Abstract

We describe an implementation of an Example-Based Machine Translation system that translates short sentences from Arabic to English. The system uses a large parallel corpus, aligned at the paragraph level, and built using many parallel Arabic-English documents from the United-Nations database.

This is a non-structural system, so examples are stored in their surface forms, with some additional morphological and part-of-speech information. Each new input sentence is matched to example patterns by using various levels of morphological data. Matched fragments are transferred to English using a rough word-level alignment algorithm, which makes use a bilingual dictionary, along with WordNet. There is no syntactical parser involved in the matching and/or transfer processes, although we do use a shallow English parser to help with specific situations.

Currently, the system first fragments any newly introduced input sentence and then translates each segment separately; recombining those translations into a final coherent form is left for future work.

We encountered several problems in the matching and the transfer steps, some of which were solved, partially or totally, sometimes by using linguistic tools for both languages. We discuss those problems and our proposed solutions.

The system has been implemented and automatically evaluated. Results are encouraging.

Table of Contents

Chapter 1
Introduction

1. Chapter 1 – Introduction

One of the oldest challenges since computers were invented is Machine-Translation (sometimes referred as "Automatic-Translation"), that is, translating a text from one natural language (the *source-language*) into another one (the *target-language*) using computers. The text might be a word, or sentence or even an entire text document. Early automatic-translation approaches focused on performing what is called "Fully Automatic High Quality Translation (FAHQT)." The output of such systems is designed to be high-quality coherent target-language text that exactly translates the source-language input text. These approaches were criticized in the famous Bar Hillel report [1], issued in 1960, claiming that developing a system for high-quality translations is utterly futile. In his report, Bar Hillel argued that the term "High Quality" should be discarded for a system that performs a fully automatic translation process. Such a system may be useful for tasks that require only rough translations, or should be considered only if there is a manually post-editing step that finalizes the entire translation process to achieve a good quality translation.

Since then, there has been much research devoted to various levels of machine-translation, introducing new, interesting approaches using different levels of linguistic analysis and/or large corpora.

In spite of the criticism of FAHQT, there are many working machine-translation systems that have achieved satisfactory results. Most existing machine-translation systems are designed to translate texts for some predefined domains, but there are also general systems that work without any domain restriction, producing translations that can not, for the most of the part, compete with high-quality human translations.

1.1 Classifying Machine Translation Systems

A machine translation system can be classified by how deeply it analyzes the source-language text. This is usually described by the classic pyramid, first presented by Vauquois [2] and shown in Figure 1.
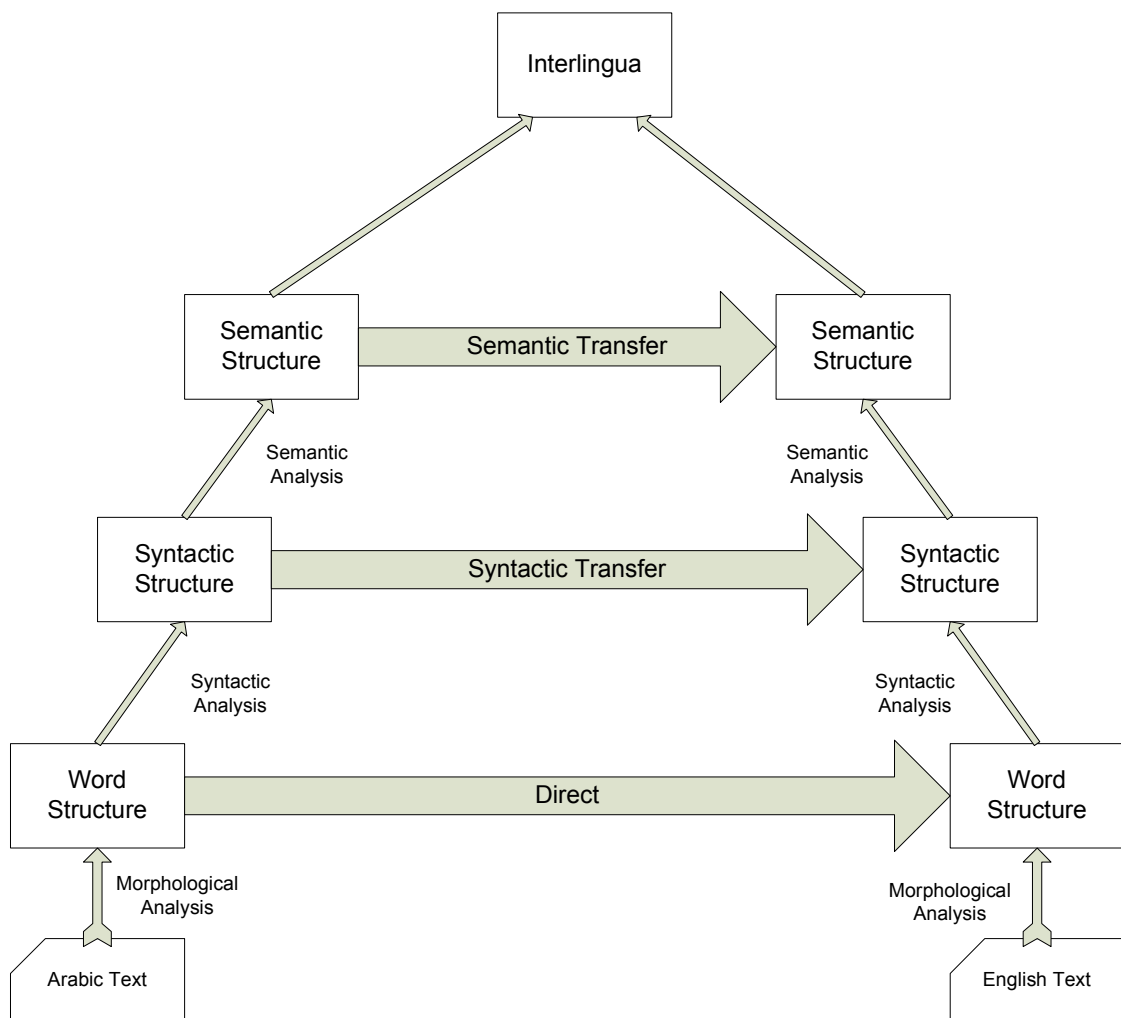
Figure 1 - Classification of a machine translation system

At the bottom of the pyramid are the *direct* machine-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 analyze 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. This may be done using various rule-based or corpus-based methods, as will be described later. 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 common-logic representation or even a slightly 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 approach were sympathized by Bar Hillel, although he actually thought that the whole idea of automatic high quality translation approach is misguided.

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

Another dimension for the classification of a machine-translation system is its research paradigm. A survey of current machine-translation paradigms [3] describes two major paradigms: linguistic-based and corpus-based. Linguistic-based paradigms mostly use a

predefined set of rules, with or without some kinds of linguistic knowledge-bases, to produce the translation. Working with such paradigms requires comprehensive knowledge of linguistic theory for both source and target languages. Linguistic-based paradigms may be used by all the different kind of system types – direct, transfer and Interlingua. Corpus-based systems exploit a large parallel bilingual corpus either for offline statistical learning, helping to predict the translation of a new given input, or for searching already translated examples and producing a recombined translation. The latter paradigm is referred as Example-Based Machine Translation (EBMT), and is the key concept of the work described in this thesis.

1.2 Example-Based Machine Translation (EBMT)

The example-based paradigm (also known as "memory-based") has become a fairly common technique for natural language processing (NLP) applications. The main idea behind an example-based machine translation paradigm is to emulate the way a human translator think in some cases, as was first introduced by Nagao [4] in 1984. Example-based machine translation systems [5-10] 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 a 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 2 presents the main steps of an example-based machine translation system.
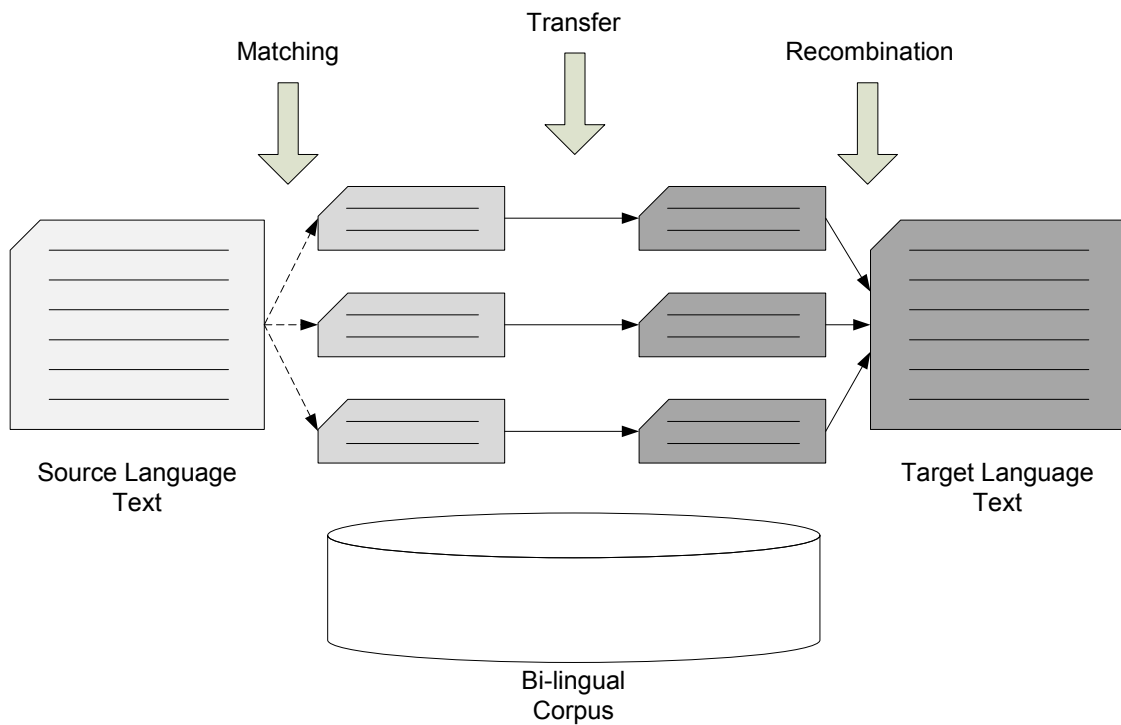
Figure 2 – Main steps of example-based translation system

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 also 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 (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.

For more information on this subject, the reader may refer to the comprehensive review of example-based machine-translation by Somers [11].

1.3 Overview of Arabic to English Machine Translator

In our work, we implemented the important parts of an example-based machine translation system that translates short Arabic sentences to English. This is a non-structural system, so it stores the translation-examples as textual strings, with some additional information, as will be described later. We used the United-Nations document inventory to build our translation-example corpus. The inventory contains a large number of documents that were manually translated into seven languages, and Arabic is one of them. We automatically aligned each document pair on the paragraph level, and each parallel paragraph was taken as a translation-example.

At this point, our system performs the fragmentation, matching and transfer steps fully automatically. The recombination step only pastes together the extracted fragments, but it does not yet smooth out the recombined text into a fully grammatical English sentence. The entire system was written as a Java application. Special GUI tools were built to allow external users to interact with the system in two modes: translation mode and corpus creation mode. Currently, those tools require the installation of the entire system, but we are planning to develop a web-interface allowing users to interact with the system remotely.

In this work, we concentrated mostly on the design and implementation of the Arabic–to-English example-based machine translation system, but did not concentrate on implementing the components efficiently. More work would be needed to reduce the total translation time.

Several external software packages were integrated into our system. The SVM-POS package [12] is used to find the part-of-speech tag of each Arabic word, and Brill's tagger is used for part-of-speech tagging English words. Buckwalter's Arabic morphological analyzer [13] is used for several analysis procedures on the word level. The BaseNP Chunker is used for shallow parsing the English part of the translation-examples, as will be explained later.

Our system is composed of several modules. Figure 3 presents a high-level view of the system's main modules and their relations.

Input: Short Arabic Sentence

Preprocessing

Preprocessed Text

Matching

General Fragments

Transfer

General Fragments +
Translations

Recombination

Examples
Corpus

Output: N Best Translations (English)
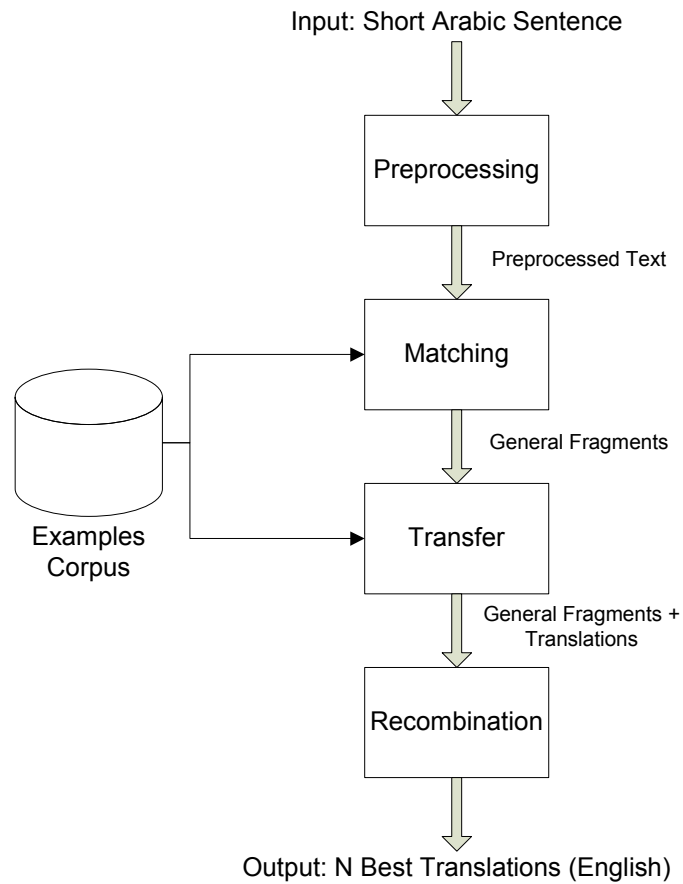
Figure 3 –High-level observation

Our system was designed to handle short Arabic sentences (up to 10 words). Actually, the
system's slow performance is the reason for the length limit; larger sentences are also
possible, but their translation generation might take an unreasonably long time.

In the first module, *preprocessing*, the input sentence is tokenized and transliterated using
the Buckwalter transliteration system [13]. The transliterated tokens are also part-of-
speech tagged using the SVM-POS tool and analyzed using the Buckwalter
morphological analyzer. In the *matching* module, the system searches for already
translated fragments of the input text, by matching word by word. Words are matched on
string, stem, lemma, and part-of-speech levels, with each level of match assigned a

different score. Exact string match receives the highest score; stem match is less, etc. The score for a fragment is calculated from the individual word-match scores, as will be explained later. For the *transfer* module, we only handle fragments whose score exceeds some predefined threshold. The first step is the extraction of the translations of the fragments from the English version of the translation examples that have been found in the *matching* module. This is done by aligning the translation example on the word level, using a bilingual lexicon. Actually, for each Arabic word, we look up its English equivalent in the lexicon, and expand it with synonyms from WordNet. Then, we search the English part of the translation example for all instances on the lemma level. Finally, we take the shortest English segment that contains the *maximum* number of aligned words. A new score is produced for each proposed translation using the length of the fragment, number of translated words and the previous matching score. After extracting a translation of the fragment, sometimes there is a need to modify it. Recall that the match of a corpus fragment to the source fragment can be inexact, so sometimes more work needs to be done to get a real translation. There are several issues that we have handled and which are presented in the following chapters of this thesis, but there remain some more unhandled cases, which are outside the scope of this work.

The *recombination* module simply generates all the best combinations of the translated fragments considering their total scores (a combination of their matching and translation scores), but – as already mentioned – it does not smooth out the recombined text. The final output contains the N-best combinations with their scores, where N is an adjustable parameter. (For clarity, a separator is placed between each translated fragment.)

1.4 Thesis Overview

In Chapter 2 we give a very short introduction to some basic elements of the Arabic language. This chapter is primarily for readers without Arabic knowledge, but intermediate and advanced speakers of Arabic may also profit from this information. Chapter 3 is a short review of several relevant natural-language-processing tools that we used in our system implementation. Chapter 4 is an explanation of the corpus creation process. The translation algorithm is described in Chapter 5 and its results and evaluations are presented in Chapter 6. The Appendix shows several example inputs and the corresponding system translations.

Chapter 2
Short Introduction to Arabic

2. Chapter 2 – Short Introduction to Arabic

In this chapter we introduce the relevant basic elements of the Arabic language. A non-Arabic speaker will find this information useful, but this is far from being a comprehensive Arabic guide. Further reading (e.g. [14]) is recommended.

Arabic is currently the sixth most widely spoken language in the world. With 22 countries that define Arabic as their official language, it is spoken by at least 250 million people. Arabic is a Semitic language; it belongs to the same family as Hebrew, Amharic, Maltese, and many more. It is one of the few languages that exhibit diaglossia [15], which is the separation between the spoken languages (dialects) and the formal language. There are Arabic dialects that are more different from each other than even French and Spanish. However, the formal language is Modern Standard Arabic (MSA), which is used in written texts and is spoken in formal settings.

Translating Arabic to English is not an easy task for a number of reasons. Arabic sentences are usually long and contain only few punctuation marks. Due to the complexity of Arabic syntax, sometimes Arabic sentences are syntactically ambiguous and require much effort when trying to resolve such ambiguities automatically. Only in the past several years has there been serious research on Arabic language understanding, and some basic Arabic natural-language-processing tools have been built. There is still a definite lack of language resources and tools, which makes it extra challenging to build working Arabic systems.

2.1 Arabic Orthography

Arabic is written from right to left and its script is also used by other languages, including Persian, Pashto, Urdu, etc. Arabic, like Hebrew, has a consonantal orthography. Due to this fact, an Arabic word without vowel diacritics may be analyzed in many ways, and sometimes it is very hard to resolve the meaning in such situations. We work with texts that are, for the most part, unvocalized.

Arabic words may be written with *tatweel* ("elongation", also called *kasheeda*), which is used for highlighting words or simply for text justification. This is done by stretching some of the letters, so the word will be spread over a bigger space than other words. For example the word الانسان (*AlAnsAn*, "the humans") may be written as الانســان (note the elongated third letter from the left).

There are several transliteration standards for Arabic. We use the Buckwalter transliteration system [13], which was developed at Xerox by Tim Buckwalter in the 1990s. This transliteration system represents Arabic orthography strictly one-to-one: Every Arabic letter or vocalization mark is assigned a single ASCII English sign.

2.2 Arabic Morphology and Syntax

Like many other Semitic languages, Arabic is highly inflected; words are derived from a *root* and *pattern*, combined with prefixes, suffixes and circumfixes (see next chapter). The root consists of 3-4 consonants, which are called radicals and the pattern is a sequence of consonants and variables. Arabic words are created by assigning the root radicals to the pattern variables. A root contains the seed meaning of the word, but the pattern may change that meaning. Thus, combinations of the same root with different patterns may have different meanings. For instance, the combination of the root ك.ت.ب

(*k.t.b*) and the pattern mXXX (here, X is a variable) results in the word مكتب (*mktb*, "office"). Combining the same root with the pattern XXAX, results in the word كتاب (*ktAb*, "book"). Verbs are also generated in the same way. Actually, there are 10 different patterns for verbs, but not all them are valid to combine with every root.

Sometimes, there are prefixes and/or suffixes attached to words. Those affixes may modify several features of the word, including its number (singular, dual, plural or collective), its gender (masculine, feminine or no gender), possession, definiteness, case (nominative, accusative or genitive), tense (past, present or future) and more. Arabic affixes have the feature of concatenating with each other according to predefined linguistic rules, which increases the overall number of affixes [16].

Due to the complexity of Arabic morphology, sometimes words may be ambiguous. The average overall morphological ambiguity in Arabic Penn Tree-Bank is 2.5 analyses per word [17]. Resolution is usually left for higher levels of analysis.

Modern-standard-Arabic sentences usually stick to the Verb-Subject-Object (VSO) structure. (English, by way of contrast, is SVO.) Within an Arabic sentence, there are several required points of agreement between the verbs and the subjects. Adjectives and nouns mostly need to agree on their gender, number and definiteness. An Arabic noun-phrase places the noun before the adjectives, which is the opposite of English noun-phrase syntax.

Chapter 3
Natural Language Processing

3. Chapter 3 - Natural Language Processing

As mentioned before, our translation process requires integration of several natural-language-processing tools, in both languages. Those tools may be categorized into two layers – morphology and syntax. This chapter explains briefly how those tools work, and describes the way we have integrated them in our implementation.

3.1 Morphological Tools

Morphological tools operate on the word level, so the input to those tools is a single word. First, let us describe some basic important definitions:

Morpheme – This is the smallest meaningful unit in a language. A word may be composed of several morphemes. A morpheme may be a word by itself, but it can also be a word affix. For example, the word "connected" is composed of the morphemes *connect* and *ed*. The latter is not a standalone word.

Lemma - A *lemma* is the canonical form of a *lexeme*, which is the set of all surface forms that represent the same word. For example, in English, "goes", "go", "went", "going", etc., are all in the same lexeme, whose lemma is *to go*. A word may be derived from a number of lemmas; each of them may have a different part-of-speech role in the sentence/phrase. For instance, in English, the possible lemmas of the word "book" are the verb *to book* (as in "to book a flight") and the noun *book*. Finding the correct lemma of a word within an Arabic sentence without diacritics (which is the situation in most of the time), may be a difficult task. For example, the word ولد (*wld*) may analyzed as وَلَدَ (*walada*, "give birth to") and therefore comes from the lemma *walada,* but it is also analyzed as وَلَّدَ (*wala~da*, "to generate") and therefore comes from the lemma *wala~da*.

Both are verbs. It is worth mentioning that there are unvocalized Arabic texts that do contains the ~ (*shada*) where needed. There are situations that it is enough to know the part-of-speech role of a word in order to find its lemma, but sometimes it is simply not sufficient.

Dictionary entries, in many languages, are usually based on lemmas. A lemma does not have to be a real word in the language; there are languages in which the lemma is only a code for the lexeme it is representing. In most English dictionaries, the lemma of a noun is represented in the singular form, for example, the lemma of "houses" is *house*. The lemma of a verb is in the first-person singular present-tense form, for example, the lemma of "built" is *build*. Adjectives and adverbs are usually represented by the positive form of the word (for example, the lemma of "unhappy" is *happy*), but sometimes, for instance in WordNet, the lemma is the only existing form of the adjective/adverb (so the lemma of "unhappy" remains *unhappy*). All other categories in English, such as prepositions, conjunctions, interjection, pronoun, etc., are represented by the only existing form. In Arabic, nouns are represented by the singular form of the noun and the lemma of a verb is the third-person singular of the past tense.

Lemmatization – By that we means the process of automatically finding the lemma of the words within a given text. It is also called *context-sensitive lemmatization*. A *lemmatized text* is a text in which every word was replaced by its lemma. Arabic lemmatization tends to be a difficult task [18] and, currently, there are no known Arabic lemmatizers (at least for the author) that are available for public use.

Stem − A stem is the base form of a word. It is a combination of a root and derivational morphemes that may be attached to the beginning of the word (prefixes), at the end

(suffixes) or somewhere in the middle (infixes). Circumfixes are discontinuous morphemes; for instance, the English word "enlighten" is built from the root *light* and the circumfix *en___en*.

Inflectional morphemes are not part of the stem. For instance, the root of the Arabic word بيوتهم (*bywthm*, "their homes") is ب.ي.ت and the stem is بيوت without the inflectional morpheme هم (*hm*, "their").

Given a word in its surface-form, *stemming* is the process of automatically revealing the word's stem. In other words, stemming a word is actually the removal of all the inflectional morphemes from the word's surface-form. Inflectional morphemes in English words are, for the most the part, attached to the end of words, so stemming an English word amounts to simply the removal of its inflectional suffixes. In Arabic, it is a much more complicated task. As mentioned above, Arabic is a highly inflectional language, so inflectional morphemes may occur as prefixes and/or suffixes.

There are several available English stemmers that produce satisfactory results (e.g., the Porter stemmer [19]), but finding a satisfactory Arabic stemmer is much more difficult. One available Arabic stemmer was built by Khoja and Garside [20], and was used by the information retrieval system of the University of Massachusetts [21]. According to the latter researchers, although this stemmer made many mistakes, it improved their system's performance immensely. Unfortunately, this stemmer is not available for public use.

3.1.1 Morphological Analysis

A morphological analysis technique is a computational process that analyzes natural language words by considering their internal structures [16]. Given a word in its surface-form, a morphological analyzer reproduces its morphemes. The output usually contains

information tags for the revealed morphemes, and their content may vary from one analyzer to another. Basically, an information tag may contain the tense of the word, the number, person and part-of-speech. For example, given the word "connective", a morphological analyzer will reproduce the morphemes *connect* [Verb] and *ive,* and the first morpheme is also the stem of the word. The second morpheme, *ive,* modifies the first morpheme, which is a verb, to an adjective.

In many cases, words may be analyzed in different ways and usually morphological analyzers produce all those possible analyses. For instance, the Arabic word ولدي (*wldy*) can be analyzed in several ways. One way is ولد + ي (*wld + y,* "my son"), where the first morpheme is the noun ولد (*wld,* "son") and the second morpheme is a first-person possession suffix. Another possible way is و + لدى + ي (*w + ldY + y,* "and with me"), where the first morpheme is the conjunction و (*w,* "and"), the second morpheme is the preposition لدى (*ldY,* "with/by") and the third morpheme ي (*y*) is a first-person pronoun.

Arabic, like other Semitic languages, introduces a high proportional rate between a single word and the number of its possible analyses, so, in most cases, resolving those ambiguities is left for the higher processing layers, syntactic and even semantic analysis.

Rule-based and corpus-based approaches are the most common paradigms for building a morphological analyzer. Rule-based morphological analyzers use large number of predefined rules, which are represented by a finite-state machine. The Buckwalter Arabic morphological analyzer [13] uses a large lexicon with 82158 entries representing 38600 lemmas (in version 1.0), 299 possible prefixes and 618 possible suffixes. It also uses a predefined table (3500 entries) that contains morphological rules for the possible combinations of those stems, prefixes and suffixes. Given an Arabic word, the analyzer

produces all possible morphological analyses of that word. Each analysis is composed of the stem, lemma, comprised morphemes with their part-of-speech tags and English glossaries.

Other analyzers for Arabic are the Beesley analyzer [22], as well as the analyzer of Berri, Zidoum and Atif [23].

Corpus-based analyzers learn how to handle new inputs using a large tagged corpus, in the case of a supervised analyzer, or a large untagged corpus, in the case of an unsupervised analyzer. Morpho3, by RDI [24], and its shallow version Sebawai, by Drweesh [25], are based on a hybrid paradigm. Those analyzers use template rules to find the root of a new given word, and also a large corpus for resolving analysis ambiguities by statistical methods. The Sebawai analyzer extracts the template rules automatically from a table that was created by another morphological analyzer. The table contains many Arabic words with their corresponding roots.

As mentioned earlier, our implementation uses the Buckwaler analyzer for Arabic morphological analysis and WordNet 2.0 for lemmatizing English words.

3.2 Syntax Tools

Syntax tools deal with the grammatical structure of a given sentence. The most common task on this level is *(syntax) parsing*. Given a sentence and a predefined grammar (in our case, a natural-language grammar), parsing is the process of finding the grammatical structure of the sentence, according to the given grammar. The structure is represented by a grammar tree, which its internal nodes represent the phrases of the sentence and its leaves represent the words with their part-of-speech [26].

Processing the syntax of a sentence helps in resolving some of the morphological ambiguities of its words. Unfortunately, it also introduces another level of ambiguity -- syntactic ambiguity, which is, in most cases, harder to resolve.

As mentioned in the previous descriptions of other computational natural-language tools, parsing an Arabic sentence is much more difficult than parsing an English one. The long average length of an Arabic sentence and the complexity of Arabic syntax are two of the reasons for that. Arabic sentences are much more ambiguous than English sentences [27]. Although recently there have been several interesting studies [28] in that field, currently there is no known available (at least to the author) satisfactory Arabic parser for public use.

Another important tool is the *part-of-speech (POS) tagger*. Part-of-speech tagging is the process of marking sentence words with their part-of-speech. It is easier than parsing, since it deals only with the words in the sentences and not with phrases. The tags are taken from a *tagset,* which is a predefined tags list. Table 1 shows the well-known Penn TreeBank tagset [29].

Tag	Role	Tag	Role	Tag	Role
CC	Coordinating conjunction	NNS	Noun plural	TO	To
CD	Cardinal number	NNP	Proper Noun singular	UH	Interjection
DT	Determiner	NNPS	Proper Noun plural	VB	Verb, base form
EX	Existential there	PDT	Predeterminer	VBD	Verb, past tense
FW	Foreign word	POS	Possessive ending	VBG	Verb, gerund/present participle
IN	Preposition	PRP	Personal pronoun	VBN	Verb, past participle
JJ	Adjective	PP$	Possessive pronoun	VBP	Verb, non-3s, present
JJR	Comparative adjective	RB	Adverb	VBZ	Verb, 3s, present
JJS	Superlative adjective	RBR	Comparative adverb	WDT	Wh-determiner
LS	List item marker	RBS	Superlative adverb	WP	Wh-pronoun
MD	Modal	RP	Particle	WPZ	Possessive Wh-pronoun
NN	Noun singular	SYM	Symbol (math, scientific)	WRB	Wh-adverb

Table 1 - The Penn TreeBank project tagset

For example, given the sentence, "Can I book one ticket for tomorrow?", the following are the tags of its words, using the Penn TreeBank tagset:

Can/*MD* I/*PRP* book/*VB* one/*CD* ticket/*NN* for/*IN* tomorrow/*NN*

A specific word may be analyzed with different part-of-speech tags according to its role in the sentence. The word *can* for instance, may be tagged with MD (modal) as in the previous sentence, but also as NN (noun) in other contexts.

Part-of-speech taggers, like other natural-language tools, have been developed based on a rule-based paradigm or a corpus-based one. Rule-based taggers use a set of rules to compute the tags of a new given sentence, while corpus-based taggers learn how to tag

new input from a large tagged corpus. Unsupervised taggers learn the necessary information from an untagged corpus. Hybrid taggers also exist.

Brill's tagger [30] is a part-of-speech tagger for English with a transformation-based error-driven learning method. The entire process of the tagger is done in two main steps. In the first step the tagger assigns the most appropriate tag to each word of the input sentence and In the second step it uses a small set of rules, which were inferred automatically from a large corpus, to transform the tag to the correct one. There is also an Arabic version of Brill's tagger developed by A. Freeman [31]. Freeman's tagger uses a tagset with 146 different tags, taken from the Brown corpus for English [32]. Another Arabic part-of-speech tagger is APT (Arabic Part-of-Speech Tagger) [33,34], which is based on a hybrid paradigm. That means it uses statistical computations and rule-based technique to produce the required tags. The tagset contains 131 tags, taken from the BNC corpus for English [35] and modified to match Arabic requirements. SVM-POS, which has recently been published, is another tagger for Arabic [12]. The SVM-POS uses a modified version of the Penn TreeBank tagset with 24 different tags and handles the tagging problem as a classification problem. Given a number of features extracted from a predefined linguistic context (such as number of tokens before the current word, number of tokens after, part-of-speech tags of the preceding tokens, etc.), the task is to predict the class of a token. The tagger uses the Support Vector Machine (SVM) to deal with the classification issue.

Our translation system uses the Brill's tagger for English and SVM-POS for Arabic.

3.2.1 Shallow Parsing

As mentioned in the previous section, parsing is a very complicated task. Moreover, Arabic syntax is more ambiguous than English, which makes Arabic parsing even harder. This is the main reason that many natural-language application developers prefer to use a limited parser version – a *shallow parser* (also called a *base-phrase chunker*). A shallow parser operates slightly above part-of-speech tagger and far below a full (tree) syntax parser. Given a sentence with its part-of-speech analysis, the shallow parser finds only its base-phrases, that is non-recursive and non-overlapping phrases. Each base-phrase contains one *head-word,* which is the base of the phrase. For example, the head-word of the phrase "the table" is *table*. A short description of English phrase structure can be found in [36]. A base-phrase may be categorized as a Noun-Phrase (NP), Verb-Phrase (VB), Adjective-Phrase (ADJP), Adverbial-Phrase (ADVP) or Preposition-Phrase (PP). For instance, take the sentence: *The house of the rising sun*.

By part-of-speech tagging the sentence (using the tagset given in Table 1), we get:

The/*DT* house/*NN* of/*IN* the/*DT* rising/*VBG* sun/*NN*.

Shallow parsing the sentence will result in:

[The/*DT* house/*NN*] of/*IN* [the/*DT* rising/*VBG* sun/*NN*]

which means that ***The/DT house/NN*** and ***the/DT rising/VBG sun/NN*** are both base noun-phrases. The word *of* was left as a standalone preposition. Relating it to the next noun-phrase would have composed a complex recursive preposition-phrase, which is out of the scope of the shallow parser task.

There are several available shallow parsers for English. We used the BaseNP parser [37]. BaseNP requires input sentences that were previously tagged by Brill's tagger. Noun-

phrases are the only base-phrases that BaseNP identifies. Arabic shallow parsers are rare. The most significant parser was developed by M.Diab, K.Hacioglu and D.Jurafsky [12], which, like their part-of-speech tagger, handles this task as a classification problem and uses SVM to solve it. Although it gives some impressive results, it is still far from being perfect.

Chapter 4
Extracting Translation Examples

4. Chapter 4 – Extracting Translation Examples

In this chapter, we describe how we created our corpus of translation-examples.

4.1 Preparing the Corpus

The translation examples were extracted from a collection of parallel Arabic-English documents that were taken from the United-Nations document inventory. The Linguistic Data Consortium (LDC) aligned many parallel-documents form this inventory on the sentence-level, and organized them as a large corpus. Unfortunately, at this point we could not afford buying a license to use it or to any other aligned bilingual corpus, so we decided to automatically align part of the UN parallel-documents and create our own corpus. The UN inventory is available under the Official-Document-System (ODS) [38] web interface and it contains official documents since 1993 until these days. The documents are available for download in six UN official languages including English and Arabic. We downloaded the documents as Microsoft Word files and took a quick look over each of them to filter out those documents with complicated format structure (bullets, tables, etc.), since it is harder to convert those documents to a pure ASCII file. Parallel-documents that survived the initial filtering were manually preprocessed in the following steps:

1. Removing all editors/translators notes from the documents.

2. Documents with more than one text column were transformed into a single column documents.

3. Removing the special Tatweel character ("_") from the Arabic documents (see Section 2.1).

The preprocessed parallel-documents were converted to text files. Arabic documents were converted to the Arabic ASCII "1256" encoding and English documents were converted to the English ASCII "1252" encoding.

As was mentioned earlier, example-based systems use bilingual corpus, aligned at least on the sentence or paragraph level. Since, in our case, manual alignment of the documents is a major time-consuming (and not feasible) task, we have decided to automatically align those documents on the paragraph level. It is worth to mention that our work is not about developing an Arabic-English paragraph alignment algorithm so the results are far from being perfect. We believe that using a better (we wish – manually) aligned corpus, our translation algorithm will produce better results.

4.2 Parallel Text Alignment

Based on our final goal – creating a translation-example corpus – we could afford using a non perfect alignment algorithm since we could take only paragraphs that were successfully aligned and without any problems or suspicions. Paragraphs that are translated into more than one paragraph in the other language (not a common situation) were discarded. All parallel-paragraphs were taken as translation-examples and were forward for further processing.

Given a preprocessed parallel-document, we used a modified version of *DK-Vec* algorithm [39] to find as many *parallel-text-anchors* as possible and use them find match for each Arabic paragraph. A parallel-text-anchor (also referred as *anchor*) is the following quadruplet –

- Arabic word;

- location of the word in the Arabic version of the parallel-document;

- English translation of the Arabic word; and

- location of the translation in the English version of the parallel-document.

Figure 4 demonstrates several anchors in a given parallel text. The colored words (also connected with arrows) are only part of the anchors of the text.

اللجنة السادسة	**Sixth** Committee
محضر **موجز** للجلسة الرابعة	**Summary** record of the 4th meeting
المعقودة في **المقر**، **نيويورك**، يوم الجمعة، 7	Held at **Headquarters**, **New York**,
تشرين الأول/أكتوبر 2005، الساعة 10/00	on 7 October 2005, at 10 a.m.

Figure 4 – *Parallel-Text-Anchor* examples

In the following sub-sections we are describing the entire process of extracting anchors in a parallel-document and the way we use them to find parallel-paragraphs.

4.2.1 Finding Parallel Text Anchors

The *DK-Vec* algorithm searches for anchors in a parallel text based on the fact that the positions of word pairs (Arabic word and its translation in the English version) are distributed similarly throughout the Arabic and English parallel versions. Although we have implemented the main steps of the *DK-Vec* algorithm, and since we are not aligning the text on the word level, we have also added another condition for filtering out anchors, so as to keep only the most reliable ones. *DK-Vec* was formerly used by Choueka, Conley and Dagan [40] as part of a word-level alignment algorithm for the Hebrew-English pair. They reported satisfied, although not perfect results.

In their implementation, as well as our version, searching for anchors is performed on the lemma level. For that reason, in our implementation, each preprocessed parallel-document was morphologically analyzed, to discover all the possible lemmas of each word and all possible glossaries for the Arabic words. Arabic and English morphological analysis is explained in Section 3.1.

Given a lemmatized parallel-document (actually, each word is represented by all its possible lemmas), the algorithm creates a *recency-vector* [39] for each lemma. In our version, a recency-vector of a lemma l contains the distances between all the possible occurrences of l in the text. To be more specific, the ith position of the vector contains the distance between the ith and the $i-1$th possible occurrences of the lemma in the text. The distance is simply the difference between the word indices, so one fact is that the first position in every recency-vector contains the index of the first possible occurrence of the lemma in the text. We refer to those occurrences as "possible" due to the fact that for each word, we extracted all the possible lemmas of the word and each of them was considered in the recency-vectors creation, but unfortunately only one lemma is the correct lemma of the word so the occurrences that were calculated from the other lemmas are erroneous. A real lemmatization of both text versions, which was left for future work, would resolve it. Figure 5 shows some recency-vectors in a given English text.

> "...*Having taken note* of the report of the Global **Environment** Facility to the Conference of the Parties,
>
> *Noting* with appreciation the efforts of the Global **Environment** Facility to operationalize the implementation phase of national adaptation programmes of action,
>
> *Noting* with appreciation the progress made so far in the preparation of national adaptation programmes of action,
>
> *Welcoming* financial and technical assistance to support least **developed** country Parties to integrate climate change issues into their **development** processes under Article 4, paragraph 1 (f) of the Convention…"
>
> Here are some examples for recency-vectors -
>
> | Lemma | Recency Vector |
> |---|---|
> | develop | 63, 10… |
> | environment | 10, 26… |

Figure 5 - English recency-vector examples of a given English text, taken from our corpus

Once the recency-vectors had been captured in both versions of the text, the first step of *DK-Vec* is to find a matched English recency-vector for each Arabic one. This step is called *matching*. The second step is extracting anchors from each matched recency-vectors pair.

4.2.2 *DK-Vec* Matching

In the matching step, the algorithm searches the best match for each Arabic recency-vector. A best match for an Arabic recency-vector is the most similar English one, in the sense of length and values. The basic assumption here is that if an Arabic word σ is translated to an English word γ, their distribution in both texts is expected to be similar. The first issue is to filter out poor English recency-vector candidates. Candidates that match any of the following events were discarded:

1. **Length event** - the ratio between the frequencies of the Arabic lemma and the candidate English lemma exceeds 2. That means that the size of one of the recency-vectors is as twice as the other recency-vector. This event is suggested

originally by Fung and McKeown in their paper and also implemented by Choueka, Conley, and Dagan.

2. **Large distance event** - the distance between the first occurrences of both lemmas is equals or greater than half of the entire document text length (measured by words count). This factor was also suggested by Fung and McKeown in their paper.

3. **Semantic distance event** – this event does not comply with the original *DK-Vec* approach. In our version, we are only keeping candidates that their lemmas are related to each other, semantically. Actually, for each candidate, we also check the semantic relation of the lemmas by lookup the English equivalent of the Arabic lemma in a bilingual lexicon, expand both lemmas with synonyms from WordNet and create two corresponding synonyms lists and then check whether those lists are intersected. The lexicon is represented by the glossary entries supplied by the Buckwalter morphological analyzer. For example, taking the recency-vector that represents the lemma سلام (*slAm*, "peace"), lookup its English equivalent in the lexicon returns the lemma *peace*. Expanding it with synonyms form WordNet generates the following list: *ataraxias, greet, greeting, heartsease, military_greeting, pacification, peace_of_mind, peace_treaty, peacefulness, pledge, present, public_security, recognize, repose, Sallam, salutation, salute, serenity, toast, wassail*. Now, every English vector that contains one of these lemmas in its corresponding synonyms list, is a possible candidate. Currently, we deal only with one word lemmas (for instance, in the last example, *peace_treaty* is discarded). Applying this event, causes a dramatically reduction of candidates

number, but also improves significantly the reliability of the resulted anchors. As mentioned above, since do not use *DK-Vec* for word-level alignment neither for dictionary generation, we can afford such reduction.

For the remaining candidates of each Arabic recency-vector, we calculated a matching-score in the same way as described by Fung and McKeown and also by Choueka, Conley, and Dagan. Let v_s be the recency-vector of an Arabic lemma l_s and let v_t be the recency-vector of an English lemma l_t. By definition, $v_s[i]$ is the distance between the $i-1$ th and i th occurrences of l_s in the Arabic version and $v_t[j]$ is the distance between the $j-1$ th and the j th occurrences of l_t in the English side. The matching-score should reflect the similarity of the distances $v_s[i]$, $v_t[j]$. In other words, the best English vector candidate should have the smallest values of $|v_s[i]-v_t[j]|$. Given a pair of recency-vectors, we could find its matching-score by calculating the basic one-to-one Euclidean distance of the positions of the vectors, but due to the linearity of that metric, sometimes it compares distances that do not correspond. Fung and McKeown suggested using the *Time-warping* method instead. This method solves this problem by recovering the optimal alignment between the distances. It is called time-warping because it warps the time axes of the two vectors in such a way that corresponding distances appear at the same location. Figure 6 shows it graphically.
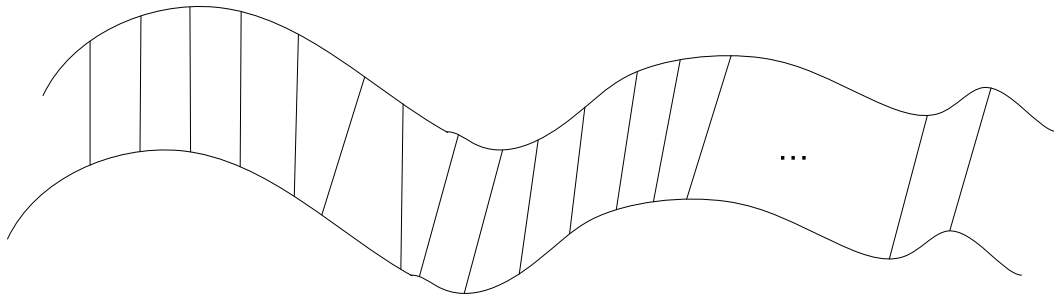
Figure 6 - Time Warping distance of two vectors

The calculation is done in a dynamic programming fashion, which is called Dynamic-Time-Warping (DTW). On each iteration, we calculate the value $C[i,j]$, which represents a partial matching-score of the candidate; actually $C[i,j]$ is the matching-score of positions $1,2,...,i$ and $1,2,...,j$ of v_s and v_t, respectively. Easily we conclude that $C[0,0] = 0$ since it represents the score of zero matches. For $0 < i \leq size[v_s]$ and $0 < j \leq size[v_t]$, we compute $C[i,j]$ using the following recursive formula:

$$C[i,j] = \left| v_s[i] - v_t[j] \right| + \min \begin{cases} C[i-1,j-1] \\ C[i-1,j] \\ C[i,j-1] \end{cases}$$

In other words $C[i,j]$ is the minimum cumulative score of the positions i th and j th in the vectors, among the adjacent positions. By looking on $C[i,j]$ as a cell in a matrix with v_s and v_t as its axes, the computation is illustrated in Figure 7. We see that each cell in the path chooses its position according to which of the three options that it has to arrive from, minimizes the cumulative score.
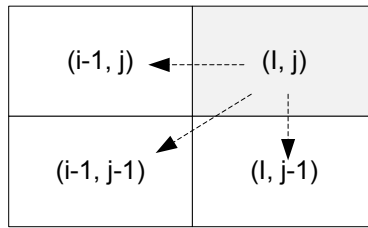
Figure 7 - Choosing the best option that minimizes the cumulative score in dynamic time warping algorithm

The calculation is continued bottom-up until reaching $C[size[v_s], size[v_t]]$ which is the final matching-score of the vectors v_s and v_t. The best match for an Arabic recency-vector v_s is the English recency-vector v_t that achieves the maximum matching-score $C[size[v_s], size[v_t]]$ among all candidates.

4.2.3 Extracting Anchors

Finding the English recency-vector that matches a given Arabic vector does not mean that all the positions in both vectors will be matched and taken as anchors. We discarded positions that were matched by cases 2 or 3 (i.e., $C[i-1, j]$ or $C[i, j-1]$). However, first case positions (i.e., $C[i-1, j-1]$), were extracted and marked as anchors.

4.2.4 Extracting Corresponding Paragraphs

The anchors that were extracted in the previous step are used to find the corresponding English paragraph for each Arabic paragraph. Given a parallel document, we divided the Arabic and the English versions into paragraphs. Fortunately, this was an easy task since the conversion of the documents from MS-Word into text places each paragraph in a single line. In the English version there was a problem only with paragraphs that were crossed a page transition in the MS-Word version, so we manually resolved such situations.

For each Arabic paragraph we matched the English paragraph that maximizes the number of common anchors. In case that more than one English paragraph has the same number of common anchors with a specific Arabic paragraph, we chose the closer one with respect to the beginning positions of the paragraphs in the document. A special word ratio parameter was used to filter unreliable paragraphs pairs. Only paragraphs that the ratio between the number of Arabic words and the number of English words is in the range 0.6 – 0.9 were used. The range was determined relying on the Language Proportion Coefficient (LPC) suggested by Choueka, Conley, and Dagan for Hebrew-English pair.

Matched paragraph pairs were morphologically analyzed, part-of-speech tagged and inserted to the corpus of translation-examples. A special look-up table that maps the Arabic words to their corresponding English words in each parallel paragraph was also created.

Chapter 5
Translation Algorithm

5. Chapter 5 - Translation Algorithm

This chapter describes the algorithm we used to develop our automated translation system. We restricted the input to be a short Arabic sentence, taken from unseen United-Nations documents published on the ODS system. The entire translation process is performed in steps. First, the input sentence is preprocessed with various morphological tasks. Then, in the *matching* step, the system searches our corpus for already translated fragments of the input sentence. The third step is *transfer*, which is the generation of the translation of each matched fragment and on the last step, *recombination*, the system generates the complete English translation, pasting together the translated fragments. In this chapter we will explain thoroughly each step.

5.1 Preprocessing and Data Preparation

Given an input Arabic sentence, first we analyze it with the Buckwalter morphological analyzer. The analyzer transliterates and analyzes each word separately, to produces all the possible morphological analyses of that word. Each analysis contains stem, lemma, comprised morphemes with their part-of-speech tags and English glossaries. The input sentence is also tagged using the SVM-POS tool, so we are considering only those analyses with the correct part-of-speech tag. For instance, if a given word is tagged as noun by the SVM-POS tool, we are considering only noun analyses of the Buckwalter analyzer. SVM-POS uses a modified version of the Penn Treebank tagset (see Table 1) and Buckwalter analyzer uses a different larger one, so we implemented a special converter module that matches Buckwalter tags to/from the corresponding Penn Treebank tags. The Buckwalter tagset is larger and more specific since it also tags word's affixes.

Therefore, for each word, we considered only its main part-of-speech tag, the tag of the stem.

The results of the preprocessing step are all the relevant analyses of each word organized as XML stream as can be shown in Figure 8.

```
<word index="1" word="AlqDA'">
    <analysis num="1" lemma="qaDA'_1" gloss="justice|judiciary" POS="DET|NOUN"/>
    <analysis num="2" lemma="qaDA'_2" gloss="extermination" POS="DET|NOUN"/>
    <analysis num="3" lemma="qaDA'_3" gloss="district|province" POS="DET|NOUN"/>
</word>
```

Figure 8 – Example for preprocessing results of the word القضاء

The figure shows the results of a single word القضاء (*AlqDA'*, <different translations are given in the figure>) taken from an input sentence.

It is worth mentioning that a similar processing step is performed on each translation-example that stored in our corpus. In the same way, our translation-examples stored with the same XML data representing the relevant analyses of their Arabic words. The English version of each translation-example is also analyzed and the relevant information is stored exactly in the same format. The tools for the English part-of-speech tagging and morphological analysis are described in Chapter 3.

5.2 Matching

In the matching step, we search our corpus for translation-examples that their Arabic version match fragments of the preprocessed input sentence. A matched fragment must contain at least two input sentence adjacent words. Congruent fragments are possible. For instance, given the input sentence: مذكرة من رئيس مجلس الأمن ("a memorandum by the president of the Security Council"), we summarize the fragments that were found in Table 2.

Fragment	Translation of the fragment
مذكرة من رئيس مجلس الأمن	`by the president of the security council`
مذكرة من رئيس مجلس الأمن	`a memorandum by`
مذكرة من رئيس مجلس الأمن	`a memorandum by the president of the security council`
مذكرة من رئيس مجلس الأمن	`a memorandum by the president`
مذكرة من رئيس مجلس الأمن	`a memorandum by the president of council`
مذكرة من رئيس مجلس الأمن	`the president of the Security Council`
مذكرة من رئيس مجلس الأمن	`the security council`
مذكرة من رئيس مجلس الأمن	`president of the council`
مذكرة من رئيس مجلس الأمن	`by the president of the council`
مذكرة من رئيس مجلس الأمن	`by president`

Table 2 –Fragments that were found for the sentence: مذكرة من رئيس مجلس الأمن

The same fragment can be found in more than one translation example, therefore a special *matching-score* is given for each pair of fragment and its matched translation example. The matching-score represents the matching quality of the fragment in the specific translation example. Fragments are matched word by word so the score for a fragment is calculated from the individual word matching-scores. Words are matched on levels, with each level assigned a different score. The levels are *text*, *stem*, *lemma*, and *part-of-speech*. Text (exact strings) and stem matches credit the words with the maximum as possible, lemma match credits less and part-of-speech level credits the fragment with the minimum amount. Table 3 summarizes the several word matching levels we used.

Match Level	Description and Example
Text Match (Surface Form)	Exact match of the words, The words are exactly the same.
Stem Match	The words match in their stems but not in the surface form. For example the following words match in their stems: 1. الدستورية (*Aldstwryp*, "the constitutionality") 2. دستوريتي (*dstwryty*, "my constitutional") The stem of both words is دستوري (*dusotuwriy*)
Lemma Match	Words share a lemma. For instance, the following words match in their lemmas: 1. مارق (*mAriq*, "apostate") 2. مراق (*mur~Aq*, "apostates") Note that the stems of these words are not equal.
Content Match	This level is planned but not yet implemented. The idea is that, for example, two location names would get a higher score than two dissimilar proper nouns.
Part-Of-Speech Match	The words match only in their part-of-speech tag. For instance, both words are nouns. Actually, we also require that both words will have the same tags for their affixes. For example, if a word is tagged as a noun and it has the definite article ال (*Al*, "the") attached at the beginning of the word, the matched word must agree on both features –

	to be a noun and to have the definite article attached at the beginning.
Common Word Group Match	This level is relevant only for common words and affixes, taken from a predefined list. These words/affixes were organized in groups that represent the same meaning. Clearly, a word/affix may be a member of more than one group. Words/affixes that are members of the same group are also matched on this level. For example the prefix ب (*b*, "with", "by", "in") is in the same group of the preposition في (*fy*, "in").

<p align="center">Table 3 –Match levels for words</p>

5.2.1 Fragment Score Calculation

The matching-score for a fragment is calculated by the individual word matching-scores.

The matching-score for a single word match varies from 0, in case of completely mismatch, to 1, in case of an exact match. The fragment score is simply the average of the words matching-scores. Table 4 shows the scores for the previously presented match levels.

Match Level	Score
Text match	1
Stem match	0.9
Lemma match	For lemma match we calculate a dynamic

	score in the following way:
	For every possible lemmas of the first word, we create pairs with all the possible lemmas of the second word. A matched pair is a pair that both lemmas are equal. The score is the rate between the number of matched pairs and the total number of pairs.
Content match	0.8
Part-of-speech match	0.3
Common word group match	1

<div align="center">

Table 4 –Matching level scores

</div>

Text and stem match have almost the same score for now since currently, we have not handled the translation modification in such situation. There are of course complicated situations when both words have the same stems but not the same lemmas, for example, the words كتب (*katab*, "wrote") and كتب (*kutub*, "books"). Such cases are not yet handled since we have not worked with a context sensitive Arabic lemmatizer and so cannot derive the correct lemma of an Arabic word. Still, the combination of the Buckwalter morphological analyzer and the SVM-POS tagger allows us to reduce the number of possible lemmas for every Arabic word so as to reduce the amount of ambiguity. Actually, by lemma match we mean that the words were matched on any one of their possible lemmas. The matching-score in such case is the ratio between the number of equal lemmas, and the total number of lemma pairs (one of each word). Further

investigation as well as developing and working with a context sensitive Arabic lemmatizer is needed in order to better handle all such situations.

For instance, let us take again the previous input sentence مذكرة من رئيس مجلس الأمن (*m*krp mn r<ys mjls AlAmn*, "a memorandum by the president of the Security Council") and assume that our system founds a match for the pattern مجلس الأمن (*mjls AlAmn*, "the Security Council") in a translation example that contains the text:

<div dir="rtl">… ويعين <u>مجلس الوزراء</u> أعضاء اللجنة ويجري…</div>

The translation example pattern that matches the source pattern is مجلس الوزراء (*mjls AlwzrA'*, "government"). The total fragment score is calculated and presented in Table 5.

Tokens	Match Level	Score
مجلس, مجلس	Text Match	1
الوزراء, الأمن	Part-Of-Speech Match (noun + determiner)	0.3

Table 5 –Word level matching scores for the fragment مجلس الوزراء (*mjls AlwzrA'*, **"government"**)

So the fragment score is simply the average $(1 + 0.3)/2 = 0.65$

Fragments with a score below some predefined threshold are discarded, since passing low-score fragments to the next step dramatically increases total running time. Note that a larger corpus, with the concomitant increase in the number of potential fragments, would require raising the threshold. Reported results in this paper are based on a corpus containing 13,500 translation examples with threshold value of 0.5.

5.2.2 Fragments Storing

Fragments are stored in the following structure:

source pattern – fragment's Arabic text, taken from the input sentence;

example pattern – fragment's Arabic text, taken from the matched translation example;

example – the matched translation example;

matching score – the matching-score of the fragment to the translation example.

Fragments with the same example pattern are collected and stored in a higher-level structure called *general-fragment*. Note that a general-fragment with only one fragment is also possible.

5.3 Transfer

The input for the transfer step is all the general-fragments that were found in the matching step, and the output of that step is the translations of those general-fragments. The translation of a general-fragment is the best generated translation among the fragments of the general-fragment. Translating a fragment is done by extracting its translation from the English version of the translation-example and performing some modifications on it. We can summarize the translation process by the following two steps:

- Extraction of the translation of the fragment's example-pattern by aligning the translation-example on the word level, using a bilingual lexicon.

- Fixing the extracted translation so that it will be the translation of the fragment's source-pattern.

5.3.1 Step 1 – Translation Extraction

The first step is to extract the translation of the fragment's example-pattern from the English version of the translation-example. Actually, as mentioned above, for each Arabic word in the pattern, we look up its English equivalents in the lexicon and expand those with synonyms from WordNet. Then we search the English version of the translation example for all instances on the lemma level. Finally, we take the <u>shortest</u> English segment that contains the <u>maximum</u> number of aligned words. Since finding all

instances of every Arabic word in the English version of the translation example is a huge time consumer task, we decided to perform it offline and store the results also in the corpus. Usually a word in some Arabic example pattern has several English equivalents, which makes the translation extraction process complicated and error prone. For this reason, we also restrict the ratio between the number of Arabic words in the example pattern and the number of English words in the extracted translation, bounded them by a function of the ratio between the total number of words in the Arabic and English versions of the translation example. Figure 9 demonstrates the translation extraction process. The following sections explain each step thoroughly.

Arabic version of the translation example	الخدمات الاستشارية والتعاون التقني في ميدان حقوق الإنسان
English version of the translation example	ADVISORY SERVICES AND TECHNICAL COOPERATION IN THE FIELD OF HUMAN RIGHTS
Alignment table (created offline using the Buckwalter glossary entries and WordNet)	SERVICES الخدمات ADVISORY الاستشارية COOPERATION والتعاون TECHNICAL التقني IN في FIELD ميدان RIGHTS حقوق HUMAN الإنسان
example pattern	ميدان حقوق الإنسان
Translation extraction	First, we mark the alignments of the relevant example

process	pattern words, using the alignment table: ADVISORY SERVICES AND TECHNICAL COOPERATION IN THE **FIELD** OF **HUMAN RIGHTS** Now, we take the shortest segment with maximum number of word alignments: ADVISORY SERVICES AND TECHNICAL COOPERATION IN THE **FIELD OF HUMAN RIGHTS** At this point of the research, we check if a determiner exists one place before the beginning of the extracted segment. If the answer on this question is true, we add it to the beginning of the extracted segment. Later, as future work, we should analyze the Arabic pattern to find if it requires the determiner. The same approach is taken for some prepositions from a predefined list. Finally we extract the translation **THE FIELD OF HUMAN RIGHTS.**

Figure 9 –Translation extraction demonstration

This is of course only a simple demonstration. As mentioned, more complicated one would contain more than one alignment for each Arabic word, as will be explain shortly.

5.3.1.1 First Task – Finding English Equivalents

This task is performed offline, at the corpus creation time. Each translation example is stored together with a look-up table that maps between every Arabic word in the translation example and all its English equivalents in the English version of the example. An English equivalent may be a word or a phrase that is a possible translation of the Arabic word. An Arabic word may have more than one English equivalent in the English version of the translation example. We use the extracted relevant Buckwalter glossaries for each Arabic word and expand these with synonyms from WordNet. Actually, a glossary entry might be composed of more than one word, so we find the synonyms for each one of them, except of common words identified from a predefined list.

Now, after creating the synonym lists, next we search the English version of the translation example for all instances on the lemma level. Of course the English equivalents may be found more than once in the English version of the translation example, that is the reason for taking the shortest English segment with maximum number of word alignments as will be explain in the following step.

There are several complicated situations in finding the English equivalents. We are now describing some of them:

1. The Arabic word is part of a phrase that is translated to a single word in the English version. Such situations are partly handled by capturing some basic Arabic language structures. For instance, let us take the Arabic example pattern غير رسمي (*gyr rsmy*, "not formal"), which -- in many cases -- we might find it as "informal" rather than "not formal". Neither the synonym list of the word رسمي (*rsmy*, "formal") nor the list of the word غير (*gyr*, "not") contain the word

"informal"[1]. Handling such situation is done by a manually defined rule that is triggered whenever the system encounter the word غير (*gyr*, "not"). The system checks the following word, and -- instead of building a synonym list -- builds an antonym list, using WordNet. In this example, the word "informal" appear as an antonym of the word "formal" in WordNet. There are more complicated structures, which are not handled yet, but capturing and writing rules for such situations seems quite feasible.

2. The translation of an Arabic word is expressed as a short phrase in the English version, for example verb and additional particle, as in the pattern يتخذ التعاون (*ytx* AltEAwn*), which was translated in one of our translation examples to "the cooperation should take on". We see that the word يتخذ (*ytx**, "take") is translated to "take on". This is partially solved using the large number of English phrases and particles list for every verb contained by WordNet. We defined a *window* around each extracted English equivalent in the translation example. The left boundary of the window is placed before the location of the equivalent and the right boundary is placed after it. The size of the window is the total words taken from the left border to the right border, ignoring the equivalent text itself, in a way that the number of words on both sides is equal. For example, consider the sentence: "promoting participation of developing countries in new and dynamic sectors of world trade". Assume that the extracted equivalent text is the word "countries". A six-word window is the text: "participation of developing **countries** in new and". We check the existence of English phrases that contain the

---

[1] Recall that we search the English version on the lemma level, and since we lemmatize the English text using WordNet, the lemma of the word "informal" is also "informal".

word "countries" inside the window. First we check the phrase that is composed of the word "countries" concatenated with the next word. Then, we concatenate the next word and check again and so on until reaching the right boundary of the window. The same process is done backward by checking "countries" with the previous word and so on until reaching the left boundary of the window. Eventually, we extract the larger phrase that was found. Large windows are better for dealing with long phrases but also increase running time dramatically. Empirically, we are working with four-word windows.

3. There are situations were an Arabic phrase is translated to an English phrase, but the individual translation of each Arabic word is different. For instance, the phrase إنهاء الاستعمار (*<nhA' AlAstEmAr*) usually is translated to "decolonization" and not to the individual translation of each word: إنهاء *(<nhA'*, "termination") and الاستعمار (*AlAstEmAr*, "the colonialism"). In addition, there are phrases that are always translated to the same English phrase, or maybe to a finite set of phrases, for example, the phrase من أجل (*mn >jl*) may translated to "in order to" or to "because of". Further investigation is needed for handling such cases. For instance, the latter cases may be handled using a look-up table that map Arabic phrases to their corresponding English phrases.

4. As mentioned earlier, sometimes we have problems with determiners or other functional words that should be placed at the beginning of the extracted phrase. This happens especially in noun-phrases. For example, consider the pattern from Figure 10, ميدان حقوق الإنسان ("the field of human rights"). The determiner ال (*Al*, "the") is on the word الإنسان (*AlAnsAn*, "the human") but in English the determiner

is on "the field". Currently, when we find a text with a preceding determiner, we always extract the text with it. Unfortunately, other functional words are not handled yet. In the future, we are planning to analyze Arabic noun-phrases to find whether they determined (or related by other functional words) and extract the text respectively.

5.3.1.2 Second Task – Fragment Alignment

The second task is to extract the translation of the fragment using the equivalent texts we have just found. Basically, this is done in the following two steps:

1. Extraction of all possible *sequences* using the corresponding English equivalents. A sequence is an English segment, extracted from the English version of the translation example and contains (all/part of) the equivalents of the Arabic pattern words.

2. Choosing the <u>shortest</u> English segment that contains <u>maximum</u> number of equivalents, as the translation of the Arabic pattern.

5.3.2 Step 2 – Fixing the Translation

After extracting the translation of the pattern, sometimes there is a need to modify it. Remember that the match of an example pattern to the input pattern is not always exact, so sometimes more work needs doing, to get a real translation. In this section we will discuss several modification issues.

5.3.2.1 Dealing with Words that were Matched on the Part-Of-Speech Level

Recall that the match of a corpus fragment to the input fragment can be inexact: words may be matched on several levels. Exactly matched words are assumed to have the same translation, but stem or lemma matched words may require modifications (mostly

inflection and prepositions issues) to the extracted translation. These issues were left for future work. Words matched on the part-of-speech level require complete change of meaning. For example, in Figure 10, we take the input fragment مجلس الامن (*mjls AlAmn*, "the Security Council"), matched to the fragment مسؤولية الامن (*ms&wlyp AlAmn*, "the security responsibility") in some translation example. The words مجلس (*mjls*, "council") and مسؤولية (*ms&wlya*, "responsibility") are matched on the part-of-speech level (both are nouns). Assume that the extracted translation from the translation example is "the security responsibility", which is actually a translation of مسؤولية الامن (*ms&wlyp AlAmn*, "the security responsibility") and is not the translation of the input pattern at all. But, by replacing the word "responsibility" from the translation example with the translation of مجلس (*mjls*, "council") from the lexicon, we get the correct phrase: "the security council". The lexicon is imitated using the glossaries extracted from the Buckwalter morphological analyzer and expanded with WordNet synonyms as was explained above.

Input Phrase:
مجلس الأمن (mjls AlAmn, *the security council*)

⬇

مسؤولية الأمن (ms&wlyp AlAmn, *the security responsibility*)

⬇

the security **_responsibility_** ⟹ **Fixing** ⟹ *The security **_council_***

⬆

| مجلس | council |
| مسؤولية | responsibility |
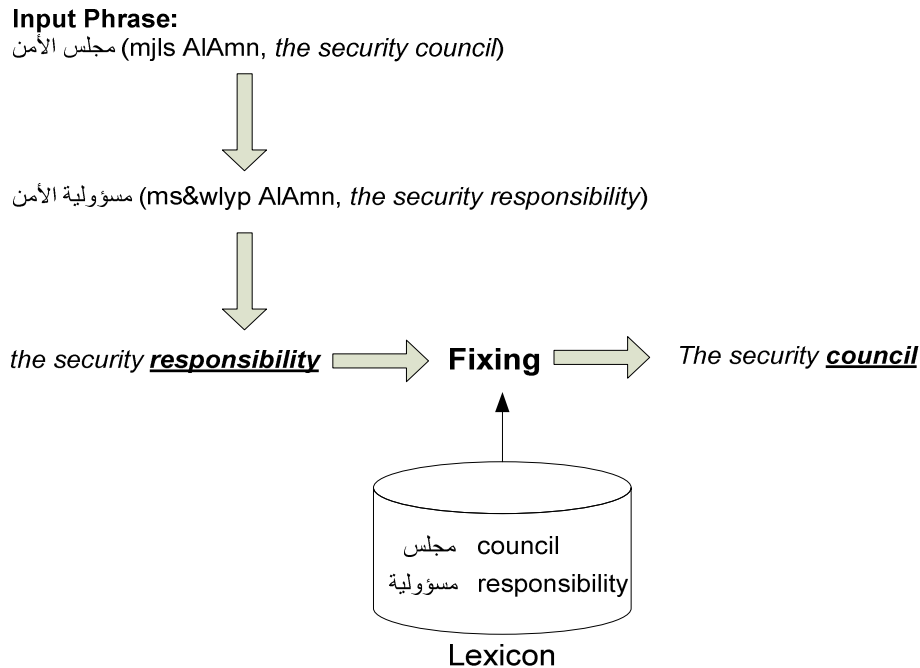
Lexicon

Figure 10 — Fixing part-of-speech level matches

The lexicon is imitated using the glossaries extracted from the Buckwalter morphological analyzer and expanded with synonyms from WordNet as was explained in Section 5.3.1.1.

5.3.2.2 Dealing With Unnecessary Words in the Middle

Sometimes the extracted translation contains extra unnecessary words in the middle. Those words appear mostly because of the different syntax of a noun-phrase in both languages. For example, given the translation-example - موضوع الامن الاقليمي (*mwDwE AlAmn AlAqlymy*) and its translation: "the subject of regional security", and extracting the translation of the pattern موضوع الامن (*mwDwE AlAmn*) we obtain: "the subject of regional security" (since it is the shortest segment that contains maximum word alignments). Clearly, the word "regional" is unnecessary in the translation because it is the translation of the word الاقليمي (*AlAqlymy*, "the regional"), which does not exist in the pattern. So by removing the word from the translation we are getting the correct translation of the pattern. The word "regional" appears in the extracted translation due to the fact that Arabic adjectives come after the nouns, which is the opposite of English syntax. The noun-phrase الامن الاقليمي (*AlAmn AlAqlymy,* "the regional security") is translated so that the translation of the word الاقليمي (*AlAqlymy*, "regional") appears first and then the translation of the word الامن (*AlAmn*, "security"). Identifying such situation is done by searching the translation of the word "regional" in the Arabic text that comes immediately after the pattern in the translation example. Currently we search only three words after the pattern because of time-complexity reasons, but basically in the future we will further investigate the use of an Arabic chunker to find the boundaries of that noun-phrase and search the word within that phrase.

Removing the unnecessary words from the extracted translation must preserve the correct English syntax of the remaining translation, which in some cases seems to be a difficult task. For that purpose, we compiled several rules to deal with different situations. The rules are based on the syntax structure of the English extracted translation to identify cases that need special care. First, we chunk the translation to discover its basic noun-phrases, using the BaseNP [37] chunker. To do that, we first apply Brill's part-of-speech tagger [41] to the translation. By looking on the chunked English text, we check the effects of removing the unnecessary word. In the previous example, removing the word "regional" from the English text: "the subject of regional security" may be done without any further modifications, since by tagging and chunking the segment, we get (phrases in brackets are noun-phrases):

[the/DT subject/NN] of/IN [regional/JJ security/NN],

and "regional" is simply an adjective within a noun-phrase which still has the same head. Prepositions and other function-words (in this case the preposition "of") that relate to the phrase are still necessary, so we keep them.

Consider another example: المراقبة عن المناظمة (*AlmrAqbp En AlmnAZmp*, "the observation on the organization") with the following extracted translation: "observer for the world tourism organization**".** The words "world" and "tourism" should be removed because they are not a translation of any of the Arabic pattern words. So again, by chunking the text we get:

[Observer/NNP] for/IN [the/DT World/NNP Tourism/NNP Organization/NN],

both words "world" and "tourism" are noun-modifiers within a noun-phrase. Since we work with base (non-recursive) noun-phrases, which their head word is located last in the

phrase, we identify that the noun-phrase still exists after the removal of the unnecessary words so we keep the definite article "the" and the preposition "for". By removing both words we almost get the correct translation: "observer for the organization". The pattern "observer for" is not correct, but this happened as a result of a lemma-level matching of the word المراقبة (*AlmrAqbp*, "the observation") and the word مراق (*mrAqb*, "observer").

There are situations when there is a need to remove the head of the noun-phrase. For instance, consider the fragment: الانسان والقانون الدولي (*AlAnsAn wAlqAnwn Aldwly*, "the humans and the international law"). Suppose that the extracted translation is: "to human rights and international law". Note that the word "rights" is unnecessary here so by chunking the translation we get:

to/TO [human/JJ rights,/NN] and/CC [international/JJ law/NN].

As we promised, the word "rights" is the head of a noun-phrase. That situation happens because of the fact that the word "human" is an adjective in the extracted translation, but it is also a noun in the required translation. Although the word "human" should be left as is, currently no work has been done to find out what to do with prepositions and other function words that related that noun-phrase. This and more complicated situations are left for future work.

5.3.3 Choosing a General Fragment's Translation

As mentioned above, a general-fragment may contain several fragments sharing the same Arabic example pattern. Among the extracted translations of the fragments, which are all translations of the same Arabic pattern, we choose the translation that covers the maximum number of Arabic words to represent the general-fragment. A *translation-score* is calculated for the chosen translation, which is actually the ratio between the

number of covered words and the total number of words in the Arabic pattern. For instance, the translation-score of a fragment with the Arabic pattern وخلال الاجتماع الخاص (*wxlAl AlAjtmAE AlxAS*, "and during the special meeting") and a translation: "and during the meeting" is 0.666, since the text "and during" covers the word وخلال (*wxlAl*), the text "the meeting" covers the word الاجتماع (*AlAjtmAE*), but the word الخاص (*AlxAS*) left without any matched English translation, so the score is the ratio between the number of covered words, which is 2, and the total number of pattern words, which is 3.

5.4 Recombination

In the recombination task, we paste together the extracted translations to form a complete translation of the input sentence. The recombination task is generally composed of two subtasks. The first is finding the *N*-best combinations of the extracted translations that cover the entire input sentence and the second is smoothing out the recombined translations into a fully grammatical English sentence. Currently we handle only the first subtask, while the second one left for future work.

5.4.1 The Recombination Algorithm

The input for the recombination step is all the extracted translations of the general-fragments from previous steps. Remember that a general-fragment actually covers a specific fragment of the input sentence, but there may be more then one general-fragment that covers the same one, maybe even with different lengths. The main goal of the recombination algorithm is to choose the best sequence of general-fragments that together combine a full cover of the input sentence. Our recombination algorithm actually provides the *N* best full cover sequences.

The algorithm uses the *total*-score of the general-fragments to recombine the required sequences. The total-score of a general-fragment is calculated by multiplying the previously introduced matching-score of the general-fragment (see Section 5.2.1) and the translation-score (see Section 5.3.3) of the chosen fragment's translation.

The main strategy of our recombination algorithm is greedy: choosing the best general-fragment for every position in the recombined sentence. In other words, we have a number of general-fragments, were each one may have a different length and starting-index (the position of the first covered word within the input sentence). It starts with index i=0, which is the beginning of the input sentence. Now it takes the higher ranked general-fragment among all the general-fragments that start at the same index and also have the same length (in their Arabic pattern they cover). Then it searches the next best general-fragment among all general-fragments that start exactly where the last one ends or is congruent with a single word (the last covered word of the previous general-fragment is the first covered word of the next general-fragment), and have the same length. This process is last until it covers the entire input sentence. Actually, on each step of the algorithm, it chooses several best general-fragments, each one among a different length group.

For example, consider the following input sentence: وخلال الفترة المشمولة بالتقرير (*wxlAl Alftrp Alm$mwlp bAltqryr*, "and during the reporting period"). Table 6 presents all the general-fragments that were found for the pattern وخلال الفترة (*wxlAl Alftrp,* "and during the period"):

Matched Arabic Pattern	Extracted Translation	Total-Score
وفي الفترة	Time	0.325
وخلال فترة	During	0.5

وخلال المناقشة	During	0.325
وخلال الفترة	During the period	1.0
وخلال فترات	The time	0.5
وأثناء الفترة	Period and during	0.65
خلال الفترة	And throughout the period	1.0

Table 6 — All general-fragments that were found for the pattern وخلال الفترة (*waxlAl Alftrp*, "and during the period")

Among all of those general-fragments, which are all of the same length, the recombination algorithm arbitrarily chooses one[2] of the higher ranked ones خلال الفترة (*xlAl Alftrp*, "and throughout the period"). Now, on the next steps it keeps choosing the best general-fragments among those that start either with index 1 or index 2. Table 7 summarizes those steps and presents all the chosen general-fragments from all length groups.

Start Index	General Fragment Cover	General Fragment Translation	General Fragment Score
0	وخلال الفترة	and throughout the period	1.0
0	وخلال الفترة المشمولة بالتقرير	and during the reporting period	0.75
0	وخلال الفترة المشمولة	during the past phase	0.533
1	الفترة المشمولة	included a period	0.65
1	الفترة المشمولة بالتقرير	the reporting period	0.66
2	المشمولة بالتقرير	report	0.5

Table 7 — The chosen general-fragments from all collections that were found for the pattern وخلال الفترة (*waxlAl Alftrp*, "and during the period")

Clearly, the first row is the general-fragment that was chosen in the first step from those of length 2 and start-index 0, and the second row presents another chosen general-fragment with the same start-index but from a different length group.

---

[2] Another option is to iterate separately for each one of the higher ranked general-fragments, but it seems to cause a major increase in running-time. Anyway, further investigation of the entire recombination step was left for future work.

Among the chosen general-fragments, the recombination algorithm generates all the possible sequences. Each sequence is called *translation*. The translations are ranked using a *final-translation-score* that is calculated by multiplying the total-score of the comprised general-fragments. Table 8 shows the five best translations of the previous example. The character "|" marks the beginning positions of a new general-fragment.

Generated Sequence (Translation)	Translation Score
and during the reporting period	0.75
and throughout the period | the reporting period	0.66
and throughout the period | report	0.5
and throughout the period | included a period | report	0.325
during the past phase | report	0.266

Table 8 — Five best translations and translation-scores for the input sentence:
وخلال الفترة المشمولة بالتقرير (*wxlAl Alftrp Alm$mwlp bAltqryr*, **"and during the reporting period"**), **created by our system**

The best among the *N* best translations is the one with the highest rank. The final-translation-score is a multiplication of the comprised general-fragments total-scores since it reduces the score of translations that are composed of several general-fragments. Recall that for now, we are not smoothing out the translations, so their quality reduces as the number of comprising general-fragments increases.

5.5 Example

In this section we summarize the different steps in producing *N* best translations for the sentence وخلال الفترة المشمولة بالتقرير (*wxlAl Alftrp Alm$mwlp bAltqryr*, "and during the reporting period"). As you realized, this may be a complicated task, so we present only the important relevant steps.

Input sentence: وخلال الفترة المشمولة بالتقرير

Translation: "and during the reporting period"

5.5.1 Preprocessing and Data Preparation

In this step, the sentence is analyzed by several tools, as was explained above, producing the following information on each word:

```
<word index="0" word="wxlAl">
    <analysis num="1" lemma="xilAla_1" gloss="during|through" POS="CONJ|PREP"/>
</word>


<word index="1" word="Alftrp">
    <analysis num="1" lemma="fatorap_1" gloss="phase/time period/interval"
                                        POS="DET|NOUN|NSUFF_FEM_SG"/>
</word>


<word index="2" word="Alm$mwlp">
    <analysis num="1" lemma="ma$omuwl_1" gloss="included/contained/implied"
                                        POS="DET|ADJ|NSUFF_FEM_SG"/>
</word>

<word index="3" word="bAltqryr">
    <analysis num="1" lemma="taqoriyr_1" gloss="decision/determination"
                                            POS="PREP|DET|NOUN"/>
    <analysis num="2" lemma="taqoriyr_2" gloss="report/account"
                                            POS="PREP|DET|NOUN"/>
</word>
```

5.5.2 Matching

In the matching step, the system searches for all the relevant Arabic fragments in the corpus. The matching is done in levels (exact, lemma, stem and part-of-speech) as was explained above. Table 9 shows some of the fragments that were found by the system for the given input sentence. The left column contains the input sentence pattern; the middle column contains the matched translation example patterns and the right column provides their matching-score.

Source Pattern	Example Pattern	Match Score
الفترة المشمولة	المسائل المشمولة	0.65
	المنطقة المشمولة	0.65
	الفترة الزمنية	0.65

	الفترة التحضيرية	0.65
	الفترة اللاحقة	0.65
	الفترة المقبلة	0.65
	الفترة الحالية	0.65
	الفترة المشمولة	1.0
	الفترة المالية	0.65
	الفترة الممتدة	0.65
	الفترة المذهلة	0.65
	الفترة الانتخابية	0.65
	الفترة الانتقالية	0.65
	الفترة التالية	0.65
	الفترة السابقة	0.65
	الفترة الاستعمارية	0.65
	الفترة الأخيرة	0.65
	الفترة الحرجة	0.65
	الفترة القصيرة	0.65
الفترة المشمولة بالتقرير	فالفترة المشمولة بالتقرير	1.0
	الفترة المشمولة بالتقرير	1.0
	الفترة المشمولة بالاستعراض	0.76
	الأطفال المشمولة بالاتفاقية	0.53
	للفترة المشمولة بالتقرير	1.0
وخلال الفترة	وفي الفترة	0.65
	وخلال فترة	1.0
	وخلال المؤتمر	0.65
	وخلال المناقشة	0.65
	وخلال الفترة	1.0
	وخلال فترات	1.0
	وأثناء الفترة	0.65
	خلال الفترة	1.0
المشمولة بالتقرير	المشمولة بالاتفاقية	0.65
	المشمولة بالتقرير	1.0
	المشمولة بالاستعراض	0.65
وخلال الفترة المشمولة بالتقرير	وخلال الجزء الأول للدورة	0.475
	وفي الفترة المشمولة بالتقرير	0.825
	وفي الفترة المشمولة بالاستعراض	0.65
	وطوال الفترة المشمولة بالتقرير	0.825

	فخلال الفترة المشمولة بالتقرير	1.0
	خلال الفترة المشمولة بالتقرير	1.0
وخلال الفترة المشمولة	وخلال الجولة الرابعة	0.53
	وخلال العام الماضي	0.53
	وخلال الأعوام العديدة	0.53
	وخلال الأشهر القليلة	0.53
	وخلال الاجتماع الخاص	0.53
	وخلال الصيف الماضي	0.53
	وخلال الفترة المشمولة	1.0
	خلال الفترة التحضيرية	0.76
	خلال الفترة الممتدة	0.76
	خلال الفترة الانتقالية	0.76
	خلال الفترة التالية	0.76
	خلال الفترة المشمولة	1.0
	خلال الفترة الأخيرة	0.76

**Table 9 – Fragments that were found by the system for the input sentence وخلال الفترة المشمولة بالتقرير
(*wxlAl Alftrp Alm$mwlp bAltqryr*, "and during the reporting period")**

Some of the patterns were found in more than one translation example. Those were collected together to a general-fragment.

5.5.3 Transfer

In the transfer step we find the translation for each general-fragment. The translation is created by first extracting the translation of the pattern from the matched translation examples and then fixing the translation so it will actually be a translation of the input pattern. For instance, let us take the general-fragment with the example-pattern الفترة الانتخابية (*Alftrp AlAntxAbyp*, "the electoral period") that was matched to the input pattern الفترة المشمولة (*Alftrp Alm$mwlp*, "the included period"). First by extracting its translation from the translation examples we get: "electoral period". The second step is to fix the translation so that it will be the translation of the input pattern. We notice that the word الانتخابية (*AlAntxAbyp*, "electoral") from the example pattern was matched on the part-of-

speech level to the word المشمولة (*Alm$mwlp*, "included") so its translation should be replaced with the translation of the word المشمولة (*Alm$mwlp*, "included") using the Buckwalter analyzer as explained above. As a result, the final translation of the general-fragment is: "included period". Table 10 summarizes the translations of all the general-fragments that were found in the previous matching step.

Input Pattern	Example-Pattern	Translation	Translation-Score	Total-Score
الفترة المشمولة	المسائل المشمولة		0.0	0.0
	المنطقة المشمولة	included in the area	0.8	0.52
	الفترة الزمنية	included	1.0	0.65
	الفترة التحضيرية		0.0	0.0
	الفترة اللاحقة	with the phase out	0.5	0.325
	الفترة المقبلة	in the period to come	1.0	0.65
	الفترة الحالية	to formulate	0.5	0.325
	الفترة المشمولة	carrying	0.5	0.5
	الفترة المالية		0.0	0.0
	الفترة الممتدة	the period	0.5	0.325
	الفترة المذهلة	included and terrifying period	1.0	0.65
	الفترة الانتخابية	included period	1.0	0.65
	الفترة الانتقالية	the transitional period	1.0	0.65
	الفترة التالية	period of 12 months included	1.0	0.65
	الفترة السابقة	period	0.5	0.325
	الفترة الاستعمارية	period	0.5	0.325
	الفترة الأخيرة	included	0.5	0.325
	الفترة الحرجة	included period	1.0	0.65
	الفترة القصيرة	included a period	1.0	0.65
الفترة المشمولة بالتقرير	فالفترة المشمولة بالتقرير	report	0.33	0.33
	الفترة المشمولة بالتقرير	the reporting period	0.66	0.66
	الفترة المشمولة بالاستعراض	the period under decision	0.66	0.51
	الأطفال المشمولة		0.0	0.0

	بالاتفاقية			
	للفترة المشمولة بالتقرير	report	0.33	0.33
وخلال الفترة	وفي الفترة	time	0.5	0.325
	وخلال فترة	during	0.5	0.5
	وخلال المؤتمر		0.0	0.0
	وخلال المناقشة	during	0.5	0.325
	وخلال الفترة	during the period	1.0	1.0
	وخلال فترات	the time	0.5	0.5
	وأثناء الفترة	period and during	1.0	0.65
	خلال الفترة	and throughout the period	1.0	1.0
المشمولة بالتقرير	المشمولة بالاتفاقية		0.0	0.0
	المشمولة بالتقرير	report	0.5	0.5
	المشمولة بالاستعراض	decision	0.5	0.325
وخلال الفترة المشمولة بالتقرير	وخلال الجزء الأول للدورة		0.0	0.0
	وفي الفترة المشمولة بالتقرير	the reporting period	0.5	0.4125
	وفي الفترة المشمولة بالاستعراض	the period under decision	0.5	0.325
	وطوال الفترة المشمولة بالتقرير		0.0	0.0
	فخلال الفترة المشمولة بالتقرير		0.0	0.0
	خلال الفترة المشمولة بالتقرير	and during the reporting period	0.75	0.75
وخلال الفترة المشمولة	وخلال الجولة الرابعة		0.0	0.0
	وخلال العام الماضي	during the past phase	1.0	0.53
	وخلال الأعوام العديدة	phase	0.33	0.177
	وخلال الأشهر القليلة		0.0	0.0
	وخلال الاجتماع	during the meeting	0.66	0.35

	الخاص			
	وخلال الصيف الماضي	during the past	0.66	0.35
	وخلال الفترة المشمولة		0.0	0.0
	خلال الفترة التحضيرية		0.0	0.0
	خلال الفترة الممتدة	included range of participants at this stage	0.66	0.51
	خلال الفترة الانتقالية	and during the transition period	0.66	0.51
	خلال الفترة التالية	period of 12 months included	0.66	0.51
	خلال الفترة المشمولة		0.0	0.0
	خلال الفترة الأخيرة	included	0.33	0.25

Table 10 –System translations for the general-fragments that were found for the input sentence: وخلال الفترة المشمولة بالتقرير (*wxlAl Alftrp Alm$mwlp bAltqryr*, **"and during the reporting period")**

The input pattern and example pattern columns are the same as the previous table. The translation column presents the translations that were created for each general-fragment. The translation-score and the total-score are also given.

5.5.4 Recombination

In this step the system pastes together the translations of the general-fragments in order to produce the final translation of the input sentence. As mentioned above, currently we are not smoothing out the recombined translation to form a coherent grammatical English sentence. The recombination step is described in Section 5.4. Table 11 presents the final *N* best translations with their scores.

Translations	Score
and during the reporting period	0.75
and throughout the period | the reporting period	0.66
and throughout the period | report	0.5

and throughout the period | included a period | report	0.325
during the past phase | report	0.266

Table 11 – Final five best translations and total scores for the input sentence:
وخلال الفترة المشمولة بالتقرير (*wxlAl Alftrp Alm$mwlp bAltqryr*, **"and during the reporting period"**)

Chapter 6
Experimental Results and Conclusions

6. Experimental Results and Conclusions

In this chapter we present some experimental results, future work and conclusions.

6.1 Results

Experiments were conducted on a corpus containing 13,500 translation examples. The following results are based on 400 Arabic short sentences (5.5 words per sentence on average) that were taken from unseen documents of the United-Nations inventory. The ten best results were evaluated by some of the common automatic criteria for machine translation evaluation, although our system does not perform the entire translation process (discarding the important smooth out task, as explained). All evaluations computed case-insensitive (lower-case), with no punctuation marks and substituting hyphens '-' with blank space.

BLEU score [42]: Comparing a mechanical system's translation to human-produced reference translations, the BLEU score uses unigram, bigram, trigram and fourgram co-occurrence precision in combination with a brevity penalty for short sentences. We evaluated our results using only two translation references. We expect a significant improvement of the results as the number of references will increase. The fact that our system may recombine translated fragments that intersect at a single word and the lack of the smoothing-out step, means that sometimes translations are not fluent enough, so in addition to the common 4-gram BLEU score, we also include 1,2,3-grams BLEU score.

NIST score [43]: This score is similar to BLEU. It also uses one- to four-gram co-occurrence precision, but it takes the arithmetic mean of the n-gram counts. Both BLEU and NIST measure accuracy, so higher scores are better.

METEOR score [44]: This score is based on unigram co-occurrence precision. Here, unigrams can be matched based on their surface forms, stemmed forms and also their meanings taken from WordNet.

Table 12 presents the results for the experiments.

	BLEU				NIST	METEOR
	1-gram	2-gram	3-gram	4-gram		
Best translation chosen by the system	0.4421	0.3183	0.2375	0.1849	4.1792	0.4851
Best translation chosen by a human referee	0.5229	0.3960	0.3100	0.2488	5.1281	0.5363

Table 12 – Evaluation of 400 Arabic UN sentences (5.5 words per sentence in average)

The first row of the table presents evaluations of the system's *highest* ranked result for each input sentence. The second row shows the same evaluation but for the best result that was chosen by a human referee from the ten best results reported by the system for each input sentence. In most cases, the best result that was chosen by the referee had a close (or even the same) translation-score as the system's best result.

Just for example, Table 13 shows the 2005 report of the annual NIST evaluation event for machine translation systems that translates Arabic text to English [45]. Participating systems were required to translate 100 Arabic articles into English. Each system was evaluated by the BLEU score comparing it to four sets of high-quality independently generated human translation.

Site	BLEU (4-gram) Score
GOOGLE	0.5131
ISI	0.4657
IBM	0.4646
UMD	0.4497
JHU-CU	0.4348
EDINBURGH	0.3970
SYSTRAN	0.1079
MITRE	0.0772
FSC	0.0037

Table 13 –The report from 2005 of the annual NIST evaluations for machine translation systems that translates Arabic text to English

Of course, at this point it is impossible to compare those scores to ours, since, besides the fact that our system is not completed yet, we also evaluated the system on a different set of sentences.

As mentioned, currently, we did not consider running time as an important factor, so translating a sentence may sometimes take an unreasonable amount of time. This happens especially due to the fact that during the matching step, we pass many fragment candidates to the transfer step for translation. (Actually, in our 400-sentence experiment, the average number of fragments given to the transfer step per input sentence is about 1900.) Transfer is a time-consuming step, since it uses several algorithms for different cases, and some of them involve calling external time consuming tools, such as WordNet and the base-phrase chunker. Surely, more work needs to be done to dramatically reduce the amount of time that is required for a simple translation.

6.2 Future Work

The first planned issue is to enrich and enlarge the existing corpus and check how it will affect system's results. We are also checking the possibility of buying an existing sentence-aligned bilingual corpus.

We will further investigate the matching and transfer steps in order to find a way to reduce the number of fragment candidates for the transfer step, so that the entire running time will be significantly reduced.

On the Arabic side, we will investigate many complicated situations that have not yet handled; on the English side, the most important issue is the development of the missing last recombination step.

When we finish implementing all the components, especially the recombination step, we will be able to participate in the NIST annual evaluation event, and compare our results with other existing Arabic systems.

6.3 Conclusions

We believe that we have demonstrated the potential of the example-based approach for Arabic, with only minimum investment in Arabic syntactical and linguistic issues.

We found that matching fragments on the level of lemma and stem, and also part-of-speech, enabled the system to better exploit the small number of examples in the corpus we used.

As mentioned, more work is needed to enlarge and enrich the corpus, as well as rules to deal with various problematic situations that are not yet handled. This all appears quite feasible.

Finally, we do not claim that the example-based method is sufficient to handle the complete translation process. It seems that, for Arabic, it should work together with some kind of a rule-based engine as part of a multi-engine system, so as to better handle more complicated situations.

References

[1] Y. Bar Hillel, "The Present Status of Automatic Translation of Languages". Advances in Computers 1, pp. 91-163, 1960.

[2] B. Vauquois, "A Survey of Formal Grammars and Algorithms for Recognition and Translation in Machine Translation". FIP Congress-68, Edinburgh, pp. 254-260, 1968.

[3] B. J. Dorr, P. W. Jordan and J. W. Benoit, "A Survey of Current Paradigms in Machine Translation ". Advances in Computers, Vol. 49, M. Zelkowitz (Ed), Academic Press, London, pp. 1-68, 1999.

[4] M. Nagao, "A Framework of Mechanical Translation between Japanese and English by Analogy Principle". In A.Elithorn and R.Banerji, editor, *Artificial and Human Intelligence*. North-Holland, 1984.

[5] S. Nirenburg, S. Beale and C. Domashnev, "A Full-Text Experiment in Example-Based Machine Translation". In Proceedings of the International Conference on New Methods in Language Processing, NeMLap Manchester, UK pp. 78-87, 1994.

[6] S. Sato and M. Nagao, "Toward Memory-Based Translation". In the 13th International Conference on Computational Linguistics (COLING '90), Vol. 3, pp. 247-252, 1990.

[7] R. Brown, "Example-Based Machine Translation in the Pangloss System". In COLING '96, pp. 169-174, Copenhagen, 1996.

[8] K. Imamura, H. Okuma, T. Watanabe and E. Sumita, "Example-Based Machine Translation Based on Syntactic Transfer with Statistical Models". In the 20th International Conference on Computational Linguistics (COLING '04), Vol. 1, pp. 99-105, 2004.

[9] E. Sumita and H. Iida, "Experiments and Prospects of Example-Based Machine Translation". In Proceedings of the ACL 29th Annual Meeting, pp. 185-192, Berkeley, CA, 1991.

[10] C. Brockett, T. Aikawa, A. Aue, A. Menezes, C. Quirk and H. Suzuki, "English-Japanese Example-Based Machine Translation Using Abstract Linguistic Representations". In Proceedings of the Workshop on Machine Translation in Asia, Taiwan, 2002.

[11] H. L. Somers, "Review Article: Example-based Machine Translation". Machine Translation **14**, pp. 113-157, 1999.

[12] M. Diab, K. Hacioglu and D. Jurafsky, "Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks", The National Science Foundation, USA, 2004.

[13] T. Buckwalter, "Buckwalter Arabic Morphological Analyzer Version 1.0". Linguistic Data Consortium, Philadelphia, 2002. URL http://www.ldc.upenn.edu/Catalog/ CatalogEntry.jsp?catalogId=LDC2002L49 (viewed on 21/11/2006)

[14] J. Kremers, "The Arabic Noun Phrase", LOT (Dutch Graduate School of Linguistics), Utrecht University, The Netherlands, 2003. URL http://www.lotpublications .nl/index3.html (viewed on 27/11/06).

[15] A. Freeman, "Andrew Freeman's Perspectives on Arabic Diaglossia", 1996. URL http://www-personal.umich.edu/~andyf/digl_96.htm (viewed on 26/11/06)

[16] I. A. Al-Sughaiyer and I. A. Al-Kharashi, "Arabic Morphological Analysis Techniques: A Comprehensive Survey". In JASIST (Journal of the American Society for Information Science and Technology) Vol. 55, Num. 3, pp. 189-213, 2004.

[17] N. Habash, "Introduction to Arabic Natural Language Processing". Tutorial in the ACL 43th annual meeting, University of Michigan – Ann Arbor, 2005.

[18] J. Dichy, "On Lemmatization in Arabic, A Formal Definition of the Arabic Entries of Multilingual Lexical Databases". In Proceedings of the ACL 39th Annual Meeting. Workshop on Arabic Language Processing; Status and Prospect. Toulouse, pp. 23-30, 2001.

[19] M. F. Porter, "An Algorithm for Suffix Stripping". In Readings in Information Retrieval, K. Sparck Jones and P. Willett, Eds. Morgan Kaufmann Multimedia Information And Systems Series. Morgan Kaufmann Publishers, San Francisco, CA, pp. 313-316, 1997.

[20] S. Khoja and R. Garside, "Stemming Arabic Text". Computing Department, Lancaster University, Lancaster, U.K, 1999.

[21] L. S. Larkey and M. E. Connell, "Arabic Information Retrieval at UMass in TREC-10". In Proceedings of the TREC 2001. Gaithersburg, Maryland, pp. 562-570, 2001.

[22] K. Beesley, "Finite-State Morphological Analysis and Generation of Arabic at Xerox Research: Status and Plans in 2001". Xerox Research Centre Europe, 2001. URL http://www.xrce.xerox.com/Publications/Attachments/2001-009/finite-state.pdf (viewed on 21/11/2006)

[23] J. Berri, H. Zidoum and Y. Atif, "Web-Based Arabic Morphological Analyzer", In Gelbukh, A (Ed.): CICLing 2001, LNGS 2004, pp. 216-225. Springer-Verlag Berlin Heidelberg, 2001.

[24] A. M. Attia, "A Large-Scale Computational Processor of the Arabic Morphology, and Applications." A Master Thesis, Faculty of Engineering, Cairo University, Cairo, Egypt, 2000.

[25] K. Darwish ,"Building a Shallow Arabic Morphological Analyzer in One Day". In Proceedings of the ACL 40th Annual Meeting. Workshop on Computational Approaches to Semitic Languages, Philadelphia, PA, 2002.

[26] J. Allen, "Natural Language Understanding", Benjamin Cummings, 2nd Edition, Chp. 3, pp. 41-74, 1994.

[27] K. Daimi, "Identifying Syntactic Ambiguities in Single-Parse Arabic Sentence". Journal of Computers and the Humanities 35, pp. 333–349, 2001.

[28] E. Othman, K. Shaalan and A. Rafea, "A Chart Parser for Analyzing Modern Standard Arabic Sentence". In Proceedings of the MT Summit IX Workshop on Machine Translation for Semitic Languages: Issues and Approaches, New Orleans, Louisiana, USA, September, 2003.

[29] The Penn Treebank Project , Computer and Information Science, Penn University. URL http://www.cis.upenn.edu/~treebank/ (viewed on 27/11/06)

[30] E. Brill, "Some Advances in Transformation-Based Part-of-Speech Tagging". In Proceedings of the 12th National Conference on Artificial Intelligence, Seattle, pp. 722-727, 1994.

[31] A. Freeman, "Brill's POS Tagger and a Morphology Parser for Arabic". In Proceedings of the ACL 39th Annual Meeting. Workshop on Arabic Language Processing; Status and Prospect. Toulouse, France, 2001.

[32] The Brown Corpus, URL http://icame.uib.no/brown/bcm.html (viewed on 27/11/06)

[33] S. Khoja. "APT: Arabic Part-Of-Speech Tagger". In Proceedings of the Student Workshop at the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL2001), Carnegie Mellon University, Pennsylvania, 2001.

[34] S. Khoja, R. Garside and G. Knowles, "A Tagset for the Morphosyntactic Tagging for Arabic". In Proceedings of the Corpus Linguistics. Lancaster University (UK), Vol. 13 - Special Issue, 341, 2001.

[35] British National Corpus (BNC), URL http://www.natcorp.ox.ac.uk (viewed on 27/11/06)

[36] J. Allen, "Natural Language Understanding", Benjamin Cummings, 2nd Edition, Chp. 2, pp.25-36, 1994.

[37] L. Ramshaw and M. Marcus, "Text Chunking Using Transformation-Based Learning". In Proceedings of the 3rd ACL Workshop on Very Large Corpora, MIT, June, 1995.

[38] United Nations Official Document System (ODS), URL http://www.ods.un.org (viewed on 29/11/06)

[39] P. Fung and K. McKeown, "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), pp. 81-88, Columbia, Maryland, USA, 1994.

[40] Y. Choueka, E. S. Conley and I. Dagan, "A Comprehensive Bilingual Word Alignment System. Application to Disparate Languages: Hebrew and English". In J.

Veronis, "Parallel Text Processing: Alignment and Use of Translation Corpora", Kluwer Academic, 2000.

[41] E. Brill, "A Simple Rule-Based Part of Speech Tagger". In Proceedings of the DARPA, Speech and Natural Language Workshop. pp. 112-116. Morgan Kauffman. San Mateo, California, 1992.

[42] K. Papineni, S. Roukos, T. Ward and W. J. Zhu, "Bleu: a Method for Automatic Evaluation of Machine Translation". In Proceedings of the ACL 40th Annual Meeting, pp. 311-318, Philadelphia, PA, July, 2002.

[43] G. Doddington, "Automatic Evaluation of Machine Translation Quality Using n-gram Co-occurrence Statistics". In Proceedings of Human Language Technology, pp. 128-132, San Diego, California, March, 2002.

[44] S. Banerjee and A. Lavie, "Meteor: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments". In Proceedings of the ACL 43th Annual Meeting. Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization, pp. 65-72, Ann Arbor, MI, June, 2005.

[45] NIST Evaluations Report for 2005. URL http://www.nist.gov/speech/tests/mt/mt05 eval_official_results_release_20050801_v3.html (viewed on 26/12/06)

Appendix

Here are some input/output examples using our system. For every example we present two different human-made reference translations. System results are shown with their total scores. Fragment translations are separated by '|'.

Input: أفرجت السلطات عن سفينة الشحن
Reference Translation 1: *the cargo was released*
Reference Translation 2: *the authorities have released the cargo ship*

System Results:

Result	Total Score
authorities have also release | from ships |	0.39
authorities have also release | some of whom have responsibility for competition ship |	0.2773
authorities have also release | authorities about the development |	0.2773
authorities have also release | authorities about | from ships |	0.2535
authorities have also release | authorities about | some of whom have responsibility for competition ship	0.1802

Input: والتعاون في أوروبا
Reference Translation 1: *and cooperation in europe*
Reference Translation 2: *and the cooperation in europe*

System Results:

Result	Total Score
cooperation | in europe |	0.65
cooperation in a europe |	0.511

Input: مسؤولية جماعية مشتركة
Reference Translation 1: *common and shared responsibility*
Reference Translation 2: *shared and common responsibility*

System Results:

Result	Total Score
collective responsibility | social common market |	0.65
social common responsibility |	0.533

Input: في سياق منظمات إقليمية

Reference Translation 1: *within regional organizations*
Reference Translation 2: *in the context of regional organizations*

System Results:

Result	Total Score
the context in | regional organizer |	0.4225
the context in | context regional organizer |	0.3466
organizations or context | regional organizer |	0.3322
regional organizer |	0.325
the context in | organizations or context | regional organizer |	0.2746

Input: المعنية بآثار الإشعاع الذري

Reference Translation 1: *on the effects of atomic radiation*
Reference Translation 2: *that is concerned with the effects of the atomic radiation*

System Results:

Result	Total Score
effects of atomic radiation |	0.75
regarding the bamako convention on the ban | of atomic radiation |	0.65
regarding the bamako convention on the ban | effects of atomic radiation |	0.65
radiation s groups in india effects | of atomic radiation |	0.355

Input: تضع في اعتبارها

Reference Translation 1: b*earing in mind*
Reference Translation 2: *taking into consideration*

System Results:

Result	Total Score
to take into consideration |	0.511
by | viewed |	0.5