayohi*, literally "attached to"), marked with the *genitive* case. Therefore, a noun can be in either of three states: definite, indefinite, or construct.

Another important characteristic for a noun is its case. In general, there are three cases in MSA: nominative, genitive, and accusative. For the most part, cases are marked by additional short vowels on the last letter, which are usually omitted in writing. How-

ever, sometimes this process requires changing or adding long vowels. For example, the nominative case of the definite word *Albayot*, "house" is *Albayot**u***, adding the short vowel *u* to the last letter. However, the accusative case of the word *walad*, "child", is *wal-ad**FA***, when the short vowel *F* is added with the long vowel *A*. (Essentially, the purpose of cases is to indicate the grammatical function of every word in a sentence. The nominative case is for indicating the subject; the accusative case is typically for indicating the object of a transitive verb; and the genitive case is typically for indicating the object of a preposition and the possessor of a construct state, as mentioned above.)

Cases are applied only to nouns and their modifiers; similarly, Arabic verbs are characterized by their mood. There are four moods: indicative, subjunctive, jussive, and imperative. Like cases, moods are usually marked with short vowels, which are omitted in writing, but sometimes result in adding/removing long vowels. For example, the indicative mood of the verb **ahaba*, "he walked", in the plural, male, 2nd person, imperfective form is *ta*ohabuwn**a***, "you (*pl+male*) are walking", adding the short vowel *a* to the last letter. However, the subjunctive mood of the same verb form is *ta*ohabuw**A***, "you (*pl+male*) are walking", removing *n* and adding the long vowel *A*.

To summarize, Arabic nominal words are usually characterized by their state and case; and verbs by their form (e.g., perfective, imperfective; also known as *aspect*), mood, and voice (passive, active). Then, all types are inflected for gender, number, and person.

Arabic has many other interesting and unique characteristics. Here we only summarize the features that may help the reader understand the computational processes that we use in the remaining chapters of this work. This section is by no means meant to be a comprehensive guide to Arabic morphology and grammar. For a comprehensive coverage of Arabic processing, we refer the reader to the book *Introduction to Arabic Natural Language Processing* (Habash, 2010).

In order to deal with the complexity of the morphology expressed by Arabic, we employ some of the following linguistic tools:

LDC Standard Arabic Morphological Analyzer (SAMA) Version 3.1

Short name: SAMA 3.1 (also known as the Buckwalter Analyzer, the predecessor versions of SAMA, abbreviated BAMA 1.0 and 2.0)

Reference: Maamouri et al., 2010

This is a context-*insensitive* morphological analyzer that is capable of delivering the entire set of analyses given a single Arabic word in its surface form. SAMA does not con-

sider context, hence when processing a full sentence, the words are processed each one at a time, regardless of the original word order. Here is an example of SAMA's output, processing the word *Altrqym*, "the numbering":

```
<token_Arabic>
 <variant>Altrqym
  <solution>
   <lemmaID>taroqiym_1</lemmaID>
   <voc>Alt~aroqiym</voc>
   <pos>Al/DET+taroqiym/NOUN</pos>
   <gloss>the + numbering/numeration</gloss>
  </solution>
  <solution>
   <lemmaID>taroqiym_1</lemmaID>
   <voc>Alt~aroqiymu</voc>
   <pos>Al/DET+taroqiym/NOUN+u/CASE_DEF_NOM</pos>
   <gloss>the + numbering/numeration + [def.nom.]</gloss>
  </solution>
  <solution>
   <lemmaID>taroqiym_1</lemmaID>
   <voc>Alt~aroqiyma</voc>
   <pos>Al/DET+taroqiym/NOUN+a/CASE_DEF_ACC</pos>
   <gloss>the + numbering/numeration + [def.acc.]</gloss>
  </solution>
  <solution>
   <lemmaID>taroqiym_1</lemmaID>
   <voc>Alt~aroqiymi</voc>
   <pos>Al/DET+taroqiym/NOUN+i/CASE_DEF_GEN</pos>
   <gloss>the + numbering/numeration + [def.gen.]</gloss>
  </solution>
  <x_solution>
   <voc>Altrqym</voc>
   <pos>Altrqym/NOUN_PROP</pos>
   <gloss>NOT_IN_LEXICON</gloss>
  </x_solution>
 </variant>
</token_Arabic>
```

AMIRA Tools for Arabic Processing

Short name: AMIRA (also known as ASVMTools, the predecessor version of AMIRA).

Reference: Diab et al., 2004

AMIRA is a statistical analyzer that is capable of tokenizing every word in a given Arabic sentence, and finding its part-of-speech and base-phrase chunk, considering the

local context. Unlike SAMA, words must be processed in order, and for every word only one result is returned. However, AMIRA does not find the lemma of a word nor its stem.

Second Generation AMIRA Tools for Arabic Processing

Short name: AMIRA 2.0

Reference: Diab, 2009

AMIRA 2.0 is a modified and improved version of AMIRA. Essentially, AMIRA 2.0 combines AMIRA output with morphological analyses provided by SAMA. It is also enriched with Named-Entity-Recognition (NER) class tags provided by (Benajiba et al., 2008). For every word, AMIRA 2.0 is capable of identifying the clitics, lemma, stem, full part-of-speech tag excluding case and mood, base-phrase chunks and NER tags.

MADA+TOKAN

Short name: MADA

Reference: Habash and Rambow, 2005; Roth et al., 2008; Habash et al., 2009

MADA is a statistical tool for finding the correct SAMA analysis for every word within a giving sentence, considering the local context. MADA is combined with TOKAN, a tool for sentence tokenization, based on the SAMA analysis selected by MADA for every word. TOKAN works based on a configured tokenization schemes, which provide the guidelines on which morphemes to split from the original word. There are some standard schemes, which were defined by Habash et al. (2009). For example, ATB is a tokenization scheme, used in the Arabic Treebank (Maamouri et al., 2004), in which affixival conjunctions, possessive forms, definite articles, and prepositions are separated from their conjoined word, forming individual tokens. D3 is similar to Arabic Treebank tokenization but additionally separating out the definite article *Al* from nouns and modifiers, and the future particle *s* from verbs.

There are several releases that we have used in our work; we indicate the release number in each case.

1.4 Machine Translation

Machine translation is probably one of the most thoroughly investigated topics in the field of natural language processing. Essentially, a translation system gets as input text in one language, the *source* language, and returns its translation in another language, the *target* language.

There are two basic paradigms that address the problem of machine translation. The first is rule-based, in which a system uses manually crafted rules and applies them on the input text for generating the translated output. The second paradigm is corpus-based, also known as the *empirical* approach (Somers, 2003), in which the system translates the input text using a corpus of bilingual parallel texts. Parallel texts are pairs of documents, aligned on the sentence level, with one document written in the source language and the other one is its translation in the target language. Figure 1-6 shows an example of an Arabic-English parallel corpus.

Rule-based translation was considered the main paradigm until the late 1980s; however, during the 1990s (Somers, 2003) when the Internet started growing, large amounts of digital data became available, and the corpus-based paradigm took over. During the 1990s and early 2000s, the most prominent corpus-based research direction was Example-Based Machine Translation (EBMT). The main idea behind EBMT is to translate fragments of the source-language input, based on similar translations found in parallel texts. Such a process presumably emulates the way a human translates in some cases. Since translations are based on actual manually created samples, the results are usually more fluent than ones created artificially using other corpus-based paradigms. A short time after that, the Statistical Machine Translation (SMT) approach emerged and quickly became the most common approach for machine translation. Nowadays, most commercial systems are based on statistical calculations on parallel texts. Figure 1-7 summarizes the most important approaches for machine translation. Please note that other, less popular methods for machine translation exist that were not mentioned here. To see the bigger picture of the machine translation research field, we encourage the reader to refer to the comprehensive survey of Somers (2003).

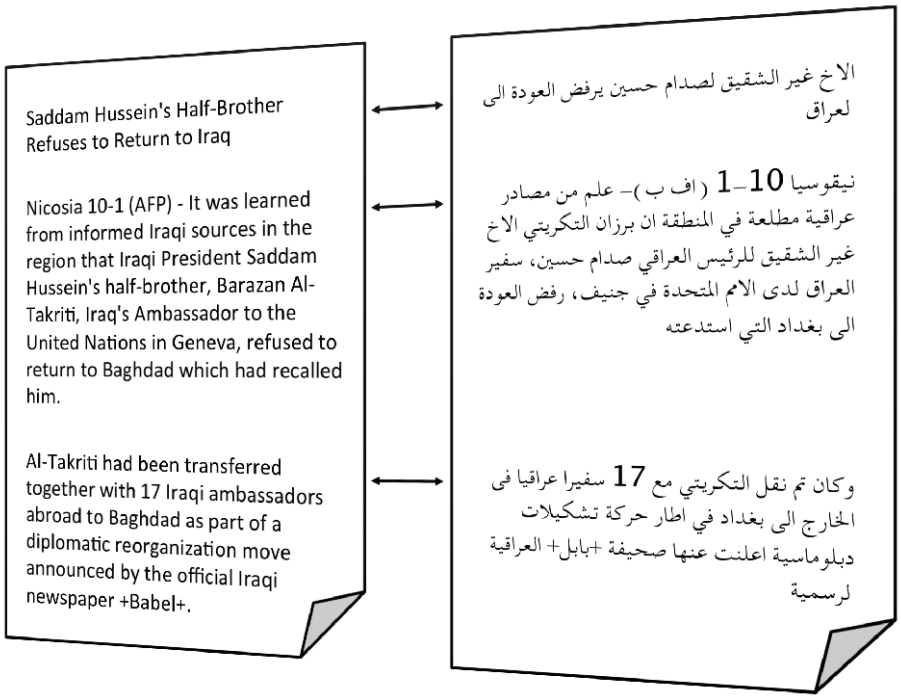


FIGURE 1-6 – An example for an Arabic-English parallel corpus.

In this work, we experiment with both, example-based and statistical translation systems. As a first step, we used our own implementation of an example-based Arabic-to-English translation system (Bar et al., 2007), investigating the possibility of improving its accuracy using Arabic synonyms. In the second step, we use Moses (Koehn et al., 2007), a well-known implementation of phrase-based statistical machine translation, with multiword Arabic paraphrases.

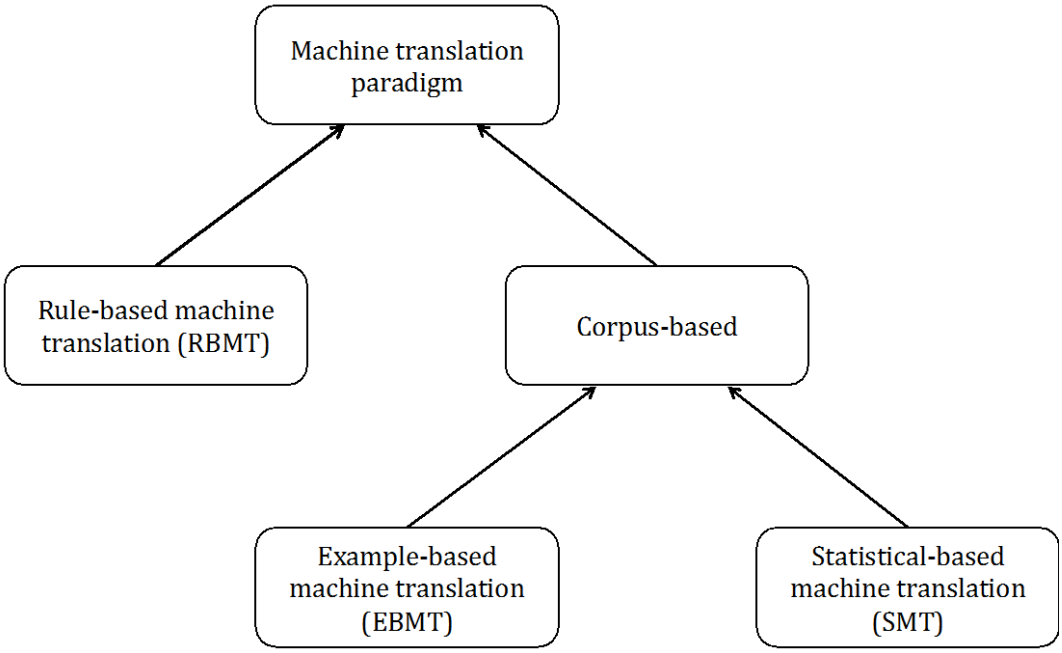


FIGURE 1-7 – Relevant machine translation paradigms.

Another important characteristic of a translation system is its level of analysis. In particular, we look at the pre- and post-processing steps the system takes in order to handle the translation process. Vauquois (1968) proposed the well-known machine-translation pyramid to address the analysis characteristic, shown in Figure 1-8. At the bottom of the pyramid are the *direct* translation systems that produce the translation using only some kind of word-level analysis. Those systems are good to for dealing with a single target language.

Higher up on this chart are the *transfer* systems, which first analyse the source-language text and create corresponding structures. The analysis can be on the syntactic level or even on the semantic level. Once the syntactic or semantic structure of the source-language text has been captured, the system transfers the source-language structure to the corresponding target-language structure. The last step is to generate the translated text from the transferred target-language structure.

At the apex of this taxonomy of translation methods there are the *Interlingua* systems, which first translate the source-language text into some kind of "universal" intermediate language (either a formal logic representation of some sort or else a modified natural-language, along the lines of Esperanto), and then generate the target-language text from the universal representation. Systems that were based on this ap-

proach were supported by Bar Hillel (1960), although he argued strenuously that the very idea of automatic high-quality translation is misguided.

Usually, transfer and Interlingua systems are designed to deal with more than one pair of languages, while direct systems are better suited to a single language-pair translation system.

Note that regardless of the level of analysis a specific translation system uses, it can be based on any one of the paradigms mentioned above. Having said that, only recently have we seen experiments on semantic transfer corpus-based machine translation (Jones et al., 2012).

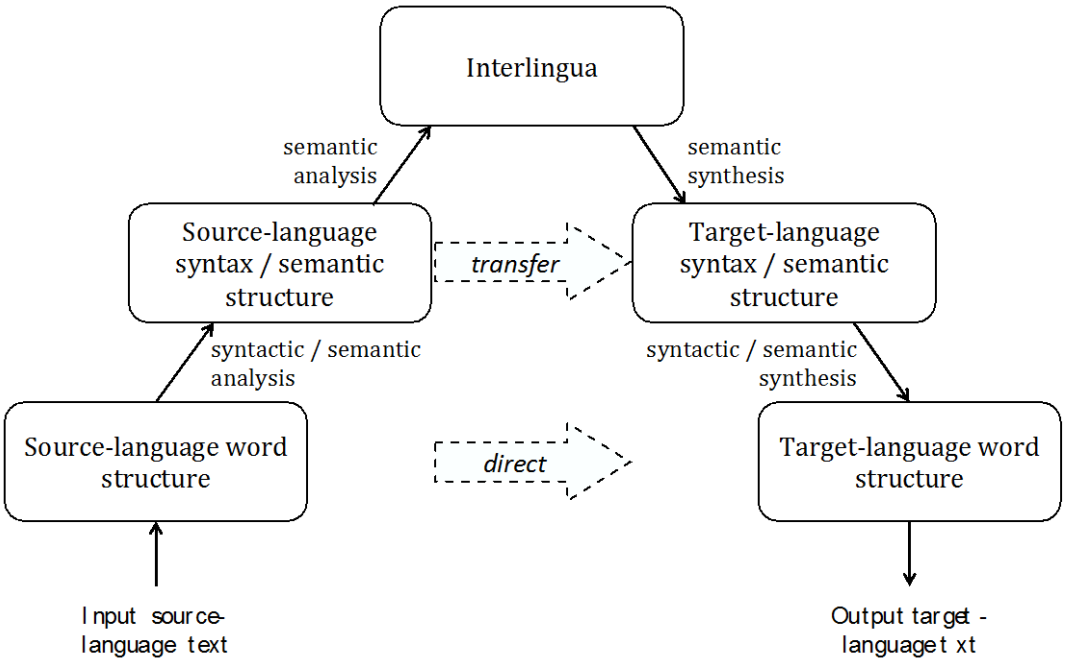


FIGURE 1-8 – Level of analysis of a translation system.

1.4.1 Example-Based Machine Translation

The main idea behind the example-based machine translation paradigm is to emulate the way a human translator think in some cases, as first introduced by Nagao (1984). Example-based translation systems exploit a large bilingual translation-example corpus to find translations for fragments of the input source-language text. This step is called *matching*. The corpus is created by aligning bilingual parallel texts on the phrase, sentence or paragraph level. Given a group of matched fragments, the next step is to extract their possible translations from the target-language side of the corpus.

This step is called *transfer*. The last step is *recombination*, which is the generation of a complete target language text, pasting together the translated fragments. Figure 1-9 presents the main steps of an example-based translation system.

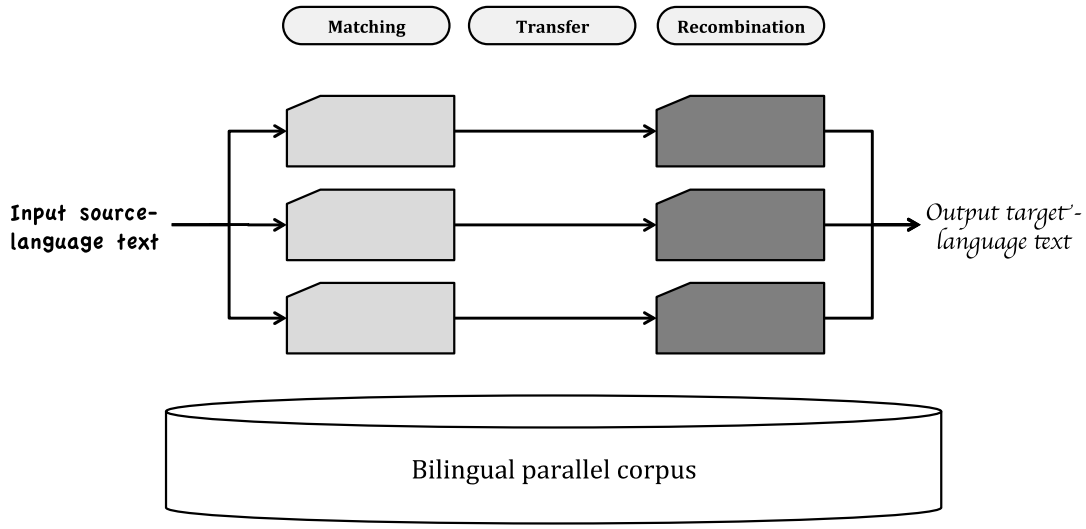


FIGURE 1-9 – Main steps of an example-based translation system.

There are example-based machine translation systems that parse the translation-examples and also store their syntactic structure. Such systems are called *structural*. The matching step in structural systems is done by first analyzing the input source-language sentence to discover its syntactic structure and then finding translation examples that match on the syntactic level. *Non-structural* systems, on the other hand, store the translation-examples as pair of strings, with some additional information, usually morphological and/or part-of-speech tags. Recalling the pyramid described above, a non-structural system is considered to be a direct system, since it translates the source-language text using only word-level information. However, a structural system is considered to be a transfer system, since it first analyzes the source-language text to create some sort of syntactic structure, transfers this structure to a target-language one, and finally generates the translation of the entire input source-language text. There are also structural systems that learn from the translation-example corpus the syntactic differences between the two languages and build set of syntactic transfer rules. Those systems are called "Example-Based Syntactic Transfer Systems."

In the matching step, the system searches for corpus fragments that match a fragment of the input text. In some systems, the match is performed on several levels, with

each level assigned a different score. Words levels may be morphologic (stem), syntactic (part of speech [POS] tags), semantic (ontology/thesaurus distance), etc. Structural systems may also try to find syntactic matches for entire sentences. Generating translations of those fragments would involve modifications and fixes of the translation extracted from the target-language side of the translation-examples.

Probably the main difference between the example-based and statistical-based paradigms is in the way they use their parallel corpora. While statistics-based systems use the corpus offline, to calculate probabilities on text fragments and their corresponding translations, example-based systems try to look for those text fragments in the parallel corpus in real-time, that is, upon receiving an input sentence for translation, considering the context in which the text was found in the corpus.

The EBMT paradigm was found to perform poorly than SMT (Groves and Way, 2005). As a result of that, most commercial and academic research refers to the statistics-based paradigm. Nevertheless, the example-based approach is still being investigated, especially as a component within a statistical system (e.g., Dandapat et al., 2011).

For more information on this subject, the reader may refer to the comprehensive review of example-based machine translation by Somers (1999).

1.4.2 Statistical Machine Translation

Brown et al. (1990), working at IBM, suggested treating the translation process as a probabilistic model. This suggestion was based on their work on speech recognition applications, where probabilistic models were found to perform very well.

The initial models proposed by Brown et al. (1990) were based on the individual words. Och and Ney (2002) and Koehn et al. (2003) introduced the phrase-based models, also known as phrase-based statistical machine translation (PBSMT); currently, this approach serves as the most prominent paradigm for machine translation.

In the word-based model that was proposed by Brown et al. (1990) the best translation sentence $\hat{e}$ is found among all possible translations *e*, given the input source-language sentence *f*, using:

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

By applying Bayes rule on this formula, we get:

$$p(e|f) = \frac{p(f|e) \cdot p(e)}{p(f)} \tag{1.1}$$

Since $p(f)$ does not change for a given input *f*, we ignore it, resulting in what is considered the general word-based statistical machine translation formula:

$$\hat{e} = argmax_e p(f|e) \cdot p(e) \quad (1.2)$$

The first part, $p(f|e)$, is often referred to as the *translation model* reflecting the faithfulness of the translation to the input; the second part, $p(e)$, is known as the *language model*, measuring the fluency of the target-language translation. For every potential translation e, $p(e)$ measures how well it is structured in terms of the style and grammar of the target language. In this sense, we want $p(e) \to 0$ in cases where e is a random word sequence, and $p(e) \to 1$ in cases of well-formed sentences. For $p(e)$, Brown et al. (1990) suggested the n-gram model, which had already been used by them in speech recognition applications. An n-gram model multiplies the probability of every individual word with all the others, considering its contextual history. The contextual history is modelled with an (n-1)th-order Markov chain (n here is a parameter), that is, for every word we measure its probability to occur after its n-1 preceding words. This is explained by the following formula:

$$p(e) = \prod_{i=1}^{m} p\left(e_i \middle| e_{i-n+1}^{i-1}\right)$$

where e_i is the ith word of the sentence e (of size m), and e_{i-n+1}^{i-1} is the sequence of the n-1 preceding words. Following the maximum-likelihood estimation approach, the probabilities $p(e_i|e_{i-n+1}^{i-1})$ are calculated from a large monolingual corpus of target-language texts, in the following way:

$$p\left(e_i \middle| e_{i-n+1}^{i-1}\right) = \frac{C(e_{i-n+1}^{i})}{C(e_{i-n+1}^{i-1})}$$

here, $C(x)$ represents the number of occurrences of the sequence x in the corpus. This function measures the relative chance of e_i occurring after e_{i-n+1}^{i-1}. Note that $\sum_e p\left(e \middle| e_{i-n+1}^{i-1}\right)$ =1, thus $p\left(e_i \middle| e_{i-n+1}^{i-1}\right)$ is a probability function.

It is not a rare situation when $C\left(e_{i-n+1}^{i}\right) = 0$ for some sequences e_{i-n+1}^{i} that do not occur in the given corpus, resulting in $p\left(e_i \middle| e_{i-n+1}^{i-1}\right) = 0$; this obviously zeros the probability of the entire sentence, hence cannot be tolerated. To deal with such situations, the probability function $p\left(e_i \middle| e_{i-n+1}^{i-1}\right)$ is often smoothed.

Similarly, we want that the translation model $p(f|e) \to 0$ when e does not relate to f in any sense, and $p(f|e) \to 1$ when e is a perfect translation of f. Brown et al. (1990) suggested using word-level *alignments* to model this probability function. In principle, given a pair of sentences (one in each language), a word-level alignment is a function $a: s \mapsto t$, where s is a single index of a source-language word and t is a single index of the aligned target-language word. Defining a as a function implies that either a single

target-language word is aligned with a single source-language word (one-to-one), or several source-language words are matched with a single target-language word (one-to-many). Normally this includes words that do not translate to any of the other words. This is illustrated in Figure 1-10 showing alignments of Arabic and English sentences. Note that, theoretically, a single source-language word may also be aligned with several target-language words; however, since the idea was to keep *a* represented as a function, such alignments are ignored, resulting with a unidirectional translation models. Thus, phrase-based translation systems, as we are about to show, often use both translation models, one for each direction. The number of target-language words a source-language word is aligned to is usually referred to as the *fertility* of the source-language word. Many-to-many alignments, where *m* source-language words are aligned with *n* target-language words, exist as well.

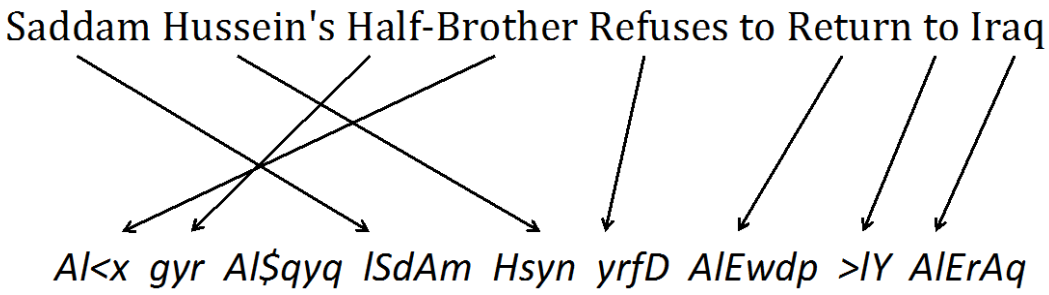


FIGURE 1-10 – An example for word alignment of Arabic and English.

Back to the estimation of $p(f|e)$ suggested by Brown et al. (1990) They, therefore, formulated the following equation:

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

where $p(f, a|e)$ is the probability that a specific word-level alignment a is the correct alignment for the sentence pair *f* and *e*. Since we do not really know which alignment, out of all the possible alignments, is the correct one, we sum over all the possibilities. In order to estimate the values of $p(f, a|e)$, we need to learn the probabilities of each source-language word to be aligned with every sequence of target-language words. These probabilities lie in a parallel corpus; thus can be extracted automatically. However, since bilingual parallel texts are usually not word-level aligned but only sentence-level aligned, Brown et el. (1993) suggested taking an unsupervised maximum-likelihood-estimation approach to learn these probabilities. This is usually done using

the Expectation-Maximization (EM) algorithm, working on a sentence-aligned parallel corpus.

In order to capture different alignment situations, where a source-language word is aligned with one or many target-language words, Brown et al. (1993) define $p(f, a|e)$ as a product of conditional probabilities, which may be calculated by several models, known as IBM Models 1-5. The first two models, 1 and 2, are the basic ones, trying to estimate the probability of $p(f, a|e)$ using a product of $t(f_i|e_{a(i)})$, calculated for every source-word individually; that is:

$$p(f, a|e) = \prod_{i=1}^{m} q(a(i)|i, l, m) \cdot t(f_i|e_{a(i)})$$

where $t(f_i|e_{a(i)})$ is the probability of the individual source-language word f_i aligned by a with the target-language word $e_{a(i)}$. Then, $q(a(i)|i, l, m)$ is the probability that $a(i) = j$, considering the length l and m of the target and source language sentences, respectively. IBM Model 1 removes the dependency on m and simply assigns $q(a(i)|i, l, m) = \frac{1}{l+1}$, assuming uniform distribution over all l+1 possible target-language words (it is l+1, and not just l, simply because we consider the empty word as another possibility for source-language words that do not get translated to any of the target-language words).

Essentially, neither model 1 nor 2 considers the fertility of the words. Models 3-5 consider additional conditional probabilities that, among other things, take into consideration the fertility of the words, modeled by random variables, and distortion probabilities, allowing words to be reordered in translations.

For more information on word-based statistical machine translation, we point the reader to Brown at al., 1990; Brown et al., 1993; and Lopez, 2008.

The more up-to-date approach to machine translation is phrase-based statistical machine translation, where the translation system uses word sequences rather than single words as the basic translation unit. In this sense, "phrase" merely means a sequence of words and not a syntactic constituent. Phrase-based models were found to perform better (Koehn et al., 2003; Och and Ney, 2002) than the corresponding word-based ones; hence, in our work, we use an implementation of a phrase-based system.

We now continue by providing some background on phrase-based statistical machine translation.

Using the same Bayes decomposition from Equations 1.1 and 1.2, phrase-based systems define the translation model as follows:

$$p(f|e) = \prod_{i=1}^{I} p(\bar{f}_i|\bar{e}_i) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i) d(start_i - end_{i-1} - 1)$$

This leads to the following best translation equation:

$$\hat{e} = argmax_e p(f|e) \cdot LM(e)$$

$$= argmax_e \prod_{i=1}^{I} [\phi(\bar{f}_i|\bar{e}_i)\, d(start_i - end_{i-1} - 1)] \cdot LM(e)$$

The language model probability, $LM(e)$, is similar to the one introduced by Brown et al. (1990), and is referred above as $p(e)$. However, the translation model $p(f|e)$ is now modeled as a multiplication of I phrase-translation probabilities $p(\bar{f}_i|\bar{e}_i)$, each breaks down into a product of two probability functions ϕ and d. The phrase translation probability is measured by $\phi(\bar{e}|\bar{f})$, that is, the probability of the target-language phrase $\bar{e}$ being a translation of the source-language phrase $\bar{f}$. These probabilities are learned from a corpus of bilingual parallel texts by taking a maximum-likelihood-estimation approach, that is, counting the number of times $\bar{e}$ is translated to $\bar{f}$ in the corpus, compared to the total number of times $\bar{e}$ appears in the target-language side of the corpus. Formally, we have:

$$p(\bar{e}|\bar{f}) = \frac{C(\bar{e} \leftrightarrow \bar{f})}{C(\bar{e})}$$

where $C(\bar{e} \leftrightarrow \bar{f})$ is the number of times $\bar{e}$ gets translated to $\bar{f}$ in the corpus, and $C(\bar{e})$ is the number of times $\bar{e}$ appears in the corpus, regardless of the translation it carries. To count phrase translations, the corpus needs to be aligned on the phrase level, so that for every phrase $\bar{e}$, the system knows its corresponding phrase $\bar{f}$. Phrase-based alignment is usually obtained on top of the individual word-level alignment (Och and Ney, 2003) resolved by the above-mentioned IBM models. Typically, it begins by running the unidirectional word-based alignment algorithm twice: once to retrieve matches of individual source-language words with any number of target-language words, and another time to retrieve matches of individual target-language words matched with any number of source-language words. For every aligned pair of sentences, the results of both unidirectional word-based models are combined, forming a unified symmetrized alignment, as illustrated in Figure 1-11. In fact, the unified word-based alignment does not include many-to-many word matches, but only one-to-many matches, in both directions.

According to Och and Ney (2003), phrase-based alignment is obtained by locating pairs of phrases that are *consistent* with the unified word-based alignment. Consistency, in this sense, refers to the situation where all the words of one side of the pair are

aligned with words located within the other side of the pair, and vice versa. In Figure 1-12, we demonstrate consistent and inconsistent pairs, extracted from the unified alignment presented in Figure 1-11.

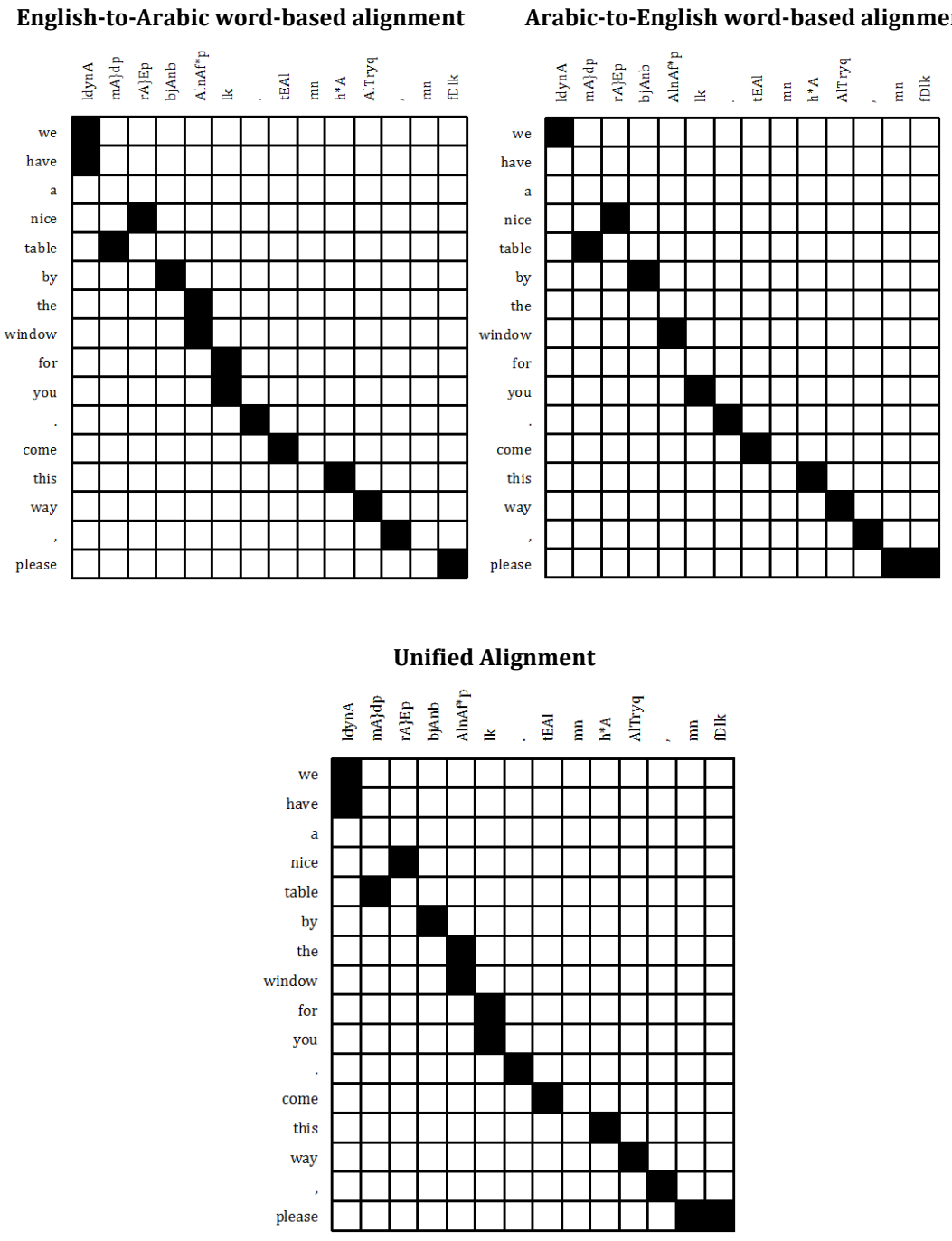


FIGURE 1-11 – Unified word-based alignment of Arabic and English sentences.

The entire set of extracted phrase pairs is then placed in what is called a *phrase table*. In the phrase table, every source-language phrase and its translation is assigned a score, such as the maximum-likelihood translation probability $\phi(\overline{f_i}|\overline{e_i})$.

The quantity $d(start_i - end_{i-1} - 1)$, known as the reordering (distortion) probability, is a score that models the distance between the start index of the source-language phrase covered by the current target-language phrase $\overline{e_i}$ and the end index of the previous phrase pair. In other words, given two translation phrases, the reordering model refers to the distance (in number of words) between the two source-language parts that were translated by the two phrases. This model simply assigns a probability to any possible distance, typically not longer than six or seven words. The probability is learned from the phrase-aligned bilingual texts, as before, using maximum-likelihood-estimation. Real-world systems tend to employ more complicated reordering techniques that model the distance of the individual words as well as the phrases. Such models are referred to as *lexical reordering*.

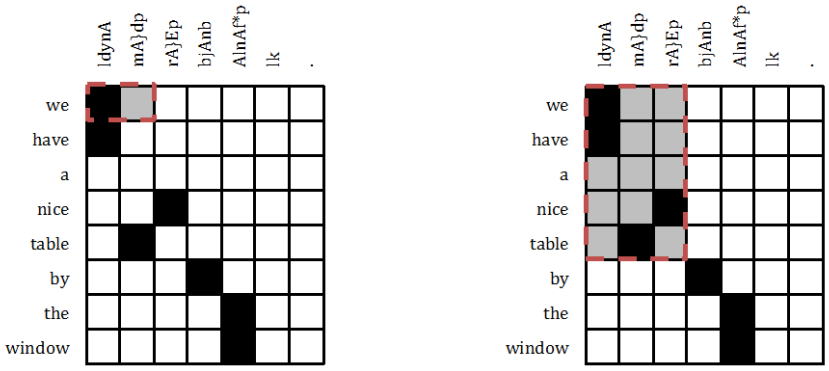


FIGURE 1-12 – Phrase alignment consistency. (a) inconsistent; (b) consistent.

In word-based models, the translation and language models make equal contributions to the final score; however, depending on the specific setting, it is plausible that one model is more important than another. In phrase-based models, this assumption is taken care of by adding weights to the probability functions, in the following way:

$$\hat{e} = argmax_e \prod_{i=1}^{I} [\phi(\overline{f_i}|\overline{e_i})^{\lambda_\phi}\, d(start_i - end_{i-1} - 1)^{\lambda_d}] \cdot LM(e)^{\lambda_{LM}}$$

where $\lambda_\phi, \lambda_d, \lambda_{LM}$ are the assigned weights for the models ϕ, d, LM, respectively.

In fact, this equation can be easily translated into a *log-linear* model (Och and Ney, 2002), of the form

$$p(e|f) = \exp\left\{\sum_{h\in H} \lambda_h\, h(e,f)\right\}$$

A log-linear model is composed of a set of weighted feature functions, with each function represents one of the discussed probabilities; in our case, the feature functions are: $\log \phi$, $\log d$, and $\log LM$ (the *log* is a result of the log-linear arrangement). However, there are more feature functions being employed by real-world systems. Since $\phi(\bar{e}|\bar{f})$ works in one direction, the reverse phrase translation probability is added as another feature function, represented by $\phi(\bar{f}|\bar{e})$ and calculated similarly to $\phi(\bar{e}|\bar{f})$ from the bilingual parallel data. It is common to use the lexical translation probability, based on the word-aligned bilingual texts, modeled by $l(\bar{e}|\bar{f})$. Then, the reverse lexical model, $l(\bar{f}|\bar{e})$, is added as well. Word and phrase counts are also considered; the first function is needed to control the natural preference of the *n*-gram language model to short translations, and the second function is necessary to control the segmentation of the final translation, that is, the number of phrases it is composed of. The phrase table usually contains all the relevant feature function scores for every phrase-pair, calculated offline during the *training* stage.

Assigning weights to every feature function is usually done automatically using a process known as Minimum Error Rate Training (MERT), as suggested by Och (2003). MERT uses a relatively small corpus of bilingual parallel text, also known as the *development* or *tuning* set, excluded from the bilingual texts that were used for building the phrase table. Essentially, MERT looks for the best weighting, measured with an automatic evaluation function, such as BLEU (Papineni et al., 2002). It runs in iterations; in every iteration it assigns different weights to the feature functions employed, based on the results of the previous iteration, and uses the log-linear model to translate the source-language part of the development set. The obtained translations are compared vis-à-vis the target-language part of the development set, for calculating the automatic score. After several iterations, MERT concludes with resulted weights that produced the best results, based on the evaluation metric. In Chapter 5, we use MERT to adjust the weight of the feature function that controls the paraphrasing component within the translation process. Our experiments show that MERT has an important impact on the translation quality, at least when measured by BLEU.

The process of finding the best translation $\hat{e}$, using the log-linear model, is called *decoding*. Given an input sentence for translation, the decoder begins by looking up in the phrase table for all the existing translations of the input's phrases, as demonstrated in

Figure 1-13, and then searches for the best translation path out of all the translation possibilities. The best translation is chosen using a translation score, calculated from the component individual phrase scores, as previously assigned by the log-linear model. In particular, at the first step, the decoder generates translation *hypotheses*, each composed of the input phrase, translated text, and a score, based on the log-linear model. Then, the decoder chooses the best hypothesis path among all the possibilities, as illustrated in Figure 1-14. The conventional search space in this case is quite big, thus cannot be addressed in a reasonable amount of time. Therefore, the decoder uses pruning techniques for eliminating paths that are unlikely to be part of the final translation.

FIGURE 1-13 – The first step of phrase-based SMT decoding, demonstrated on German-to-English translation. This figure is borrowed from the book, *Statistical Machine Translation* (Koehn, 2010).

For more information on this subject the reader may refer to the book *Statistical Machine Translation* (Koehn, 2010).

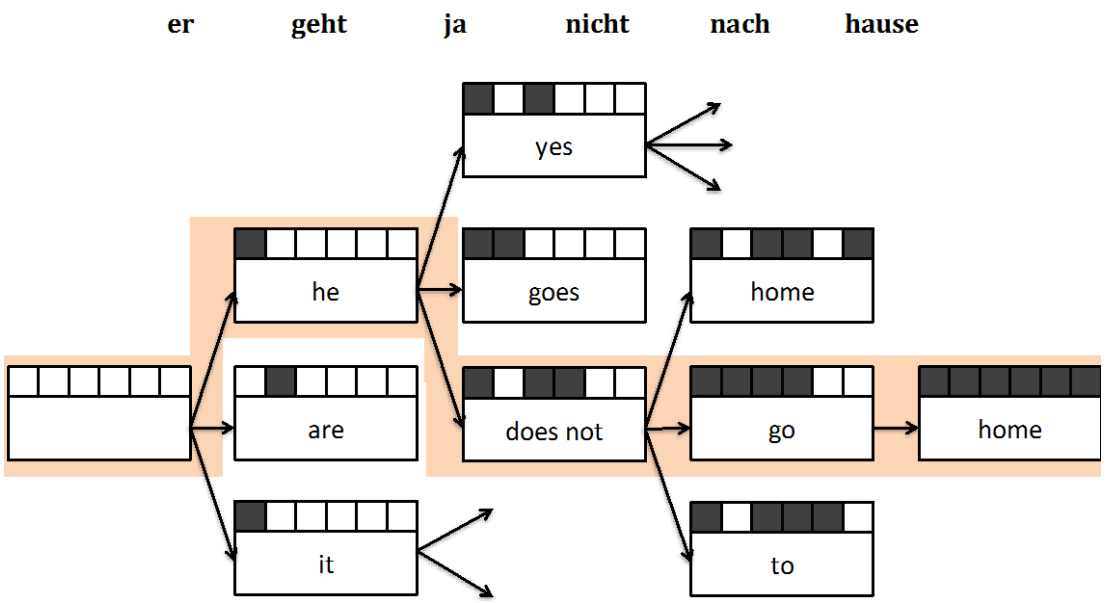


FIGURE 1-14 – Finding the best path having the best score by a phrase-based SMT decoder, demonstrated on German-to-English translation. This figure is borrowed from the book, *Statistical Machine Translation* (Koehn, 2010).

1.4.3 Translating Morphologically Rich Languages

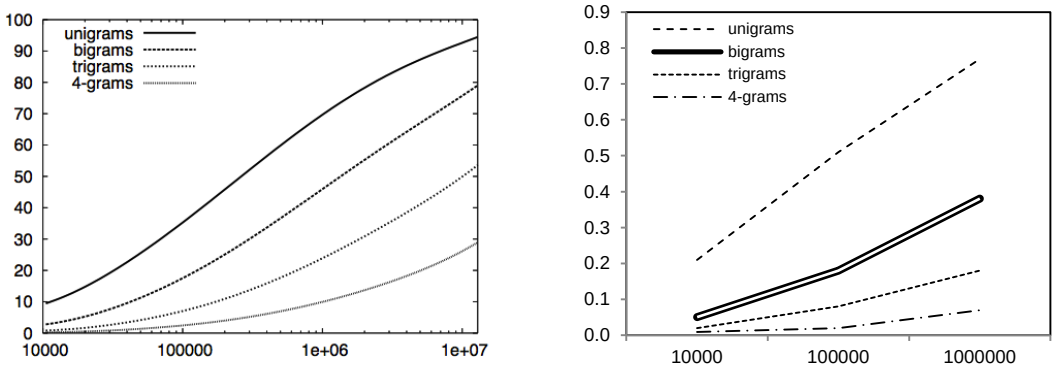
Machine translation is considered very challenging. Bar Hillel (1960), in his well known report, addressed this issue, arguing that building a "fully high-quality automatic translation" (FHQAT) system is impossible, as any such system will have to know things about the world and not just how to decode words. His best-known example for supporting this claim is: "*Little John was looking for his toy box. Finally he found it. The box was in the pen. John was very happy*" (Bar Hillel, 1964). The difficulty that Bar Hillel was referring to in his example is the inability of a machine to determine the correct sense of the word *pen*. According to Bar Hillel, a human translator immediately understands that a box cannot be placed within a handwriting tool, probably because of size issues, therefore that sense of *pen* is overruled. This process requires common knowledge, which can never be modelled by an automatic system. In spite of his pessimism, Bar Hillel expressed his hope and special interest in systems that incorporate semantic linguistics for dealing with such problems.

Fast forwarding to contemporary corpus-based approaches, this problem translates to the lack of parallel data, the main fuel for corpus-based systems. Ideally, if we had access to an enormous corpus of text, written in one language and manually translated into another language, we would probably be able to build a nearly perfect translation

system. However, the current situation is not even close to that. Bilingual parallel texts are very hard to obtain, thus the translation quality of existing machines is far from perfect. The lack of parallel data for corpus-based translation systems is usually referred to as the *data insufficiency* problem, or *data sparseness*.

Morphologically rich languages challenge the corpus-based paradigm even more. A highly inflected language modifies words for different grammatical roles, and by that increasing the number of words that one can find in a corpus of texts written in that specific language. Moreover, as we have seen, Arabic uses diacritic marks to indicate short vowels, which are usually omitted in writing, increasing the level of word ambiguity and by that making the data sparseness even more noticeable.

Callison-Burch (2007), in his thesis, demonstrated the effect of using different sizes of corpora of bilingual texts in a Spanish-to-English phrase-based statistical translation system. His demonstration is presented in Figure 1-15(a). Although Spanish is considered a morphologically rich language, we repeated this experiment with a similar system, translating Arabic to English, as demonstrated in Figure 1-15(b). In both charts, the abscissa is the number of source-language words in the corpus of bilingual parallel texts given to the translation system as training data and the ordinate is the percentage of unique test-set *n*-grams that were translated by the system, regardless of the translation's quality. In other words, this time we are not interested in the quality of the translation, but only in learning the number of phrases (here referred to as *n*-grams) for which the system could find a complete translation in the phrase table.



(a) (b)

FIGURE 1-15 – Demonstrating data sparseness: (a) results borrowed from Callison-Burch (2007) for a Spanish-to-English translation system; (b) results of an Arabic-to-English translation system.

In both demonstrations, we clearly see that as the corpus size grows larger, so does the number of translated *n*-grams. On the other hand, we see that even when the corpus is relatively large, the system could not translate a large number of *n*-grams, especially tri- and four-grams. The results also show that more Arabic *n*-grams remain unseen than Spanish *n*-grams, when using the same corpus size. This may be ascribed to the fact that Arabic has richer inflective morphology than Spanish.

Data sparseness is relevant to other text processing applications that use a textual corpus as a resource; namely, question answering, textual search, information retrieval, and others. One way to deal with this problem is to obtain more data. For machine translation, it was shown in previous works that using additional parallel data helps increase the quality of the translation. Although this may seem labor intensive, commercial applications often take this direction, preferably combines with other techniques.

Another possibility is to use some techniques that enable inexact matches of input phrases in the corpus, for instance allowing words to be matched on their lemma level. This approach is often referred to as *generalization*. A common technique, called *segmentation* (Koehn and Knight, 2003; Lee, 2004; Goldwater and McClosky, 2005; Habash and Sadat, 2006; Singh and Habash, 2012), may be seen as generalization. Segmentation, also known as "tokenization", refers to the pre-processing step of breaking the source-language space-delimited words into smaller morphemes before being given to the translation system. The morphemes are then treated by the system as individual words. In some languages, such as Arabic, segmentation is considered a challenging task because of word-level ambiguity. For example, the Arabic word *brd* (short vowels omitted) may be segmented as *b+ rd*, "with/in" + "reply", tearing off the preposition *b*, "with/in", from the noun *rd*, "reply"; or as *brd,* "cold". Resolving this ambiguity is usually done by considering a larger context. An alternative approach is to consider all the possibilities and place them on what is called a *word lattice* (Dyer et al., 2008), which is then given to the decoder for translation.

Regardless of the ambiguity problem, deciding which morphemes are better treated as individual words and which are not is another research topic. For example, Habash and Sadat (2006), and Al Haj and Lavie (2010) experimented with different segmentation schemes of Arabic input to an Arabic-to-English translation system. Similarly, Singh and Habash (2012) experimented with various segmentation schemes of Hebrew texts. The rationale behind segmentation, is to reshape the input text so that it will "look like" a target-language text; for example, splitting the Arabic coordinating conjunction prefix

w+ to match the individual English word *and*. In other words, it is assumed that choosing the best segmentation scheme depends highly on the target language.

There are many works on using the syntactic information of the input and/or output texts (e.g. Chiang, 2005). Such systems are sometimes referred to as string-to-tree, tree-to-string, tree-to-tree, or hierarchical translation systems.

Another way to tackle the problem of data sparseness is to enrich the source language with semantic equivalents, so that a system can use them to translate unseen phrases. Figure 1-16 illustrates this technique by way of example.

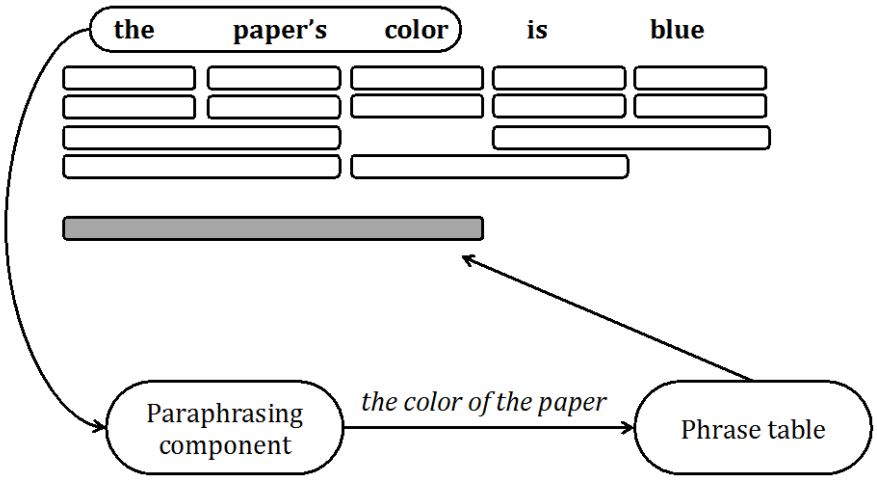


FIGURE 1-16 – Using source-language paraphrases in machine translation.

There are several works that follow along this path. Callison-Burch et al. (2006) use source-language paraphrases to improve phrase-based statistical machine translation. The paraphrases were extracted using bilingual parallel corpora that pair source-language texts with translations in languages other than the target language as configured in the translation system. Marton et al. (2009) followed the same direction with paraphrases that were extracted from a large monolingual corpus in the source language. We elaborate on both works in detail in the following section, but one observation for now is that neither of these works was applied on a highly inflected language such as Arabic. In our work, we introduce a novel paraphrasing technique focusing on Arabic as source language.

1.4.4 Challenges of Arabic-to-English Translation

Translating an Arabic sentence into English raises some interesting challenges. One of the challenges is the richness of the Arabic morphology compared to English. Due to

its rich morphology, Arabic words, especially verbs, may translate into several English words; therefore, given a parallel Arabic-English text, the English part is usually longer. By way of example, we compiled several bilingual Arabic-English parallel texts and counted their words; the results are provided in Table 1-1. We counted words under two conditions, each depicted in its own row. Under the first condition, we counted words using a simple tokenization approach that breaks the text at white spaces and punctuation marks. In the second condition, the Arabic text was morphologically analysed with MADA 3.1 and then tokenized following the D3 Arabic tokenization standard (Habash et al., 2009), which is similar to Arabic Treebank tokenization, but additionally separating out the definite article *Al* from words; hence in the D3 scheme affixival conjunctions, possessive forms, definite articles, and prepositions are separated from their conjoined word, forming individual tokens. The English part was further processed as well; affixival possessive forms (e.g. *'s*) were separated from their carriers. We clearly see that under the white-space condition, an Arabic word is translated into 1.23 English words on average. When we tokenize the text, this number is diminished, resulting in an almost even rate.

Tokenization	*Arabic*	*English*	*Average per word*
White-spaces and punctuations	5,712,574	7,074,213	1.23
Using morphological analysis	6,729,163	7,106,001	1.05

TABLE 1-1 – Comparing word counts of bilingual Arabic-English parallel text, using the D3 tokenization scheme for Arabic (Habash et al., 2009), and splitting *'s* in English.

Arabic and English have different word-order structures. While an English sentence is structured following the Subject-Verb-Object (SVO) scheme, Modern Standard Arabic follows the Verb-Subject-Object (VSO) structure. This structural difference repeatedly causes skips, or gaps, which are usually handled with a distortion model, as described above. The gaps are relatively large, as demonstrated in the following example:

(Arabic): *<Eln AlmtHd* AlEskry AlmSry, AlEqyd <Hmd mHmd Ely, <n…*

(English): *The Egyptian military spokesman, Col. Ahmed Mohammed Ali, announced that…*

The first word is the verb *<Eln*, "announced", marked in boldface; its translation comes only after eight English words, a relatively large gap. Larger gaps are not unlikely; thus working with a relatively large distortion value may help for handling such

events. However, increasing the distortion value also increases the search space for the decoder and as a result of that, it takes more time for a sentence to be translated.

Arabic adjectives, as opposed to English ones, come after the noun they modify, introducing another potential cause for gaps. Here, the gaps can grow as large as the number of adjectives used to modify the head noun. For example, take the noun phrase from the previous sentence: *AlmtHd* AlEskry AlmSry*, "The Egyptian military spokesman". The word *AlmSry*, "The Egyptian", occurring as the last word in the noun phrase, is translated into the first definite word in the corresponding English noun phrase.

These are just a few of the challenges related to the automatic translation of Arabic text to English.

1.4.5 Machine Translation Evaluation

The evaluation of translations is sometimes considered almost as difficult as the translation process by itself. Starting in the early 1990's, the machine-translation community continuously sought a standard automatic evaluation procedure, so as to be able to compare the results of one translation system with another (Turian et al., 2003). Basically, the main challenge comes from the fact that a translation can be phrased in various ways; hence there is no gold standard that can be used for comparison.

Generally speaking, existing evaluation algorithms seek overlapping parts between the *hypothesis* translation (generated by the translation system) and the *reference* translation (generated by a human translator). The most common metrics are probably BLEU (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005; Denkowsky and Lavie, 2011). Since all these metrics are based on co-existing n-grams, they can use more than one reference translation to capture language variations and by that increase coverage.

The BLEU score uses 1- to 4-gram co-occurrence precision in combination with a brevity penalty for short sentences. Nowadays, despite all the known limitations, BLEU is still the dominant metric in the machine translation field. The NIST score, a modified version of BLEU, uses 1- to 4-gram co-occurrence precision as well, but takes the arithmetic mean of the *n*-gram counts. METEOR extends this behaviour by considering single-word variations: unigrams are matched on their surface form, stemmed form, and their meaning, as found in WordNet.

These metrics, although widely used, have been criticized for their inability to capture the true quality of translations. Callison-Burch (2006), in his thesis, compared the

performance of BLEU with a human evaluator. Following (Doddington, 2002; Coughlin, 2003), he argued that there are two serious drawbacks that make BLEU inconsistent with human evaluation. The first issue is related to the fact that BLEU matches *n*-grams regardless of their location in the hypothesis and reference translations. He showed that being flexible in ordering causes wrong variations to be counted, incorrectly. The second issue is that synonyms and paraphrases are considered only when they specifically occur in one of the reference translations, which makes the quality of the evaluation strongly depends on the number of reference translations as well as their style. METEOR, by definition, addresses the second issue by extending single-word matching with synonymous words, extracted from WordNet. Obviously this limits METEOR to be used only on target languages for which WordNet or other similar resource exists.

Besides being used for improving the translation quality, paraphrases have also been applied to improve machine-translation evaluation. For example, several works (Kauchak and Barzilay, 2006; Owczarzak et al., 2006; Zhou et al., 2006) derive paraphrases for the reference translations in order to increase the reliability of BLEU. Recently, in (Denkowski and Lavie, 2010a; Denkowski and Lavie 2010b), METEOR was improved to consider paraphrase matches between words and phrases.

For the same reason, other works (Madnani et al., 2007; Madnani et al., 2008; Madnani and Dorr, 2013) inferred paraphrases for the reference translations so as to improve parameter tuning and, concomitantly, translation quality. They obtained a significant improvement in BLEU over a baseline system that did not use paraphrases for tuning.

There are alternative evaluation techniques. For instance, Padó et al. (2009) attempted to treat the evaluation process as a textual entailment problem. They used systems for recognizing textual entailments for finding semantic similarity between the hypothesis and the translation references, and showed some effective preliminary results. The limitation of this approach is the time complexity of the entailment algorithms, which makes it infeasible to be used as a target function in tuning processes, such as MERT.

Recently, there is a trend to focus more on manual evaluation, by asking human translators to count how many modifications they need to do on a given hypothesis so that it conveys the same meaning as a human translation.

In this work, we compare the results of our translation system with and without using Arabic synonyms and paraphrases. Although we do use the standard automatic metrics for evaluation, we always spend more time to evaluate the results manually in order to gain a greater sense of the true performance.

1.5 Multiword Expressions

A *multiword expression*, or *expression* (also known as MWE), refers to a multiword unit or a collocation of words that co-occur together statistically more often than chance. An expression is a cover term for different types of collocations, which vary in their transparency and fixedness. Expressions are pervasive in natural language, especially in web-based texts and speech genres. Identifying expressions and understanding their meaning is essential to language understanding, hence they are of crucial importance for any natural language processing applications that aim at handling robust language meaning and use. In fact, the seminal paper (Sag et al., 2002) refers to this problem as a key issue for the development of high-quality applications. Typically, expressions are classified based on their syntactic constructions. Among the various classes, one can find the Verb-Noun Idiomatic Constructions (VNIC), as in *spill the beans*, Noun-Noun Constructions (NNC), as in *traffic light*, and others. An expression typically has an idiosyncratic meaning that is more or different from the meaning of its component words. The meaning of an expression is *transparent*, or *compositional*, if its meaning as a unit can be predicted from the meaning of its words, such as in the English expression *prime minister*. On the other hand, idiomatic expressions, which are *non-compositional*, are expressions whose overall meaning is impossible or difficult to predict from the individual component word senses. In Figure 1-17, we treat compositionality with a measurement scale, provided with some examples.

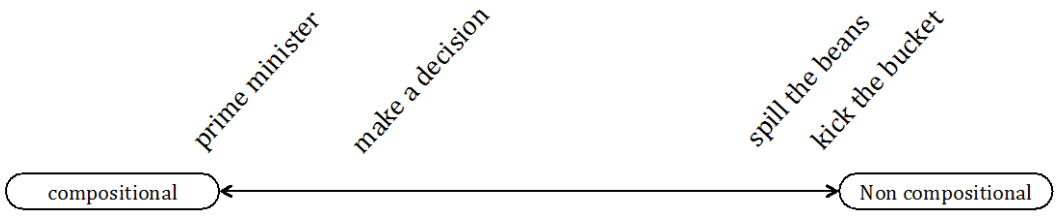


FIGURE 1-17 – Expression-compositionality scale.

In contrast with their English equivalents, Arabic expressions can be expressed in a large number of forms, expressing various inflections and derivations of the words

while maintaining the exact same meaning, for example, *>gmD* [*flAn*] *Eynyh En* [*Al>mr*], "[one] disregarded/overlooked/ignored [the issue]", literally, closed one's eyes, vs. ***>gmDt*** [*flAnp*] ***EynyhA*** *En* [*Al>mr*], "[one_*fem*] disregarded/overlooked/ignored_*fem* [the issue]", where the predicate takes on the feminine inflection. However, in many cases, there are morphological features that cannot be changed in different contexts, for example, *mkrh >xAk lA bTl*, "forced with no choice", in this example, regardless of context, the words of the expression do not agree in number and gender with the surrounding context; these are *frozen* expressions.

In Chapter 6, we address the problem of automatic multiword expression boundary detection and classification in Arabic running text. We take a supervised machine-learning approach using a relatively small manually annotated data set, augmented with an increasing quantity of automatically annotated data, labeled using a deterministic algorithm. We investigate the impact of explicitly modeling morpho-syntactic features and address the problem of handling gapped expressions in running text.

1.6 Co-training

Our paraphrasing algorithm, as will be seen, is based on the co-training technique (Blum and Mitchell, 1998). The main idea of the co-training approach applied to unlabeled data is to use two classifiers on different views of the same data. This technique is mainly used for bootstrapping, when only a small set of the examples are labeled, and the data can be partitioned into two distinct views, that is, each example has two different sets of features. It is also assumed that if a supervised-learning approach be taken, each feature set could have been used individually for classifying the data. For instance, in their original paper, Blum and Mitchell (1998) experimented with the binary classification problem of a web page whether it is an academic course page. Each page has two different feature sets: (1) bag of all words appearing in the page text; and (2) bag of all words appearing in the hyperlinks pointing to the page from other locations. The learning algorithm is presented in Figure 1-18.

Co-training (*Labeled*, *Unlabeled*)

for $i = 1$ to k

train C_1 on E_1 of *Labeled*;

train C_2 on E_2 of *Labeled*;

classify *Unlabeled* with C_1 and select the most reliable p positive and n negative examples;

classify *Unlabeled* with C_2 and select the most reliable p positive and n negative examples;

move the $2p + 2n$ classified examples to *Labeled*;

end

FIGURE 1-18 – A typical co-training algorithm.

The *Unlabeled* set contains a large number of unclassified examples. The *Labeled* set contains the small set of labeled examples. The algorithm runs in iterations; in each iteration, it trains two classifiers C_1 and C_2 independently on the two feature sets E_1 and E_2, respectively. Then, it uses C_1 and C_2 to classify the unlabeled examples and select the most reliable ones, based on a confidence score produced by the machinery of each classifier. The parameters p and n are used to indicate that the number of positive examples does not have to equal the number of negative examples. Blum and Mitchell (1998) argued that making C_1 and C_2 classifying examples from a reduced set U' that is randomly selected from *Unlabeled*, rather than the entire *Unlabeled* set, improves the performance of the overall algorithm. Finally, the $2p$+$2n$ classified examples are pushed into the *Labeled* set, before continuing to the next iteration.

Blum and Mitchell demonstrated with their web page classification experiment that the co-training algorithm outperforms a supervised algorithm that uses the labeled examples as training set and the unlabeled set as testing set. Moreover, they showed that the error rate produced by the co-training algorithm decreases as the number of iterations grows larger.

Co-training has been widely used in applications from different domains. For example, Gupta et al. (2008) and Guillaumin et al. (2010) used co-training for image classification; Wan (2009) used co-training for sentiment analysis; and Chen et al. (2011) used co-training for domain adaptation.

Blum and Mitchell (1998) discussed the relation of co-training with the Expectation-Maximization (EM) learning algorithm, which is often used in unsupervised settings. Most of the existing machine-learning approaches for paraphrase extraction from an unstructured textual resource use some sort of word-based alignment algorithms, which are usually operated by EM. Such alignment algorithms, though, require that the text be sentence aligned beforehand. As we argue in the next section, aligning comparable documents on the sentence level dramatically reduces the amount of data that is made available to the word-based alignment algorithm.

1.7 Related Work

1.7.1 Paraphrases

While in this work we are only interested in data-driven approaches for deriving paraphrases, there are other works that use rule-based frameworks. For example, Fujita et al. (2004) created rules for inferring Japanese structural paraphrases by capturing various syntactic transfers, such as rewriting light-verb constructions. Fujita et al. (2005; 2007) applied those rules to a monolingual corpus for generating a repository of Japanese paraphrases.

We classify data-driven works addressing the problem of deriving paraphrases using two main parameters. The first one refers to the type of corpus used as a resource for paraphrasing and the second parameter refers to the level of the extracted paraphrases, namely single word synonyms, phrases (sub-sentential), or sentences. The following are the most common corpus types used for paraphrasing:

Monolingual corpus Contains texts written merely in the language of interest.

Monolingual parallel corpus Contains multiple monolingual translations of the same foreign-language resource. The translations are typically generated by different translators and are written in the language of interest.

Monolingual corpus of comparable documents Contains pairs of textual documents discussing the same topic. For example, multiple news articles that cover the same story. The documents are all written in the language of interest.

Bilingual parallel corpus Contains pairs of documents where one document is the translation of the other. In particular, the first document is written in a source language and the second one is written in a target language.

Bilingual corpus of comparable documents Contains pairs of documents discussing the same event, with each document written in a different language.

Resource Type	***Synonyms***	***Phrases***	***Sentences***
Monolingual corpus	Glickman and Dagan (2003)	Lin and Pantel (2001); Marton et al. (2009)	
Monolingual parallel corpus		Barzilay and McKeown (2001), Ibrahim et al. (2003)	Pang et al. (2003)
Monolingual corpus of comparable documents	**Our work**	Quirk et al. (2004), Wang and Callison-Burch (2011); **our work**	Barzilay and Lee (2003); Dolan and Brockett (2005)
Bilingual parallel corpus	Dyvik (2004); van der Plas and Tiedemann (2006)	Bannard and Callison-Burch (2005), Callison-Burch (2008), Zhao et al. (2008)	Ganitkevitch et al. (2011)

TABLE 1-2 – Examples of works in the field of paraphrase extraction.

In Table 1-2, we list some of the relevant works. The idea for this layout is borrowed from Wang and Callison-Burch (2011).

There are two additional interesting parameters. The first refers to the extent of the level of linguistic analysis the investigated system performs to capture the features of the language of interest, and the second parameter refers to the size of the corpus the system requires for producing reasonable results. Figure 1-19 puts some of the techniques we examine on a double scale, where the abscissa measures the corpus size in number of words, and the ordinate measures the level of linguistic analysis performed by each technique. We now elaborate further on some of the aforementioned works.

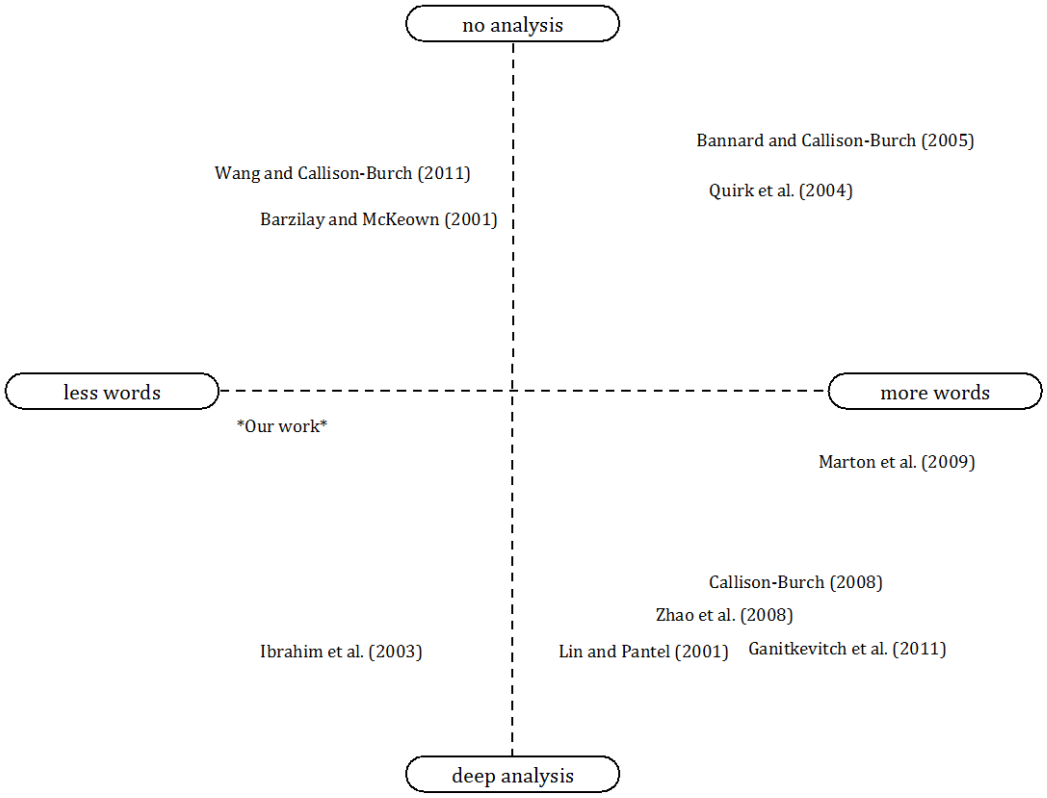


FIGURE 1-19 – Paraphrasing techniques classification for corpus size on the abscissa, and the level of linguistic analysis on the ordinate.

1.7.1.1 Bannard and Callison-Burch (2005) – Using Bilingual Parallel Corpus

Bannard and Callison-Burch (2005) used parallel corpora of English paired with other languages for extracting sub-sentential paraphrases, an approach known as "pivoting". Their main idea, borrowed from phrase-based machine translation, is to use alignment methods, applied on the phrase level, and select source-language phrases that translate into the same target-language phrase as paraphrases. Figure 1-20 demonstrates this method, using English-German parallel text.

Given a phrase e_1, the system looks for its paraphrases using phrase-aligned bilingual parallel text. Since e_1 may appear several times in the entire corpus, every time aligned with a different translation, they define a probability score for every candidate pair in the following way:

$$p(e_2|e_1) = \sum_f p(f|e_1)p(e_2|f) \tag{1.3}$$

namely, the probability of e_2 being a paraphrase of e_1 depends on the set of their common translations in the parallel corpus. The probability models $p(f|e_1)$, $p(e_2|f)$ are calculated using maximum-likelihood estimation, that is

$$p(e|f) = \frac{count(e,f)}{\sum_e count(e,f)}$$

The value $count(e,f)$ is calculated using multiple corpora of parallel texts, with each one pairing English with a different language. Therefore, searching for the best paraphrase(s) $\hat{e}$, given e_1, is done by means of

$$\hat{e} = argmax_{e_2} p(e_2|e_1)$$

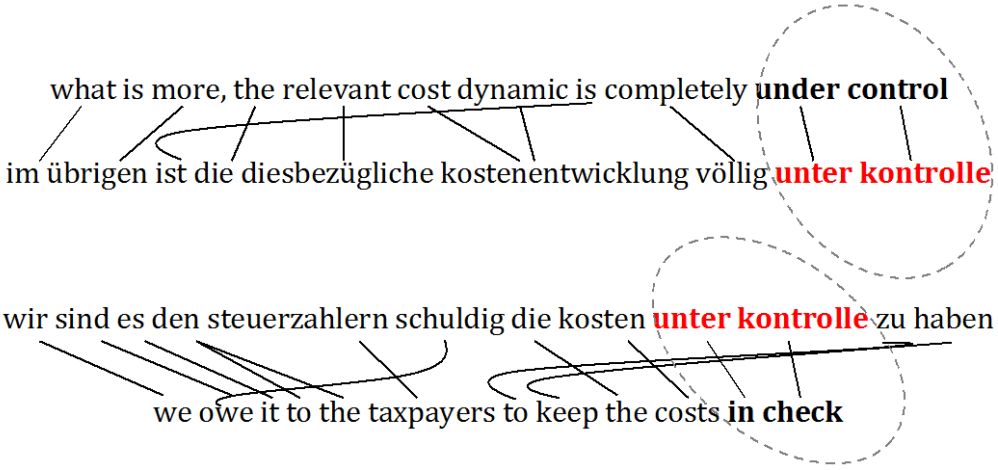


FIGURE 1-20 – Using bilingual parallel text to extract paraphrases (Bannard and Callison-Burch, 2005).

In the experiment, they used multiple corpora containing four million sentence pairs altogether, and manually evaluated the quality of the extracted paraphrases in different conditions. The evaluation was done on 289 input phrases by two native English speakers. For each input phrase, their system created several paraphrase candidates and each was examined considering the original context where they were found. In their best settings, 70% of the candidates were identified as correct.

Callison-Burch et al. (2006) experimented with Spanish and French and the extracted paraphrases were used to improve a Spanish-to-English and French-to-English phrase-based statistical translation systems, respectively. They defined a new feature function combined with the log linear model of the translation system that assigns the probability scores, as defined in Equation 1.3, for every candidate. They reported on an improvement of BLEU score when the translation system employed a relatively small

corpus of bilingual parallel; for larger corpus sizes, the improvement became less significant. In addition to BLEU, they did some manual evaluation that showed a greater improvement over a baseline system that did not use paraphrases. They argued that BLEU was insensitive to their improvement, because most of the translated phrases that were generated by some extracted paraphrases were not found in any reference translation.

Extracting paraphrases from a bilingual resource was tried earlier by Wu and Zhou (2003). Instead of using a word-aligned bilingual parallel corpus, they first built a simple translation system that was capable of translating English text into Chinese based on a bilingual dictionary. They derived English paraphrase patterns from a parsed monolingual corpus using syntactic dependencies and synonyms from WordNet. In other words, their templates were composed of two groups of synonyms connected by a single word that was found to be syntactically dependant on at least one word from each group. Then, potential paraphrases were generated from the template, each composed of a different combination of synonyms. In the next step, they translated each paraphrase into Chinese and used the translations to determine whether the original English phrases are true paraphrases.

To the best of our knowledge, the pivoting technique was never applied to Arabic, mainly because of the lack of sentence-aligned bilingual texts that pair Arabic with languages other than English. Madnani and Dorr (2010) tried to use Arabic as a pivot language for deriving paraphrases for English. Among true paraphrases, they found pairs of different morphological variants conveying the same meaning (e.g., *caused clouds* ⇔ *causing clouds*) and pairs of phrases that only share partial meaning (e.g., *accounting firms* ⇔ *auditing firms*). They argued that, besides the linguistic differences between English and Arabic, the main reason for obtaining such pairs is incorrect word and phrase alignments.

Several works further extended the pivoting technique. Zhao et al. (2008) used a dependency parser to parse the English side of the corpus for finding paraphrase patterns that capture a class of words of the same part-of-speech tag; for example: *consider NN* ⇔ *take NN into consideration*. In another work, Callison-Burch (2008) requested that a phrase and its paraphrases be of the same syntactic type, modeled by Combinatorial Categorical Grammar (CCG) (Steedman, 1999). They have managed to improve the quality of the results by 19% over the original work of Bannard and Callison-Burch (2005), measured by human evaluators. In a recent work (Ganitkevitch et al., 2011),

this technique was extended to capture structural paraphrases on the sentence level, using Synchronous Context-Free Grammar (SCFG).

There are a number of works on automatic thesaurus creation using bilingual parallel texts. Similarly to what we have seen so far, they used the parallel texts mostly for finding source-language words that share the same translations. One interesting work was done by Dyvik (2004) who uses an English-Norwegian parallel corpus for building a lattice of semantically related English and Norwegian words. Then, relations like synonyms and hyponyms were discovered. Another related work (van der Plas and Tiedemann, 2006) uses multilingual sentence-aligned parallel texts, for extraction of synonyms, antonyms and hyponyms for Dutch.

In general, bilingual parallel texts are not available for many language pairs. Therefore, this technique is limited to work only on languages that have this kind of resource available. Existing parallel corpora for Arabic are usually translated to English and since we currently focus on improving Arabic-to-English machine translation, any additional parallel corpus could have been pre-processed by a translation system in the usual way. The contribution of additional equivalents that were extracted from a parallel corpus, which is loaded into the system in the traditional way, is expected to be very limited. Since Arabic is one of the UN official languages, we could have built such corpora using the formal published documentation by the UN, provided in seven different languages. Using the pivoting algorithm on automatically sentence-aligned Arabic-(the other 6 UN languages) corpora is something that should be considered in the future.

1.7.1.2 Marton, Callison-Burch, and Resnik (2009) – Using Monolingual Corpus

In this work, paraphrases were generated to improve Spanish-to-English and English-to-Chinese phrase-based statistical machine translation. They based their inference approach on a concept known as *distributional similarity,* that is, the hypothesis that words/phrases that occur in the same contexts tend to have similar meanings. A significant portion of research on this topic has been devoted to finding clever ways of modeling the context of the words (e.g., Dagan et al., 1995; Dagan et al., 1999; Lee, 1999).

Particularly, in this work, for each phrase that was left without a translation, they looked for it in a monolingual corpus and recorded the contexts in which it appeared. They modeled the contexts using *distributional profiles*, that is, vectors that capture phrase occurrences with their context words, and searched for other phrases with the

most similar distributional profiles to improve the translation. In particular, a distributional profile (*DP*) of a phrase *p* is calculated by:

$$DP_p = \{< v, LLR(p, v) > | v \in V\}$$

where *V* is the entire vocabulary and *v* is a single word in the vocabulary. $LLR(p, v)$ implements Dunning's (1993) Log-Likelihood Rate score, which essentially examines the similarity between the event of seeing *v* near *p* and the event of not seeing *v* in the environment of *p*. In other words, it addresses the question of "how likely is it that *v* occurs near *p*". The notion of nearness, in this sense, was defined as words that occur in a window of three words to the left and three words to the right of *p*. Following McDonald's work (2000), the semantic similarity of two phrases p_1, p_2 is calculated using the cosine similarity value of their *DP* vectors.

To summarize, given a phrase *p* for paraphrasing, their system first calculates the *DP* of *p* and then looks for all phrases with similar *DP*s in the monolingual corpus, ranked by their cosine similarity value. They experimented with Spanish and English, where paraphrases were generated for improving Spanish-to-English and English-to-Chinese phrase-based statistical translation systems. In both experiments they improved BLEU scores in cases of using a relatively small bilingual parallel corpus by the translation system. Upon increasing the size of the corpus to 6.4 million English words in the English-to-Chinese experiment and to 2.3 million Spanish words in the Spanish-to-English experiment, they stopped observing any improvement.

One issue with this method is the need for a relatively large amount of monolingual texts for this method to be effective. However, obtaining monolingual texts is considered relatively easy, compared to obtaining bilingual texts. Another issue that is worth noting is the fact that this method captures antonym phrases as well.

1.7.1.3 Lin and Pantel (2001) – Using Monolingual Corpus

In this work, the authors extracted paraphrases from a large monolingual English corpus by measuring the similarity of the syntactically dependent parts. Essentially, they used a dependency syntax parser, in this case MiniPar (Lin, 1993), to parse every sentence in their corpus and measured the similarity between paths in the dependency trees using mutual information, as proposed by Lin (1998). Paths with high mutual information were defined as paraphrases. In fact, this technique produces inference rules, such as

X is author of *Y* ≈ *X* wrote *Y*,

which can then be used to generate a large number of structural paraphrases.

One disadvantage of this technique is that the extracted paraphrases can have the opposite meaning. Another drawback of this method is the level of analysis it requires. Although nowadays the field of dependency parsing is well established and many parsers are developed for other languages, finding a robust dependency parser for a language other than English is still considered challenging.

Glickman and Dagan (2003) described an algorithm for finding synonymous verbs in a monolingual corpus. They also used a syntax parser for building a vector containing the subject, object and other arguments for every verb they find in their corpus. Later they use these vectors to look for similarities between verbs. Overall, this technique showed competitive results to the one introduced by Lin and Pantel (2001). Nonetheless, since both techniques may perform differently on a given case, they suggested combining them to get better results.

1.7.1.4 Barzilay and McKeown (2001) – Using Monolingual Parallel Corpus

This is the most inspiring work for us. In this work, the authors extracted paraphrases from a monolingual parallel corpora, that is, multiple English translations of the same novels, published originally in foreign-languages. Naturally, monolingual parallel corpora are a valuable resource for learning paraphrases, as they have two independent translators using their own words to convey the same meaning.

They began by finding parallel sentences, that is, pairs of sentences that have the same meaning in the context of the original novel, using a sentence-alignment algorithm (Gale and Church, 1991)**,** which considers the number of identical words shared by a potential pair of sentences. They ended up with about 44K pairs of sentences, corresponding to about 1.8M words, from which they extracted paraphrases using co-training.

They used two classifiers: one that models the context of potential paraphrases, based on a context of three words to the left and right, and another one that models the paraphrases' words. They considered identical words shared by sentences of a given pair as positive examples for training the context classifier. All non-identical words' pairs were deemed negative examples. Based on the resulting context, the other classifier was trained on the phrases' words, and this co-training process was repeated until no new paraphrases were extracted.

The classifiers' machinery is quite simple. For every instance (context or phrase words), they checked how many times it occurs within a positive/negative examples, divided by its total frequency. This number is called the *strength* of the instance, so the most powerful *k* positive and negative instances (*k*=10 in their experiments) were selected as the outcome model. The instances for both classifiers are composed of pairs of texts, represented by the part-of-speech tags of the individual word, and their lemma-based similarity.

Their system extracted 9,483 paraphrases, from which they sampled 500 for evaluation by two native English speakers. The evaluation was performed twice: once with showing the context of the paraphrases, and another time without. The evaluators were equipped with a definition that describes paraphrases as "approximate conceptual equivalence". The results were 85-87% accuracy when the context was not shown to the evaluators, and 91.4-91.8% when the context was given. However, 70.8% of the paraphrases were single words.

Ibrahim et al. (2003) extracted English paraphrases from the same corpus of monolingual parallel documents. Essentially, they used a similar technique to that of Lin and Pantel (2001) for extracting structural paraphrases using dependency parsing. By working with a monolingual parallel corpus, they hoped to reduce the number of cases in which pairs of phrases that have the opposite meaning are deemed paraphrases--one of the main drawbacks of Lin and Pantel's approach. They manually evaluated 130 pairs of paraphrases (with three evaluators) and obtained an average precision of 41.2%. A disadvantage of this technique, like that of Lin and Pantel, is its dependency on the availability of a high-quality parser.

This is an interesting and straightforward technique for extracting paraphrases from such a valuable resource. However, finding resources like this is challenging. In our work, we follow a similar idea, implemented on Arabic. Since there are no monolingual parallel corpora available for Arabic, we used a corpus of comparable documents. Considering that Arabic is a morphologically rich language, we incorporated morphological features of the surrounding words as well as the paraphrase patterns themselves.

1.7.1.5 Barzilay and Lee (2003) – Using Monolingual Comparable Documents

In this work, Barzilay and Lee used English comparable corpora to create sentence-level paraphrases. They employed a multi-sentence alignment technique to each corpus for finding similarities between sentences. Based on the those similarities, they gener-

ated lattices that capture common words and phrases of the aligned sentences, and then keep only the nodes that at least 50% of the sentences use. The reminder of the nodes, corresponding to words that are used by less than 50% of the sentences, were replaced by a generic node, referred to as *slot*. Figure 1-21, borrowed from the original paper of Barzilay and Lee (2003), demonstrates a lattice and its resulted slotted lattice.

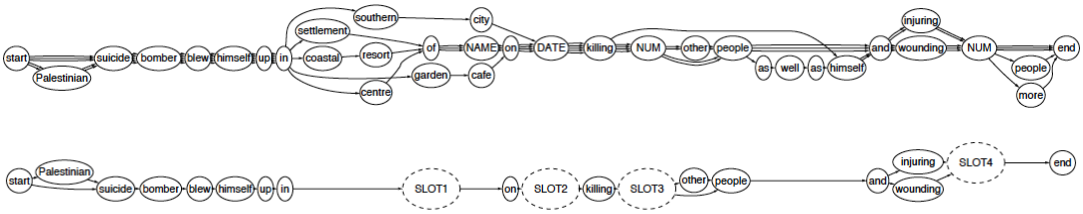


FIGURE 1-21 – An example for a lattice (top) and its corresponding slotted lattice (bottom). Presented in (Barzilay and Lee, 2003).

In the next step they compare lattices from different comparable corpora, calculating a similarity score for every potential lattice pair. Pairs that pass a predefined threshold are deemed paraphrases. In fact, every such a pair is a template for paraphrasing rather than a single paraphrases pair. The similarity between two lattices is determined by the number of similar arguments their sentences have for the slots. For example, let us look at the lattices *SLOT1 bombed SLOT2*, and *SLOT3 was bombed by SLOT4*, derived from different comparable corpora. We look at the sentences that were used for generating the first lattice and find the sentence *the plane bombed the town*. Similarly, we find the sentence *the town was bombed by the plane* in the cluster of sentences that generated the second lattice. Based on those two sentences they see a similarity between the lattices, such that SLOT1 and SLOT4 are assigned with the same argument *the plane*, and SLOT2 and SLOT3 are assigned with the same argument *the town*. Therefore, those two lattices are considered as a paraphrasing template.

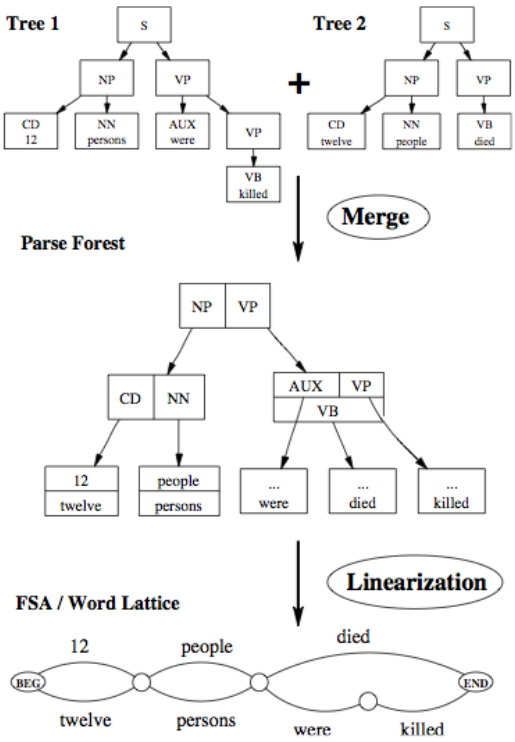


FIGURE 1-22 – An example for merging two syntactic trees for generating a paraphrasing word lattice. Presented in (Pang et el., 2003).

The actual paraphrases are generated as follows: given a sentence for paraphrasing, they first try to match it to one of the generated lattices. Once a relevant lattice is found, they used the corresponding matched templates to generate paraphrases, replacing the slots with the original sentence's arguments. In their experiments they generated 6,534 template pairs.

Pang et al. (2003) extended this work by generating lattices based on an alignment method that use syntactic properties, extracted with the help of a syntax parser. In fact, they worked on a different corpus type, multiple English translations of the same Chinese original sentence, a corpus that was developed for machine translation evaluation purposes. Altogether, their corpus contains 11 English translations of 889 Chinese original sentences, from which they worked on every pair of English sentences among the available 11 ones. They worked on the syntactic trees of the sentences and merged trees from the same sentence group. Then, some of the merged nodes were treated as potential paraphrases. Figure 1-22, borrowed from (Pang et al., 2003) illustrates this process.

1.7.1.6 Dolan and Brockett (2005) – Using Monolingual Comparable Documents

In this work, a repository of 5,801 English paraphrases on the sentence level is introduced. The pairs were extracted from a corpus of comparable news articles broken into sentences, gleaned from various online sources in a period of about 2 years. Two heuristics were used to collect candidate sentence pairs for paraphrasing: (1) the two sentences have a Levenshtein edit distance (Levenshtein, 1966) limited by 20, and their length ratio is 66%; (2) each of the two sentences is one of the three first sentences in the article from which they were extracted. With these heuristics, they managed to distil an initial collection of sentence pairs to 49,375 pairs, which were then used as candidates for consideration as paraphrases.

Simultaneously, they manually labelled a set of 10,000 sentence pairs with two labels: paraphrase or not paraphrase. This set was used for training a supervised classifier, based on Support-Vector Machines (SVM) (Vapnik and Cortes, 1995), modeling various feature sets calculated on the word level. By applying the trained classifier on the distilled 49,375 sentence pairs, they managed to generate 20,574 paraphrases; 5,801 were randomly selected for manual evaluation.

The evaluation was conducted by two evaluators, with a third one consulted when the first two did not agree. The majority decision was selected as the label for each pair. For every candidate pair, the evaluators were requested to answer the question whether they are semantically equivalent. Not surprisingly, as working with paraphrases on the sentence level, they had to use a more relaxed definition for paraphrases. Recall that the rigid definition for paraphrases is based on the two-way textual entailment concept; however, with sentences, this definition rules out many pairs, mainly due to small pieces of information that exist only in one of the sentences, for example:

> *David Gest has sued his estranged wife Liza Minelli for %MONEY% million for beating him when she was drunk*
>
> and
>
> *Liza Minelli's estranged husband is taking her to court for %MONEY% million after saying she **threw a lamp** at him and beat him in drunken rages*

The minor event of throwing a lamp is missing in the first sentence. Therefore, they had

to ask the evaluators to treat such cases as paraphrases. Overall, 67% of the 5,801 sentence pairs were judged as paraphrases.

Quirk et al. (2004) used this repository as a resource for extracting paraphrases on the phrase level. They applied a word-based alignment algorithm, similar to the one used by phrase-based statistical machine translation, to align the sentence pairs on the word and phrase levels. Then, given a source input for paraphrasing, a simple machine-translation decoding algorithm was used in a way that every aligned word or phrase composed of non-identical words was deemed paraphrases.

Wang and Callison-Burch (2011) created their own corpus of parallel English sentences, extracted from English Gigaword (Graff and Cieri, 2003). They began with collecting comparable documents from which they extracted the parallel sentences by setting a threshold on the number of shared words. A weight was assigned to every word, referred to as term, based on its *tf-idf* score. The tf-idf score is the term frequency multiplied by the inverse document frequency, which is essentially the reciprocal number of documents in which the term is mentioned. The parallel sentences were extracted using a different heuristic than those used by Dolan and Brockett (2005): they counted the number of shared *n*-grams (with $1 \leq n \leq 4$) and experimented with different minimum and maximum threshold values. They evaluated both, the document and sentence pairs, using Amazon's Mechanical Turk[4], and concluded with encouraging results.

The sentence pairs were used further for learning new paraphrases on the phrase level. They followed Munteanu and Marcu (2006), who used machine-translation alignment methods applied on bilingual comparable documents, for extracting additional sentence and phrase translations in order to enrich the parallel corpus used by a statistical translation system, hence increasing its coverage. Inspired by this approach, Wang and Callison-Burch (2011) applied different alignment methods on the monolingual parallel sentences for learning paraphrases. Basically, they considered the alignment score given for every word by the alignment algorithm, and used that to detect the most reliable aligned patterns to be considered as paraphrases. They also tried this technique with Dolan and Brockett's repository of parallel sentences, and manually evaluated the extracted parallel phrases from 1051 parallel sentences with Mechanical Turk. Using their best configurations, 62% of the phrase pairs, extracted from their own

[4] http://www.mturk.com

corpus of parallel sentences, were indeed paraphrases. Similarly, when using Dolan and Brockett's corpus, this number grew to 67%. As previously seen, those numbers are based on a relaxed definition of paraphrases. In this case, their evaluators even used two different labels to distinguish between "related" phrases, that is, "almost" paraphrases, and phrases that completely match on semantic level. If only considering the latter type, they were left with 36% correct pairs from their own corpus, and 49% correct pairs extracted from Dolan and Brockett's.

In a previous work, Shinyama et al. (2002) extracted sub-sentential Japanese paraphrases from comparable sentences, which were extracted from comparable documents. They focused on finding similarities between comparable sentences mainly based on mentions of the same named entities. They reported on 49% precision on one domain and 94% precision on another domain. Both domains were modeled by a relatively small set of sentences.

In all the above-mentioned works, there is always a step in which similar sentences are extracted from the comparable documents and used as a resource for paraphrasing on the phrase level. Obviously, the main advantage of doing that is to enable the usage of word alignment algorithms, which cannot be applied on random selection of sentence pairs. However, this step dramatically reduces the size of the resource, and, by that, the number of extracted paraphrases. For example, Dolan and Brockett started with 13,127,938 sentence pairs and ended up only with 49,375 pairs. Moreover, although Wang and Callison-Burch extracted many more sentence pairs than did Dolan and Brocket, the quality of the extracted phrase pairs was poorer. In our work, we skip this step by using the comparable documents directly for finding paraphrases on the phrase level.

1.7.2 Multiword Expression

Among all the MWE-related tasks, identifying MWEs in a running text is probably the most common one. There are many works addressing different angles of MWE identification (e.g. Baldwin et al., 2003; Schone and Juraksfy, 2001). Some works took unsupervised approaches to the MWE classification problem (Fazly and Stevenson, 2007; Cook et al., 2007).

Our work is mostly inspired by Diab and Bhutada (2009), who applied a supervised learning framework to the problem of classifying English verb-noun constructions (VNCs) as idiomatic or literal in running text, on the token level. They adopted a chunking approach, indicating the label of every token using the Inside-Outside-Beginning

(IOB) notation, and used the annotated corpus provided by Cook et al. (2008), a resource of almost 3,000 English sentences annotated with VNCs. Hashimoto and Kawahara (2008) addressed token classification into idiomatic versus literal for Japanese MWEs of all types. They annotated a corpus of 102K sentences, and used it to train a supervised classifier for MWEs. Katz and Giesbrecht (2006) carried out a vector similarity comparison between the context of an English MWE and that of the constituent words using Latent Semantic Analysis (LSA) (Deerwester et al., 1990) to determine if the expression is idiomatic or not. Various other works addressed the same problem using an unsupervised framework (Birke and Sarkar, 2006; Diab and Krishna, 2009a; Diab and Krishna, 2009b; Sporleder and Li, 2009).

Kim and Baldwin (2009) identified English verb particle constructions in raw text using syntactic and semantic features of the subject and object(s) of the verb, as input for a supervised learning algorithm. They covered both continuous and discontinuous VPC instances, allowing non-MWE words to appear between the MWE words (e.g. *he **put** the sweater **on***), also known as *gaps*. In this thesis, we address the issue of gappy MWEs, but in addition to handling them for VPCs, we extend our approach to handle gappy MWEs in general.

Using MWEs in machine translation is another application. Carpuat and Diab (2010) studied the effect of integrating English MWEs with a statistical translation system, using WordNet 3.0 as the main source for MWEs. In a recent work by Lancioni and Boella (2012), a small number of idiomatic Arabic MWEs were extracted from Arabic-English translation memories using CCG.

Arabic MWEs have been already investigated in previous research. Attia et al. (2010) extracted Arabic MWEs from various resources. They focused only on nominal MWEs and used diverse techniques for automatic MWE extraction from cross-lingual parallel Wikipedia titles, machine-translated English MWEs taken from the English WordNet, and Arabic Gigaword (4th ed.) (Parker et al., 2011a). They found a large number of MWEs; however, only a few of them were evaluated.

There are some works on MWE in Hebrew, another key highly inflected Semitic language. Tsvetkov and Wintner (2011) used various linguistic features that capture word inflections, combined with some statistical parameters, in a Bayesian Network framework, for identifying MWEs in a monolingual text. In another work, Tsvetkov and Wintner (2012) developed a system for learning a repository of Hebrew MWEs from a word-aligned bilingual parallel corpus, and applied it to a Hebrew-English parallel corpus.

As part of our work, we introduce a repository of Arabic MWEs containing about 5,000 expressions, each one assigned with its syntactic class, and every word with its context-sensitive SAMA morphological analysis. Additionally, we develop a deterministic pattern-matching algorithm, to annotate MWEs in Arabic texts considering different morpho-syntactic variations. Ultimately, we employ the pattern-matching algorithm to generate a noisy supervised training set, which is then used to augment a manually annotated data set for building an Arabic MWE classifier.

Chapter 2

Discovering Arabic Noun Synonyms Using WordNet

2 Discovering Arabic Noun Synonyms Using WordNet

In this chapter we present a deterministic approach for extracting Arabic synonyms from existing linguistic resources, forming a thesaurus for Arabic. As a case study, we focus only on nouns. Intuitively, dealing with verbs seems to be more difficult than nouns, since the meaning of Arabic verbs usually changes when used with different prepositions; for example, the meaning of the verb *qDY* is "judged", and when followed by the preposition *ElY*, "on", is "brought an end to". We handle verbs in the following chapter.

The thesaurus was extracted from the list of stems provided by the Buckwalter (version 1.0) Arabic Morphological Analyzer (Buckwalter, 2002), also known as BAMA 1.0, and then organized in levels of perceived synonymy. Even until now, to the best of our knowledge, there is no publicly available Arabic thesaurus. The current efforts of building the Arabic WordNet (Black et al., 2006) are not over yet.

We continue as follows: In Section 2.1 we give a detailed description of BAMA 1.0 stems repository, and in Section 2.2, we describe how we use it for building a collection of Arabic noun synonyms. Section 2.3 describes the way we use the synonyms to improve an Arabic-to-English example-based translation system, and Section 2.4 presents our experimental results. Finally, we conclude in Section 2.5.

2.1 BAMA 1.0 Stems Repository

BAMA 1.0 is provided with a list of Arabic stems; one stem entry contains the following morpho-syntactic information:

1. The unvocalized stem (without short vowels);
2. the vocalized stem;
3. the stem's lemma;
4. the stem's morphological category (for controlling its compatibility with prefixes and suffixes);
5. its English gloss(es).

In fact, the English glosses contain some semantic information about the stem that we exploit for building our thesaurus. Table 2-1 shows an example for a couple of stem entries where the five columns of the table show the five items described above. The first two entries form together an example for a vocalized stem that is affiliated with

two different lemmas: *liwA'_1* and *liwA'_2*. The two different affiliations are described by two stem entries differing in their lemma information and English glosses. The morphological category of each entry contains codes that were defined by the creators of BAMA 1.0, so that one will know what are all the possible affixes that may be combined with the stem; for instance, NduAt enables all feminine and dual noun suffixes. The morphological category also implies the part-of-speech of the stem; for example, PV is always a verb. However, some categories, like most of those that begin with the letter N, can be used on stems with different part-of-speech tags. Fortunately, in such cases, the part-of-speech tag is mentioned explicitly with an additional item.

Unvocalized stem	***Vocalized stem***	***Lemma***	***Morphological category***	***English gloss(es)***
lwA'	*liwA'*	*liwA'_1*	N0_Nh_L	banner;flag
lwA'	*liwA'*	*liwA'_2*	N0_Nh_L	major general;brigade
tbll	*tabal~ul*	*tabal~ul_1*	NduAt	moistness;humidity
blg	*bal~ag*	*bal~ag_1*	PV	communicate;convey

TABLE 2-1 - Examples of BAMA 1.0 stem entries.

2.2 Building the Thesaurus

For building our thesaurus, every noun stem from the stems repository was compared to all the other stems for looking for synonym relations. As mentioned previously, each stem entry contains one or more English glosses. A naïve approach to find synonyms is by saying that every two stems sharing some of their glosses are synonyms. However, sharing English glosses is insufficient for determining that two stems are synonymous, mainly because of polysemy expressed by English glosses: we do not know which of a gloss's possible senses was intended for any particular stem. For example, take the gloss *banner* from Table 2-1, does it mean a flag or a slogan?

Therefore, we attempt to determine stem senses automatically by asking the English WordNet for all (noun) *synsets* of every English gloss of a specific stem. A synset in WordNet is defined as a set of words, representing a specific sense. A word in WordNet, is a member of all the synsets that represent its different senses. In our case, a synset containing two or more of the glosses is taken to be a possible sense for the given stem. This assumption is based on the idea that if a stem has two or more different glosses

that semantically intersect, it should probably be interpreted as their common meaning. Back to the previous example, the glosses *banner* and *flag* are members of the same synset, representing the sense "a piece of cloth bearing a symbol", hence we choose this sense to represent the Arabic lemma *liwA'_1*.

Based on this technique, finding senses, therefore, is possible only for those Arabic noun stems that are provided with more than one English gloss; fortunately, there are 26,913 of them in BAMA 1.0. We decided also to consider the hyponym-hypernym relation between the glosses' senses and understand a stem to have the sense of the shared hyponym in this case. It worth mentioning that using this technique, we assign English WordNet senses to Arabic stems.

Based on the above information, we define five levels of synonymy for Arabic stems:

Level 1 Two stems have more than one gloss in common.

Level 2 Two stems have more than one sense in common, or they have just one sense in common but this sense is shared by all the translations.

Level 3 Each stem has one and the same gloss.

Level 4 Each stem has exactly one gloss and the two glosses are English synonyms.

Level 5 The stems have one gloss in common.

Every stem pair is assigned the highest possible level of synonymy, or none when none of the above levels applies.

The resultant thesaurus contains 22,621 nouns; Table 2-2 summarizes the amounts of relations as we found, classified by their level.

Type	***Amount***
Level 1	20,512
Level 2	1479
Level 3	17,166
Level 4	38,754
Level 5	137,240

TABLE 2-2 – Amounts of relations of each level.

The quality of an Arabic-to-English translation system is tested for each level of synonymy, individually, starting with level 1, then adding level 2 and so forth. The results are reported in the subsequent section. Figure 2-1 shows an example of a relation between two Arabic stems. In this example, the stem *AEAdp*, "return", is matched to the

stem *krwr*, "return", on level 2 because the first stem is translated as both "repetition" and "return", which share the same synset. The second stem is translated as "return" and "recurrence", which also share the same synset as the first stem. Therefore level 2 is the highest appropriate one. Table 2-3 shows some extracted synonyms and their levels.

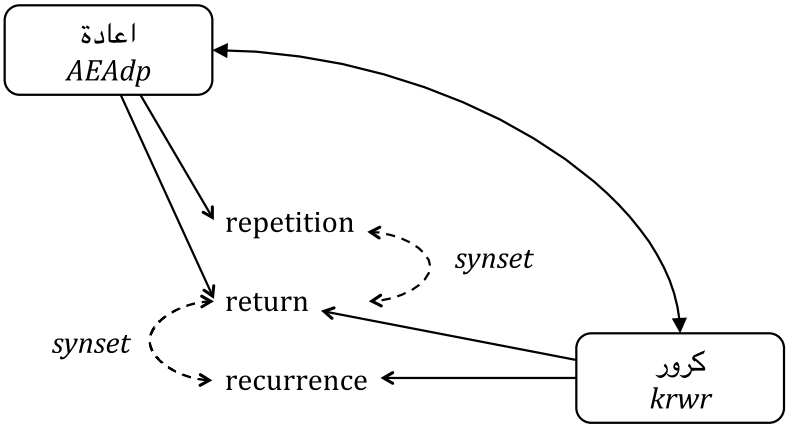


FIGURE 2-1 – Synonym relation, level 2 example.

Synonyms	***Level***
n$yj ⇔ *dmE* ("crying")	4
sTH ⇔ *sqf* ("ceiling")	5
zlEwm ⇔ *Hlqwm* ("throat")	1
njdp ⇔ *AEAnp* ("help;support")	2
AbtdA' ⇔ *ftH* ("beginning")	5
AxtrAE ⇔ *AbtkAr* ("invention")	3

TABLE 2-3 – Examples of extracted synonyms.

2.3 Using Noun Synonyms in Translation

We now use the repository of noun synonyms in trying to improve the quality of an Arabic-to-English translation system. In this chapter as well as the following one, we use our own implementation of a simple Arabic-to-English example-based system (Bar et al., 2007) as a test case (implemented as part of the author's MSc thesis). In Chapter 5, we use longer paraphrases for improving a *Moses* (Koehn et al., 2007) implementation of a

phrase-based statistical machine translation. We begin by describing our simple example-based system implementation. We suggest the reader to read the background information on the example-based paradigm, provided in the previous chapter, before reading the next section.

2.3.1 Example-based Translation System Description

2.3.1.1 Translation Corpus

The translation examples in our system were extracted from a collection of parallel, sentence-aligned, unvocalized Arabic-English documents. All the Arabic translation examples were morphologically analyzed using the BAMA 1.0, and then part-of-speech tagged using AMIRA (Diab et al., 2004) in such a way that, for each word, we consider only the relevant morphological analyses with the corresponding part-of-speech tag (at the time we performed our experiments, MADA was still premature).

Each translation example was aligned on the word level, using Giza++ (Och and Ney, 2003), which is an implementation of the IBM word alignment models (Brown et al., 1993). The Arabic version of the corpus was indexed on the word, stem and lemma levels. So, for each given Arabic word, we were able to retrieve all translation examples that contain that word on any of those three levels.

2.3.1.2 Matching

Given a new input sentence, the system begins by searching the corpus for translation examples for which the Arabic version matches fragments of the input sentence. In the implementation we are describing, the system is restricted to fragmenting the input sentence so that a matched fragment must be a combination of one or more complete adjacent base phrases of the input sentence. The base phrases are initially extracted using the AMIRA tool. The same fragment can be found in more than one translation example. Therefore, a *match score* is assigned to each fragment-translation pair, signifying the quality of the matched fragment in the specific translation example. Fragments are matched word by word, so the score for a fragment is the average of the individual word match scores. To deal with data sparseness, we generalize the relatively small corpus by matching words on text, stem, lemma, morphological, cardinal, proper-noun, and synonym levels, with each level assigned a different score. These match-levels are defined as follows:

Text level An exact match. It credits the words in the match with the maximum possible score.

Stem level A match of word stems. This match-level currently credits words with somewhat less than a text-level match only because we do not have a component that can modify the translation appropriately.

Lemma level Words that share a lemma. For the same reasons as stem-level matches, an imperfect match score is assigned in this case. As previously shown, when dealing with unvocalized text, there are, of course, complicated situations when both words have the same unvocalized stem but different lemmas, for example, the words *ktb*, "wrote", and *ktb*, "books". Such cases are not yet handled accurately, since we are not working with a context-sensitive Arabic lemmatizer, such as MADA, and so cannot unambiguously determine the correct lemma of an Arabic word. Actually, by "lemma match", we mean that words match on any one of their possible lemmas. Still, the combination of BAMA 1.0 and the AMIRA part-of-speech tagger allows us to reduce the number of possible lemmas for every Arabic word, so as to reduce the degree of ambiguity. We believe that repeating this experiment working with MADA, will allow us to better handle such situations.

Cardinal level Numeric words. Correcting the translation of the input word is trivial.

Proper-noun level Words that are both tagged as proper nouns by the part-of-speech tagger. In most cases, the words are interchangeable and, consequently, the translation can be easily fixed in the transfer step.

Morphological level Words that match based only on their morphological features. For example, two nouns that have the definite article *Al*, constitute a morphological match. This is a very weak level, since it basically allows a match of two different words with totally different meanings. In the transfer step, some of the necessary corrections are done, so this level appears, all the same, to be useful when using a large number of translation examples.

Synonym level The additional feature investigated in the current work, are words that are deemed to be synonyms, according to our automatically extracted thesaurus. Since synonyms are considered interchangeable in many cases, this level credits the words with 95%, which is almost the maximum possible. Using a score of 100% reduces translation results because sometimes synonym-based fragments hide other text-based fragments, and the latter are usually more accurate.

At this point in our experiments, we are using ad-hoc match-level scores, with the goal of a qualitative evaluation of the effect of including the synonym level for match-

ing. Exact-text matches and cardinal matches receive full weight (100%); synonyms, just a tad bit less, namely 95%; stems and proper nouns, 90%; lemmas and stems are scored at 80%; morphological matches receive only 40%.

Fragments are stored in a structure comprising the following:

Source pattern The fragment's Arabic text, taken from the input sentence.

Example pattern The fragment's Arabic text, taken from the matched translation example.

Example The English translation of the example pattern;.

Match score The score computed for the fragment and its example translation. Fragments with a score below some predefined threshold are discarded, because passing low-score fragments to the next step would dramatically increase the total running time and sometimes make it unfeasible to process all fragments.

2.3.1.3 Transfer

The input to the transfer step consists of all the collected fragments that were found in the matching step, and its output is the translations of those fragments. Translating a fragment is done in two main steps: (1) extracting the translation of the example pattern from the English version of the translation example; and (2) fixing the extracted translation so that it will be the translation of the fragment's source pattern.

2.3.1.4 Recombination

In the recombination step, we paste together the extracted translations to form a complete translation of the input sentence. This is generally composed of two subtasks. The first is finding the best recombination of the extracted translations that cover the entire input sentence, and the second is smoothing out the recombined translations to make a fully grammatical English sentence. Our simple implementation handles only the first subtask. By multiplying the total-scores of the comprised fragments, our system calculates a final score for each generated recombination.

2.3.2 Using Noun Synonyms

The extracted thesaurus was used for matching Arabic fragments based on synonyms. Finding a synonym for a given word is not a simple task, considering that input sentences are not given with word senses. Matching input words based on synonymy without knowing their true senses is error-prone, because one might match two synonym words based on a specific sense that is not the one used by the author. One way to

handle this issue would be to use a Word-Sense-Disambiguation (WSD) tool for Arabic to uncover the intended sense of each input sentence word. Although there has been some research in this area, we could not find any available tool that produces reasonable results.

Another option for matching synonyms is to use the immediate context of a candidate word for matching. Given a pair of words, a window of several words appearing around each may be compared on several WordNet levels and a final score can be computed on that basis. Candidate pairs crossing a predefined threshold can be considered as having the same sense. This direction was left for future investigation.

In this work, we decided to experiment with a different route. We classify each input sentence by topic, as well as all the corpus translation examples. For each translation example, we consider synonyms only if their topic-set intersects with that of the input sentence. The classification was done using the manually tagged Reuters-21578[5] corpus for English, since we could not find a similar corpus for Arabic. The topics covered in that corpus relate to the newswire domain, which is the one we used to test our translation system. First, we trained a simple classifier on the training-set given by Reuters, building statistical model for every topic of the predefined Reuter topic list. We used the support-vector machine (Vapnik and Cortes, 1995) model for this classification task, it having proved to be one of the most appropriate one for classification for this corpus. Feature-vectors consisted of *tf-idf* values for English stems, extracted from English WordNet 2.0 by a morphological analyzer, ignoring stems of stop words. The classifier was tested on 1219 documents from the test-set provided by Reuters, producing accurate results in the 94% range.

In the next step, we used this classifier to classify the English half of all the translation examples in our parallel corpus, allowing for more than one topic per document. In addition, the Arabic part of those translation examples was used as a training-set for training another classifier for the same topic list for Arabic. Like its English equivalent, it uses stems as features, ignores stem of stop words, and creates feature-vectors using the *tf-idf* function. Stems were extracted using BAMA 1.0 (as MADA was still premature at the time we performed this research); since BAMA is not context sensitive, in case of ambiguity, we selected the first stem heuristically. The accuracy of this classifier was not measured due to the lack of any manually tagged test-set.

[5] http://archive.ics.uci.edu/ml/datasets/Reuters-21578+Text+Categorization+Collection

Returning to the translation process: Given a new sentence from an input document, the system begins by classifying the *entire document* using the Arabic classifier and determining its topic-set, which is assigned to all sentences within that document. Finally, during the matching step, we allow the system to consider synonyms only in the case of a non-empty intersection of topic-sets of the input sentence and the examined translation example. The efficiency of this classification feature was examined and results show a slight improvement in final translations compared to the same conditions running without classification. We elaborate further on this in the next section.

2.4 Experimental Results

Experiments were conducted with two corpora of bilingual texts. The first contains 29,992 translation examples (~1.3 million Arabic words) and the second one contains 58,115 translation examples (~2 million Arabic words). The system was tested on all levels of synonyms relations and the effect of using the classification feature on every level was examined.

The following results are based on a test-set taken from the 2009 NIST OpenMT Evaluation set (LDC2010T23), containing 586 sentences corresponding to 20,671 tokens (17,370 words) and compared to four reference translations. We evaluated the results under some of the common automatic criteria for machine-translation evaluation: BLEU (Papineni, 2002) and METEOR (Banerjee and Lavie, 2005). Table 2-4 shows some experimental results, presented as BLEU and METEOR score.

From these results, one can observe that, in general, the system performs slightly better when using synonyms. The most prominent improvement in the BLEU score was achieved when using all levels, 1 through 5, on the small corpus. However, the same experiments using the large corpus did not show significant improvements. This was expected: the larger corpus has more translation examples that might match more fragments exactly. Using synonyms at level 5 caused reductions in all scores in the large corpora. This is probably because level 5 gives synonyms of low confidence, thereby introducing errors in matching corpus fragments, which may hide better fragments that could participate in the output translation. On the other hand, when using level 5 synonyms on the small corpus, the system performed even better than when not using them. That can be explained by the fact that the small corpus probably produces fewer fragments, and the ones based on synonyms can cover ranges of the input sentence, which were not covered by other fragments. However, when using the classification

feature over the large corpus, the system was able to remove some of the problematic fragments, resulting in better scores.

Test	***Small Corpus***				***Large Corpus***			
	+CLAS		-CLAS		+CLAS		-CLAS	
	BLEU	MTOR	BLEU	MTOR	BLEU	MTOR	BLEU	MTOR
Level 1	11.86	47.48	11.76	47.56	15.15	51.83	15.06	51.85
Levels 1 – 2	11.76	47.69	11.73	47.48	15.15	51.83	15.05	51.86
Levels 1 – 3	11.86	47.62	11.76	47.70	15.20	51.86	15.10	51.89
Levels 1 – 4	11.87	47.59	11.79	47.56	15.19	51.84	15.09	51.88
Levels 1 – 5	**11.92**	47.46	11.77	47.51	15.00	51.81	14.84	51.70
No synonym			10.84	44.60			14.85	51.94

TABLE 2-4 – Translation results – BLEU and METEOR (MTOR) scores. CLAS refers to the classification feature as described below.

In general, when synonyms are used and contribute significantly, this classification feature did show some improvement. We can also see that experiments in which synonyms did not help improve translations significantly show a reduction in final scores when using classification. This strengthens our intuition that real synonyms are more likely to be found in documents dealing with similar subject matters. We expect that taking the words' local context into consideration, as mentioned above, would result in even better performance.

	w/ synonyms	***w/o synonyms***
Unigrams	733	738
Bigrams	(+3.2%) 7612	7864
Trigrams	11554	11632
4-grams	11224	11243

TABLE 2-5 – Unseen *n*-grams in the small corpus, tested with and without using synonyms.

In addition to the traditional automatic evaluation for the resulted translations, we have measured the effect of using synonyms, on the corpus coverage. Table 2-5 summarizes the number of unseen (were not translated at all) 1-4 grams when using synonyms

vs. without using synonyms on the small corpus. The results show that when using synonyms the system was able to find an additional 252 bigrams; however, on longer *n*-grams the system did not show significant improvement. As expected, increasing the size of the corpus reduced the positive effect on *n*-gram coverage.

2.5 Summary

The system we are working on has demonstrated a promising potential for using contextual noun synonyms in an example-based approach to machine translation, for Arabic, in particular. Although our BLEU scores are relatively low, we found that noun synonyms benefit from being matched carefully by considering the topic of the sentence in which they appear. Comparing other ways of using context to properly match the true senses of ambiguous synonyms is definitely a direction for future investigation.

Another interesting observation is the fact that using synonyms on a large corpus did not result in a significant improvement of the final results, as it did for a smaller corpus. This suggests that synonyms can contribute to translation systems for language pairs lacking large parallel corpora.

Though the standard scores achieved by our system remain low, primarily because of the limited system implementation, a detailed examination of numerous translations suggests that the benefits of using matches based on synonyms will carry over to more complete translation systems. What is true for our automatically generated thesaurus is even more likely to hold when a quality Arabic thesaurus will become available for mechanical use. In the meanwhile, we will continue working on different methods for automatic extraction of semantic equivalents for Arabic, starting in the next chapter.

Chapter 3

Extracting Arabic Verb Synonyms from Comparable Documents

3 Extracting Arabic Verb Synonyms from Comparable Documents

In this chapter we automatically create a list of synonymous Arabic verbs for the purpose of improving an Arabic-to-English automatic translation system. This time we use a corpus of Arabic *comparable documents* for learning synonym pairs. Comparable documents are texts dealing with the same event, but which are not necessarily translations of the same source. The basic learning technique we describe here is implemented merely on verbs, as a motivation for the next chapter dealing with alternative approaches for learning large sets of (multiword) paraphrases and using them to extend the coverage of a statistical translation system. Our approach for extracting synonyms and later paraphrases is based on the inspiring work by Barzilay and McKeown (2001) on finding paraphrases in a parallel monolingual corpus for English, that is, multiple English translations for several known novels published originally in other foreign language. Understanding how powerful such a resource can be for paraphrasing, but finding no such resource for Arabic, we begin by generating a corpus of Arabic comparable documents.

We continue as follows: Section 3.1 describes the corpus preparation; the extraction technique is then explained in Section 3.2 followed by the evaluation part, described in section 3.3. In Section 3.4 we report on our experiment to use the synonymous verbs in our own simple implementation of an example-based translation system. Our conclusions are in Section 3.5.

3.1 Corpus Preparation

We extract Arabic verb synonyms using a corpus of comparable documents. The evaluation of this process was performed both manually and automatically. As in the noun experiment, we measure the translation quality of a system that uses the extracted synonyms in the matching step. In subsequent chapters we will extend our work for the purpose of finding longer equivalents, extracted from similar comparable corpora.

Our corpus was derived from Arabic Gigaword (4th ed.) following only a simple technique. Arabic Gigaword is a large collection of original Arabic news reports provided with their publication date and the name of their publisher. We pre-process the corpus with MADA 2.1 (Habash and Rambow, 2005; Roth et al., 2008) to discover the words' stems and lemmas. In our case, among all articles published only by one of the

two publishers al-Nahar and al-Hayat on the same day, we took those whose titles matched lexically. The matching criterion was simple: for every candidate pair of articles, we count the number of matched lemmas appearing in their titles and for each single article we choose another article having the larger number of matched lemmas to be its match. For the time being, we eliminated cases in which one article matched more than one document.

3.2 Extracting Verb Synonyms

Given the corpus of comparable documents, the first task is to obtain a (partial) word alignment of every document pair. Since Barzilay and McKeown (2001) in their work used various English translations of the same source, the alignment could be obtained with less effort. This is not the case when using a corpus of comparable documents as a source for synonym extraction. News articles covering the same story are not necessarily a translation of the same source, thus finding word alignments is challenging.

Therefore, in our case we pair every two verbs that are generated from the same lemma. Clearly, such a naïve approach is expected to result in a relatively large number of incorrect aligned pairs due to polysemy. However, since we are working with comparable documents, we believe that for the most part, words tend to carry the same meaning in that particular context.

In addition to this initial alignment, we created a list of potential synonym relations for a large list of Arabic verbs. This list was extracted using the English glosses provided with the Arabic stem list of BAMA 1.0, similar to our noun experiment from the previous chapter, using English WordNet. In this case, stems that share at least one gloss in common or whose glosses are English synonyms, according to WordNet, were deemed to be synonyms. This is equivalent to taking all the synonyms of level 5 as defined in the previous chapter. Unlike nouns, Arabic verbs tend to change their senses with different attached prepositions; therefore, the extracted list of synonymous verbs is error-prone. Note that original stem list does not contain prepositional information, so it cannot be used as a thesaurus.

Given the initial alignment for every document pair, we start to look for those contexts in which verb synonyms exist (at this stage, only similar verbs). A context is defined as a list of features extracted from the n (n, a parameter, determines the context size) words on the left and right sides of the verb, and mainly contains morpho-syntactic information. The features that we use are the words' lemma (since Arabic is a

highly inflected language) and part-of-speech tags, estimated automatically by AMIRA (Diab et al., 2004). Figure 3-1 shows an example for a verb context when *n*=2.

In this example, the verb *nfy*, "denied", appears in the first sentence in its imperfective dual form, while in the second sentence it is in its imperfective singular form. The lemmas are not used directly, but only to indicate equality of words located in both context parts. In such a case, the part-of-speech tag will contain the index of the matched word in the other part of the context, as can be seen in this example: On the right-hand side, the two following words (*xbrA*, *En*) are exactly the same, however on the left-hand side there are no lemma-based similar words.

One may consider adding other features. Matching content words (not functional) based on a direct WordNet hypernym relation is possible. In such cases we will match the Arabic equivalents to "blue" and "green" as both words being part of synsets with direct hyponym relation to the same synset, namely, *color*. The main challenge here is the lack of a robust and extensive WordNet for Arabic. Instead, we will use the English WordNet for the gloss entries of the words' stem.

Sentence 1: *mktb Alsnywrp wdywAn <wlmrt* ***ynfyAn*** *xbrA En lqA' fy $rm Al$yx.*
Translation: Seniora's office and Olmert's administration deny a story about a meeting in Sharm al Sheikh.

Sentence 2: *mktb Alsnywrp* ***ynfy*** *xbrA* E*n lqA}h ms&wlyn <srA}ylyyn.*
Translation: Seniora's office denies a story about a meeting with Israeli officials.

Verb: *nfy* (*ynfyAn*, *ynfy*)

Context:
Left-1: (NN, NNP) Right-1: (NN_1, IN_2)
Left-2: (NN, NNP) Right-2: (NN_1, IN_2)

FIGURE 3-1 - An example for a context.

Arabic verbs often use additional prepositions to mark their objects. Different prepositions can completely change the meaning of the verb. For instance, the meaning of

the direct-object version of the transitive verb *qDY* is "judged" and the meaning of the same verb using the preposition *ElY* to mark the object is "put an end to". Therefore, we should also use the word that appears right after the first preposition as part of the context.

Although parsing Arabic text is a difficult task, there have been recent works on dependency parsing in Arabic that may be used to locate the subject and object of each verb and then consider them as the context, instead of choosing the immediate words surrounding the verb. Arabic sentences are often written with many noun and verb modifiers and descriptors, so we think that using such parser will help produce more accurate synonyms.

Based on the ideas of Barzilay and McKeown (2001), we can identify the best contexts using the strength and frequency of each context, which is no more than using maximum-likelihood estimation. The strength of a positive context is defined as p/N and the strength of a negative context is defined as n/N, where p is the number of times the context appears in a positive example (similar verbs), n is the number of times it appears in a negative example (non-similar verbs), and N is simply the frequency of the context in the entire corpus. We then select the most frequent k positive and negative contexts (k, a parameter) that their strength is higher than a predefined threshold and use them for extracting synonymous verbs. We do this by finding all instances of each selected positive context that are not covered by a negative context in every document pair. The verbs that are surrounded by those contexts are deemed synonymous. Since we do not use word alignment of any kind, finding negative examples seems to be non-trivial. For this reason, we previously created the potential synonym-verbs list. In every document pair, we look for verb-pair candidates, which are not even synonyms based on the potential list. Such verb pairs are marked as negative examples.

To evaluate this process, we will examine a random number of the resultant synonyms with their extracted contexts.

An expert will evaluate our results by examining some number of pairs, given along with the contexts in which they were found, and decide whether the verbs are synonyms (at least in one context) or non-synonyms (wrongly identified as synonyms in all the given contexts). We want to measure the precision and relative recall, based on manually tagging paraphrase relations between all candidate pairs within a limited set of compared documents.

3.3 Experimental Results and Evaluation

We use 5,500 document pairs, extracted from Arabic Gigaword (4th ed.). The total number of words is about three million. The context window that we use is of size 2. That is, we consider all the possible contexts surrounding a verb with the limitation of one or two words before the candidate verbs and after. The strength threshold for selecting the best contexts of both categories is 0.95, as suggested by Barizlay and McKeown (2001), and the number of best contexts (defined as *k* above) we use is 20. (We found that, *k*=10, as used by Barzilay and McKeown (2001), is too restrictive in our settings, and that, therefore, the result set was relatively small).

Unique Candidates	***Unique Synonyms***	***Expert 1: Correct Synonyms***	***Expert 2: Correct Synonyms***
15,101	139	120 (86% precision)	103 (74% precision)

TABLE 3-1 – Expert's evaluation.

Two experts evaluated the resultant verb pairs. The verb pairs were given along with the different contexts in which they were found. For each candidate pair, each expert was requested to make one of the following decisions: *correct*—verb instances are exchanged in some contexts; or *incorrect*—verb instances are not exchangeable at all. There needed to be at least one context in which a verb pair is semantically replaceable in order for them to be marked as correct paraphrases by the experts. They were also allowed to say that verbs are correct paraphrases even if their meaning is modified by another word in the context. Recall that the system was instructed to identify whether two verbs have the same meaning in a given context. Therefore, it decides so even if the meaning is defined by an expression of more than one word, including the target verb. Table 3-1 shows the expert decisions.

In all, we found about 15,000 unique (based on the verbs' lemmas) candidates. Of these, the classifier decided that only 139 are synonyms. While 120 were found to be correct by the first expert, the second expert found only 103 to be correct, yielding precision values of 86% and 74%, respectively. Since we do not know how many of the 15,000 candidates are actually synonyms, we have not calculated recall. Table 3-2

shows some synonym examples that were extracted by this technique and Table 3-3 shows two of the best contexts for the positive candidates.

Synonyms
<Etql / <wqf (“arrest”)
bv~ / nq~l (“broadcast”)
<stqbl / <ltqy (“meet”)

TABLE 3-2 - Examples of extracted synonyms.

Best Contexts
Left-1: (NN_0) Right-1: (IN, NN)
Left-2: (NN_0) Right-2: (IN)
Left-1: (NN, WP_0) Right-1: (NN_0)
Left-2: (WP_1) Right-2: (NN_0)

TABLE 3-3 - Some of the best contexts for the positive candidates.

3.4 Using Synonyms in Translation

Once we found the above-mentioned synonyms, we used them in translation, under settings similar to those described in the noun experiment in the previous chapter. In this case, we tested the system only using the corpus containing about 1.2 million Arabic words and on the same test-set, taken from the 2009 NIST OpenMT Evaluation set (LDC2010T23), containing 586 sentences corresponding to 20,671 tokens (17,370 words) comparing to four reference translations. We automatically evaluated the results under BLEU and realized that there was only a slight insignificant improvement in the final results. Remember that our current example-based system is using a very simple recombination technique; therefore, we decided to examine the results of the matching step manually. Overall, we found:

- 193 input-sentence fragments for which at least one of the words matched on a synonym level. For simplicity, we call these *syn-fragments*.
- For 162 syn-fragments (83% precision), the synonym matching is correct under the sentence context.

- Out of these syn-fragments, 57 are covering parts in the input sentence that are not covered by other fragments of at least the same size. That means they might help to better cover the input sentence in the matching step, however our current recombination algorithm was not able to capture that.

We further looked at the extracted translation for the 193 syn-fragments and found that only 97 (~50%) were actually translated correctly. All the other syn-fragments received wrong translations by the system. From a first look, in most cases, the synonyms were not the main reason for the wrong translation. It seems more like the traditional problem of word alignment affecting the translation of the fragments.

Only 63 syn-fragmens participated in the final translations; out of them only 42 were translated correctly (based on our observation). Seeing these results, one can conclude that, unsurprisingly, the system is making bad choices when it tries to select the best fragments for incorporation in the final translations.

3.5 Summary

As in the previous chapter, the system we are working on has demonstrated a promising potential, although limited at this time, for using contextual verb synonyms in an example-based approach to machine translation. As demonstrated in this chapter, the classifier, which has been trained to find new verb synonyms in a corpus of comparable documents, performs pretty well in terms of precision. Even though we have not yet calculated the recall, by manually overlooking at the candidates, we could see some that there are true synonyms that were not yet discovered. In Chapters 4 and 5, we improve this technique even more, by considering more context features that model the complex morphology expressed by Arabic.

The experiment described here is just a first step toward the larger goal of deriving longer paraphrases for Arabic and using them to improve machine translation, as described in Chapters 4 and 5.

Chapter 4

Deriving Multiword Paraphrases from Comparable Documents

4 Deriving Multiword Paraphrases from Comparable Documents

In this chapter we extend our work to extract multiword paraphrases for Arabic. Paraphrases are derived from comparable documents, that is, distinct documents dealing with the same topic. As before, a co-training approach is taken, with two classifiers, one designed to model the contexts surrounding occurrences of paraphrases, and the other trained to identify significant features of the words within paraphrases. In particular, we use morpho-syntactic features calculated for both classifiers, as is to be expected when working with highly inflected languages. We provide some experimental results for Arabic, and for the simpler English, which we find to be encouraging.

There are several existing approaches for inferring paraphrases from a corpus, which differ from one another in the type of corpus they employ. Some require bilingual parallel texts (Callison-Burch et al., 2006; Zhao et el., 2008), some need monolingual parallel texts (Barzilai and McKeown, 2001), some need general monolingual texts (Marton et al., 2009) and others need corpora of comparable documents (Rui and Callison-Burch, 2011; Dolan et al., 2004). Bilingual parallel texts, pairing Arabic with languages other than English, are very hard to obtain.

To the best of our knowledge, ours is the first work on paraphrasing in Arabic. Salloum and Habash (2011) developed a rule-based algorithm for generating Modern Standard Arabic (MSA) paraphrases for dialectical Arabic phrases given to a statistics-based automatic translation system. They focused only on input phrases that do not exist in the translation table used by the translation system, for the purpose of improving its coverage. The MSA paraphrases were generated mostly using different morphological variations of the input words. They reported a slight improvement in BLEU score (Papineni, 2002) over a baseline system that did not use their generated paraphrases. In another work, by Denkowski et al. (2010), 726 Arabic paraphrases were manually generated and confirmed using Amazon's Mechanical Turk, from the NIST OpenMT 2002 development set (Garofolo, 2002). That was mainly done with the purpose of improving the evaluation of an English-to-Arabic machine translation system.

In this chapter, we continue with the co-training approach. We repeat the process of creating a corpus of comparable documents and use it as a resource for paraphrasing. Considering that Arabic is a morphologically rich language, we incorporate morphological features of the surrounding words as well as the paraphrase patterns themselves.

We continue as follows: Sections 4.1 discuss the corpus preparation; In Section 4.2 we elaborate on our proposal, followed with some experimental results reported in Section 4.3. Conclusions are given in the last section.

4.1 Preparing the Corpus

As before, our approach is inspired by the work of Barzilay and McKeown (2001) on finding paraphrases in different English translations of the same source text. Here we improve the process of building a corpus of comparable documents, extracted from Arabic Gigaword (4th ed.). Pairing documents, based on their topic, was done automatically using cosine similarity over the lemma-frequency vector of every document, with the lemma of every word extracted using MADA 3.1 (Habash and Rambow, 2005; Roth et al., 2008). More formally, given two documents P and Q, we build their lemma-frequency vectors V_P and V_Q respectively. The dimension of those vectors equals to the size of the entire vocabulary, and each index i contains the frequency of the specific lemma-i from the vocabulary, as occurring in the specific document. Typically, those lemma-frequency vectors are quite sparse. The similarity of P and Q is then calculated by the following formula:

$$CosSimilarity(P,Q) = \frac{V_P \cdot V_Q}{\|V_P\|\|V_Q\|} = \frac{\sum_{i=1}^{n} V_P[i] \times V_Q[i]}{\sqrt{\sum_i^n V_P[i]^2} \times \sqrt{\sum_i^n V_Q[i]^2}} \quad (4.1)$$

This formula measures the cosine value of the angle between the two lemma-frequency vectors. The resulting value ranges between 0 and 1, with 0 indicates completely different, and 1 indicates identical (the angle in between, equals to zero). We use lemmas rather than words, in order to handle the Arabic rich morphology.

We considered candidates for document pairs only when they were published by different news agencies on the same day. For every document published by one agency, we pair it with a document from the agency that maximizes the similarity score over all the other documents published by the same agency on the same day. Not only that, we require that the score be higher than a predefined threshold. We also tried using lower thresholds for which we retrieved additional pairs; however, precision decreased linearly. It is obvious, then, that this approach prefers precision to recall; in other words, we probably miss a large number of potential candidates, while the candidates that we do extract are likely correct.

All together, we created 690 document pairs, comprising about half a million words. Our corpus of comparable documents was manually evaluated by two Arabic speakers. We randomly selected 120 document pairs out of the 690 and, for each, asked the eval-

uators for a simple "yes" or "no" answer to the question, "Do both documents discuss the same event?" The results are encouraging: out of the 120 pairs, 100 were classified as correct by both evaluators. Of the other 20 instances, 5 were classified "yes" by one evaluator. The rest of the pairs actually dealt with the same general domain but were not specifically discussing the same event. This positive evaluation allowed us to use this corpus in the next step of our inference technique.

Every document was pre-processed with AMIRA 2.0 (Diab et al., 2004, 2007) before being given to the inference classifier, described in the next section. Recall, AMIRA 2.0, is a tool for finding the context-sensitive morpho-syntactic information. For every word, it is capable of identifying the clitics, lemma, stem, full part-of-speech tag (excluding case and mood), base-phrase chunks and named-entity-recognition tags. The corpus is obviously not annotated with paraphrasing-related information and there is no alignment indication included at any level.

4.2 Inference Technique

To infer new paraphrases from the corpus, we follow the "co-training" technique, training two different classifiers: one for modeling the context of a potential paraphrase and another for modeling the features of the paraphrase pattern itself. The main idea of the co-training approach applied to unlabeled data is to use the two classifiers on different views of the data. In our case, the two views are the context (CX) and the pattern (PT), with one classifier labeling the most reliable unlabeled data items for training the second classifier. Then, the second classifier can label some of the data items for training the first one. This process is repeated several times, and the labeled data collected during the entire run is returned. The algorithm runs in iterations; each iteration increases the number of words a potential paraphrase may contain, that is, in the first iteration only single-word paraphrases are allowed to be found, in the second one, paraphrases composed of up to two words are allowed, and so on. The input of the algorithm is the pairs of documents that we found on the previous section, from which we extract pairs of phrases.

A *pair of phrases* is composed of two phrases, one from each of a pair of comparable documents. Since alignment at any level does not exist for comparable documents, we consider all the possible pairs of phrases, given one pair of documents. To avoid too much noise, we restrict a phrase for consideration using the following two heuristics: (1) a phrase has to be composed of at least one non-function word; and (2) the phrase does not break a base-phrase in the middle, similar to (Wang and Callison-Burch, 2011).

Function words, in our case, are identified based on their part-of-speech and base-phrase tags, as provided by AMIRA 2.0. Otherwise, a huge number of pairs containing only function words, not too important for paraphrasing, would be considered. The number of iterations, and concomitantly, the maximum length of the output phrases, is a parameter we control. As implied before, we start with single words and increase this parameter with every iteration. During the entire run of the algorithm, we maintain two sets of pairs of phrases:

Labeled Containing pairs of phrases with their label, "true" to indicate paraphrases and "false" to indicate that the phrases are not paraphrases of each other. This set starts off empty.

Unlabeled Containing pairs of phrases that are still waiting for their label assignment by the algorithm.

In every iteration, the algorithm performs the following steps:

1. Deterministic labeling of potential paraphrases;
2. training the CX classifier using the labeled set as training data;
3. running CX on unlabeled pairs and labeling the most reliable ones;
4. training the PT classifier using the labeled set as training data;
5. running the PT classifier on the labeled set;
6. labeling some unlabeled pairs, based on the labels provided by both classifiers.

Figure 4-1 illustrates this process. We now describe these steps in greater detail. It is difficult to estimate in advance the weight of the selected features and their effect on the predictions of the classifiers; therefore, we chose to use support-vector-machine classifiers (Vapnik and Cortes, 1995) because of their good generalization property. Technically, the classifiers are trained on the WEKA platform (Hall et al., 2009) running with the LibSVM library (Chang and Lin, 2011). One drawback of using support-vector machine in this kind of setting is the long running time of the training algorithm. Because we are running the trainer twice during every iteration, this drawback becomes even more pronounced.

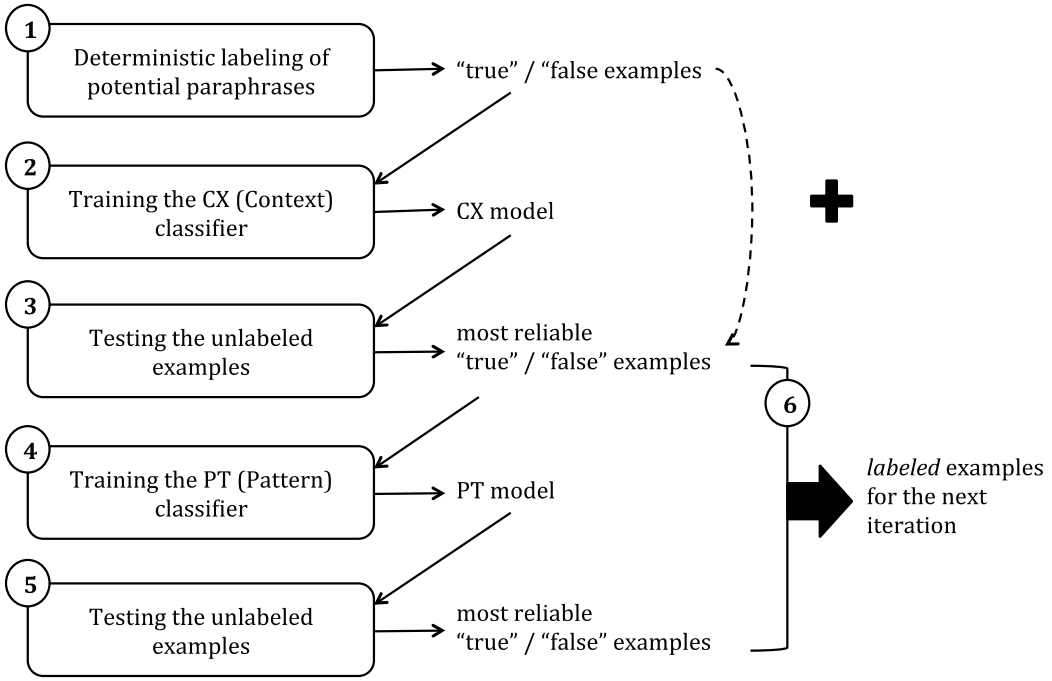


FIGURE 4-1 - An overview of the paraphrasing co-training algorithm.

The labeled pairs are used as training data for both classifiers, with every pair formatted as a feature vector. The features for the CX classifier capture some morpho-syntactic information expressed by the window-based context words. In the current experiment, we use a window of size three, that is, three words before each word sequence from the pair, and three words afterward. That gives us twelve words from which we extract features for representing a single pair and that number does not change during the entire learning process. Following the same example from Figure 3-1, the context of the two sentences for this experiment is shown in Figure 4-2. In this case, the emphasized texts are the actual paraphrases while the surrounding words are composing the context, which is described in the last part. In this example, the paraphrase pair is composed of a single identical lemma, inflected differently for person. The context of a paraphrase pair is composed of four parts: left and right words of each of the paired texts.

Sentence 1: *mktb Alsnywrp wdywAn <wlmrt **ynfyAn** xbrA En lqA' fy $rm Al$yx.*
Translation: Seniora's office and Olmert's administration **deny** a story about a meeting in Sharm al Sheikh.

Sentence 2: *mktb Alsnywrp **ynfy** xbrA En lqA}h ms&wlyn <srA}ylyyn.*
Translation: Seniora's office **denies** a story about a meeting with Israeli officials.

Phrases: *ynfyAn*, *ynfy*

Context:
Sentence 1: *Alsnywrp wdywAn >wlmrt* [...] *xbrA En lqA'*
Sentence 2: *mktb Alsnywrp* [...] *xbrA En lqA}h*

FIGURE 4-2 - An example for a context.

The PT classifier makes its predictions based on the phrases themselves; their size (number of words) varies as the iteration number increases. For both classifiers, we use a quadratic kernel for capturing the common effect of all the features on prediction.

Tables 4-1 and 4-2 summarize the features we currently use for building the feature vectors for the CX and PT classifiers, respectively. NER tags are assigned to persons, organizations, geo-political organizations and locations. The gloss-match rate is calculated for both sides of the context. In the example of Figure 4-2, there is no word that matches on the left side (note that proper nouns usually do not have glosses). However, on the right side *xbrA En lqA'*, "a story about a meeting", matches *xbrA En lqA}h*, "a story about his meeting", with all three words on the gloss level; therefore, the left gloss-match rate is 0 and the right one is 1. The same calculation works with lemma-match rates on the lemma level. Using the gloss and lemma match rates enables us to consider a level of similarity of the context words, surrounding a potential paraphrase pair. We intend to improve this simple technique using alternative ways to model the semantic distance of the two contexts, for example, using cosine similarity score calculated on the two context vectors containing left and right words and glosses.

To model the Arabic rich morphology, we use some morphological features calculated on each phrase word individually for the PT classifier. They are all Boolean values indicating whether the word expresses the feature or not. For example, the word *wbktAbh*, "and in his book", expresses conjunction, preposition and possessive. When working with Arabic, a highly inflected language, morphological features may contribute to the classification performance. Take for example the Arabic pattern *qAm b-*, literally, "did something". When it followed by some verbal nouns, it gets the meaning of the corresponding verb, as in *qAm bzyArp*, "he visited". Therefore, accounting the verb *qAm* with the preposition proclitic *b* as features for paraphrasing seems like a good idea.

Feature	***Description***
Lemma, POS, NER, BP	of each context word
Gloss-match rate	the rate of gloss match on each side of the context (left and right)
Lemma-match rate	the rate of lemma match on each side of the context

TABLE 4-1 - The features we use for training the CX classifier on Arabic.

Feature	***Description***
n-gram score	normalized *n*-gram frequency score for word sequences up to 4 words (2-4 grams)
POS, NER, BP	of each sequence word
Boolean morphological features (exists/does not exist): Conjunction, Possessive, Determiner and Prepositions	of each sequence word
Sequence length	the number of words in each sequence

TABLE 4-2 - The features we use for training the PT classifier on Arabic.

The *n*-gram score is a log-probability language-model score for capturing the co-occurrence of the candidate sequence words, calculated using SRILM (Stolcke, 2002).

The first time one of the classifiers is trained, it needs some labeled items. With "co-training", those items are usually provided by manual annotation of a relatively small fraction of the data or, in this case, by using an automatic deterministic annotation algorithm. Therefore, in the first step of every iteration, the algorithm enriches the labeled set with additional "true" labeled pairs following a deterministic approach. Since it is very difficult to obtain a word or sentence-level alignment of two given comparable documents, our algorithm simply adds all the pairs whose phrases match on the lemma level, word by word. If the lemma does not exist, we use the word's surface form for matching. We use this deterministic approach before starting every iteration. The lengths of the phrases are determined by the iteration number, so in the first iteration only phrases of size 1, that is, single words, are added, in the second iteration phrases of size 2 are added, and so forth. Such a pair of single words, matched on the lemma level, is shown in Figure 4-2. The same example is used in Figure 4-3 to demonstrate a pair of longer phrases, of size 4, that match on the lemma level word by word.

Note that paraphrases work on the sense level, rather than on the surface form; however, our assumption is that, because we are using phrases from comparable documents, their senses may be the same with a reasonable high probability. Note that, since we are using the context-sensitive lemmas for matching, one can think of that as matching words on the sense level. However, AMIRA 2.0 was trained mostly with morpho-syntactic features and therefore achieves good performance in identifying the common lemma of a context-sensitive part-of-speech tag for every word. When a word may have two or more different lemmas for the same part-of-speech tag that have different senses, AMIRA 2.0 does not perform as well. For example, the word *>mAnp* has three different noun lemmas: *>amAnap_1* ("faithfulness"), *>amAnap_2* ("secretariat") and *>amAnap_3* ("deposit").

That approach leaves us with some deterministically selected positive examples; however, it does not provide us with the necessary negative examples. In the first iteration, we consider phrases of size 1 only. Our assumption is that word pairs sharing some of their senses in common may be considered paraphrases, thus cannot be naturally selected as negative examples. Currently we use the English gloss of every word, as provided by AMIRA 2.0, to select word pairs with different gloss values as negative examples. Therefore, under this condition, the Arabic word pair *mjAl* and *mnTqp* is not considered as a negative example because they share the same gloss value: "area". An alternative approach, which we plan to take in the future, would be using Arabic WordNet. It implies that, in our first iteration, only word pairs that have the same English

gloss and not the same Arabic lemma are put in the unlabeled set. That dramatically reduces the amount of paraphrases of size one, better referred to as synonyms, that we can find. Since we are more interested in longer paraphrases, we can live with this limitation.

Sentence 1: *mktb Alsnywrp wdywAn <wlmrt **ynfyAn xbrA En lqA'** fy $rm Al$yx.*
Translation: Seniora's office and Olmert's administration **deny a story about a meeting** in Sharm al Sheikh.

Sentence 2: *mktb Alsnywrp **ynfy xbrA En lqA}h** ms&wlyn <srA}ylyyn.*
Translation: Seniora's office **denies a story about a meeting** with Israeli officials.

FIGURE 4-3 - An example of similar phrases as marked by the deterministic algorithm.

In subsequent iterations, negative examples are assigned automatically by the classifiers of the previous one, in the following way: after training the CX classifier in step 2, we use the classifier to tag the unlabeled pairs in step 3. Some pairs are labeled as positive and some as negative. Those for which the classifier has a "good sense" are added to the labeled set with their corresponding label. "Good sense" is measured with a confidence score that is provided by LibSVM along with every tested pair. Since this score is based on margin length calculations and it is basically representing the distance of the pair from the separating hyperplane, one should use it carefully. Considering other approaches for estimating the confidence of the classifier, such as (Wu et al., 2004), is a good plan for the future. Currently, we only set some threshold values for adding pairs to the labeled set, with a high score, empirically determined. The unlabeled set is also updated with additional examples of length not exceeding the iteration number. In that sense, the iteration number is actually an upper bound on the length of the examples, allowing the algorithm to select phrases of a lower length paired with longer phrases. For example, in the second iteration, the unlabeled set also contains examples that pair a phrase of one word with a phrase of two words.

The labeled set after step 3 contains positive as well as negative pairs, added by both the deterministic algorithm and the CT classifier, for training the PT classifier.

Steps 4 and 5 train and test the classifier PT on the labeled and unlabeled sets respectively. Finally, in step 6, pairs that receive the same label from both classifiers, with a confidence score higher than the predefined threshold, are added to the labeled set with their corresponding label and stay there forever. This labeled set is used as part of the training data in the next iteration. The number of iterations is manually configured upon initialization of the algorithm and at the end, the positive pairs are deemed paraphrases. The entire process is summarized in Figure 4-1.

Feature	***Description***
Lemma, POS, NER, BP	of each context word
Lemma-match rate	the rate of lemma match on each side of the context

TABLE 4-3 - The features we use for training the CX classifier on English.

Feature	***Description***
n-gram score	normalized *n*-gram frequency score for word sequences up to 4 words (2-4 grams)
POS, NER, BP	of each sequence word
Possessive form	of each sequence word
Sequence length	the number of words in each sequence

TABLE 4-4 - The features we use for training the PT classifier on English.

To get a feeling for the robustness of the methodology, we applied the same technique to the task of generating paraphrases in English. English has shallow morphology as compared with Arabic, on one hand, but, on the other hand, uses more words than Arabic to convey the same meaning. Based on this observation, for English, we changed the settings of the data for using a window of size 4 instead of 3 and removed most of the morphology-related features. The English set of features for the CX and PT classifiers are summarized in Tables 4-3 and 4-4, respectively.

Comparable documents were extracted using the same technique from a relatively small part of English Gigaword (5th ed.) (Parker et al., 2011b). We preprocessed the

documents using the OpenNLP[6] library. For every word, we determined its part-of-speech, base-phrase and named-entity tags. The lemma of each word was retrieved from WordNet by providing it with the surface form and the part-of-speech tag as inferred by OpenNLP. Overall, we found 294 document pairs, containing about 220,000 words. Similar to the evaluation step of the Arabic corpus, we randomly selected 80 document pairs for vetting their correspondence to each other. Out of the selected 80 document pairs, 65 were classified as "yes" instances by both evaluators. Of the other 15 instances, 3 were classified as "yes" by only one evaluator. As for Arabic, the rest of the pairs were actually dealing with the same general domain but not specifically discussing the same event. The inference algorithm for English worked exactly as described above. Recall that in the first iteration on Arabic, we used a deterministic algorithm for labeling some of the data for training the classifiers for the first time. For Arabic, we used the English gloss values of the Arabic words for finding "false" examples; for English, we use WordNet for the same task in such a way that synonyms are not considered as "false" examples.

4.3 Experimental Results

4.3.1 Experimental Approach

Our initial experiments perform only five iterations on both corpora (Arabic, as well as English), which means that we find paraphrases of no longer than five words. The two classifiers are configured with different thresholds. The confidence score given by LibSVM for every classification is a value between 0 and 1; therefore, we experimented with different threshold values and realized that the best settings in this case are obtained when using 0.85 for positive pairs for the CX classifier and 0.75 for the PT classifier. For the negative pairs, we use 0.75 for both classifiers. Since we noticed that the number of negative pairs is much larger than the number of positive ones in the training data of every iteration, we defined another parameter (currently 6) that limits the factor of negative pairs allowed in the training data with respect to the positive pairs. This parameter helps us to configure our algorithm to prefer precision over recall.

In the next section, we show some results when running over 240 document pairs in Arabic, containing about 165,000 words, and 40 English document pairs containing about 11,000 words.

[6] http://opennlp.apache.org

4.3.2 Results

First, we give some statistics on the results obtained by the inference algorithm on both the Arabic and English corpora, in Tables 4-5 and 4-6, respectively.

	"false" pairs	*"true" pairs*	*Unique paraphrase pairs*	*Unlabeled pairs*
Initialization	22,885,104	66,317		19,480
After iteration 1	23,799,787	(+1,726) 68,043		3,166,935
After iteration 2	24,759,791	(+3,757) 71,800	954	2,790,574
After iteration 3	25,349,489	(+2,623) 74,423	416	2,198,253
After iteration 4	26,221,889	(+451) 74,874	331	1,557,931
After iteration 5	26,900,833	(+101) 74,975	72	878,987
Total			**1,773**	

TABLE 4-5 - Statistics and final results of the inference algorithm running on the Arabic corpus.

In both tables, the initialization row shows the number of positive and negative examples as was labeled by the deterministic algorithm and the size of the unlabeled examples set. In the following rows, the numbers refer to the results of the specific iteration. The numbers of positive and negative pairs reported on every line are the aggregated numbers collected from all previous iterations. Recall that at the beginning of every iteration, a deterministic algorithm adds pairs of phrases that match on the lemma level, word by word; hence, the number of positive pairs in every line is the sum of the pairs from the previous iterations, the pairs added by the deterministic algorithm for the next iteration and the paraphrase pairs inferred by the current iteration. The third column, unique paraphrase pairs, is merely the number of unique paraphrase pairs inferred during the current iteration. The parenthesized numbers indicate the difference in the quantity of positive pairs from the previous iteration. So, the total number of extracted paraphrases is the number written on the total line in the unique paraphrases column.

In Arabic, we found 1,773 paraphrase pairs and in English we found 525. This process can be scaled up for finding more paraphrases.

We do not include paraphrases generated after the first iteration because, by definition, they are composed of synonymous words. Recall that, during initialization, the deterministic algorithm adds pairs to the unlabeled set if their paired words are synonyms in English or share the same English gloss, in Arabic. Table 4-7 shows some statistics for the entire inference process.

	"false" pairs	***"true" pairs***	***Unique paraphrase pairs***	***Unlabeled pairs***
Initialization	876,947	32,972		3,597
After iteration 1	960,840	(+868) 33,840		86,648
After iteration 2	1,058,970	(+1,633) 35,473	230	58,312
After iteration 3	1,109, 746	(+1,194) 36,667	177	21,332
After iteration 4	1,127,643	(+339) 37,006	94	6,677
After iteration 5	1,128,475	(+52) 37,058	24	1,490
Total			**525**	

TABLE 4-6 - Statistics and final results of the inference algorithm running on the English corpus.

The raw data corpus size is a rough estimation of the amount of words we had in the corpus at the beginning. Note that currently we did not use the entire Gigaword corpora: in Arabic we used about 30% of the entire set and in English we only used about 10% of the documents. The following column shows the number of comparable document pairs we found using the pairing algorithm described above. Since the pairing algorithm was designed to prefer recall over precision, the number of comparable documents is lower than might be expected considering the relatively large number of words we had in the raw corpus. We expect that this number will grow larger once we improve the pairing algorithm. The next column, number of words used in inference, sums up the number of words of the entire set of comparable document pairs from the previous column. The last column shows the number of paraphrase pairs extracted by the inference algorithm.

Comparing the results to the results retrieved by other works is difficult because there is neither a shared task for paraphrase extraction nor common resources for comparison. Therefore, we show some manual evaluations of our results. The evaluation was performed by two Arabic/English speakers by going over the reported

paraphrases one by one. For each pair, we assigned one label: *P* – indicating correct paraphrase, *E* – indicating unidirectional entailment, *R* – related (for other semantic relations except antonyms, e.g. San Diego/Los Angeles) and *F* – wrong (including antonyms). Table 4-8 and 4-9 summarizes our preliminary evaluation report on Arabic and English, respectively.

	Raw data corpus size	***Extracted comparable document pairs***	***Comparable documents used in inference***	***Number of words used in inference***	***Number of inferred unique paraphrases***
Arabic	~20,000,000	690	240	165,369	1,773
English	~1,000,000	294	40	11,600	525

TABLE 4-7 - General statistics on the entire inference process.

Length	***Evaluated***	**P**	**E**	**R**	**F**	***Precision***
2	120	49	12	25	34	71%
3	95	45	10	11	31	69%
4	70	26	4	5	35	50%
5	50	24	2	7	20	66%
Total	335	144	28	48	120	**66%**

TABLE 4-8 - Manual evaluation summary for Arabic. *P*: paraphrases, *E*: unidirectional entailment, *R*: related, *F*: wrong, i.e. unrelated or antonyms.

Length	***Evaluated***	**P**	**E**	**R**	**F**	***Precision***
2	120	23	11	37	49	59%
3	60	28	6	9	17	71%
4	50	15	8	8	21	62%
5	25	8	5	2	10	60%
Total	255	74	30	56	97	**63%**

TABLE 4-9 - Manual evaluation results for English. *P*: paraphrases, *E*: unidirectional entailment, *R*: related, *F*: wrong, i.e. unrelated or antonyms.

The evaluation results reported in both tables are based on the agreement of the two evaluators; in other words, we report here only on pairs that were annotated by both evaluators with the same tag. Note that the first column, length, indicates the number of words of the largest phrase included in the evaluated paraphrase pair. Paraphrase pairs containing a single word in both phrases were not evaluated at all. In the last column, we calculate the precision, considering pairs tagged with *P*, *E* and *R* as positive instances. The last row summarizes the results. In Arabic, 66% of the generated paraphrase pairs are at least considered as semantically related; among them, about 43% are considered real paraphrases. In English, only 63% of the paraphrase pairs are considered related, out of which 30% are real paraphrases. As can be seen from the tables, there is no preferred length for the inference algorithm. We see a slight improvement in the precision of paraphrases up to length three; however, this improvement does not seem significant, considering the relatively small amount of evaluated pairs.

When we increase the threshold on the confidence that is used by the PT classifier on English to 0.9, the number of paraphrases reported by the inference algorithm decreases to 330 and the average number of similar words in a pair, increases. As a results of that, the overall precision is improved to 72%, calculated over 250 evaluated pairs. These results help us understand the effect of the PT classifier on performance. The pairs with a high confidence score, as reported by the PT classifier, are most likely to be real paraphrases; however, in most cases, the phrases of such a pair, share more words in common than do other pairs (e.g. “the U.S. Air Forces” ⇔ “the United States Air Force”).

In order to measure the effect of the PT classifier and the contribution of the morphological features to the overall performance, we run an additional smaller experiment. We test the algorithm running on the same corpus of 40 Arabic document pairs in three different conditions:

1. Using both classifiers PT and CX with all features;
2. using only the CX classifier;
3. using both classifiers PT and CX, without morphological features.

Experiment	*Extracted pairs*	*Precision*
CX + PT	653	68%
CX only	7,405	23%
CX + PT without morphological features	211	62%

TABLE 4-10 – Testing the contribution of the PT classifier and the morphological features.

Table 4-10 summarizes the results. We can see that when we put the PT classifier off we get a larger number of paraphrases, but their precision is relatively low. However, when we use the PT classifier but remove all the morphological features, we get much less paraphrases, reported with a reasonable range of precision. We learn from this experiment that the morphological features of the paraphrase candidates are important for this task. Investigating the contribution of each morphological feature individually is another direction for future investigation.

Some examples for Arabic as well as English pairs that were inferred by our co-training algorithm are mentioned in Appendix A.

4.4 Summary

The method suggested here has demonstrated its potential for inferring paraphrases from a corpus of comparable documents, using co-training. As we have seen, incorporating morphological features for a highly inflected language, such as Arabic, is very effective. SVM with its generalization property was a natural option for dealing with combinations of features that can play an important role for identifying paraphrases. Finding more features that help to match the true senses of phrases properly is a direction for future investigation. In a similar experiment performed on English, we also obtained encouraging results, despite the smaller corpus. In the next chapter we use paraphrases within an Arabic-to-English translation system to improve the quality of the translations.

Chapter 5

Translating with Paraphrases

5 Translating with Paraphrases

In this chapter we use a paraphrasing technique similar to the one introduced in the previous chapter, to improve a phrase-based statistical translation system. We modify the paraphrasing algorithm described in the previous chapter to meet the requirements of the translation process, and to produce a large number of paraphrases in a reasonable amount of time. Here, we use comparable documents only for modeling the context of potential paraphrase pairs, using similar features to the ones used in the previous chapter. Then, given a phrase for paraphrasing, we apply that classifier on a large monolingual corpus for finding equivalent phrases.

We employ our paraphrasing technique to improve a *Moses* (Koehn et al., 2007) implementation of an Arabic-to-English phrase-based statistical translation system, and evaluate the results vis-à-vis a similar system that does not use paraphrases in translation. The paraphrases are given to the translation system organized in a *word lattice* (Dyer et al., 2008), a directed acyclic graph that captures different input variants, each assigned with its prior weight or probability.

We experiment with two levels of paraphrases: nominal and verbal synonyms, as extracted using the techniques described in Chapters 2 and 3 respectively, and full multiword paraphrases. The results are encouraging: our best system shows an increase of 1.73 in BLEU.

We continue as follows: in Section 5.1 we describe how we derive paraphrases at a large scale, a technique that we use to embed paraphrases in the translation process, as discussed in details in Section 5.2. Section 5.3 describes our experimental approach, followed by the results provided in Section 5.4. Finally, we conclude in Section 5.5.

5.1 Scaling up our Paraphrasing Approach

Our previous approach for paraphrasing focused on learning new paraphrases from comparable documents. Overall, we identified only a few thousands of paraphrase couples, using a relatively small amount of comparable documents. In order to use this technique for improving machine translation, we had to scale up its ability to produce a larger number of paraphrases, so that we can get a better coverage of the input text given to the translation system. Increasing the amount of comparable documents given to the paraphrasing algorithm, assuming it will increase the resulting paraphrases num-

ber, introduces performance complexity issues, which makes this option almost infeasible. In particular, our algorithm is based on two support-vector-machine classifiers, running in tandem in every iteration. Increasing the number of comparable documents, introduces large number of potential candidates, which are then used as training examples for the classifiers. Finding no clever way of pruning some of them (except for random pruning), we ended up with large training sets, which slowed down the overall execution. Moreover, extracting a large number of comparable documents while keeping a reasonable similarity rate was found to be challenging. In other words, changing the thresholds of the extraction algorithm, described in Section 4.1, so that it will find more document pairs, was found to be error-prone.

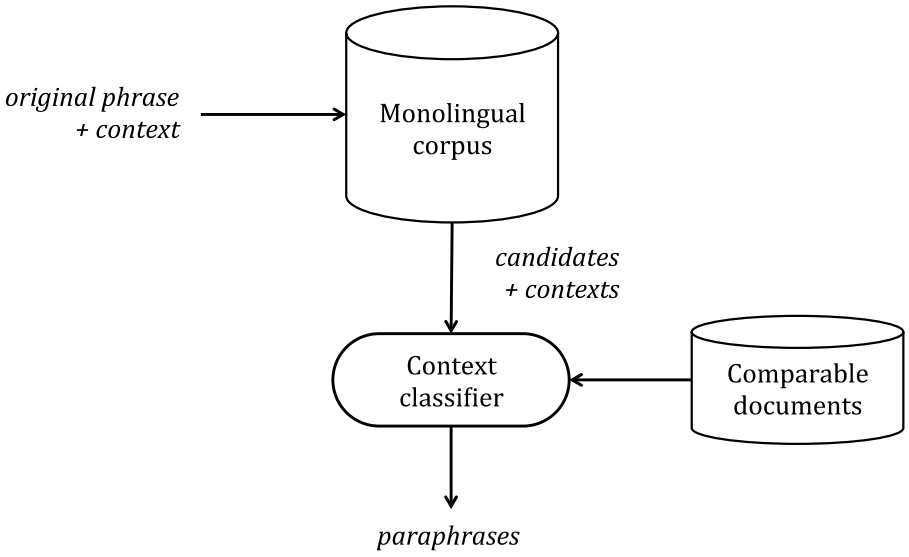


FIGURE 5-1 – Paraphrasing process, in the context of machine translation.

In this section, we simplify our paraphrasing algorithm to work at a larger scale. In order to do that, we abandon the co-training approach and use the small set of comparable documents only to model the context of potential paraphrases. In other words, we measure the distributional similarity of phrases using only a classifier applied on the context of the phrases. The context, like before, is defined to be the surrounding words of the two candidate phrases. Unlike the previous algorithm, here, we use a broader context and some additional features. Broadly, our algorithm works as follows: (1) we train a classifier that is capable of distinguishing between paraphrases and non-paraphrases, based on their context, using a relatively small set of comparable docu-

ments; (2) given a phrase for paraphrasing, we look for all potential phrases in a large monolingual corpus, based on co-occurring lemmas; and (3) we use the context classifier to remove non-paraphrases. Figure 5-1 illustrates this process. We now elaborate further on each step individually.

5.1.1 Training a Context Classifier

As described in the previous chapter, we used the set of 690 pairs of comparable documents to extract some phrase pairs, which are then used as examples for training our context classifier. In this case, we do not intend to extract paraphrases from this step, but only some labeled examples, positive (pairs of paraphrases) as well as negative. Finding labeled examples is done similar to our deterministic labeling from Section 4.2. Namely, given one pair of comparable documents, we begin by extracting all phrases (i.e., word sequences) of up to *N* words (here, *N* is a variable; *N*=6 for the rest of this chapter) from each document. Then, we pair each phrase from one document with all the phrases from the other document, resulting in a relatively large set of phrase pairs. Among all those pairs, we keep only the ones that we can tag either as positive or negative, so that we can use them as training material.

5.1.1.1 Obtaining Training Examples

A positive pair must comply with the following rules:

1. Both phrases do not break a base phrase in the middle;
2. both phrases contain at least one content word (non-functional, determined using the part-of-speech tag);
3. both phrases match on the lemma level, word by word.

The main reason for the first rule is to increase the chance that both phrases are grammatical.

Unsurprisingly, the number of positive pairs decreases as the length of the phrases increases; in our experiments, we intend to keep their numbers balanced. Like before, since we work with words rather than word senses, similar phrases do not always have the same meaning, given their context. However, the fact that the phrases are taken from comparable documents provides support in the determination that they share similar senses.

For negative examples, we select pairs of phrases that do not comply with the last rule, and also not with one of the two others. This gives us enough confidence to believe that such phrase pairs are not composed of paraphrases. Some positive examples are provided in Table 5-1. The first example demonstrates the matching of two similar phrases, inflected differently for number. In the second example, the same phrase is matched, with each instance using a different proclitic. The third example is the trivial case of exact match.

wqAl AlmSdr ↔ *wqAlt AlmSAdr*
"and the source said" ↔ *"and the sources said"*
bt$kyl Hkwmp ↔ *wt$kyl Hkwmp*
"with establishment of a government" ↔ "and establishment of a government"
wzyr AlxArjyp ↔ *wzyr AlxArjyp*
"the minister for foreign affairs"

TABLE 5-1 – Examples of positive pairs.

Working on the lemma level, we include among paraphrases other pairs of phrases that express the same meaning, regardless of their inflection for number, gender, and person. (Recall our modified definition in he introductory chapter.) The motivation is that such pairs often have similar English renderings.

5.1.1.2 Feature Extraction

Unlike our previous algorithm, here we do not consider features calculated on the actual phrase words. Generally speaking, the main purpose of our extracted features is to model the similarity between the window-based context words of potential paraphrases; in the reported experiments, the size of the window varies depending on each feature.

For each phrase, we build a lemma-frequency vector, as explained in Section 4.1, representing the lemma of the contextual words. In this case we consider 8 words to the left and right of the phrase. Each lemma is further multiplied by the *inverse document*

frequency to reduce the effect of functional words that typically occur more frequently, resulting in the standard *tf-idf* score for every contextual lemma. A document, in this sense, is defined to be a context window of sixteen words.

Looking at a single phrase pair, we have two *tf-idf* vectors, one for each phrase's context. We measure their similarity using a cosine-similarity score (defined in Equation 4.1 in the previous chapter). Furthermore, since we are using relatively sparse vectors, each containing merely sixteen non-zero values at most, we want to increase the effect of context lemmas that co-occur with their corresponding phrase more often than by chance. In other words, we wish to allocate more weight for lemmas that have a better chance to represent their phrase.

Lin (1998) investigated some similarity metrics of two events, focusing on the semantic similarity of words that co-occur in a large corpus. Following his ideas, we use Pointwise Mutual Information (PMI) for every contextual lemma as a tool for measuring the amount of information it shares with the surrounded phrase.

Generally speaking, let *x* and *y* be two specific events of the random variables *X* and *Y*, respectively; their PMI score is defined as follows:

$$\mathrm{PMI}(x,y) = \log\frac{p(x,y)}{p(x)p(y)} \tag{5.1}$$

where $p(x,y)$, $p(x)$, and $p(y)$ are probabilities calculated based on the distributions of *X* and *Y*. Intuitively, PMI measures the level of independence of *x* and *y*, given their distributions. When they are completely independent, then $p(x,y) = p(x)p(y)$ by definition, hence $PMI(x,y) = 0$; on the other hand, when they perfectly associated we have $p(x,y) = p(x) = p(y)$, hence $PMI(x,y) = -\log p(x)$. The latter value is also known as the *self-information*, or *entropy* of *x*, denoted as $h(x)$, that is, the amount of uncertainty of *x*. In fact, the PMI formula can be mathematically rephrased as:

$$\mathrm{PMI}(x,y) = \log\frac{p(x,y)}{p(x)p(y)} = h(x) + h(y) - h(x,y)$$

Using this version, one may get more intuition about PMI's definition: it is the amount of individual uncertainty of *x* and *y* left after removing the amount of uncertainty of *x* and *y* occurring together. If nothing is left, it means that there is no uncertainty in the event of *x* and *y* occurring individually; in other words, they are completely independent. Marton et al. (2009), working on paraphrasing, and following McDonald (2000), were using Dunning's (1993) Log-Likelihood Rate (LLR) in a similar situation. To calculate the LLR of two specific events *x* and *y*, one needs to multiply their PMI with the

total number of occurrences, namely the aggregated number of occurrences of *x* and *y*. Intuitively, it helps to handle situations of high values of PMI when *x* and *y* occurs rarely in the data, but mostly together. This topic is widely investigated and some aspects remain controversial (Pedersen et al., 1996; Agresti, 1990, p. 246).

Following Equation 5.1, we calculate the PMI of every lemma l occurring in the context of a phrase *P*, in the following way:

$$\text{PMI}(l, P) = \log \frac{p(l, P)}{p(l)p(P)}$$

where $p(l, P)$, $p(l)$, and $p(P)$, are calculated using a simple maximum-likelihood estimation, that is:

$$p(l) = \frac{C(l)}{N_L}$$

$$p(P) = \frac{C(P)}{N_{Ph}}$$

$$p(l, P) = \frac{C(l \text{ occurs in the context of } P)}{N_L}$$

where *C* is the *count* function, returning the number of occurrences of the argument in the a corpus of monolingual documents. The amount N_L is the number of words (or lemmas) in the entire corpus, and N_{Ph} is the number of phrases that exist in the entire corpus.

Finally, the *tf-idf* value of each lemma is multiplied by the PMI. Every phrase *P* is then represented by the following vector V_P:

$$V_P = \{< l, \text{tf-idf}(l, P) \times \text{PMI}(l, P) > | l \in context(L)\} \quad (5.2)$$

Note that in our case, $\text{tf-idf}(l, P) > 0$ only if l occurs within the context of *P*.

Contextual words that are either not derived from a lemma, or the morphological analyser was failed to find it, are considered with their surface form. This situation happens mostly (but not only) with named entities; hence each named entity occurring in the context of *P* is replaced by a placeholder lemma representing the entity type (e.g. person, organization, location). The named entities are found by AMIRA 2.0, as described in the previous chapter.

Table 5-2 describes the features used for modeling the contexts of a given phrase pair (P_1, P_2). In addition to cosine similarity, we use the part-of-speech and base-phrase tags of 6 words to the left and right of each phrase, by keeping the order of the words,

as they appear, similarly to our previous algorithm. For part-of-speech tags, currently we use a reduced tag set (RTS), distributed with the Arabic Treebank (ATB) (Maamouri et al., 2004). Alternatively, we could use the enriched RTS (ERTS) (Diab, 2007), explicitly encodes definiteness, number, and gender information increasing the number of tags from 25 in RTS to 75 tags.

In order to confirm our hypothesis that paraphrases occur in similar contexts, we checked the distribution of the context similarity score over a sample of 1,735 phrase pairs corresponding to 867 positive and 868 negative pairs. Figure 5-2 shows the distribution, where the abscissa represents sub-intervals of the context-similarity value. As expected, we note that a greater mass of the negative pairs is concentrated in the low part of the scale, while the positive pairs move toward the right-hand side. Based on that, we learn that the context-similarity feature cannot be used deterministically for deciding positive or negative; however it can be combined with more relevant features and potentially help in classification.

Feature	***Description***
Context similarity of the representing vectors V_{P_1}, V_{P_2}	as described in Equations 4.1 and 5.2
POS	the part-of-speech tag of each contextual word, considering word order
BP	the base-phrase tag of each contextual word, considering word order

TABLE 5-2 - The features we use for training the context classifier.

We use SVM (Vapnik and Cortes, 1995) as the machinery for training the context classifier. Like before, we employ a quadratic kernel, to enable the learning process to consider any combination of features. We used WEKA (Hall et al., 2009) as a framework combined with LibSVM (Chang and Lin, 2011). The code is available for download.[7]

[7] http://cs.tau.ac.il/~kfirbar

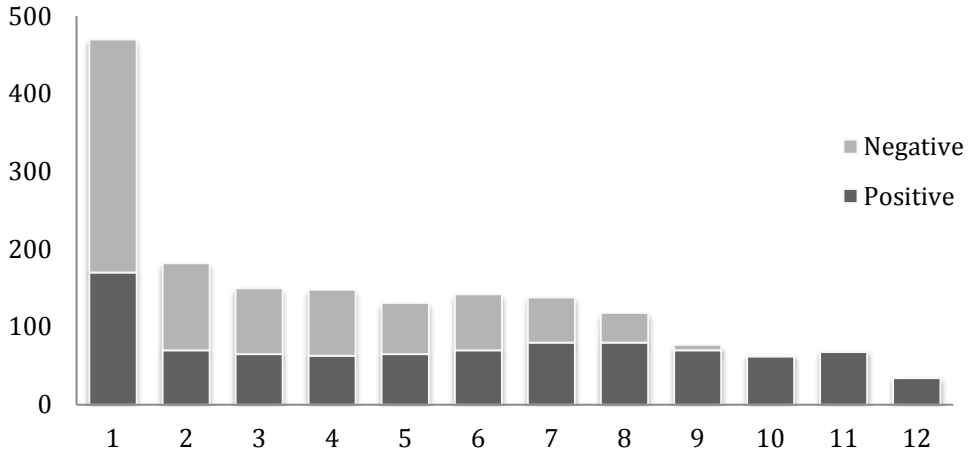


FIGURE 5-2 – The distribution of the context-similarity feature. The abscissa represents twelve sub-intervals of the context-similarity value, ranging from 0 to 1, that is, the first column represents the values between [0, 0.08), the second column represents the values between [0.08, 0.16), and so on. The dark part of each column is the number of positive pairs, while the light parts represent the negative pairs.

5.1.2 Evaluation of the Context Classifier

In our experiment, overall we extracted about 12,000 phrase pairs of various sizes. In order to manipulate the prior probability and to prefer precision over recall, the number of negative examples was selected to be twice as many as the positive examples. To evaluate the performance of the classifier on the training data, we run a 10-fold cross-validation check; the precision, recall, and $F_{\beta=1}$-measure results are presented in Table 5-3.

	Precision	***Recall***	***F-measure***
Positive	84.7	79.0	81.7
Negative	89.8	92.9	91.3

TABLE 5-3 – Evaluation results of the context classifier, using 10-fold cross-validation.

Essentially, we observe that our classifier does prefer precision over recall. It means that it is capable of distinguishing between contexts of identical phrases, on the lemma level, and the context of non-paraphrases. However, since the paraphrases we are look-

ing for are not part of this training set, and, in fact, are not composed of identical phrases, we cannot estimate the performance on the real data, based on these results.

5.1.3 Paraphrasing at a Large Scale

In order to work at a large scale, we use large parts of Arabic Gigaword (4th ed.), a monolingual Arabic corpus, as a resource for paraphrasing.

5.1.3.1 Pre-processing the Monolingual Corpus

We pre-process the corpus in a similar way to the comparable documents:

1. Morpho-syntactic analysis using AMIRA 2.0;
2. extraction of phrases of up to N (recall that N=6 in our experiment) words, to be used as paraphrasing candidates;
3. indexing the large amount of phrases, so that they can be searched in a reasonable amount of time, given an input phrase for paraphrasing.

We extract phrases following the first two rules we mentioned above, namely, phrases are not allowed to break a base phrase in the middle, and they have to contain at least one content (non-functional) word.

Every phrase was indexed in a phrase repository, so that it can be searched by each of its composing lemmas. Since we consider a large amount of data, too much to be stored in memory, we used MySQL,[8] a relational database, for storing those indexes.

5.1.3.2 Finding Paraphrases for a Given Input Phrase

Given a phrase P with C_L and C_R, its left and right contexts, respectively, for paraphrasing, our algorithm works in two steps:

1. Searching the indexed repository for finding potential candidate paraphrases;
2. pairing each candidate phrase with P, and deciding whether they are paraphrases or not using the context classifier.

Theoretically, any phrase should be considered as a potential paraphrase of P; however, checking every phrase from the repository is infeasible due to computational reasons. Therefore, in the first step we select phrases that have a reasonable chance to

[8] http://www.mysql.com

be identified as paraphrases for *P*. We do that by selecting phrases that have at least some percentage of words in common with *P*, matched on the lemma level; then, we work with different rate values and measure their effect on the results. In Table 5-4 we provide some examples for phrase pairs that match on different values of the common-lemma rate. In particular, each number represents the percentage of common lemmas with respect to the longest phrase among the two in each pair. Using this heuristic, we were able to force a decrease in the number of phrases that go on to the next phase. A disadvantage of using this technique is that the extracted paraphrases must have words in common. Considering better approaches, such as, matching lemmas on the synonym level, is definitely a direction for future investigation.

Common-lemma rate	***Extracted phrases***
0.7	***Ajl*** *AldfE* ***bEmlyp AlslAm***, "in order to push the peace process" ⇔ ***Ajl*** *mwASlp* ***Emlyp AlslAm*****,** "in order to reach a peace process"
0.5	*Aljy$* ***AlAmyrky***, "the American army" ⇔ *jndyyn* ***Amyrkyyn****,* "American soldiers"
0.4	***AlAslHp Alnwwyp*** *mn kwryA Al$mAlyp,* "the nuclear weapon of North Korea" ⇔ *Ant$Ar* ***AlAslHp Alnwwyp***, "the nuclear weapon proliferation"
0.3	***AlwlAyAt AlmtHdp*** *ms&wlp En*, "the United States is responsible for…" ⇔ ***AlwlAyAt AlmtHdp*** *AlAmyrkyp ttwly*, "the United States holds…"
0.1	*Aljy$* ***AlAmyrky*****,** "the American army" ⇔ *wqAlt Al$rTp An jndyyn* ***Amyrkyyn*** *ASybA bjrwH*, "and the police said that American soldiers were injured"

TABLE 5-4 – Examples for phrase pairs extracted from the phrase repository, using different common-lemma percentage values. The matching lemmas are in boldface.

In the second step, each extracted phrase is paired with the input phrase *P* and their feature vector is created using both phrases' context. Then, we employ the context classifier for deciding which of the pairs is a positive instance; the phrases that are found to

be part of a positive pair are deemed paraphrases. Note that a phrase may be retrieved from the repository more than once, potentially with a different context each time. The context classifier examines each instance individually and considers the corresponding contexts.

Like before, we use support-vector-machine as the underlying technology, and consider the confidence score returned by LibSVM. The confidence score, basically representing the distance of the pair from the separating hyperplane, is used to measure the quality of the returned paraphrases in the next step where we actually use those paraphrases in translation. In other words, the confidence score reflects the "good sense" of the context classifier.

Recall that we include part-of-speech as well as base-phrase tags of the contextual words in our feature set. They may help the classifier to find syntactic patterns in the contextual environment of the phrases. However, the resultant paraphrases cannot always replace the phrase without being detrimental to the input-sentence structure. Therefore, for every potential paraphrase P' we calculate two additional scores, based on the text that is composed of the context of the original phrase *P*, replacing *P* with P'. More formally, $C_L P' C_R$ is the text in which P' replaces *P*, which we consider to calculate the following scores:

Left-LM A language-model log-probability of the string $C_L {P'}_1$, where ${P'}_1$ is the first word of P'.

Right-LM A language-model log-probability of the string ${P'}_{\text{last}} C_R$, where ${P'}_{\text{last}}$ is the last word of P'.

Both scores measure how likely is to find P' in the same context as *P*. We refrain from using a single language-model score calculated over the entire string $C_L P' C_R$, because we do not want P' to affect the score, as every P' may have a different language-model probability, which we do not want to consider. We are after measuring only the matching of P' to the context of *P*, so we focus only on the left and right ends of P'. The language model is generated using a large monolingual corpus, on the lemma level, with SRILM (Stolcke, 2002).

Overall, for every paraphrase we calculate three scores: context-similarity confidence, Left-LM, and Right-LM.

We provide some examples for Arabic phrases and their generated paraphrases in Appendix B.

5.2 Embedding Paraphrases in Translation

In this section we describe how we use paraphrasing in translation. We experiment with an Arabic-to-English implementation of *Moses*, a well-known open-source statistical-machine-translation platform, aiming to improve its translation quality using different levels of paraphrases of fragments of the input sentence.

Using paraphrases in translation involves two main steps: (1) finding paraphrases of some fragments of the input text; and (2) embedding the derived paraphrases into the translation process. Paraphrases can be derived either for any fragment of the input text, or only for fragments that the system could not find a translation for, here referred to as *unseen phrases*. To be clear, an unseen phrase is a phrase that the system could not locate as a whole in the phrase table, but it still could be translated by the system, using the translations of (some of) its fragments.

Even if a phrase is identified in the phrase table, still there is a chance that its translation may be improved by translating one of its paraphrases. Callison-Burch et al. (2006), and Marton et al. (2009) extracted paraphrases of unseen phrases and enriched the phrase table with additional entries that pair the original unseen phrase with its paraphrases' translations. Despite the assumed benefit of considering paraphrases of phrases that exist in the phrase table, there is a risk that the system will prefer a translation of one of the paraphrases, which was incorrectly identified, to the translation of the original phrase. One way to deal with this risk is to assign scores to the paraphrases that reflect the quality of their equivalence, so that the system will consider them accordingly. However, this does not completely help, as demonstrated in Figure 5-3. In this example, the original sentence is *AElnt AlAdArp Almrykyp An*..., "the American administration announced that...", and the phrase *AlAdArp Almrykyp*, "the American administration", got paraphrased by *Alr}ys Almryky*, "the American president", incorrectly. As demonstrated, regardless of the fact that the original phrase *AlAdArp Almrykyp* is covered, the translation system may prefer to use *Alr}ys Almryky*, as its translation may have higher scores for both, the target-language language-model and the translation model.

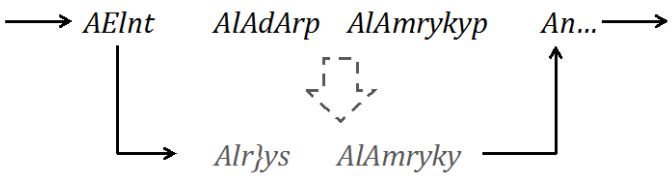


FIGURE 5-3 – Preferring a wrong paraphrase to a covered original phrase, resulting in an incorrect translation.

Jinhua et al. (2010) considered paraphrases also for input phrases that exist in the phrase table. They formatted the input text, augmented with paraphrases, as a *word lattice* (Dyer et al., 2008), and showed that it performs better than a system that merely calculates paraphrases for unseen phrases. Inspired by that work, we restructure the input sentence as a word lattice and augment it with paraphrases of all the composing phrases, regardless of their presence in the phrase table.

5.2.1 Translating with a Word Lattice

A word lattice is a directed acyclic graph, with every node uniquely labeled and every edge containing a word and a weight. A word lattice is mainly used when some parts of the input sentence are ambiguous and instead of selecting merely one interpretation in the usual way, the lattice encodes multiple interpretations, with each one encoded with its plausibility weight. In this sense, translating a word lattice is equal to the process of translating all the interpretation paths individually, and then selecting the one with the highest translation score. With a word lattice, the system is able to prune some paths that are unlikely to be selected as the final translations, resulting in a more efficient translation process. An example for a common usage of a word lattice is when the morphological representation of a word is ambiguous, and instead of using a context-sensitive morphological analyser to decide on a single reading, we compile the different readings into a word lattice, as illustrated in Figure 5-4. In this example, the word *lljnp* can be read in three different ways, depending on the context: (1) *l+ Al+ jnp*, "to heaven"; (2) *l+ ljnp*, "to/for/of committee"; and (3) *lljnp*, although unlikely, it can be used as a proper name. Instead of using the context to decide on the correct interpretation, all three interpretations are placed in a word lattice.

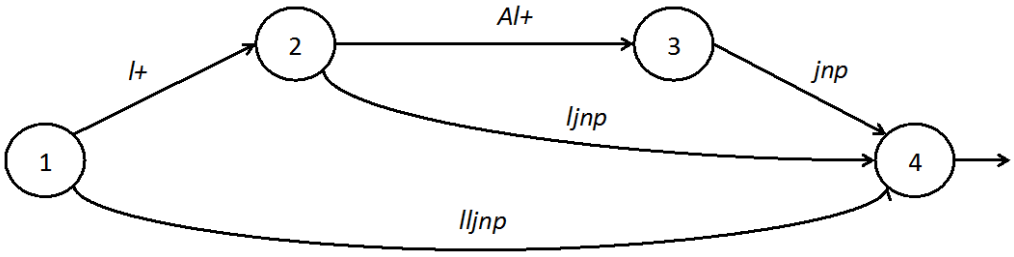


FIGURE 5-4 – Using a word lattice to represent ambiguous morphological representations.

One issue related to word lattices when used in statistical machine translation is the question of how to calculate the distortion model. Since the reordering issue is significant for the translation of Arabic text to English (see Section 1.4.4), let us try to shed some light on the way it is addressed.

Recall that the distortion model refers to some reordering situations, where the translation of a specific source phrase comes before the translation of its previous source phrase. In particular, it assigns a score that reflects the distance between the start index of the source-language phrase covered by a target-language phrase and the end index of the previous phrase pair, that is, $d(start_i - end_{i-1} - 1)$. Usually, a maximum number is set to limit the search space (e.g. allowing the system to skip up to six words). With a word lattice, it is difficult to know the correct value of this distance, because sometimes we do not know what the previous input phrase was, as illustrated in Figure 5-5, borrowed from Dyer et al. (2008). If *c* is translated into the first target word, the distortion model value should be either 3 or 2, depending on what path is chosen to translate the span [0,3]. In situations of large lattices, like in our case, every edge may span many nodes, and this problem becomes more pronounced. Dyer et al. (2008) suggest to always prefer the shortest path over all the other possibilities; eventually, we probably want to choose a translation with minimum reordered parts.

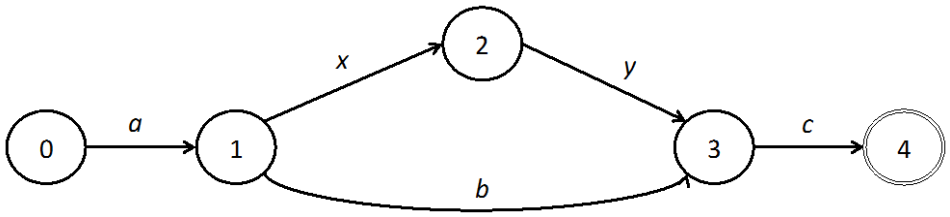


FIGURE 5-5 – Distance-based distortion problem when used on a word lattice (borrowed from Dyer et al., 2008).

5.2.2 Building a Semantic Lattice

As we have previously mentioned, Jinhua et al. (2010) described a way of building a lattice that captures the different paraphrases of parts of the input sentence. They use some straightforward construction guidelines, which we follow in our construction process. Our construction process is composed of two main steps. Given a tokenized input Arabic sentence for translation, we begin by initiating a lattice that captures the transition of the individual tokens, in a linear fashion. Let N represent the number of input tokens; we create a lattice of N+1 nodes and N edges, each representing one of the N tokens. Every edge should have a weight for representing the probability of considering the corresponding word as part of the input sentence. At this point, we assign the value 1 for every edge, keeping the lattice faithful to the input text.

In the second step, we add some bypass routes to the lattice, reflecting the paraphrases that were found by our paraphrasing algorithm. By way of example, if our paraphrasing algorithm finds a paraphrase P' of 4 tokens, for the phrase P starting at the second token and ending at the third token, we add a path of four edges from the node originating with the second token, to the target node of the third token, as illustrated in Figure 5-6.

Throughout our description, we have mentioned tokens rather than words. Particularly, we pre-process the Arabic text with MADA 3.1 for discovering the context-sensitive morphological analysis of every word, and then tokenize the text following the D3 tokenization scheme (Habash et al., 2009) as described in the introduction. Each morpheme is considered as a token for the purpose of lattice construction. When adding a new path to the lattice, we consistently consider tokens rather than words. We use the morphological analysis that was applied to the paraphrases within their original context, mainly because analyzing the words within the lattice context is difficult, as several paths may lead to the same phrase and we cannot be sure which one will actually be chosen.

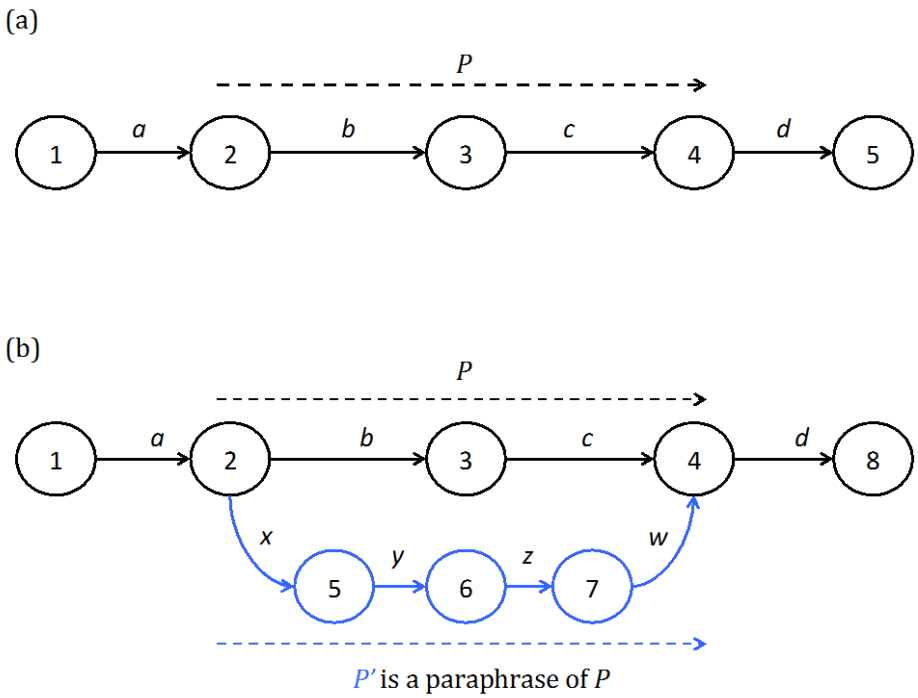


FIGURE 5-6 – Constructing the semantic lattice: (a) initiating the lattice with the input tokens; and (b) adding a paraphrase path to the lattice.

5.2.3 Assigning Weights to the Edges

We assign weights to every edge in the lattice in order to reflect the chance that a specific paraphrase represents its corresponding input phrase. Those weights are considered by the decoder as part of the log-linear model of the translation system. In particular, Moses introduces an additional feature function, referred to as *InputFeature*, which represents the input type; the weight of that feature function, as combined in the log-linear model, allows the decoder to consider different paths of the input lattice. The decoder considers the weights on the edges, as long as the InputFeature function allows it; in other words, considering the edge weights highly depends on the weight of InputFeature. The weight of the InputFeature function may be either set manually or learned from a tuning procedure, such as MERT, as described in Section 1.4.2. In our experiments, we take both approaches and measure the effect of the tuning procedure on the final translation results.

The weight assigned to each edge does not have to be a probability score. In fact, it can be any number, as long as it reliably reflects the chance of the edge to be selected among all outbound edges of a given node. Figure 5-7 shows a legitimate weight as-

signment to all outbound edges; in this example, the weights clearly prefer the word *a* to all the others. However, selecting the path depends also on other feature functions, for example, the translation model. In other words, not always the edge with the higher weight is eventually selected.

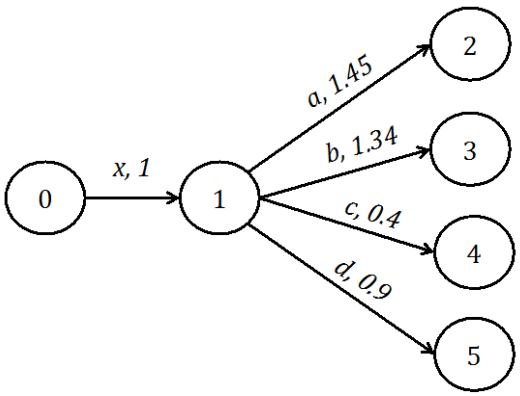


FIGURE 5-7 – Assigning weights to edges.

As mentioned before, adding paths that represent paraphrases of covered phrases may pose a risk in the decoding process. Let us say, for example, that we add a path representing an incorrect paraphrase P' of a specific phrase. If the decoder finds a highly ranked translation (according to the translation and language models for example) for P', the decoder may prefer this path even if the weight on the edge that initiates it is relatively small, resulting in what is considered an eloquent but unreliable translation. To deal with that, the decoder has to learn how seriously it should consider the weights on the edges. Tuning the decoder on semantic lattices, and by that, adjusting the weight of the InputFeature function may help to handle this issue.

Given a node *v*, we assign scores to its outgoing edges, as follow:

- If *v* has only a single outgoing edge, its score is 1. This can happen only when *v* either does not start a paraphrase path, or participates in a paraphrase path as an internal node.
- If *v* has $N > 1$ outgoing edges, it means that *v* starts *N*-1 paraphrase paths, plus one path that refers to the original word sequence. In this case, we calculate the score for each edge following two steps: (1) the edge that refers to the original sentence gets the score 1, and all the other edges get a score reflecting the quali-

ty of the paraphrase (we experiment with different scoring techniques, as described in Table 5-5 below); (2) we normalize the edge scores by dividing each value by the total sum, resulting in a final probability score.

Weighting condition	***Description***
CONF	the confidence score returned by the context classifier
AVG_CONF_LM	a single weight, calculated by the average of the confidence score, and the two language-model scores (calculated as described above)
CONF+AVG_LM	two individual weights: the first one is the confidence score of the context classifier, and the second one is the average value of the two language-model scores
CONF+Left-LM+Right-LM	three individual weights: the first one is the confidence score of the context classifier, and the other two weights are the scores of the left and right language-models respectively

TABLE 5-5 – The different weighting conditions we use in our experiments.

Moses does not limit the number of weights assigned to an edge. In fact, one can assigns as many weights as needed and connects each to a different score of the InputFeature function. Then, each score may be set or adjusted by MERT individually. We experiment with different weighting conditions; we list them in Table 5-5. In Figure 5-8 we demonstrate lattices using some of the weighting conditions.

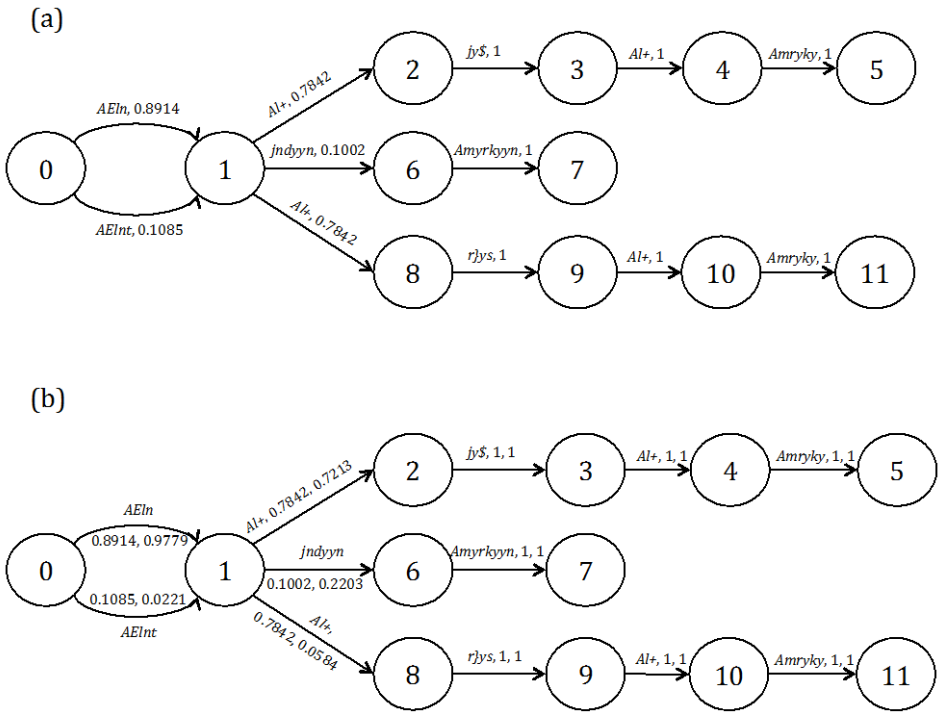


FIGURE 5-8 – Examples of semantic lattices using (a) CONF; and (b) CONF+AVG_LM.

5.3 Experimental Approach

Generally speaking, we want to use paraphrases in translation and to measure their effect on the translation quality. Our baseline is a Moses implementation of a phrase-based statistical machine translation from Arabic to English, using different sizes of bi-lingual corpora, focusing on the newswire domain. The implementation details are summarized in Table 5-6.

In addition to our paraphrasing algorithm, we consider verbal and nominal synonyms, as derived in Chapters 2 and 3, respectively. The synonyms are added to the lattice similarly to the paraphrases, creating different levels of semantic lattices, which are then used in translation.

We experiment under different configuration conditions, related to the following parameters:

- Bilingual corpus size;
- semantic-lattice level (verbal/nominal synonyms, paraphrases);
- tuning with semantic lattices;

- weighting condition (see Table 5-5);
- confidence-score threshold of the paraphrasing algorithm;

We use the same testing set, 2009 NIST OpenMT Evaluation set, through out the entire set of experiments.

Parameter	***Description***
English language-model	5-gram, using 10 million words extracted from English Gigaword (5th ed.)
Bilingual parallel corpora	LDC Arabic-English corpora: LDC2004E07, LDC2004E08, LDC2004T17, LDC2006E25, LDC2004T18, LDC2009T22
Tuning	MERT on a development set containing 5,000 sentences, corresponding to 132,699 Arabic words
Arabic tokenization scheme	D3 (Habash et al., 2009)
English tokenization scheme	using Moses internal tokenizer for English
Input type	word lattice
Evaluation set	the newswire part of the 2009 NIST OpenMT Evaluation set (LDC2010T23), containing 586 sentences corresponding to 20,671 tokens (17,370 words), with four English references

TABLE 5-6 –Baseline implementation details.

For paraphrasing, we use a monolingual Arabic corpus of about 10 million words, corresponding to 2.7 million phrases, extracted from Arabic Gigaword (4th ed.). For now, we set the common-lemma rate to 0.4, based on observations.

5.4 Results and Evaluation

In this section we report on our experimental results and analysis.

5.4.1 Experimenting with Different Thresholds for the Confidence Score

We investigate how the threshold on the confidence score of the paraphrasing algorithm affects the overall translation performance. Recall, the confidence score is the

value returned from the context classifier. Clearly, as we decrease the threshold, the number of generated paraphrases grows larger; however, the quality of the paraphrases is likely to decrease. To control the size of the lattice, because of efficiency concerns, we restrict the algorithm to generate at most three paraphrases for every input phrase.

The confidence-score values as returned from WEKA, reflect the distribution of the classification results; hence, the values are expected to be in the range [0, 1]. However, the confidence-score values that we obtain are mainly concentrated in the range [0.91, 1]. This may be related to the fact that we work with a binary classifier that was trained on a highly dimensional large-number of instances (the dimension is around 450), resulting in a relatively large number of support vectors. Therefore, we use several threshold values within that range.

Figure 5-9 shows the BLEU scores calculated for a system that uses a bilingual corpus containing one million Arabic words, running under different threshold values. The edges are weighted according to the CONF weighting condition, as defined in Table 5-5; the weight of the InputFeature function is arbitrarily set to 0.1. An alternative approach, which we consider in the following experiments, is to use tuning, as mentioned above.

It is clear from the results that, when using a relatively high threshold, the translation quality gets better, while the number of generated paraphrases decreases. At 0.97 we see a steep drop in the number of qualified paraphrases, and as a result the BLEU score slightly declines. Overall, the results are encouraging, as we may learn from this that the confidence score affects the translation results as expected. With low thresholds, we get a relatively large number of paraphrases that do not appropriately reflect the meaning of its generating phrase; hence may be detrimental to the final result. Moreover, since in this experiment we do not use the language-model scores that we calculate for every paraphrase, some paraphrases may result in an inarticulate sentence.

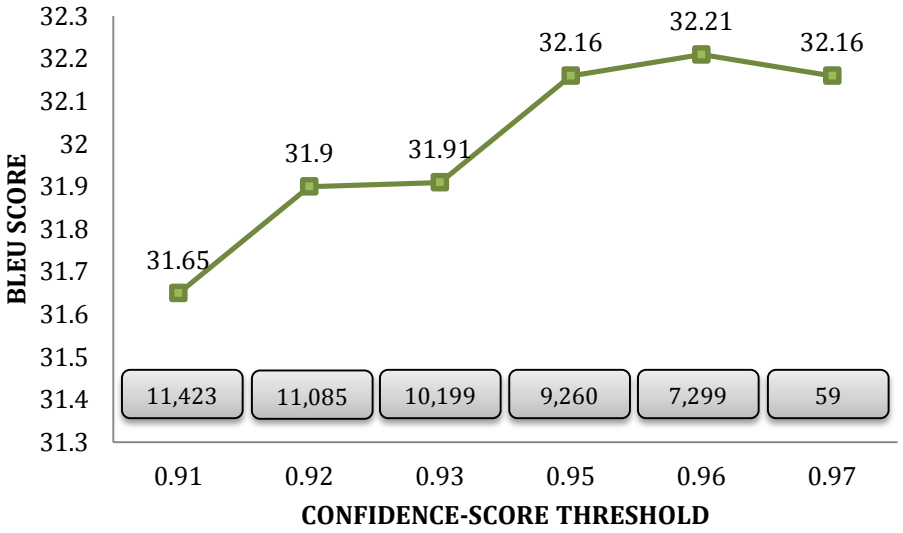


FIGURE 5-9 – BLEU scores using different threshold values for the paraphrasing-algorithm confidence score. The numbers in the rectangles indicate the number of paraphrases generated using each threshold value.

5.4.2 Experimenting Under Different Weighting Condition

To confirm our last hypothesis, we repeat the previous experiment, but this time running under several weighting conditions, as defined in Table 5-5. The conditions CONF, CONF+AVG_LM, and CONF+Left-LM+Right-LM use the same threshold values, which applied to the confidence-score value returned by the context classifier. Recall that AVG_CONF_LM is the average score of the confidence-score value and the average of the two language-model scores. As a result of that, the values of AVG_CONF_LM are in a lower range. Therefore, we use two sets of threshold values: one for the three conditions CONF, CONF+AVG_LM, and CONF+Left-LM+Right_LM, and one for AVG_CONF_LM. Figure 5-10 presents BLEU scores of using different threshold values applied on the confidence-score value as returned from the context classifier, with each weighting condition represented by its own curve. Similarly, Figure 5-11 provides BLEU scores of using different threshold values applied on the AVG_CONF_LM weighting condition.

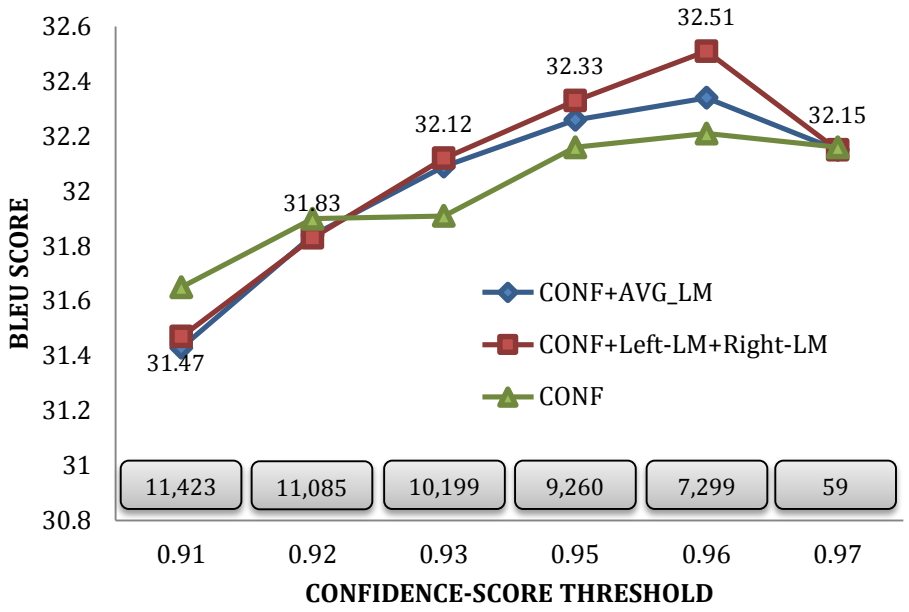


FIGURE 5-10 – BLEU scores of a system running with different thresholds on the confidence score as returned by the context classifier. Here, comparing different weighting conditions. The numbers in the rectangles indicate the number of paraphrases generated using each threshold value.

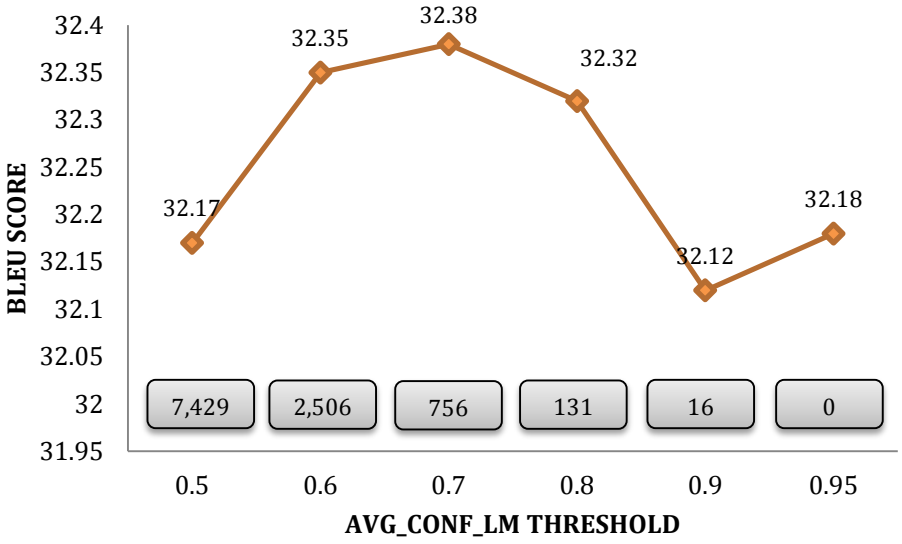


FIGURE 5-11 – BLEU scores as a result of using different thresholds applied on the AVG_CONF_LM weighting condition. The numbers in the rectangles indicate the amount of paraphrases generated using each threshold value.

The best score is obtained by the system running under CONF+Left-LM+Right-LM, that is, the one that assigns three individual weights to every edge, which in addition to the confidence score of the context classifier, represent the right and left language-model scores, as mentioned in Section 5.1.3.2. That system succeeded in improving the baseline's BLEU by 0.33 (the baseline system does not use the paraphrasing algorithm; it's BLEU score is 32.18); hence using the language-model scores as a tool for deciding whether to use a particular paraphrase in translation is productive. Interestingly, we observe that merely using two weights, replacing the two language-model scores by their average, was slightly less effective.

The three weighting conditions from Figure 5-10 behave similarly in terms of improvement per threshold value. As the threshold value increases, the number of qualified paraphrases decreases, and the translation quality increases. The best result is obtained on 0.96. However, this behavior was not seen when running under AVG_CONF_LM, as illustrated in Figure 5-11. It looks like this score does not behave as expected; perhaps this is related to the fact that we are trying to average two unrelated numbers: the confidence score as returned by the context classifier, and the average of the two language-mode scores.

We proceed by using CONF+Left-LM+Right-LM in subsequent experiments.

5.4.3 Measuring the Effect of Using Different Sizes of Bilingual Corpora

We experiment with different sizes of bilingual corpora using different variations of semantic lattices. We use the following lattices:

Baseline Containing merely the input text, formatted as a word lattice.

Verb synonyms Containing the input text, augmented with some verbal synonyms, as extracted in Chapter 3.

Noun synonyms Containing the input text, augmented with some nominal synonyms, as extracted in Chapter 2.

Noun+Verb synonyms Combining the verb and noun synonyms lattices.

Morph gen Containing the input text, augmented with morphologically generated verb forms (see below).

Paraphrases Containing the input text, augmented with paraphrases, identified using our paraphrasing algorithm.

Synonyms+paraphrases Combining the synonyms and paraphrases lattices.

As for paraphrases, synonyms of input words are added to the lattice. Since synonyms are single words, we allow any number of synonyms to be generated for a single input word. As we do not have a confidence score under this setting, we assign equal weight to all synonyms, including the word itself. The language-model scores are calculated in the same way as for paraphrases. Note that synonyms are provided on the lemma level; hence, they must be inflected to reflect the form of the original input word. For example, given an input verb *EvrwA*, "they discovered", derived from the lemma *Eavar-u_1*, the thesaurus returns the synonym *ka$af-i_1*. To generate the required form *k$fwA*, we employ Almor (Habash, 2005), an Arabic morphological generator, and provide it with the morphological features of the original word, as extracted by MADA.

Given a verb, our paraphrasing algorithm very often identifies different inflected forms of the same verb as paraphrases. For the most part, such forms can be generated deterministically, regardless of the context in which they occur. That was the motivation for building the *Morph Gen* lattice, which contains all the inflected forms generated by Almor that have the same English translation as the original form. In general, a typical Arabic verb has two forms: (1) *perfective*, to indicate the past tense; and (2) *imperfective*, to indicate the present and future tenses. Arabic verbs are characterized by their mood; as mentioned in the introductory chapter, there are four moods: indicative, subjunctive, jussive, and imperative. Arabic verbs are inflected for number, gender, and person; in addition to the singular and plural forms, they are inflected differently for duals. Moreover, Arabic verbs may be augmented with direct objects, which are also modified for number, gender, and person. Finally, proclitics may be attached to the verb to indicate conjunction and various prepositions. This complicated structure gives us an extremely large number of possible forms for every Arabic verb. For instance, the verb *kataba*, "wrote", has 2,556 different inflected forms if short vowels are considered, and 1,812 if they are not. On the other hand, English verbs have a relatively small number of forms, which mostly modified for tense and person. Given an Arabic verb, we are interested only in the inflected forms that keep the same English translation. Currently, given a verb *V*, we generate its additional forms using the following rules:

- *V* is in its perfective form, singular, 3rd person → we generate only the form that is inflected for the opposite gender. This is because in English, singular 3rd person, past-tense verbs are combined with *s* or *es*.
- *V* is in its perfective form, and NOT in the 3rd person → we generate the following perfective forms: 1st and 2nd person; singular, plural, and dual number; and both genders. All those forms normally have the same English translation.
- *V* is in its imperfective form → we generate all the possible imperfect forms.

Currently we do not handle direct objects. Dealing with other forms, as well as words with other part of speech, is left for future work.

Figure 5-12 shows an example for a lattice that was generated for one of the input sentences. For simplicity, we provide only the CONF weight, rounded to the second digit after the dot.

In the following set of experiments we use the CONF+Left-LM+Right-LM weighting condition with confidence score above 0.96, and a limitation of at most three paraphrases generated for each input phrase. Table 5-7 provides BLEU (Papineni et al., 2002) and METEOR 1.4 (Denkowski and Lavie, 2011) results; Figure 5-13 focuses on BLEU for convenience.

	BLEU				***METEOR***			
Corpus size →	**0.5**	**1**	**1.5**	**4.5**	**0.5**	**1**	**1.5**	**4.5**
Baseline	31.48	32.18	32.75	34.20	32.07	33.09	33.97	36.10
Verb Syns	31.34	32.06	32.38	34.11	31.82	32.95	33.91	35.97
Noun Syns	31.60	32.20	32.26	33.97	32.32	33.36	34.02	35.54
Noun+Verb Syns	31.50	32.31	32.30	34.07	32.77	33.75	34.43	35.72
Morph Gen	31.44	32.00	32.40	34.02	31.87	32.81	33.73	35.92
Paraphrases	32.28	32.51	33.19	34.21	32.83	34.30	34.71	36.23
Syn+paraphrases	32.39	32.72	33.28	34.12	33.06	33.73	34.67	35.92

TABLE 5-7 – Evaluation results for different size (in millions of Arabic words) bilingual corpora on different lattices. Boxes highlighted in light-green indicate improvement over the baseline.

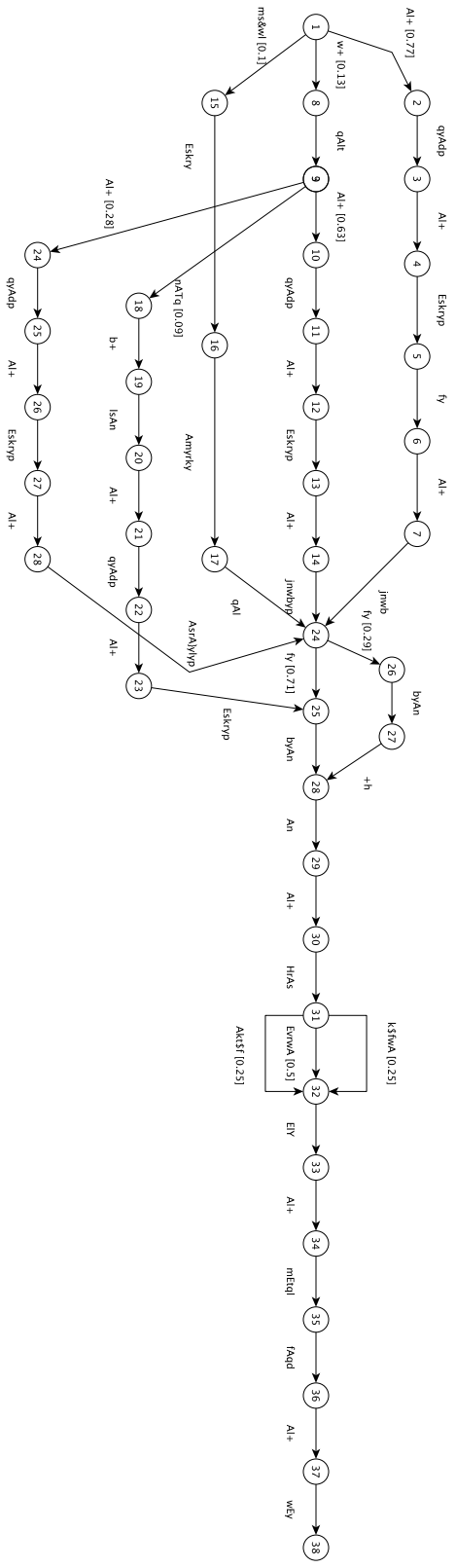


FIGURE 5-12 – An example of a lattice.

Table 5-8 summarizes the number of paraphrases/synonyms that were extracted in each method, including phrase-length distribution.

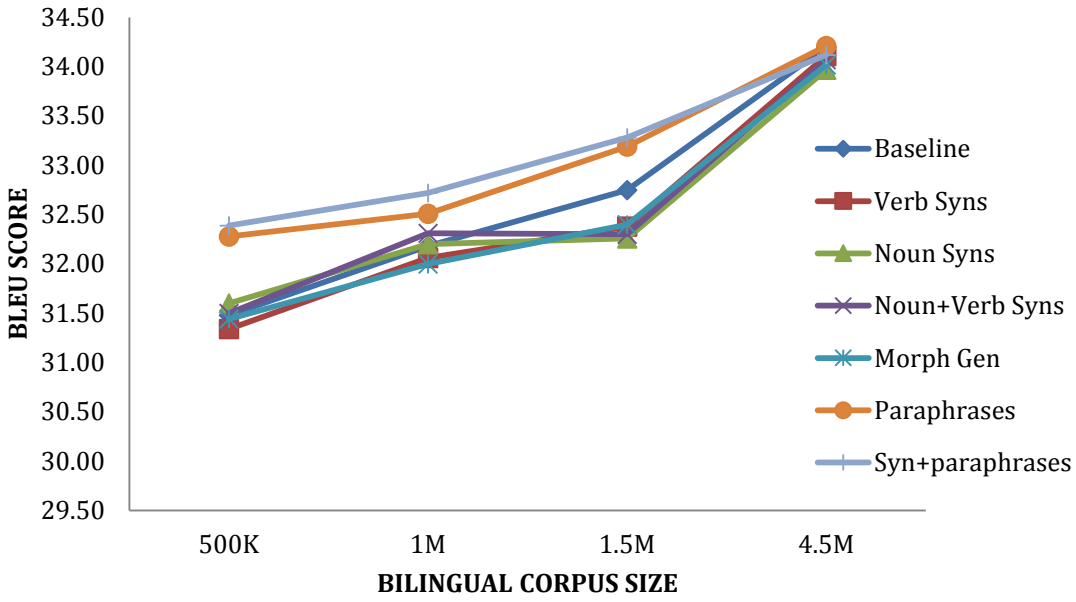


FIGURE 5-13 – BLEU scores, corresponding to the results presented in Table 5-7.

Method	***Total***	***1***	***2***	***3***	***4***	***5***	***6***
Verb Syns	466	466					
Noun Syns	649	649					
Noun+Verb Syns	1,115	1,115					
Morph Gen	642	642					
Paraphrases	7,299	1,194	3,791	1,773	442	73	26
Syn+paraphrases	8,414	2,309	3,791	1,773	442	73	26

TABLE 5-8 – The number of paraphrases/synonyms that were generated in each method, including phrase-length distribution. The *total* column contains the total number of paraphrases/synonyms, and columns 1-6 contain the amount of generated paraphrases of the specific size.

The best improvement over the baseline is 0.91 in BLEU score, observed when using a lattice containing paraphrases and all synonyms, running with a bilingual corpus of

500K Arabic words. When we increase the size of the bilingual corpus, the improvement is eroded, although it still exists. This observation complies with the observations made in similar works (Callison-Burch et al., 2006; Marton et al., 2009). As opposed to those works, which only use paraphrases of unseen input phrases, we used paraphrases to augment any input phrase, as long as the confidence score of the generated paraphrases passes the threshold.

We used paired bootstrap resampling (Koehn, 2004) for calculating statistical significance ($p < 0.05$) over the baseline. The improvements we get by *Paraphrases* and *Syn+paraphrases* on all corpus sizes are all statistically significant. The improvements we get by the synonym lattices are not significant, however.

The numbers of augmenting paraphrases per seen/unseen input phrases are summarized in Table 5-9. These numbers were calculated by counting the phrases for which at least one paraphrase was generated. We see a total of 8,414 paraphrases that were generated by the *Syn+paraphrases* method, used by the 500K system. Generating three paraphrases at most for each of the 390 unseen phrases (from Table 5-9), we have 1,170 paraphrases that were generated for unseen phrases. That leaves us with at least 7,244 seen phrases that were augmented with one paraphrase.

Method	**0.5**	**1**	**1.5**	**4.5**
Verb Syns	34	32	32	29
Noun Syns	25	22	18	14
Noun+Verb Syns	59	54	50	43
Paraphrases	331	217	211	193
Syn+paraphrases	390	225	219	199

TABLE 5-9 – The amounts of unseen phrases that were augmented with paraphrases. Each column represents the number of unseen phrases corresponding to the size (in millions of Arabic words) of the bilingual corpus used by the translation system.

Going back to the translation results reported in Table 5-7, we observe that using synonyms, verbal as well as nominal, moderately improves the final translations when using a relatively small bilingual corpus. When we increase the size of the corpus, the impact diminishes, possibly due to a decrease of unseen words, as observed in Table 5-

9. Synonyms helped also to improve the translations when combined with the generated paraphrases.

Using different inflected forms, represented by *Morph-gen*, was found to be counterproductive. In fact, the results obtained from using the *Morph-gen* lattice are consistently a little worse than the baseline results. This teaches us that the improvement obtained by using paraphrases was not due to verbs that get paraphrased simply as different inflected forms, although such cases do exist.

As a side note, the METEOR scores in Table 5-7 sometimes express better improvements compared to the corresponding BLEU scores; our assumption is that it happens due to the capability of METEOR to match words on the stem/lemma level, hence to account for translations that were generated by paraphrases that represent different inflected forms of the original phrases.

5.4.4 Tuning with MERT

So far, all our experiments were executed on a system that was merely tuned on the original sentences, formatted as word lattices, but including neither paraphrases nor synonyms. The weight of the InputFeature function, which affects the preferences of the decoder, was assigned arbitrarily to be 0.1. As a next step, we repeat the same experiments, minus the less productive ones, this time with a system that was tuned with MERT on the same development set, formatted as semantic lattices and augmented with paraphrases.

We repeat the same experiments as depicted in Section 5.4.3 and whose results are presented in Table 5-7, but this time running MERT for adjusting the weight of all the feature functions, including InputFeature. We omitted the *Morph gen* method, as it was not found to be competitive enough as a baseline. The two methods that use verbal and nominal synonyms individually were removed at this point and, instead, we keep the method that combines them and generating verbal as well as nominal synonyms in the same lattice. As in the previous experiments, we use the CONF+Left-LM+Right-LM weighting condition. Table 5-10 summarizes the results, and Figure 5-14 shows the improvement of each method using MERT on semantic lattices.

Clearly, there are some significant improvements when using MERT to determine the InputFeature score. With our best settings, our system got an improvement of +1.73

BLEU points over the baseline. This was obtained by the system that uses 500K Arabic words.

	BLEU			
	0.5	**1**	**1.5**	**4.5**
Baseline	31.48	32.18	32.75	34.20
Noun+Verb Syns	31.50	32.31	32.30	34.07
Noun+Verb Syns *Tuned*	**31.89**	**32.47**	**32.45**	**33.73**
Paraphrases	32.28	32.52	33.19	34.21
Paraphrases *Tuned*	**33.01**	**33.11**	**33.46**	**34.19**
Syn+paraphrases	32.39	32.72	33.28	34.12
Syn+paraphrases *Tuned*	**33.21**	**33.43**	**33.68**	**34.10**

TABLE 5-10 – Evaluation results of using different sizes (in millions of Arabic words) of bilingual corpora on different semantic lattices, with *MERT applied on semantic lattices* to adjust the weight of the InputFeature function. Boxes highlighted in light-green indicate improvement over the untuned system.

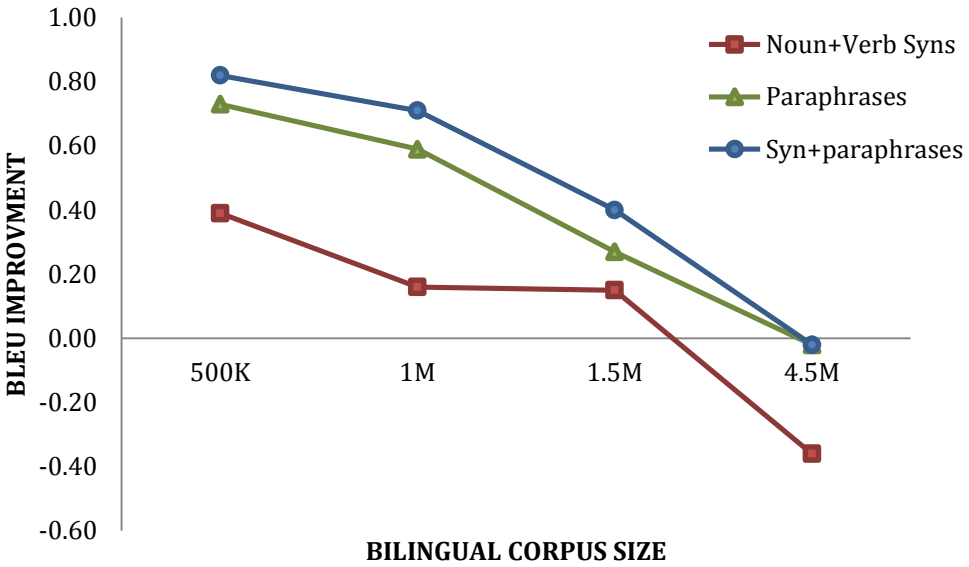


FIGURE 5-14 – Improvement in BLEU scores when using MERT to determine the weight of the InputFeature function, corresponding to the results presented in Table 5-10.

The results we get by *Paraphrases* and *Syn+paraphrases* on all corpus sizes except the largest one are all statistically significant ($p < 0.05$). The results we get by Noun+Verb Syns on the smallest corpus is statistically significant as well; the rest of the results are not significant, however.

For the most part, tuning the parameters for paraphrases helps improve the translations. But we see a slight drop for the larger corpus, suggesting that the weights assigned to other features were slightly miscalculated. To learn more about the behavior of the improvement curves, we look at the weight given for the InputFeature function in each experiment. For example, in the *paraphrases* method, the weights given for the InputFeature function, are 0.91, 0.23, 0.27, and 0.25 for the systems 500K, 1M, 1.5M, and 4.5M respectively. We see that in the 500K system, where the number of unseen phrases is relatively large, the InputFeature function gains more importance. In the larger systems, the weight of the InputFeature function dramatically reduces, likely, to compensate for the problem of preferring incorrect paraphrase paths to an original covered phrase.

5.4.5 Qualitative Evaluation

In a qualitative evaluation of the results, we discover some examples that are worth mentioning. We list some of them in Table 5-11.

Different number/gender/person Those are cases in which the unseen original phrase got translated by the same phrase, inflected differently for number, gender, and/or person; for instance, Example 1 from Table 5-11. In such cases, the English translation is inflected as the paraphrase rather than as the original phrase. It is likely that one will find *Polish diplomat<u>s</u>* in a reference translation, and not *Polish diplomat* as translated by the system. BLEU may not get improved in such cases; however, from a reader's point of view, it is better to have this translation than nothing. METEOR does get improved, as it is working also on the English-stem level.

	Original phrase	***Baseline translation***	***Paraphrase***	***Translation***
1	*dblwmAsywn bwlndywn*	NA	*dblwmAs bwlndy*	Polish diplomat
2	*Al+ r}ysyyn*	NA	*Al+ r}ysy*	main
3	*w+ yHAwl mHmd slymAn AlHzyn HmAyp*	Mohammad Suleiman AlHzyn and try to protect	*yHAwl mHmd sly-mAn AlHzyn HmAyp*	Mohammad Sulei-man AlHzyn tries to protect
4	*Abnh*	NA	*AbnhA*	her son
5	*w+ y}n Al+ Tfl*	the Child and y}n	*bkY Alwld*	the child cried
6	*wADAf >n AlEskryyn*	The military wADAf	*ADAf >n Aljy$*	he added that the army
7	*AntHAr*	NA	*<st$hAd*	Suicide
8	*Al+ r}ys Al+ sAbq bwrys yltsyn*	the former president Boris Yeltsin	*Al+ r}ys Al+ sAbq mHmd xwnA*	the former president Muhammad xwnA
9	*Al+ A$hr Al+ Axyrp*	the last months	*Al+ AyAm Al+ Axyrp*	the last days
10	*w+ qAl Al+ mtHdv*	the spokesman said	*w+ qAl Al+ mtHdv b+ Asm wzArp Al+ xArjyp*	and the spokesman of the ministry for foreign affairs said
11	*Al+ r}ys, Al+ Amyrky*	the American president	*Al+ AdArp Al+ Amyrkyp*	the American administration
12	*nzE Al+ slAH Al+ nwwy*	The elimination of weapon for mass destruction	*Ant$Ar Al+ AslHp Al+ nwwyp*	the spreading of weapon for mass destruction

TABLE 5-11 – Examples of paraphrases and their translations. The Arabic text is tokenized according to the D3 scheme (Habash et al., 2009).

Wrong tokenization Cases of incorrect tokenization of the original words, resulting in a bad translation. In Example 6 from Table 5-10, the first word *wADAf*, "and he added", was not tokenized properly; the conjunction proclitic *w+* was not separated from the base word as necessary. Hence, the entire phrase was not translated as whole. However, using its paraphrase, which completely omitted the conjunction proclitic, and also replaced the word *AlEskryyn, "the military/the armed people"* with its synonym *Aljy$*, "the army", improved the final translation.

Entailment Those are cases in which the paraphrases entail the original meaning of their corresponding phrase, but at the same time contain more details that get translated. In cases where the original phrase does not have a translation, the translated paraphrase, even if adding some incorrect information, may help. However, when the original phrase does have a reasonable translation, the overall translation may be damaged. This is demonstrated by Example 10 from Table 5-10.

Antonyms Our paraphrasing algorithm works on the context level. As mentioned in Chapter 4, other semantically related phrases may be found and reported as paraphrases, incorrectly. Antonym is one such semantic relation that usually degrades the overall translation results, as demonstrated in Example 12. However, even though we have not observed that in our evaluation, sometimes the automatic scores may be incorrectly improved, when the original phrase is not covered by the system, and the paraphrase's translation contains some correct words that are used in the reference translation.

Other situations exist; here we only mention some of the most frequent cases.

We learn from the above that BLEU and METEOR scores do not necessarily reflect the improvement in translation. This goes in two directions, however; there are examples that show improvement in translation but not on BLEU/METEOR, as well as examples that may contribute to the BLEU/METEOR scores but in fact, harm the final translation.

Even though we have not conducted a manual evaluation, in which several human evaluators are vetting the resulted translations, like we did in the previous chapter, we believe that the improvement that we have achieved using the paraphrasing technique is larger than the improvement we have seen by BLEU and METEOR. This assumption is made based on the qualitative-evaluation process that we performed and reported above. Callison-Burch et al. (2006) performed such an evaluation in a similar setting resulting in the same conclusions.

5.4.6 Results of Other Works

To the best of our knowledge, this is the first work to apply automatically generated paraphrases in Arabic for improving translation results. As mentioned above, there are other works that derive paraphrases in other languages and use them in translation.

Callison-burch et al. (2006) generated paraphrases of up to ten words from bilingual corpora, as described in the introduction chapter. They reported an improvement of +1.5 BLEU points when running with a Spanish-to-English translation system, and almost +2.0 BLEU points when running with a similar French-to-English system. They only generated paraphrases for unseen phrases by simply enriching the phrase table with the translations of the generated paraphrases. They defined a new feature function that assigned a score for every generated paraphrase corresponding to its confidence score, and used MERT to adjust its weight.

Marton et al. (2009) reported an improvement of < 1 BLEU point in a Spanish-to-English system, and +1.67 BLEU score on a Chinese-to-English system. Similarly to Callison-burch et al. (2006), they were only generating paraphrases to unseen phrases.

Jinhua et al. (2010) experimented with paraphrases, which were formatted as word lattices, with the InputFeature weight determined manually. They generated Chinese paraphrases following the method of Bannard and Callison-burch (2005), using bilingual corpora, and used them in a Chinese-to-English translation system. They reported an improvement of ∼2 BLEU points with their best settings.

In all the abovementioned works, it was found that paraphrases did help to improve the translation quality, especially when the bilingual corpus used by the translation system was relatively small. When the size of the corpus grows larger, we usually see only a slight improvement over the baseline, or no improvement at all. Our experiments showed the same.

5.5 Summary

In this chapter we have demonstrated the potential of using paraphrases and synonyms to improve the results of a phrase-based statistical translation system. Like before, we focused on Arabic, a highly inflected language. The paraphrases were derived for input phrases and embedded in word lattices, considering paraphrases for covered as well as unseen original phrases. We used a relatively large monolingual corpus for deriving new paraphrases, using a classifier that works on the contextual words. The classifier was trained on a set of examples that were automatically extracted from a relatively small set of comparable documents.

Based on our observations, we may conclude the following:

1. Paraphrases help to improve the translation quality. The improvement was less significant when we increased the size of the bilingual corpus used by the translation system.
2. Word lattices were found to be an effective way of considering paraphrases in a statistical translation system, enabling the usage of paraphrases for any input phrase, regardless of its coverage by the original phrase table. There is a complexity limitation though: considering paraphrases for covered phrases may result in large lattices, which may be infeasible to translate. Therefore, we restricted our lattice to have at most three paraphrases for every original phrase.
3. Using MERT to adjust the weight of the InputFeature function helped improve the final translation results. To the best of our knowledge, this is a novel idea.
4. Synonyms that were derived from dictionaries, as described in Chapters 2 and 3, helped to improve the translation quality. However, the improvement was less prominent than the one we obtained using paraphrases. Unsurprisingly, the best results were obtained by combining paraphrases and synonyms.
5. Applying language-model scores, which measure the matching level of each paraphrase to the context of the original phrase, was found to be effective. The improvement was obtained only when the language-model scores were used as part of a weight vector, considering each score individually.
6. Although some of the derived paraphrases were in fact different inflected forms of their corresponding original phrases, we learned that this was not the salient reason for improvement.
7. Although the paraphrasing technique that we used in this chapter is far from being accurate in terms of recall, it could still generate some useful paraphrases that helped to improve the translation quality. Remember that we configured our algorithm to prefer precision to recall by merely considering phrases that have some lemmas in common with the subject phrase. Improving this technique is expected to improve the results even more.

Our qualitative evaluation, although was not quantified, uncovered some patterns in which paraphrases either improve or damage the translation results.

Due to Arabic's rich morphology we work on the lemma level. The lemma groups together all the inflected perfective and imperfective forms of a verb, and all the inflect-

ed singular, dual and plural forms of a noun. Consequently, our "paraphrases" include pairs with shared meaning, regardless of inflection for number, gender, and person. The motivation was that such pairs often have similar English renderings. Currently we do not encode inflection-related features for training the context classifier.

Considering the scope of this chapter, here are some potential research directions:

1. Applying this paraphrasing technique to other languages, preferably with complex morphology.
2. Exploring other matching techniques for finding candidate phrases, before using the context classifier, so as to capture additional paraphrases potentially not sharing the same lemmas. One option is to use synonym-level matches, rather than merely the simple lemma level.
3. Exploring linguistic approaches in determining the context of a phrase instead of using the surrounding words. For that purpose, one may use a syntax parser to uncover syntactic dependencies and include the connected words among the contextual ones. For more information on dependency parsing, we suggest the book *Dependency Parsing* by Sandra Kübler, Ryan McDonald, and Joakim Nivre (2009). Obviously, other linguistic approaches may be considered.

Chapter 6

Arabic Multiword Expressions

6 Arabic Multiword Expressions

A multiword expression (MWE) or simply "expression", refers to a multiword unit or a collocation of words that co-occur together statistically more than chance. Sag et al. (2002) define a multiword expression as "idiosyncratic interpretations that cross word boundaries (or spaces)". They cited Jackendoff (1997), estimating that the number of multiword expressions in a speaker's lexicon is of the same order of magnitude as the number of single words. They found out that in WordNet 1.7, for example, 41% of the entries are multiword expressions.

Typically, MWEs are classified based on their syntactic construction type. Among the various classes, one can find the Verb-Verb Construction (VVC), Verb-Noun Construction (VNC), Verb-Particle Construction (VPC), Noun-Noun Construction (NNC), and Adjective Noun Construction (ANC). A MWE typically has an idiosyncratic meaning that is more or different from the meaning of its component words. A MWE meaning is transparent, also referred to as *compositional*, if the meaning of the expression as a unit can be predicted from the meaning of its words, such as in the English MWE *prime minister*. On the other hand, idiomatic expressions, also referred to as *non-compositional*, are expressions that their overall meaning is difficult to predict from each individual component word sense, such as in the English MWE *spill the beans*. The complexity of identifying MWEs in running text comes from the fact that an idiomatic MWE may occur with the literal meaning of it's individual words, as in cases of *spill the beans* when someone literally did it.

In this chapter we address the general task of finding Arabic MWEs in running text, by looking at two subtasks: (1) MWE Identification: finding the boundaries of the MWEs in the text, that is the first and last word of every expression; and (2) MWE Classification: finding the construction type of every identified MWE. To deal with first subtask, we use a pattern-matching algorithm to generate a relatively noisy supervised training set, which is then used to augment a manually annotated data for building an Arabic MWE classifier. The main contribution of this work is:

- Improving accuracy on the identification subtask, using "noisy" annotated data augmenting a small manually annotated data;
- Dealing with gappy MWEs of all construction types;
- Showing the impact of explicitly modeling morpho-syntactic features on the identification and classification subtasks;

- Integrating identification and classification into a single task.

We continue as follows: In Section 6.1 we explain how we created a repository of Arabic MWEs; in Section 6.2 we describe our annotation scheme. Our identification and classification approach is described in Sections 6.3 and 6.4, followed by Section 6.5 in which we elaborate on the way we create the training data for our experiments, depicted in Section 6.6. Finally, we conclude in Section 6.7.

6.1 Building a Collection of Arabic Multiword Expressions

We collect a large number of MWEs from various Arabic dictionaries (Abou Saad, 1987; Sieny et al., 1996; Dawood, 2003; Fayed, 2007), and classify them based on their syntactic constructions. The MWEs are manually annotated with the context-sensitive SAMA morphological analysis for every word to assist an automated identification of MWEs in a large corpus of text.

We pre-process the MWEs, taking the following steps: 1) cleaning punctuations and unnecessary characters; 2) breaking alternative expressions into individual entries; and 3) running MADA 3.1, combined with SAMA 3.1 on each MWE individually for finding the context-sensitive morphological analysis for every word. Some of the extracted MWEs are originally enriched with placeholder generic words that play the same semantic role in the context of the MWE. That set of generic words is manually normalized and reduced to two main types, as shown in Table 6-1.

Generic Type	***Semantic Role***	***Example***
flAn "so-and-so", a person	Agent/Patient	*qr* ***flAn*** *EynA* "pleased **someone**"
<mr "something", an issue	Object	***<mr*** *Abn ywmh* "**something** very new"

TABLE 6-1 – Generic types.

Generic words are sometimes provided with or without additional clitics. For example, in the expression *lEbt* **[*bflAn*]** *AldnyA*, literally, "the world played [*passive*] with so-and-so", which could be translated as "life played havoc with so-and-so", the word *bflan* has the preposition *b*, "with", cliticized to it. Every word that substitutes a generic word (an instantiation) has to comply with the morphological features of the context surrounding it.

We realize that the short context we had for every MWE was not sufficient for MADA 3.1 to find the correct analysis with a reasonable accuracy. Therefore, we manually select the correct SAMA 3.1 analysis for each word within every MWE. Generic words are also assigned with their correct analysis in context.

The construction type of every MWE is also assigned manually; in this chapter we focus only on the most frequent types: VVC, VNC, VPC, NNC, and ANC.

Verb-Verb Construction (VVC) As in *>x** [*flAn*] *w>ETY*, "[someone] gave and took" as in "discussed in detail/haggled".

Verb-Noun Construction (VNC) This category includes both, Verb-Noun Idiomatic Constructions, (VNIC), as well as Light Verb Constructions (LVC), that is, a verb conveying almost no meaning by itself, modified by a noun. *md~* [*flAn*] *Aljswr*, "[someone] built bridges" as in "extending the arms of peace/bridged the communication divide", is an example of a VNIC. The following is an example of a LVC: *qAm* [*flAn*] *bzyArp*, "[someone] visited", literally, [someone] carried out a visit, where the verb *qAm*, "carried out", is modified by the noun *zyArap*, "a visit", taking the preposition *b+*, "by/with", as part of the expression. Another example is *<lqY* [*flAn*] *mHADrp*, "[someone] lectured", literally, [someone] delivered a lecture.

Verb + Prepositional-Phrase Construction (V+PP) For example, *mDY* [*flAn*] *fy*, "[someone] continues working on". Similar to English, the preposition may completely modify the meaning of the verb in its simplex form, as in *qDY* [*flAn*] *ElY*, "[someone] put an end to" vs. *qDY* [*flAn*], "[someone] judged". In English and other Germanic languages, there are Verb-Particle Constructions (VPC), such as *put on*. The particle is an integral part of the construction, which may appear before or sometime after the direct object. Identifying English VPCs in running text is not a trivial task, as sometimes a combination of a verb and a particle can be used either as a VPC, or as a simplex verb combined with a prepositional phrase; for example: 1) ***put on*** *the sweater*, or 2) ***put*** *the book* ***on*** *the table*. In the first sentence, *put on* represent a VPC with the meaning of wearing a cloth, and in second sentence, *put* is mentioned in its simplex form combined with the preposition *on*, indicating the place of the action. Kim and Baldwin (2009) worked on distinguishing between the two cases in running text.

In the Arabic expression *qDY ElY*, the word *ElY* introduces a prepositional phrase, and therefore is considered as a preposition and not as a particle. In other words, in Arabic, prepositions may subcategorize the verb by modifying its meaning. However, VPCs as exist in English, do not occur in Arabic.

Identifying Arabic V+PPs in running text is not trivial. The same preposition can be

used in two different contexts: (1) subcategorizing the verb and modifying its simplex meaning (corresponding to the subcategory that takes a noun-phrase), as in *qDY ElY*; and 2) introducing an additional prepositional phrase or complementing other words or phrases, while keeping the simplex meaning of the verb. For example, let's take the verb *<H**, "took". The preposition *ElY*, "on/about", subcategorizes the verb and changes its meaning to "criticized". However, in the following sentence:

>H AltEwyD ElY AlHAdv mn $rkp Alt>myn*,

"Taking the compensation for the accident from the insurance company"

The preposition *ElY* does not change the meaning of the simplex verb form, but rather complements the verbal noun *AltEwyD*.

As another example, let's take the verb *HDr*, "attended". The preposition *<lY*, "to/into", subcategorizes the verb and changes its meaning to "arrived to". However, in the following sentence:

HDr Alm&tmr <lY jAnb AlbTryrk lHAm r&sA' AlTwA}f fy lbnAn,

"The leaders of the communities of Lebanon attended the conference with Patriarch Laham"

The preposition *<lY* does not change the meaning of the simplex verb form. Without referring to the word senses, one can translates this into "the conference **arrived to**…", which does not seem reasonable. Therefore, involving semantic concepts, following Kim and Baldwin (2009), seems like one of the ways to go about this issue.

Noun-Noun Construction (NNC) As in *Enq {lzjAjp*, "bottleneck". The NNC type in Arabic includes *Idafa* constructions, that is, the construct state, where the first noun dominates the second one usually to form what is known in other languages as compound nouns. Construct state in Arabic is typically used to indicate possessive forms, as in *kTAb AlTAlb*, "the student's book", literally, *book the student*.

Adjective Noun Construction (ANC) As in [*flAn*] *wAsE {l&fq*, "[someone] broad-minded".

Due Arabic's rich morphology, MWEs can be expressed in variety of forms, expressing various inflections and derivations of the words while maintaining the exact same meaning. For example, *>gmD* [*flAn*] *Eynyh En* [*Al>mr*], "[one] disregarded/overlooked/ignored [the issue]", literally, closed one's eyes, vs. ***>gmDt*** [*flAnp*] ***EynyhA*** *En* [*Al>mr*], "[one_*fem*] disregarded/overlooked/ ignored_*fem* [the issue]", where the predicate takes on the feminine inflection. However, in many cases, there are morphological features that cannot be changed in different contexts, for example, *mkrh >xAk lA bTl*, "forced with no choice", in this example, regardless of context, the words of

the MWE do not agree in number and gender with the surrounding context, these are *frozen* or *fixed* expressions.

MWE Category Type	***Number***
VVC	41
VNC	1,974
V+PP	670
NNC	1,239
ANC	285

TABLE 6-2 – Arabic MWEs by construction types.

The final collection contains 4,209 MWE types. Table 6-2 presents the total number of MWE types per every syntactic construction type.

This collection was published and released (Hawwari et al., 2012).[9] Some examples are shown in Appendix C.

6.2 Annotation

In order to identify MWEs in running Arabic text, we train a classifier on texts with labels indicating the boundaries of every MWE instance. We utilize the IOB notation for marking the boundaries of the MWE instances. With this notation, every MWE's initial word is annotated with a label starting with the letter B (Beginning), every MWE's internal word is annotated with a label starting with the letter I (Inside), and finally, the rest of the words (non MWE) are annotated with the label O (Outside). For example, the annotations of the Arabic sentence: *hd>t AlzwbEp Alty >vArhA Alfhm AlxAT'},* "The storm that was a result of the misunderstanding calmed down", is:

hd>t **/B-MWE** *AlzwbEp* **/I-MWE** *Alty* **/O** *>vArhA* **/O** *Alfhm* **/O** *AlxAT'}* **/O**

As mentioned before, sometimes a MWE occurs with intervening additional words, such as modifiers, that are not part of the original expression. For instance, the MWE *wDEt <wzArhA,* "*something[war/battle/rebellion]* is over", as in *wDEt AlHrb <wzArhA*, "the war is over", is found in the text as follows: *wDEt* ***AlHrb AlEAlmyp AlvAnyp*** *<wzAr-*

[9] http://cs.tau.ac.il/~kfirbar

hA, "the **second world war** is over". The nominal modifiers *AlHrb AlEAlmyp AlvAnyp* ("the second world war…") are not part of the original MWE, and therefore considered as intervening fillers (IF), also known as gaps. With English, IFs occur quite frequently in VPCs, when the object of the verb with its modifiers are inserted before the verb particle, as in *he closed* ***his office door*** *off*. In Arabic, the sentential structure Verb-Subject-Object (VSO) introduces additional cases of potential IFs, in particular subject words that appear between the verb and its object in VNCs, as in *Eqd* ***AlrEb*** *lsAnh*, "**the fear** left him tongue-tied". English VNICs and LVCs may also occur with IFs; for example, *make a* ***wise*** *decision*. However, we believe that such cases happen less frequently than in cases of Arabic non-MWE subject words being inserted after the verb in Arabic VNCs. In order to get an idea about the distribution of IF words among occurrences of Arabic MWEs in free text, we employ a pattern-matching algorithm (see below in Section 6.3), which considers gaps in matching, using our Arabic MWE repository to find MWE occurrences in 500K words, extracted from Arabic Gigaword (4th ed.). Table 6-3 illustrates the statistics of IF word occurrences within Arabic MWEs. Empirically, we learn that Arabic NNCs almost occur with no gaps; however, we infrequently encounter such following examples: *tTAlb Al$rkp btTbyE* ***Alm$tryn*** *ElAqAthm mEhA*, "the company asks **buyers for** normalization of relations with it". The percentage of gappy VNCs is larger than V+PPs, but the gaps in the latter type are longer in general.

	NNC	***VNC***	***V+PP***
% of MWE occurring with IF words	< 1%	32%	10%
average gap length (number of words)	3.0	1.7	2.6

TABLE 6-3 – Statistics on IF words occurrence, calculated over a corpus of 500K Arabic words.

In our annotation scheme, IF words are annotated with the O label, and each MWE fragment, that is, before and after the gap, is augmented with a sequence number; for example:

wDEt/**B-MWE-1** *AlHrb*/**O** *AlEAlmyp*/**O** *AlvAnyp*/**O** *<wzArhA*/**B-MWE-2**

The first fragment is annotated with B/I-MWE-**1**, the IF words are annotated with the label O, and the second fragment with B/I-MWE-**2**. MWEs that occur with no IFs are an-

notated with B/I-MWE-1.[10] This leaves us with the 5 following labels for the MWE identification subtask: B-MWE-1, I-MWE-1, B-MWE-2, I-MWE-2, and O. To the best of our knowledge, this is the first work to address automatic identification of gapped VNCs.

In addition to identification, we address the classification subtask for which we rely on the syntactic construction types of the annotated MWEs. In the classification subtask, the classifier is assigned to find the construction type of a given MWE instance. In particular, we train our classifier in two tagging conditions. In the first condition, CASCADED, we perform MWE boundary detection followed by classification. Specifically, in the first step the classifier detects the span of the MWEs, and then classifies them using the identified boundaries as features. The second tagging condition, INTEGRATED, is where we perform boundary detection and classification in one fell swoop. The labels that we use for this condition, reflect the MWE's syntactic construction class. For example, the annotation of the abovementioned sentence is:

hd>t/**B-VNC-1** *AlzwbEp*/**I-VNC-1** *Alty*/**O** *>vArhA*/**O** *Alfhm*/**O** *AlxAT'}*/**O**

In this work we focus only on three construction types: V+PP, VNC, and NNC. Therefore, we use the 13 following labels: B/I-VPP-1/2, B/I-VNC-1/2, B/I-NNC-1/2, and O.

6.3 Pattern-Matching Algorithm for MWE Boundary Detection

We developed a pattern-matching algorithm for discovering MWEs in Arabic running text. The main goal of this algorithm is to deterministically identifying instances of MWEs from our repository, in free texts, considering their morpho-syntactical variations. Then, we use this algorithm to generate training data for supporting our supervised machine-learning framework.

The algorithm recognizes AMIRA 2.0 output files, scanning word-by-word, comparing to the SAMA 3.1 analysis of each MWE word, and considering different morphological variations and potential gaps**.**

Morphological variations Our algorithm was designed to handle various word's representations, as provided by AMIRA 2.0, and compare them to the manually selected SAMA 3.1 morphological analysis of every MWE word. Although the matching levels are

[10] Although cases of more than one gap may occur, in this work we ignore them since they are very infrequent.

configurable, here we match words on the lemma level. Furthermore, we noticed that in some cases proclitics, such as conjunctions (*w+*, *f+*) and prepositions (*b+*, *l+*, *k+*), may be important for matching MWE words, hence we configured the algorithm to take them into consideration as well. For example, in the MWE: *<x* bAlv<r*, "requited", the proclitic *b+*, "with", expressed in the last word, is important for matching.

Gaps To consider gaps of MWEs in context, the pattern-matching algorithm uses the words' part-of-speech and base-phrase tagging information. In particular, it allows every MWE noun to be matched with a complete non-recursive noun-phrase appearing in the text. In the previous example, *AlHrb AlEAlmyp AlvAnyp*, "the Second World War", is a noun-phrase, hence matches as a gap in the MWE *wDEt AlHrb AlEAlmyp AlvAnyp <wzArhA*, "the war is over". Currently, only noun-phrases are considered as potential IFs; other phrase types need to be considered in the future.

The output of the pattern-matching algorithm is an enriched version of the input AMIRA file, adding the IOB MWE label tags. Figure 6-1 shows an example for a complete annotated sentence.

Word	**Lemma**	**BPC**	**RTS**	**ERTS**	**3-gram Prefix**	**3-gram Suffix**	**NER**	**Label**
nftH	fataH-a	B-VP	VBP	VBPP1+S	nft	ftH	O	B-MWE-1
mE	maEa	B-NP	NN	NN	mE	mE	O	O
<yrAn	>yraAn	B-NP	NNP	NNP	<yr	rAn	B-GPE	O
SfHp	SafoHap	B-NP	NN	NNFS	SfH	fHp	O	B-MWE-2
jdydp	jadiyd	I-NP	JJ	JJFS	jdy	ydp	O	I-MWE-2

FIGURE 6-1 – Annotation and features: a complete sentence example.

6.4 Supervised Framework

For IOB classification, we designed a sequential classifier running on top of the WEKA platform that is capable of processing texts annotated with the IOB notation, similar to YamCha (Kudo and Matsumoto, 2003). We use LibSVM, an implementation of support vector machines, as the underlying technology, with degree 2 polynomial kernels. We use features from a window-based context; different window sizes were tested, ranging from −/+3 to −/+5 tokens before and after the token of interest. For every word, we consider the labels of the 3-5 previous words in addition to the features we

extract from the context words.

Recall that Arabic MWEs may be morphologically and syntactically modified to agree with its surrounding context. Hence, we exploit some morpho-syntactic features and examine their contribution to the overall classification results. Those features are discovered by AMIRA 2.0. Recall that AMIRA combines tokenization and part-of-speech output with morphological analyses provided by SAMA 3.1. AMIRA 2.0 is also enriched with named-entity-recognition class tags provided by Benajiba et al. (2008). For every word, AMIRA 2.0 is capable of identifying the context-sensitive clitics, diacritized lemma, stem, full POS tag excluding case and mood, base-phrase chunks, and named-entity-recognition tags. We experiment with two different types of POS tag-sets: 1) Reduced Tag Set (RTS), distributed with the Arabic Treebank (ATB) (Maamouri et al., 2004); and 2) Enriched RTS (ERTS) (Diab, 2007), explicitly encodes definiteness, number, and gender information increasing the number of tags from 25 in RTS to 75 tags. Table 6-4 summarizes the entire set of features we use in our classification platform.

Feature	***Description***
word token	the surface form of the token of interest
lemma	the token's derived lemma
ERTS POS tag	enriched POS tag
RTS POS tag	reduced POS tag, casted from the ERTS version
(1-3)-gram prefix / suffix	first and last 1-3 characters of a token, as means of capturing the word inflectional and derivational morphology
(2-4)-gram language model	language-model log-probabilities calculated over sequences of 2-4 words, starting at the token of interest
named-entity-recognition tag	using IOB notation

TABLE 6-4 – The list of features employed.

The *n*-gram language model was trained with SRILM (Stolcke, 2002), over 30 million words from Arabic Gigaword (4th ed.), aiming to capture possible relations between MWEs and the frequency of their individual words co-occurrence. Our data sets are tokenized with the D3 Arabic tokenization standard (Habash et al., 2009).

6.5 Generating Annotated Data Sets

We manually annotated a relatively small data set of 13K tokens,[11] referred to as GOLD, corresponding to 140 sentences extracted from Arabic Gigaword (4th ed.), with gold MWE boundaries and syntactic construction types. Overall, we annotated 304 MWE types that breakdown into 102 VNCs, 100 NNCs, 42 V+PPs, and other MWE types.

Additionally, we automatically generated MWE tagged data, referred to as NOISY, by running our deterministic pattern-matching algorithm on parts of Arabic Gigaword. Obviously, as opposed to the manual annotation, the annotation created by the pattern-matching algorithm does not distinguish between literal and idiomatic occurrences of the collected MWEs. Furthermore, it contains only MWEs existed in the repository.

	Precision	***Recall***	$F_{\beta=1}$
All	***95.6***	***44.7***	***70.1***
VNC	97.2	42.1	69.6
NNC	100	46.0	73,0
V+PP	89.4	91.4	90.4
PNC	100	6.25	53.1

TABLE 6-5 – Evaluation results obtained by our deterministic algorithm.

Therefore, we evaluated the quality of the deterministic tagging system against the manually annotated data. The results are presented in Table 6-5 focusing on the most prominent syntactic construction types, while the first line aggregates the results of all classes. We clearly see that while the precision values are relatively high, the recall is low for all classes, excluding V+PP. The relatively high precision scores enable us to use the automatically tagged data as additional training data in our supervised framework.

6.6 Experimental Approach

Below is the lineup of experiments we followed in this work:

1. Determining the best feature setting;
2. measuring the contribution of the NOISY data to the identification subtask, and finding the optimal amount to be used;

[11] To get a direct access, please go to http://cs.tau.ac.il/~kfirbar

3. comparing the accuracy results of a classifier running under the two tagging conditions: CASCADED and INTEGRATED.

6.6.1 Determining the Best Feature Setting

We begin by selecting the best feature setting for the identification subtask. In order to deal with accuracy issues related to the automatically determined linguistic features, we derived a data set from the Arabic Treebank (ATB), where linguistic features are manually assigned for every word. Then, we use the deterministic pattern-matching algorithm to generate the MWE boundary labels. Our ATB data set contains 22K words, corresponding to 490 sentences, and divided into 80% for training, 10% for testing, and 10% for development. We experiment with different window sizes and feature settings, and realized that a −/+3 window achieves the best results; hence we use it in the subsequent experiments.

Table 6-6 shows the results of using different types of part-of-speech tag sets in the identification subtask. FULL refers to the most detailed tag set used in the ATB, specifically by SAMA, the ATB Arabic morphological analyser. FULL consists of a large number of tags (∼450) for capturing detailed morphological and syntactic attributes including case and mood. The FULL tag set is not supported by the automatic part-of-speech tagger AMIRA 2.0 used in our work, hence cannot be considered in our next set of experiments.

Interestingly, the results slightly decrease as the amount of information encoded in the tag set increases. We note that the best results are obtained by RTS. This suggests that MWE words are frequently modified to agree with surrounding contexts resulting in significant data sparseness.

POS set	***Precision***	***Recall***	$F_{\beta=1}$
FULL	94.4	42.5	68.4
ERTS	94.7	45.0	69.8
RTS	100	46.6	73.3

TABLE 6-6 – Results of running with different types of POS tag sets on the ATB data set.

6.6.2 Determining the Optimal Amount of Augmenting NOISY Data

We are interested in identifying how much training data is needed to improve the identification results. We augment the manually annotated data (GOLD) with portions of the automatically tagged data (NOISY) in various increments, and measure the impact on MWE identification subtask. GOLD was divided into training (80%), test (10%), and development (10%) sets. We ran several experiments incrementally increasing the size of the training data by augmenting it using portions of the NOISY data. All of the experiments were evaluated on the test set.

Figure 6-2 shows two curves of *F* measure with regard to the increasing size of the augmenting NOISY data. One algorithm curve represents our boundary detection classifier, using RTS and the rest of the features from Table 6-4, extracted from a -/+3 context window. We evaluate our classifier performance vis-à-vis a baseline algorithm that simply assigns a word its most frequent label, as observed in training regardless of context, considering the labels of the previous word.

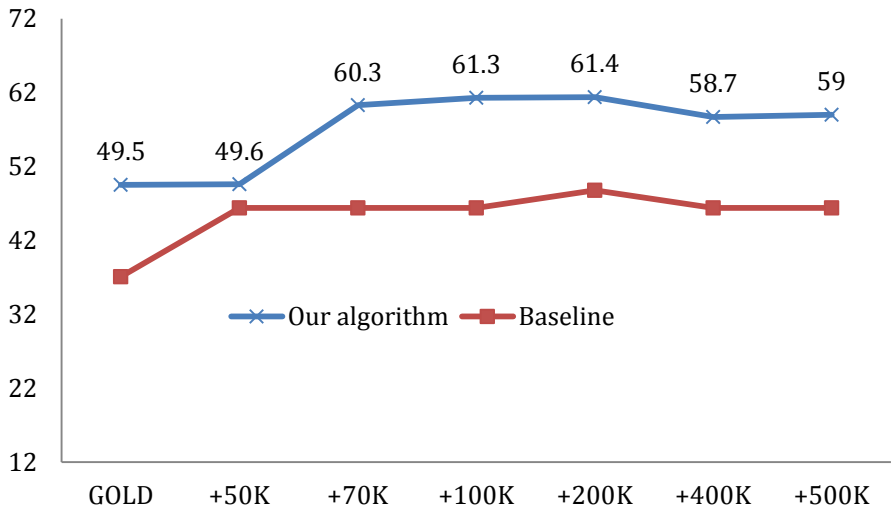


FIGURE 6-2 – Comparing our system's *F* measure with the baseline, focusing specifically on the subtask of MWE identification.

Overall, we see improvements over the baseline throughout all sizes of NOISY data. Moreover, for the most part the score increases as the size of the augmenting data set increases up to 200K, then we see a plateau effect in larger data sizes. Table 6-7 shows the detailed results. The results indicate that the drop in *F*-measure scores are due a decrease in precision rather than recall. This supports our hypothesis that with more data, even if relatively noisy, our classifier is able to discover additional, unseen MWE

tokens. In order to confirm this, we see that in the baseline, the recall did not improve, suggesting significant generalization power for our classifier, learned from the contextual features.

Upon qualitative error analysis on the data, although this was specifically tested in the identification subtask, we find that the classifier does best on VNCs, followed by NNCs with the weakest performance being on the V+PPs, which is an interesting complement to the deterministic algorithm. For instance, in the +200K classifier condition, the system yields 83.3%, 80%, 77.7% recall for VNCs, NNCs, and V+PPs, respectively.

	Precision	***Recall***	$F_{\beta=1}$
GOLD	73.1 (53.3)	26.0 (21.0)	49.5 (37.1)
+50K	69.2 (69.2)	30.0 (23.7)	49.6 (46.4)
+70K	85.7 (69.2)	35.0 (23.7)	60.3 (46.4)
+100K	86.0 (69.2)	36.7 (23.7)	61.3 (46.4)
+200K	**83.3 (71.4)**	**39.5 (26.3)**	**61.4 (48.8)**
+400K	78.0 (69.2)	39.5 (23.7)	58.7 (46.4)
+500K	77.0 (69.2)	41.1 (23.7)	59.0 (46.4)

TABLE 6-7 – Results of augmenting with varying sizes of NOISY data, on the test set (numbers in parentheses are the baseline results).

6.6.3 MWE Classification

In this section, we address the classification task, that is, finding the construction type of the MWEs in free text. As noted before, we trained our classifier in two tagging conditions: CASCADED and INTEGRATED.

In the CASCADED condition, we follow the next two steps: (1) run the boundary-detection classifier, trained on the optimal feature setting discovered using RTS and a context window of size –/+3 for uncovering boundaries of MWEs; and (2) use the uncovered boundaries as features, combined with the optimal feature setting, to train a classifier for finding the construction type of every MWE, on the token level.

Recall that the construction type of every MWE was mentioned next to every annotated MWE. We begin with the second step, independently, using the MWE boundaries that were determined upon annotation, as features. In Table 6-8 we provide the results obtained only on GOLD part that was allocated for training, and then augmented with

200K of NOISY data, the optimal amount from the previous experiment. The classifier is tested on the same GOLD test set as before.

Among the three constructions we were experimenting with, V+PPs were easier to predict. Surprisingly, we see that the results decline when we added the 200K of noisy words to the training set. Upon deeper error analysis on the data, we found that most of the errors occur due part-of-speech tagging errors. Since the deterministic pattern-matching algorithm uses part-of-speech information to find MWEs, the annotated MWE words are tagged with the correct part-of-speech tag. Therefore, with the 200K of noisy annotated words, the training set becomes more homogenous with respect to the relation between MWE word and its assigned part-of-speech tag, thus the capability of finding the correct construction type of a MWE word that was assigned with the wrong part-of-speech tag by AMIRA 2.0, is damaged.

Data set	***Class***	***Precision***	***Recall***	$F_{\beta=1}$
GOLD	NNC	78.8	89.7	84.2
	VNC	88.2	78.9	83.5
	V+PP	**85.7**	**100**	**92.8**
+200K	NNC	71.4	66.7	62.0
	VNC	66.7	50.0	58.5
	V+PP	75.0	100	87.5

TABLE 6-8 – Results of running a MWE classifier using gold MWE boundaries as features (i.e., the second step of the CASCADED condition).

Generally speaking, we learn that finding the correct syntactic construction class of a MWE, provided with its correct boundaries, is relatively easy. In the following experiments we use only GOLD data.

To complete our experiment under the CASCADED condition, we perform both steps sequentially, using the boundary detection classifier followed by the syntactic-construction type classifier. The results are presented in Table 6-9.

The second tagging condition, INTEGRATED, is where there are no explicit boundaries modeled in the feature set. In fact, the classifier uses 13 tag labels, as mentioned above, integrating the boundary detection and classification subtasks into the same workflow.

Table 6-9 shows the results of INTEGRATED vis-à-vis CASCADED. As before, we compare the results with those of a baseline algorithm that simply assigns a word its most frequent label as observed in training regardless of context. The three classifiers were trained with GOLD training set and evaluated on the GOLD testing set only. From the results it is clearly that INTEGRATED outperforms CASCADED on the VNC and V+PP types, but on the other hand, CASCADED shows a slightly better precision over INTEGRATED on the NNC category.

Classifier	***Class***	***Precision***	***Recall***	$F_{\beta=1}$
INTEGRATED	NNC	50.0	13.8	31.9
	VNC	**83.3**	**41.6**	**62.4**
	V+PP	**100**	**55.5**	**77.0**
CASCADED	**NNC**	**66.6**	**13.3**	**39.9**
	VNC	66.6	16.7	41.6
	V+PP	75.0	33.3	54.1
Baseline	NNC	40.0	13.3	26.6
	VNC	50.0	16.7	33.3
	V+PP	66.7	44.4	55.0

TABLE 6-9 – Results of INTEGRATED, CASCADED, and a baseline classifiers, trained on the GOLD training set.

The relatively low recall obtained by all the three classifiers on NNCs, implying that NNCs are more difficult to identify in running text, despite the fact that Arabic NNCs are hardly occur with IFs in our corpus. By taking a closer look at both the confusion matrix and the results of INTEGRATED, we see NNC instances that were incorrectly labeled with VNC. Some of those instances were misclassified by the part-of-speech and base-phrase taggers, for example: *wjE AldmAg*, "bothersome", where the lack of short vowels causes POS-related ambiguity on the first word *wjE*, which may be interpreted as a noun, "pain", as well as a verb, "feel pain; hurt". That said, there are cases of missed NNCs that were correctly tagged by the part-of-speech tagger. One possibility for those misses is the variety of NNCs, which cannot be captured merely by morpho-syntactic features. CASCADED obtained better precision on NNCs; however, by looking at the results, we noticed that some of the MWEs that INTEGRATED incorrectly classified as

NNC, were not captured by the boundary detection classifier running in step 1 of CASCADED. This implies that the improvement in precision is not significant.

The last experiment suggests that for the most part, it is better to use a token-level boundary detection classifier trained on each construction type individually, as we did in our INTEGRATED framework.

6.7 Summary

The method suggested in this chapter has demonstrated the potential of augmenting manually annotated data with automatically obtained data for dealing with Arabic MWE boundary detection on the token level. On the other hand, as we have seen, the same noisy augmenting data was counterproductive for the classification subtask, given the MWE boundaries as features. To address the task of finding MWEs in running text, we combined two subtasks, identification and classification, into a single process. We have done experiments under two tagging conditions: one performs each subtask individually by feeding the classification algorithm with the boundary labels, uncovered by a boundary detection classifier; and another condition that was proved to be more effective, in which we trained a boundary detection classifier on each construction type individually.

As we have observed, encoding additional morphological information in part-of-speech tags was not found to be effective, probably due to data sparseness. Learning the individual contribution of each morphological feature to the detection and classification subtasks is definitely a direction for future investigation. Furthermore, we plan to investigate each construction type individually, using additional feature types.

SVM with its generalization property was a natural option for dealing with combination of features. We believe that other technologies should be examined as well.

This chapter describes an initial study on Arabic expressions identification and classification.

Chapter 7

Conclusions

7 Conclusions

In this chapter, we summarize our results and arrive at some conclusions. Generally speaking, the main focus of this work has been on improving a system that translates a highly inflected language into English by using semantic equivalents of fragments of the input text. We chose to work with Arabic, mainly due to the availability of resources and tools that are relevant for pursuing our research goals.

We proceed as follows: In Section 7.1 we summarize our work and provide some conclusions, and in Section 7.2 we suggest some future directions.

7.1 Summary and Conclusions

We began our investigations with a case study of automatic extraction of nominal synonyms from dictionary glosses. We took the Arabic dictionary derived from the English glosses of Arabic stems provided by Buckwalter's analyser (BAMA 1.0). Then, we matched stems on the gloss level, resolving their senses using WordNet, and defined several levels of similarity to differentiate between stems that match on the sense level and stems that merely share some of their glosses. Overall, we found over 200K pairs corresponding to about 20K noun stems, which we used to improve an Arabic-to-English translation system, considering different similarity levels. For simplicity, we employed our own implementation of example-based machine translation (Bar et al., 2007) and modified its matching component to consider synonymous words when searching for translation examples in the example repository. Although the BLEU scores were relatively low, primarily because of the limited implementation, the results were encouraging enough to proceed. Moreover, we observed that the synonyms benefit from being matched carefully by considering the topic of the sentence in which they appear, suggesting that considering the context so as to properly match the true senses of ambiguous synonyms is an important direction for future investigation.

For the next stage of the research, we decided to explore alternative paraphrasing approaches that can derive new synonyms from text corpora rather than from a dictionary, this time focusing on Arabic verbs. We designed a simple maximum-likelihood-estimation classifier that works on the context level of potential candidates. In particular, we used a relatively small corpus of comparable documents and automatically annotated some positive and negative examples for training our simple context classifi-

er. In this sense, a positive example is a pair of verbs, each taken from one document of a comparable pair, which are considered synonyms within their contexts. We took a deterministic approach in which every pair of verbs that match on the lemma level, is deemed positive example. The rationale behind this is based on the fact that every such a pair is composed of two verbs extracted from comparable documents; hence they are likely to carry the same sense in their corresponding contexts. Negative pairs were automatically generated based on a list of all potential verbal synonyms, which was created by a similar technique that we used to extract the list of nominal synonyms. Essentially, pairs that were not on that list were annotated as negative examples. To model the context, we extracted some linguistic features, such as part-of-speech tags and words lemmas, from the surrounding words.

Our experiments showed that the classifier performs pretty well on new pairs, in terms of precision. Recall was not calculated; however, upon qualitative evaluation of the results we found synonyms that our classifier could not capture. Similarly to the noun experiment, we used the extracted synonymous verbs to improve the automatic translation process, implemented by our own example-based system, and evaluated the results. Given that our implementation is lacking the recombination component, an important tool for generating the correct translation, we decided to abandon the automatic evaluation of the translation quality and carried out a manual evaluation of translation examples where one of its words was matched on the synonym level. Similarly to our previous conclusions, we learned that matching with synonyms may improve translation quality as long as correct examples are chosen by a well-implemented recombination component.

Experimenting with synonyms over our example-based implementation encouraged us to move on and work on a broader technique to discover larger (multiword) paraphrases, and apply them to a more mature corpus-based machine-translation implementation, such as Moses (Koehn et al., 2006), a well-known implementation for a phrase-based machine translation.

Since bilingual parallel corpora, which pair Arabic with other languages other than English, are difficult to obtain, we postponed the investigation of using techniques that are based on such a resource for generating Arabic paraphrases. Monolingual parallel corpora are considered a good resource for paraphrasing; however, we could not find one for Arabic. So, we continued our investigation with a corpus of comparable documents and then simplified it to work on a relatively large monolingual corpus, enabling

paraphrases to be generated at a larger scale to improve the results of a statistical translation system.

We abandoned the approach proposed by others to use comparable corpora, which first finds comparable sentences in the comparable documents and then applies some kind of a word aligner on them to locate aligned equivalent parts. The main reason for not following this technique is the relatively large amount of data that is lost this way, not being considered for finding paraphrases. Instead, we decided to consider almost every pair of phrases, with each phrase taken from one document of a comparable pair. On one hand, it may increase the coverage of the algorithm, but at the same time it might be detrimental to precision.

We took a machine-learning approach using two classifiers, each trained on a different perspective of the same training set. This approach is also known as "co-training" (Blum and Mitchell, 1998). In other words, each classifier was trained on a different set of features extracted from the same training set. Preferably, one should choose the feature set so that each could have been assigned individually to handle the classification problem. To generate annotated examples for training the classifiers, we adopted a similar approach to the one we took in the verb experiment; that is, every pair of two similar phrases, each taken from one document of a pair of comparable documents, is annotated as a positive example. In accordance with the verb experiment, similarity between phrases was determined by matching word by word on the lemma level, because Arabic is highly inflected. Here, working with lemmas was a natural choice for us, as we wanted to generate as many positive examples as possible by matching for instance, perfective and imperfective forms of the same verb (the lemma is insensitive to the verb form). On the other hand, working with lemmas brought about phrase pairs reported as paraphrases unintentionally. The main reason for that is the insensitivity of lemmas to various inflections. Recall our modified definition for Arabic paraphrases to include pairs of phrases that express the same meaning in at least one context, regardless of their inflection for number, gender, and person. However, in addition to inflections, Arabic lemmas hide possessive attachments, the definite article, conjunctions, prepositions and direct objects attached to verbs. Therefore, even with the modified definition, some incorrect phrase pairs get reported; for example, the phrase *byth*, "his house", may be reported as a paraphrase of *Albyt*, "the house", incorrectly.

Knowing that the phrase pairs were extracted from comparable documents, we have reason to believe that such cases do not happen very often, and propose to live with a

small number of incorrect examples in our training set. In the evaluation process, we did not account such pairs as paraphrases, but only as *semantically related* phrases, as defined in Chapter 4.

Negative examples, also generated automatically, are pairs of phrases that are unlikely to be accounted as semantically equivalent, decided based on some morpho-syntactic elements.

Paraphrases were generated using the co-classifiers, with one working on the local context of the phrases, and another one working on the phrase words themselves, increasing the length of a potential paraphrase in every iteration. Both classifiers worked on features that were extracted from the words, which mostly reflected morphological and syntactical characteristics. The classifier that worked on the phrase words was fed primarily with features that express the existence of morphological elements in every phrase word, for example, the definite article, conjunctions, and various prepositions.

Based on a manual evaluation, performed by two Arabic speakers, our paraphrasing technique was found to be effective in terms of precision. We demonstrated the strength of the co-training algorithm, compared to using a single classifier under the same settings. Moreover, we found that the morphological features used by the phrase classifier were important for this task; however, we have not yet investigated each morphological element individually.

Encouraged by the results, we began to plan experiments for learning whether those derived paraphrases can actually help improve a corpus-based translation system. One drawback of our paraphrasing technique is its ability to work at a large scale by generating a relatively large number of paraphrases. The main resource that is used by our paraphrasing algorithm is the corpus of comparable documents that we have created automatically from a large collection of monolingual documents. As we followed a very simple technique in the generation of the corpus, we ended up with only a few comparable-document pairs. Had we succeeded to increase the size of the comparable documents, thereby enabling a broader coverage of our paraphrasing algorithm, it would probably be able to generate more paraphrase pairs. However, we still would encounter a time resource issue that relates to the fact that our algorithm uses a quadratic-kernel SVM as the machinery for both classifiers.

For those reasons, we simplified our paraphrasing technique so that we can use it to generate paraphrases for phrases of a sentence that are given for translation in a rea-

sonable amount of time. Viewed from a high level, we modified two main elements in the original technique: (1) we use only one classifier, the context classifier, which we train on the same training set extracted from the corpus of comparable document, similarly to the previous training procedure; and (2) paraphrases are now generated from a relatively large monolingual corpus.

Essentially, with this new paraphrasing technique, we used a classifier that merely models the context of two paraphrases, which we trained on positive and negative examples extracted from the corpus of comparable documents. In particular, given a phrase for paraphrasing, the new algorithm begins by searching for paraphrase candidates in the monolingual corpus and then applies the classifier on them for vetting their equivalency. Since the context classifier takes some time to run on a single candidate, applying the context classifier on every phrase from the monolingual corpus is infeasible. For simplicity, we limited the candidates to be those phrases that have some percentage of lemmas in common with the input phrase, in respect to the length of the longest phrase. That way, we were able to build a lemma index over the entire monolingual corpus, thereby enabling a relatively fast lookup. On the other hand, this heuristic prevents the algorithm from generating semantic paraphrases that do not share lemmas in common with the input phrase. Finding a better way to extract candidate paraphrases from the large monolingual corpus, such as using synonyms and just similar lemmas, was left for a future investigation.

Among the features we used for training the context classifier, we included the cosine-similarity score calculated on vectors of *tf-idf* multiplied the by the PMI of the contextual lemmas of the two phrases. Named entities were replaced with a generic label corresponding to their entity type, as discovered by AMIRA 2.0 (Diab et al., 2009).

Before using this paraphrasing algorithm on translation, we automatically evaluated the classifier by calculating its precision and recall over a 10-fold cross-validation framework. The evaluation resulted in 85% precision and 79% recall; however, since the evaluation was performed on the automatically annotated examples, it means that the classifier was only tested on paraphrases that were composed of two similar phrases, word by word, on the lemma level. The classifier was not evaluated on other types of paraphrases, since we did not have such annotated pairs. Instead of proceeding with a manual evaluation process, we decided on trying to use paraphrases in translation and evaluate their contribution to the translation process.

We used a Moses implementation of Arabic-to-English phrase-based machine translation, and measured the effect of using the paraphrasing algorithm for deriving paraphrases potentially for every phrase of an input sentence. We decided to allow every input phrase to be paraphrased, regardless of whether it is "seen", that is, existed in the original phrase table, or not. For that reason, we formatted the input sentence as a word lattice (Dyer et al., 2008) and augmented every phrase with its paraphrases as derived by our paraphrasing algorithm. Each paraphrase was assigned with a score reflecting its level of equivalence to the original phrase.

A BLEU-based evaluation showed that paraphrases do in fact help improve the quality of the resulted translations, compared to a baseline system that does not use paraphrases. We discovered that the decoding process benefits from the scores assigned for every input paraphrase, especially when a tuning process, executed by Minimum Error Rate Training (MERT), takes place. We experimented with a system that was fed with different sizes of bilingual corpora and learned that paraphrases contribute less when a relatively large corpus is used, an observation that was made also in other works on the same topic.

By way of summary, we wish to conclude the following, based on our experiments:

- synonyms and paraphrases may help improve the quality of a corpus-based translation system;
- when the size of the bilingual corpus used by the translation system grows larger, the contribution of the paraphrases decreases to the point of being counterproductive;
- using scores that reflect the equivalence level of the paraphrases to the original phrase helps improve the translation quality;
- the language-model scores that we combined with the equivalence scores help improve the results;
- a MERT-based tuning process to adjust the parameters of the feature functions, including the one that represents the equivalence score, helped to improve the results;
- co-training was found to be effective for dealing with the paraphrasing task, applied on a corpus of comparable documents;

- morphological features help improve the paraphrasing performance for a highly inflected language.

One chapter deals with multiword expressions, a topic closely related to paraphrasing. It describes our initial study of Arabic multiword expression (while visiting the Center for Computational Learning System). We began by creating a repository of Arabic multiword expressions, comprising fewer than 5,000 expressions that were collected manually from various dictionaries. We proceeded with training a classifier for identifying expressions in running Arabic text, adopting the Inside Outside Beginning (IOB) notation scheme. For this purpose we manually annotated a relatively small amount of examples and augmented them with automatically annotated examples generated by a deterministic pattern-matching algorithm that looks for expressions from the repository in the text. The results, evaluated by precision and recall, were relatively good, suggesting that using the automatically annotated data actually helped to improve the identification performance. We also tried to work on each expression type (e.g., noun-noun constructions, verb-noun constructions) individually, and found that the same augmenting set did not help improve the results as before.

Identifying multiword expressions is related to paraphrasing in the sense that idiomatic expressions typically get paraphrased in various ways. We believe that combining both research directions may result in improving the performance in both tasks.

7.2 Further Discussion and Future Work

In this section we lay out some future directions regarding the research topics that were covered in this work.

Our main plan is to further explore the area of using semantic tools for supporting a corpus-based translation process. We plan to keep our focus on morphologically rich languages, which are considered more challenging than other languages in terms of corpus-based processing, especially due to the coverage problem they introduce. Consequently, working with another highly inflected language, such as Hebrew, is a direction for future investigation. We are aware of several related works that deal with Hebrew, but there is very little on paraphrasing. In fact, the only one we encountered is the recent work by Stanvosky (2012), which describes a study in Hebrew paraphrasing, considering deep syntactic structures for similarity calculations. Hebrew multiword

expressions have received more attention than paraphrases, especially in the works of Tsvetkov and Wintner (2011; 2012), which we have already mentioned, and Al-Haj and Wintner (2010).

As we have seen, one main characteristic of our paraphrasing technique as well as other techniques that we explored is the fact that many pairs that are deemed paraphrases by our algorithm, are actually composed of two phrases that relate somehow in the semantic space, rather than real paraphrases. Such pairs, although usually assigned with a lower score than real paraphrases, may eventually participate in the final translation, thereby defacing its quality. The type of those relations varies from the simple unidirectional-entailment relation, through metonymy, to antonym. Some of those relations may still be used in translation, especially when there is no other translation for the input phrases, hence we wish to explore clever ways to re-rank the returned paraphrases for disqualifying cases like antonyms and keeping those paraphrases, which may result in a reasonable and actionable translation.

Mirkin et al. (2009) have been dealing with the problem of using entailment rules for translating unseen phrases. They used simple entailment rules based on synonyms and hypernyms that were retrieved from WordNet 3.0. The entailment rules were applied only on unseen phrases in a way that the Right-Hand Side (RHS) of a rule replaced its Left-Hand Side (LHS), which was essentially the source phrase itself. The replacement took place only if the RHS phrase was found in the system's phrase-table. They assigned a score for every replacement using several scoring models, such as language modeling and Latent Semantic Analysis (LSA) (Deerwester et al., 1990). Upon decoding an unseen phrase, their translation system was designed to try applying entailment rules that merely replace synonyms, those are, paraphrasing rules. Only if none of those rules was actually applied, the system continued to try executing the unidirectional entailment rules, which were set for replacing source words with their hypernyms. They performed a manual evaluation, which clearly showed an improvement of the system that used the entailment rules over a baseline system.

In fact, we used similar techniques in our experiments that use Arabic synonyms in translations. In our framework we did not use the Arabic WordNet mainly due its coverage limitation, a common case when working with a resource similar to WordNet for other languages than English and some of the main European languages. Instead, we extracted synonyms from a dictionary and a corpus of comparable documents, as described in Chapters 2 and 3 respectively. We wish to explore ways for considering

Arabic hypernyms in translation as well as other entailing phrases, which may be discovered by a more complex system for deriving entailment relations from a large Arabic textual corpus.

We found that paraphrases benefit from being used in a relevant context. Intuitively, as long as the paraphrase is, its semantic-ambiguity level is less pronounced; for example, a single word is probably more semantically ambiguous than a bigram. Therefore, we believe that the relevancy of the context to which a paraphrase is applied, should be measured according to the length of the paraphrase. In Chapter 2, where we merely used synonyms, we found that comparing the general topics of the input sentence and the translation example helped to resolve the semantic disambiguation. When we used longer paraphrases, as in Chapter 5, we used language-model scores in addition to the equivalence score and found that they actually helped to improve the translation quality. This finding suggests that the context should be carefully considered when injecting a paraphrase to the input sentence; while single-word synonyms benefit more from thematic level, longer paraphrases help more when they keep the sentence grammatically correct. Dagan et al. (2006) focused on word-sense matching for lexical substitution, by avoiding a prior usage of algorithms for Word-Sense Disambiguation (WSD). Generally speaking, they employed a classifier that uses unigrams, bigrams and words' part-of-speech tags, extracted from the near environment of the focused synonymous words. They took both supervised as well as unsupervised approaches, and reported on encouraging results. Szpektor at el. (2008) generalized the latter approach by proposing a framework, which claimed to be effective, for considering the context in which inference rules, used by many other works to find (uni- as well as bi-directional) entailment relations (e.g., Lin and Pantel, 2001; Ravichandran and Hovy, 2002; Shinyama et al., 2002; Szpektor et al., 2004), are matched. Primarily, they considered two aspects to model the context for matching inference rules to a given text: (1) global, measured by a cosine-similarity score calculated over the LSA vectors of the inference-rule components and the text to which it is applied; and (2) local, measured by semantic-similarity calculations over the named-entity types of the specific variables occurred in the inference rule. Applying similar techniques for applying paraphrases to semantic lattices, which is then given for translation, is another potential direction for investigation.

Learning how Arabic multiword expressions can contribute to a corpus-based translation system is another interesting research direction. One aspect of this could be

improving the results of the word-alignment process by treating expressions as atomic units rather than potentially aligning each of their component words individually. Several works on this topic have already appeared (e.g., Pal et al., 2010, Carpuat and Diab, 2010; Okita and Way, 2011; Pal et al., 2013). However, to the best of our knowledge, none of them worked on a highly inflected source language, considering syntactical gaps, as we introduced in the previous chapter.

Although this work focuses on the ability of paraphrases to improve machine translation, exploring the potential of paraphrasing to assist other NLP-related systems, which use text as input and text as output, is another desideratum. Using paraphrases for improving question answering (QA) has already been addressed and it is still a live topic of research (e.g., McKeown, 1983; Takahashi et al., 2003; Duclaye et al., 2003; Fader et al., 2013). However, we could not find works in that area that focus on highly inflected languages such as Arabic. Other applications, which may be considered for paraphrasing, are text summarization and text generation.

Appendix A

Examples of Arabic pairs derived by the co-training algorithm from Chapter 4, provided with the evaluation score they were assigned with by the evaluators.

Alr}ys AlflsTyny, "the Palestinian president" ⇔ *AlslTp AlwTnyp AlflsTynyp,* "the Palestinian authority"	Related
jwrj wwkr bw$,* "George Walker Bush" ⇔ *jwrj bw$, "George Bush"	Paraphrases
Alm&tmr AlsAds, "the Sixth conference" ⇔ *AlAjtmAE AlwzAry AlsAds*, "the Sixth ministerial meeting"	Paraphrases
dAnyyl jlAzr, "Daniel Glaser" ⇔ *dAny}l glAsr*, "Daniel Glaser"	Paraphrases
kyly gwnzAlyz wAngwlw, "Kaylie Gonzales and Angelo" ⇨ *AlArjntynyyn AlxTyryn*, "the dangerous Argentinians"	Unidirectional entailment
AlbrlmAn Aljdyd, "the new Parliament" ⇔ *Almjls AlwTny AlsAbE E$r*, "the Seventeenth Parliament"	Paraphrases
AlHdwd Alswryp AllbnAnyp, "the Syrian-Lebanese borders" ⇔ *AlHdwd Alswryp,* "the Syrian border"	Unidirectional entailment

Examples of English pairs derived by the co-training algorithm from Chapter 4, provided with the evaluation score they were assigned with by the evaluators.

could veto ⇔ threatened to veto	Related
the U.S. Naval Task Force ⇔ a US Naval Task Group	Paraphrases
Beijing's policy ⇔ the China's policy	Paraphrases
a poor and little-developed province ⇔ its resource-rich northwestern province	Wrong
U.S. beef and related products ⇨ beef products	Unidirectional entailment
a magnitude 6.0 earthquake ⇨ the quiver	Unidirectional entailment
will only endanger ⇔ will not only endanger	Wrong

Appendix B

Examples of Arabic paraphrases derived by the simplified algorithm from Chapter 5. The paraphrases are provided with their corresponding confidence score, obtained by the context classifier.

Original phrase: *Aljy$ AlAmyrky*, "the American Army"

AlqwAt AlAmyrkyp, "the American forces"	0.99
mE AlAmyrkyyn, "with the American"	0.99
Jnwd Amyrkywn, "American soldiers"	0.97
Altdxl AlAmyrky, "the American entrance"	0.92

Original phrase: *AEln*, "informed"

wAEln, "and (usually dropped in English) informed"	0.99
AElnt, "informed_*feminine*"	0.96
qd AEln, "*past-indicator* informed"	0.92
wAEln AbwAlgyT, "and Abu Algyt informed"	0.89

Original phrase: *mnZmAt Hqwq AlAnsAn*, "human-rights organization"

AlmHkmp AlAwrwbyp lHqwq AlAnsAn, "the European court for human rights"	0.98
Hqwq AlAnsAn wHryp, "human rights and freedom"	0.92
lHqwq AlAnsAn, "for human rights"	0.92
lHqwq AlAnsAn fy, "for human rights in"	0.90

Original phrase: *AlqyAdp AlEskryp Aljnwbyp*, "the southern military headquarter"

AlqyAdp AlEskryp fy Aljnwb, "the military headquarter of the	0.99

south”

AlqyAdp AlEskryp Al<srA}ylyp, “the Israeli military headquarter”	0.99
nATq blsAn AlqyAdp AlEskryp, “spokesman of the military headquarter”	0.90

Original phrase: *Alksndr lytyfnynkw*, “Alexander Litvinenko”

Alksndr lytfynynkw, “Alexander Litvininko” (different spelling)	0.99
Alksndr wEDw, “Alexander and member”	0.95
Alrwsy Alksndr, “the Russian Alexander”	0.94

Original phrase: *m&tmr SHAfy*, press Conference

tSryH SHAfy, press statement	0.99
wqAl lAlSHAfyyn, “and he said to the journalists”	0.98
lAlSHAfyyn, “to the journalists”	0.98

Original phrase: *AlmxAbrAt AlbryTAnyp*, “the British intelligence”

Aljy$ AlbryTany, “the British army”	0.96
DbAT bryTAnyyn, “British officers”	0.92
AlbryTAnyyn, “the British_*plural*”	0.92

Original phrase: *AlslTAt AlbryTAnyp*, “the British leadership”

jndy bryTAny, “a British soldier”	0.98
Aljy$ AlbryTAny, “the British army”	0.97
AlbryTAnyyn, “the British_*plural*”	0.93

Original phrase: *nATq bAsm AlxArjyp*, “a spokesman of the foreign affairs”

AlmtHdv bAsm wzArp AlxArjyp, "the spokesman of the ministry for foreign affairs"	0.99
nATq bAsm Aljy$ AlAsrA}yly, "a spokesman of the Israeli army"	0.98
Akd nATq bAsm, "a spokesman emphasized"	0.91

Original phrase: *mE jmyE*, "with all"

mE AsrA}yl, "with Israel"	0.98
mE bArAk, "with Barak"	0.98
mE swryp, "with Syria"	0.91

Original phrase: *sntyn*, "two years"

AlsnwAt AlAxyrp, "the last years"	0.99
Edp snwAt, "a period of years"	0.99
snp 2001, "year 2001"	0.96

Appendix C

Here are some examples for Arabic expressions from our manually created repository. Each expression has its own XML node, named MWE. Each node is provided with its category type and every word is provided with its BAMA analysis, manually assigned.

```
<MWE category="VNIC" id="68" text="$q Alskwn">
   <Word pos="VBD" tokenized="$q" value="$q">
      <![CDATA[diac:$aq~a lex:$aq~-u_1 bw:+$aq~/PV+a/PVSUFF_SUBJ:3MS
gloss:split;cut_through pos:verb prc3:0 prc2:0 prc1:0 prc0:0 per:3 asp:p vox:a
mod:i gen:m num:s stt:na cas:na enc0:0 rat:na source:lex stem:$aq~ stemcat:PV_V]]>
   </Word>
   <Word pos="NN" tokenized="Al+_skwn" value="Alskwn">
      <![CDATA[diac:Als~ukuwna lex:sukuwn_2 bw:Al/DET+sukuwn/NOUN+a/CASE_DEF_ACC
gloss:sukun_(Arabic_null_vowel) pos:noun prc3:0 prc2:0 prc1:0 prc0:Al_det per:na
asp:na vox:na mod:na gen:m num:s stt:d cas:a enc0:0 rat:y source:lex stem:sukuwn
stemcat:N]]>
   </Word>
</MWE>
<MWE category="VNIC" id="69" text="$q AlSf">
   <Word pos="VBD" tokenized="$q" value="$q">
      <![CDATA[diac:$aq~a lex:$aq~-u_1 bw:+$aq~/PV+a/PVSUFF_SUBJ:3MS
gloss:split;cut_through pos:verb prc3:0 prc2:0 prc1:0 prc0:0 per:3 asp:p vox:a
mod:i gen:m num:s stt:na cas:na enc0:0 rat:na source:lex stem:$aq~ stemcat:PV_V]]>
   </Word>
   <Word pos="NN" tokenized="Al+_Sf" value="AlSf">
      <![CDATA[diac:AlS~af~a lex:Saf~_1 bw:Al/DET+Saf~/NOUN+a/CASE_DEF_ACC
gloss:line;row;class pos:noun prc3:0 prc2:0 prc1:0 prc0:Al_det per:na asp:na vox:na
mod:na gen:m num:s stt:d cas:a enc0:0 rat:y source:lex stem:Saf~ stemcat:Ndu]]>
   </Word>
</MWE>
<MWE category="NNC" id="90" text="$hr Esl">
   <Word pos="NN" tokenized="$hr" value="$hr">
      <![CDATA[diac:$ahoru lex:$ahor_1 bw:+$ahor/NOUN+u/CASE_DEF_NOM gloss:month
pos:noun prc3:0 prc2:0 prc1:0 prc0:0 per:na asp:na vox:na mod:na gen:m num:s stt:c
cas:n enc0:0 rat:y source:lex stem:$ahor stemcat:Ndu]]>
   </Word>
   <Word pos="NN" tokenized="Esl" value="Esl">
   <![CDATA[diac:EasalK lex:Easal_1 bw:+Easal/NOUN+K/CASE_INDEF_GEN gloss:honey
pos:noun prc3:0 prc2:0 prc1:0 prc0:0 per:na asp:na vox:na mod:na gen:m num:s stt:i
cas:g enc0:0 rat:y source:lex stem:Easal stemcat:N]]>
   </Word>
</MWE>
<MWE category="VNC" id="100" text="*Ab qlb (flAn)">
```

```
   <Word pos="VBD" tokenized="*Ab" value="*Ab">

      <![CDATA[diac:*Aba lex:*Ab-u_1 bw:+*Ab/PV+a/PVSUFF_SUBJ:3MS
gloss:be_dissolved;be_melted;dwindle pos:verb prc3:0 prc2:0 prc1:0 prc0:0 per:3
asp:p vox:a mod:i gen:m num:s stt:na cas:na enc0:0 rat:na source:lex stem:*Ab stem-
cat:PV_V_intr]]>

   </Word>

   <Word pos="NN" tokenized="qlb" value="qlb">

      <![CDATA[diac:qalobu lex:qalob_2 bw:+qalob/NOUN+u/CASE_DEF_NOM
gloss:reversal;inversion pos:noun prc3:0 prc2:0 prc1:0 prc0:0 per:na asp:na vox:na
mod:na gen:m num:s stt:c cas:n enc0:0 rat:y source:lex stem:qalob stemcat:N]]>

   </Word>

   <Word pos="NN" tokenized="flAn" value="flAn" variable="true">

      <![CDATA[diac:fulAnK lex:fulAn_1 bw:+fulAn/NOUN+K/CASE_INDEF_GEN gloss:so-
and-so;such-and-such pos:noun prc3:0 prc2:0 prc1:0 prc0:0 per:na asp:na vox:na
mod:na gen:m num:s stt:i cas:g enc0:0 rat:y source:lex stem:fulAn stemcat:N-ap]]>

   </Word>

</MWE>
```

Bibliography

Abou Saad, Ahmed. 1987. A Dictionary of Arabic Idiomatic Expressions (mu'jm al-trakib wala'barat alastlahiah ala'rbiah alkdim mnha walmould). *Dar El Ilm Lilmalayin*.

Agresti, Alan. 1990. Categorical Data Analysis. *John Wiley & Sons*, New York, NY.

Al-Hag, Hassan and Alon Lavie. 2010. The Impact of Arabic Morphological Segmentation on Broad-Coverage English-to-Arabic Statistical Machine Translation. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)*, Denver, CO.

Al-Haj, Hassan and Shuly Wintner. Identifying Multi-word Expressions by Leveraging Morphological and Syntactic Idiosyncrasy. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 10-18, Beijing, China.

Attia, Mohammed, Antonio Toral, Lamia Tounsi, Pavel Pecina and Josef van Genabith. 2010. Automatic Extraction of Arabic Multiword Expressions. In *Proceedings of the 7th Conference on Language Resources and Evaluation, LREC-2010*, Valletta, Malta.

Baldwin, Timothy, Collin Bannard, Takakki Tanaka, and Dominic Widdows. 2003. An Empirical Model of Multiword Expression Decomposability. In *Proceedings of the ACL 2003 workshop on Multiword expressions*, pages 89–96, Morristown, NJ, USA.

Banerjee, Satanjeev and Alon Lavie. 2005. Meteor: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the 43rd Annual ACL Meeting. Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization*, pages 65-72, Ann Arbor, MI.

Bannard, Colin and Chris Callison-Burch. 2005. Paraphrasing with Bilingual Parallel Corpora. In *Proceedings of the Annual Meeting of Association of Computational Linguistics (ACL 2005)*, pages 597-604, Ann Arbor, MI.

Bar, Kfir and Nachum Dershowitz. 2010. Using Synonyms for Arabic-to-English Example-Based Translation. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas (AMTA 9)*, Denver, CO.

Bar, Kfir, Yaacov Choueka and Nachum Dershowitz. 2007. An Arabic to English Example Based Translation System. In *Proceedings of the Information and Communication Technologies International Symposium (ICTIS 2007), Workshop on Arabic natural language processing*, pages 355-359, Fez, Morocco.

Bar Hillel, Yehoshua. 1960. The Present Status of Automatic Translation of Languages. *Advances in Computers*, vol. 1, pages 91-163. Online version: http://mt-archive.info /Bar-Hillel-1960.pdf

Bar Hillel, Yehoshua .1964. Language and Information. *Addison Wesley*, *London and Jerusalem Academic Press*, pages x+388.

Barzilay, Regina and Kathleen McKeown. 2001. Extracting Paraphrases from a Parallel Corpus. In *Proceedings of the 39th Annual Meeting of Association of Computational Linguistics (ACL 2001)*, pages 50-57, Toulouse, France.

Barzilay, Regina and Lillian Lee. 2003. Learning to Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003)*, pages 16-23, Edmonton, Canada.

Benajiba, Yassine, Mona Diab and Paolo Rosso. 2008. Arabic Named Entity Recognition: An SVM-Based Approach. In *Proceedings of the Arab International Conference on Information Technology (ACIT-2008)*, Hammamet, Tunisia.

Birke, Julia and Anoop Sarkar. 2006. A Clustering Approach for Nearly Unsupervised Recognition of Nonliteral Language. In *Proceedings of the 11th Conference of the European Chapter of the Association of Computational Linguistics (EACL)*, volume 6, pages 329–336, Trento, Italy.

Black, William, Sabri Elkateb and Piek Vossen. 2006. Introducing the Arabic WordNet Project. In *Proceedings of the 3rd Global Wordnet Conference (GWC 2006)*, pages 295-299, Jeju Island, South Korea.

Blum, Avrim and Tom Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-training. In *Proceedings of 11th Annual Conference on Computational Learning Theory (COLT)*, pages 92–100, Madison, WI.

Buckwalter, Tim. 2002. Buckwalter Arabic Morphological Analyzer Version 1.0. LDC Catalog number LDC2002L49.

Brill, Eric. 1992. A Simple Rule-Based Part-of-Speech Tagger. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 112-116, San Mateo, CA.

Brown, Peter F., 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*, Vol. 6, 2, pages 79-85.

Brown, Peter F., Vincent J. Della Pietra, Stephen A. Della Pietra and Robert L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, Vol. 19, 2, pages 263-311.

Callison-Burch, Chris. 2007. Paraphrasing and Translation. PhD dissertation, *University of Edinburgh*.

Callison-Burch, Chris. 2008. Syntactic Constraints on Paraphrases Extracted from Parallel Corpora. In *Proceedings of the Conference for Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 196-205, Honolulu, Hawaii.

Callison-Burch, Chris, Philip Koehn and Miles Osborne. 2006. Improved Statistical Machine Translation Using Paraphrases. In *Proceedings of the North American Association for Computational Linguistics (NAACL 2006)*, New York City, NY.

Callison-Burch, Chris, Trevor Cohn, and Mirella Lapata. 2008. ParaMetric: An Automatic Evaluation Metric for Paraphrasing. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008),* 97-104.

Carpuat, Marine and Mona Diab. 2010. Task-Based Evaluation of Multiword Expressions: a Pilot Study in Statistical Machine Translation. In *Proceedings of the 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, Los Angeles, CA.

Chafe, Wallace L. 1971. Directionality and Paraphrase. *Language*, vol. 47, 1, pages 1-26.

Chang, Chih-Chung and Chih-Jen Lin. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1-27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Chen, Minmin, Kilian Q. Weinberger, John C. Blitzer. 2011. Co-training for Domain Adaptation. In *Proceedings of the 25th conference on Neural Information Processing Systems (NIPS)*, Granada, Spain.

Chiang, David. 2005. A Hierarchical Phrase-Based Model for Statistical Machine Translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 263-270, Ann Arbor, MI.

Cook, Paul, Afsaneh Fazly, and Suzanne Stevenson. 2007. Pulling their Weight: Exploiting Syntactic Forms for the Automatic Identification of Idiomatic Expressions in Context. In *Proceedings of the ACL Workshop on A Broader Perspective on Multiword Expressions*, pages 41–48, Prague, Czech Republic.

Cook, Paul, Afsaneh Fazly, and Suzanne Stevenson. 2008. The VNC-Tokens Dataset. In *Proceedings of the LREC Workshop on Towards a Shared Task for Multiword Expressions (MWE 2008)*, Marrakech, Morocco.

Coughlin, Deborah A. 2003. Correlating Automated and Human Assessments of Machine Translation Quality. In *Proceedings of the MT Summit IX*, New Orleans, LA.

Dagan, Ido and Oren Glickman. 2004. Probabilistic Textual Entailment: Generic Applied Modeling of Language Variability. In *PASCAL workshop on Learning Methods for Text Understanding and Mining*, pages 26-29, Grenoble, France.

Dagan, Ido, Lillian Lee and Fernando Pereira. 1999. Similarity-based Models of Cooccurrence Probabilities. *Machine Learning*, vol. 34 pages (1-3):43-69.

Dagan, Ido, Oren Glickman, Alfio Gliozzo, Efrat Marmorshtein and Carlo Strapparava. 2006. Direct Word Sense Matching for Lexical Substitution. In *Proceedings of the Association of Computational Linguistics (ACL 2006)*, Sydney, Australia.

Dagan, Ido, Shaul Marcus and Shaul Markovitch. 1995. Contextual Word Similarity and Estimation from Sparse Data. *Computer Speech and Language*, pages 9:123-152.

Dandapat, Sandipan, Sara Morrissey, Andy Way and Mikel L. Forcada. 2011. Using Example-Based MT to Support Statistical MT when Translating Homogeneous Data in a Resource-Poor Setting. In *Proceedings of the 15th Annual Conference of the European Association for Machine Translation*, pages 201-208, Leuven, Belgium.

Dawood, Mohammed. 2003. A Dictionary of Arabic Contemporary Idioms (mu'jm alta'bir alastlahiat). *Dar Ghareeb*.

Deerwester, Scott C., Susan T. Dumais, Thomas K. Landauer, George W. Furnas and Richard A. Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, Vol. 41, 6, pages 391-407.

Denkowski, Michael and Alon Lavie. 2010a. METEOR-NEXT and the METEOR Paraphrase Tables: Improved Evaluation Support For Five Target Languages. In *Proceedings of the ACL 2010 Joint Workshop on Statistical Machine Translation and Metrics MATR*, Uppsala, Sweden.

Denkowski, Michael and Alon Lavie. 2010b. Extending the METEOR Machine Translation Evaluation Metric to the Phrase Level. In *Proceedings of the Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL-10)*, pages 250-253, Los Angeles, CA.

Denkowski, Michael and Alon Lavie. 2011. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the EMNLP 2011 Workshop on Statistical Machine Translation*, Edinburgh, UK.

Denkowski, Michael, Hassan Al-Haj and Alon Lavie. 2010. Turker-Assisted Paraphrasing for English-Arabic Machine Translation. In *Proceedings of the Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk at the Conference of Human Language Technology-North American Association for Computational Linguistics (HLT-NAACL 2010)*, pages 66-70, Los Angeles, CA.

Diab, Mona. 2007. Improved Arabic Base Phrase Chunking with a New Enriched POS Tag Set. In *Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources, 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 89-96, Prague, Czech Republic.

Diab, Mona. 2009. Second generation tools (AMIRA 2.0): Fast and Robust Tokenization, POS Tagging, and Base Phrase Chunking. *MEDAR 2nd International Conference on Arabic Language Resources and Tools*, Cairo, Egypt.

Diab, Mona, Kadri Hacioglu and Daniel Jurafsky. 2004. Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks. In *Proceedings of Human Language Technology-North American Association for Computational Linguistics (HLT-NAACL 2004)*, pages 149-152, Boston, MA.

Diab, Mona and Madhav Krishna. 2009a. Handling Sparsity for Verb Noun MWE Token Classification. In *Proceedings of the ACL Workshop on Geometrical Models of Natural Language Semantics*, pages 96–103, Athens, Greece.

Diab, Mona and Madhav Krishna. 2009b. Unsupervised Classification for VNC Multiword Expressions Tokens. In *Proceedings of the 10th International Conference on Intelligent Text Processing and Computational Linguistics (CICLING)*, Mexico City, Mexico.

Diab, Mona and Pravin Bhutada. 2009. Verb Noun Construction MWE Token Supervised Classification. In *Workshop on Multiword Expressions (ACL-IJCNLP*), pages 17–22, Singapore.

Doddington, George. 2002. Automatic Evaluation of Machine Translation Quality Using n-gram Cooccurrence Statistics. In *Proceedings of the Human Language Technology Conference (HLT)*, pages 128-132, San Diego, CA.

Dolan, William B. and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005), Asia Federation of Natural Language Processing*.

Dolan, William B., Chris Quirk and Chris Brockett. 2004. Unsupervised Construction of Large Paraphrase Corpora: Exploiting Massively Parallel News Sources. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004),* Geneva, Switzerland.

Duboue, Pablo Ariel and Jennifer Chu-Carroll. 2006. Answering the Question you Wish they had Asked: The Impact of Paraphrasing for Question Answering. In *Proceedings of Human Language Technology-North American Association for Computational Linguistics (HLT-NAACL 2006)*, pages 33-36, New York City, NY.

Duclaye, Florence, Francois Yvon and Olivier Collin. 2003. Learning Paraphrases to Improve a Question-Answering System. In *Proceedings of the 11th Conference of*

the European Chapter of the Association for Computational Linguistics (EACL), Workshop in NLP for Question Answering, Budapest, Hungary.

Dunning, Ted. 1993. Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics* 19, 1, pages 61-74.

Dyer, Christopher, Smaranda Muresan and Philip Resnik. 2008. Generalizing Word Lattice Translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2008)*, pages 1012–1020, Columbus, Ohio.

Dyvik Helge. 2004. Translations as Semantic Mirrors: From Parallel Corpus to WordNet. *Language and Computers*, *Rodopi*, 49, 1, pages 311-326.

Eifring, Halvor and Rolf Theil. 2005. Linguistic Typology. *Linguistics for Students of Asian and African Languages*, Chapter 4.

Fader, Anthony, Luke Zettlemoyer and Oren Etzioni. 2013. Paraphrase-Driven Learning for Open Question Answering. In Proceedings of the Association for Computational Linguistics (ACL 2013), Sofia, Bulgaria.

Fayed, Wafaa Kamel. 2007. A Dictionary of Arabic Contemporary Idioms (mu'jm al-ta'bir alastlahiat). *Abu Elhoul*.

Fazly, Afsaneh and Suzanne Stevenson. 2007. Distinguishing Subtypes of Multiword Expressions Using Linguistically Motivated Statistical Measures. In *Proceedings of the ACL Workshop on A Broader Perspective on Multiword Expressions*, pages 9–16, Prague, Czech Republic.

Fellbaum, Christiane. 1998. WordNet: An Electronic Lexical Database*. MIT Press*, Cambridge, MA.

Fujita, Atsushi, Kentaro Furihata, Kentaro Inui, Yuji Matsumoto and Koichi Takeuchi. 2004. Paraphrasing of Japanese Light-Verb Constructions Based on Lexical Conceptual Structure. In *Proceedings of the ACL Workshop on Multiword Expressions: Integrating Processing*, pages 9–16, Barcelona, Spain.

Fujita, Atsushi and Kentaro Inui. 2005. A Class-Oriented Approach to Building a Paraphrase Corpus. In *Proceedings of the Third International Workshop on Paraphrasing*, pages 25–32, Jeju Island, Republic of Korea.

Fujita, Atsushi, Shuhei Kato, Naoki Kato and Satoshi Sato. 2007. A Compositional Approach toward Dynamic Phrasal Thesaurus. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 151–158, Prague, Czech Republic.

Fung, Pascale and Kathleen McKeown. 1994. Aligning Noisy Corpora Across Language Groups: Word Pair Feature Matching by Dynamic Time Warping. In *Proceedings of the 1st Conference of the Association for Machine Translation in the Americas (AMTA-94)*, pages 81-88, Columbia, MD.

Gale, William A. and Kenneth W. Church. 1991. A Program for Aligning Sentences in Bilingual Corpora. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics (ACL '91)*, pages 177-184, Stroudsburg, PA.

Ganitkevitch, Juri, Chris Callison-Burch, Courtney Napoles and Benjamin Van Durme. 2011. Learning Sentential Paraphrases from Bilingual Parallel Corpora for Text-to-Text Generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, Edinburgh, UK.

Garofolo, John. 2002. NIST OpenMT Eval. 2002. http://www.itl.nist .gov/iad/mig/tests/mt/2002/.

Garofolo, John. 2009. NIST OpenMT Eval. 2009. http://www.itl.nist. gov/iad/mig/tests/mt/2009/.

Glickman, Oren and Ido Dagan. 2003. Identifying Lexical Paraphrases from a Single Corpus: A Case Study for Verbs. In *Proceedings of* the *Recent Advantages in Natural Language Processing (RANLP-03)*, pages 81-90, Borovets, Bulgaria.

Goldwater, Sharon and David McClosky. 2005. Improving Statistical MT Through Morphological Analysis. In *Proceedings of the Human Language Technology Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, Vancouver, B.C, Canada.

Graff, David and Christopher Cieri. 2003. English Gigaword. *Linguistic Data Consortium*, Philadelphia, PA.

Guillaumin, Matthieu, Jakob Verbeek and Cordelia Schmid. 2010. Multimodal Semi-Supervised Learning for Image Classification. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, pages 902 -909, San Francisco, CA.

Gupta, Sonal, Joohyun Kim, Kristen Grauman and Raymond J. Mooney. 2008. Watch, Listen & Learn: Co-training on Captioned Images and Videos. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2008)*, Antwerp, Belgium.

Habash, Nizar. 2010. Introduction to Arabic Natural Language Processing. *Synthesis Lectures on Human Language Technologies, Morgan and Claypool Publishers*, ISBN-10: 1598297953.

Habash, Nizar and Fatiha Sadat. 2006. Arabic Preprocessing Schemes for Statistical Machine Translation. In *Proceedings of Human Language Technology Conference of the NAACL*, pages 49-52, New York, NY.

Habash, Nizar and Owen Rambow. 2005. Arabic Tokenization, Morphological Analysis, and Part-of-Speech Tagging in one Fell Swoop. In *Proceedings of the Conference of American Association for Computational Linguistics*, pages 578-580, Ann Arbor, MI.

Habash, Nizar, Owen Rambow and Ryan Roth. 2009. MADA+TOKAN: A Toolkit for Arabic Tokenization, Diacritization, Morphological Disambiguation, POS Tagging Stemming and Lemmatization. In *Proceedings of the Second International Conference on Arabic Language Resources and Tools, The MEDAR Consortium*, Cairo, Egypt.

Hall, Mark, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, volume 11, issue 1, pages 10-18.

Hashimoto, Chikara and Daisuke Kawahara. 2008. Construction of an Idiom Corpus and its Application to Idiom Identification Based on WSD Incorporating Idiom-Specific Features. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 992–1001, Honolulu, Hawaii.

Hawwari, Abdelati, Kfir Bar, and Mona Diab. 2012. Building an Arabic Multiword Expressions Repository. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 24-29, Jeju, Republic of Korea.

Ibrahim, Ali, Boris Katz and Jimmy Lin. 2003. Extracting Structural Paraphrases from Aligned Monolingual Corpora. In *Proceedings of the Second International Workshop on Paraphrasing (ACL, 2003)*, Sapporo, Japan.

Jackendoff, Ray. 1997. The architecture of the language faculty. *The MIT Press*, Cambridge, MA.

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, Stroudsburg, PA.

Jones, Bevan K., Jacob Andreas, Daniel Bauer, Karl Moritz Hermann and Kevin Knight. 2012. Semantics-Based Machine Translation with Hyperedge Replacement Grammars. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, Mumbai, India.

Katz, Graham and Eugenie Giesbrecht. 2006. Automatic Identification of Non-Compositional Multiword Expressions using Latent Semantic Analysis. In *Proceedings of the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties*, pages 12–19, Sydney, Australia.

Kauchak, David and Regina Barzilay. 2006. Paraphrasing for Automatic Evaluation. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL-06)*, pages 455–462, New York, NY.

Kim, Su Nam and Timothy Baldwin. 2009. How to Pick out Token Instances of English Verb-Particle Constructions. *Journal of Language Resources and Evaluation* 44(1-2), pages 97-113.

Koehn, Philip. 2004. Statistical Significance Tests For Machine Translation Evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, pages 388-395, Barcelona, Spain.

Koehn, Philipp. 2010. Statistical Machine Translation. *Cambridge University Press*, ISBN-10: 0521874157.

Koehn, Philipp, Franz Josef Och and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the Human Language Technology Conference (HLT-NAACL 2003)*, pages 48-54, Edmonton, Canada.

Koehn, Philipp and Kevin Knight. 2003. Empirical Methods for Compound Splitting. In *Proceedings of the 10th Conference of the European Chapter of the ACL (EACL 2003)*, Budapest, Hungary.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin and Evan Herbst. 2007. **Moses**: Open source Toolkit for Statistical Machine Translation. In *Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL)*, demonstration session, Prague, Czech Republic. http://www.statmt.org/moses/

Kubler, Sandra, Ryan McDonald and Joakim Nivre. 2009. Dependency Parsing. *Morgan & Claypool Publishers*.

Kudo, Taku and Yuji Matsumoto. 2003. Fast Methods for Kernel-Based Text Analysis. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 24-31, Sapporo, Japan.

Lancioni, Giuliano and Marco Boella. 2012. Idiomatic MWEs and Machine Translation. A Retrieval and Representation Model: the AraMWE Project. In *Proceedings of the Fourth Workshop on Computational Approaches to Arabic Script-based Languages, in the Tenth Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, San Diego, CA.

Lee, Lillian. 1999. Measures of Distributional Similarity. In Proceedings of *the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 25-32, College Park, MD.

Lee, Young-Suk. 2004. Morphological Analysis for Statistical Machine Translation. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL-04)*, Lisbon, Portugal, 57–60.

Levenshtein, V. I. 1966. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. Translated to English from *Doklady Akademii Nauk SSSR*, Vol. 163, 4, pages 845-848.

Lin, Dekang. 1993. MiniPar. http://webdocs.cs.ualberta.ca/~lindek/minipar.htm

Lin, Dekang. 1998. An Information-Theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML '98)*, *Jude W. Shavlik (Ed.). Morgan Kaufmann Publishers Inc.*, pages 296-304, San Francisco, CA.

Lin, Dekang and Patrick Pantel. 2001. Discovery of Inference Rules for Question Answering. *Natural Language Engineering* 7(4), pages 343-36.

van der Plas, Lonneke and Jörg Tiedemann. 2006. Finding Synonyms Using Automatic Word Alignment and Measures of Distributional Similarity. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics and International Conference on Computational Linguistics (COLING/ACL 2006), Main Conference Poster Sessions*, pages 866-873, Sydney, Australia.

Lopez, Adam. 2008. Statistical Machine Translation. *ACM Computing Surveys*, Vol. 40, 3, pages 8:1-49.

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

Maamouri, Mohamed, Dave Graff, Basma Bouziri, Sondos Krouna, Ann Bies and Seth Kulick. 2010. Standard Arabic morphological Analyzer (SAMA), Version 3.1. *Linguistic Data Consortium*, Philadelphia, PA.

Madnani, Nitin and Bonnie J. Dorr. 2010. Generating Phrasal and Sentential Paraphrases: A Survey of Data-Driven Methods. *Computational Linguistics*, vol. 36, 3, pages 341-387.

Madnani, Nitin and Bonnie J. Dorr. 2013. Generating Targeted Paraphrases for Improved Translation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 4, 3, pages 40:1-25.

Madnani, Nitin, Necip Fazil Ayan, Philip Resnik and Bonnie J. Dorr. 2007. Using Paraphrases for Parameter Tuning in Statistical Machine Translation. In *Proceedings*

of the Second Workshop on Statistical Machine Translation, pages 120-127, Prague, Czech Republic.

Madnani, Nitin, Philip Resnik, Bonnie J. Dorr and Richard Schwartz. 2008. Are Multiple Reference Translations Necessary? Investigating the Value of Paraphrased Reference Translations in Parameter Optimization. In *Proceedings of the Conference of the Association for Machine Translation in the Americas (AMTA)*, Waikiki, Hawaii.

Marton, Yuval, Chris Callison-Burch and Philip Resnik. 2009. Improved Statistical Machine Translation Using Monolingually Derived Paraphrases. In *Proceedings of the Conference for Empirical Methods in Natural Language Processing (EMNLP 2009)*, Singapore.

McDonald, Scott. 2000. Environmental Determinants of Lexical Processing Effort. Ph.D. thesis, *University of Edinburgh*.

McKeown, Kathleen R. 1983. Paraphrasing Questions Using Given and New Information. *Computational Linguistics* 9, 1, pages 1-10.

Miller, George A. 1995. WordNet: A Lexical Database for English. *Communications of the ACM* Vol. 38, No. 11, pages 39-41.

Mirkin, Shachar, Lucia Specia, Nicola Cancedda, Ido Dagan, Marc Dymetman and Idan Szpektor. Source-Language Entailment Modeling for Translating Unknown Terms. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing (ACL-IJCNLP-2009)*, pages 791–799, Singapore.

Munteanu, Dragos Stefan and Daniel Marcu. 2006. Extracting Parallel Sub-Sentential Fragments from Nonparallel Corpora. In *Proceedings of the Association for Computational Linguistics Conference (ACL 2006)*, Syndey, Australia.

Nagao, Makoto. 1984. A Framework of Mechanical Translation Between Japanese and English by Analogy Principle. In A. Elithorn and R. Banerji, eds., *Artificial and Human Intelligence*. North-Holland, 173-180.

Och, Franz Josef. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 160-167, Sapporo, Japan.

Och, Franz Josef and Hermann Ney. 2002. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 295-302, Philadelphia, PA.

Och, Franz Josef and Hermann Ney. 2003. The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, Vol. 30, 4, pages 418-449.

Okita, Tsuyoshi and Andy Way. Given Bilingual Terminology in Statistical Machine Translation: MWE-Sensitve Word Alignment and Hierarchical Pitman-Yor Process-Based Translation Model Smoothing. In *Proceedings of the 24th International Florida Artificial Intelligence Research Society Conference (FLAIRS-24)*, pages 269-274, Palm Beach, Florida.

Owczarzak, Karolina, Declan Groves, Josef Van Genabith and Andy Way. 2006. Contextual Bitext-Derived Paraphrases in Automatic MT Evaluation. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 86–93, New York, NY.

Pado, Sebastian, Michel Galley, Dan Jurafsky and Christopher D. Manning. 2009. Textual Entailment Features for Machine Translation Evaluation. In *Proceedings of the EACL Workshop on Machine Translation*, Athens, Greece.

Pal, Santanu, Sudip Kumar Naskar, Pavel Pecina, Sivaji Bandyopadhyay and Andy Way. 2010. Handling Named Entities and Compound Verbs in Phrase-Based Statistical Machine Translation. In *Proceedings of MWE 2010 - Workshop on Multiword Expressions: from Theory to Applications*, Beijing, China.

Pal, Santanu, Sudip Kumar Naskar and Sivaji Bandyopadhyay. 2013. MWE Alignment in Phrase Based Statistical Machine Translation. In *Proceedings of the 2nd Workshop of Hybrid Approaches to Machine Translation (HyTra 2013), ACL-2013*, Sofia, Bulgaria.

Pang, Bo, Kevin Knight and Daniel Marcu. 2003. Syntax-Based Alignment of Multiple Translations: Extracting Paraphrases and Generating New Sentences. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003)*, Edmonton, Canada.

Papineni, Kishore, Salim Roukos, Todd Ward and Wei-Jing Zhu. 2002. Bleu: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the ACL 40th Annual Meeting*, pages 311-318, Philadelphia, PA.

Parker, Robert, David Graff, Ke Chen, Junbo Kong and Kazuaki Maeda. 2011. Arabic Gigaword, Fourth Edition. *Linguistic Data Consortium*, LDC2009T30, ISBN 1-58563-532-4, Philadelphia, PA.

Parker, Robert, David Graff, Ke Chen, Junbo Kong and Kazuaki Maeda. 2011. English Gigaword Fifth Edition. *Linguistic Data Consortium*, LDC2011T07, Philadelphia, PA.

Pedersen, Ted, Mehmet Kayaalp and Rebecca Bruce. 1996. Significant Lexical Relationships. In *Proceedings of the 13th National Conference on Artificial Intelligence*, Portland, Oregon.

Peter Brown, John Cocke, Stephen Della Pietra, Vincent Della Pietra, Frederick Jelinek, Robert Mercer and Paul Poossin. 1990. A Statistical Approach to Language Translation. *Computational Linguistics*, 16(2).

Quirk, Chris, Chris Brockett and William Dolan. 2004. Monolingual Machine Translation for Paraphrase Generation. In *Proceedings of the 2004 conference of Empirical Methods in Natural Language Processing (EMNLP 2004)*, pages 142-149, Barcelona, Spain.

Ravichandran, Deepak and Eduard Hovy. 2002. Learning Surface Text Patterns for a Question Answering System. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002), Philadelphia, PA.

Riezler, Stefan, Alexander Vasserman, Ioannis Tsochantaridis, Vibhu Mittal and Yi Liu. 2007. Statistical Machine Translation for Query Expansion in Answer Retrieval. In *Proceedings of Association for Computational Linguistics (ACL 2007)*, pages 464-471, Prague, Czech Repulic.

Rosenbaum, Gabriel M. 2000. “Fushāmmiyya: Alternating Style in Egyptian Prose”, *Journal of Arabic Linguistics (ZAL)* (38) pages 68-87.

Roth, Ryan, Owen Rambow, Nizar Habash, Mona Diab Cynthia and Rudin. 2008. Arabic Morphological Tagging, Diacritization, and Lemmatization Using Lexeme

Models and Feature Ranking. In *Proceedings of Association for Computational Linguistics (ACL 2008)*, pages 117-120, Columbus, Ohio.

Sag, Ivan A. and Timothy Baldwin, Francis Bond, Ann A. Copestake, and Dan Flickinger. 2002. Multiword Expressions: A Pain in the Neck for NLP. In *Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing*, pages 1–15, London, UK.

Salloum, Wael and Nizar Habash. 2011. Dialectal to Standard Arabic Paraphrasing to Improve Arabic-English Statistical Machine Translation. In *Proceedings of the Dialects workshop at the Conference for Empirical Methods in Natural Language Processing (EMNLP 2011)*. Edinburgh, UK.

Schone, Patrick and Daniel Juraksfy. 2001. Is Knowledge-Free Induction of Multiword Unit Dictionary Headwords a Solved Problem? In *Proceedings of Empirical Methods in Natural Language Processing*, pages 100–108, Pittsburg, PA.

Shinyama, Yusuke, Satoshi Sekine, Kiyoshi Sudo and Ralph Grishman. 2002. Automatic Paraphrase Acquisition from News Articles. In *Proceedings of the Hu- man Language Technology Conference (HLT–02)*, San Diego, CA.

Sieny, Mahmoud Esmail, Mokhtar A. Hussein and Sayyed A. Al-Doush. 1996. A Contextual Dictionary of Idioms (almu'jm alsyaqi lelta'birat alastlahiah). *Librairie du Liban Publishers*.

Singh, Nimesh and Nizar Habash. 2012. Hebrew Morphological Preprocessing for Statistical Machine Translation. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation (EAMT)*, Trento, Italy.

Somers, Harold. 1999. Review Article: Example-Based Machine Translation. *Machine Translation 14*, pages 113-157.

Somers, Harold. 2003. Machine Translation: Latest Developments. In *Ruslan Mitkov (ed) The Oxford Handbook of Computational Linguistics*, Oxford: Oxford University Press, pages 512-528.

Sporleder, C. and L. Li. 2009. Unsupervised Recognition of Literal and Non-Literal Use of Idiomatic Expressions. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL)*, pages 754–762, Athens, Greece.

Stanovsky, Gabriel. 2012. A study in Hebrew Paraphrase Identification. *MSc thesis. Ben Gurion University of the Negev, Israel*. Online version: http://www.cs.bgu.ac.il/~nlpproj/paraphrase/Thesis.pdf.

Stolcke, Andreas. 2002. SRILM -- An Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, Vol. 2, pages 901-904, Denver, CO.

Szpektor, Idan, Hristo Tanev, Ido Dagan and Bonaventura Coppola. 2004. Scaling Web-Based Acquisition of Entailment Relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, pages 41–48, Barcelona, Spain.

Szpektor, Idan, Ido Dagan, Roy Bar-Haim and Jacob Goldberger. 2008. Contextual Preferences. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2008)*, Columbus, OH.

Takahashi, Tetsuro, Kozo Nawata, Kentaro Inui and Yuji Matsumoto. 2003. Effects of Structural Matching and Paraphrasing in Question Answering. *IEICE Transactions on Information and Systems*, Vol. E86-D, 9, pages 1677-1685.

Tsvetkov, Yulia and Shuly Wintner. 2011. Identification of Multiword Expressions by Combining Multiple Linguistic Information Sources. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 836-845, Edinburgh, Scotland.

Tsvetkov, Yulia and Shuly Wintner. 2012. Extraction of Multiword Expressions from Small Parallel Corpora. *Natural Language Engineering* 18(4), pages 549-573.

Turian, Joseph, Luke Shen and I. Dan Melamed. 2003. Evaluation of Machine Translation and its Evaluation. In *Proceedings of MT Summit IX*, pages 23-28, New Orleans.

Vapnik, Vladimir and Corinna Cortes. 1995. Support Vector Networks. *Machine Learning*, vol. 20, pages 273-297.

Vauquois, B. 1968. A Survey of Formal Grammars and Algorithms for Recognition and Translation in Machine Translation. In *FIP Congress-68*, pages 254-260, Edinburgh.

Wan, Xiaojun. 2009. Co-training for Cross-Lingual Sentiment Classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, Singapore.

Wang, Rui and Chris Callison-Burch. 2011. Paraphrase Fragment Extraction from Monolingual Comparable Corpora. In *Proceedings of Fourth Workshop on Building and Using Comparable Corpora (BUCC)*, Istanbul, Turkey.

Wu, Hua and Ming Zhou. 2003. Synonymous Collocation Extraction Using Translation Information. In *Proceedings of the ACL Workshop on Multiword Expressions: Integrating Processing*, pages 120–127, Sapporo, Japan.

Wu, Ting-Fan, Chih-Jen Lin and Ruby C. Weng. 2004. Probability Estimates for Multiclass Classification by Pairwise Coupling. *Journal of Machine Learning Research*, Vol. 5, pages 975-1005.

Zhao, Shiqi, Haifeng Wang, Ting Liu and Sheng Li. 2008. Pivot Approach for Extracting Paraphrase Patterns from Bilingual Corpora. In *Proceedings of the Annual Meeting of Association of Computational Linguistics (ACL 2008*), pages 780-788, Columbus, OH.

Zhou, Liang, Chin-Yew Lin, and Eduard Hovy. 2006. Re-evaluating Machine Translation Results with Paraphrase Support. In *Proceedings of the Conference for Empirical Methods in Natural Language Processing (EMNLP 2006)*, pages 77–84, Sydney.