The Impact of Morphological Stemming on Arabic Mention Detection and Coreference Resolution

Imed Zitouni, Jeff Sorensen, Xiaoqiang Luo, Radu Florian
{izitouni, sorenj, xiaoluo, raduf}@watson.ibm.com
IBM T.J. Watson Research Center
1101 Kitchawan Rd, Yorktown Heights, NY 10598, USA

Abstract

Arabic presents an interesting challenge to natural language processing, being a highly inflected and agglutinative language. In particular, this paper presents an in-depth investigation of the entity detection and recognition (EDR) task for Arabic. We start by highlighting why segmentation is a necessary prerequisite for EDR, continue by presenting a finite-state statistical segmenter, and then examine how the resulting segments can be better included into a mention detection system and an entity recognition system; both systems are statistical, build around the maximum entropy principle. Experiments on a clearly stated partition of the ACE 2004 data show that stem-based features can significantly improve the performance of the EDT system by 2 absolute F-measure points. The system presented here had a competitive performance in the ACE 2004 evaluation.

1 Introduction

Information extraction is a crucial step toward understanding and processing language. One goal of information extraction tasks is to identify important conceptual information in a discourse. These tasks have applications in summarization, information retrieval (one can get all hits for Washington/person and not the ones for Washington/state or Washington/city), data mining, question answering, language understanding, etc.

In this paper we focus on the Entity Detection and Recognition task (EDR) for Arabic as described in ACE 2004 framework (ACE, 2004). The EDR has close ties to the named entity recognition (NER) and coreference resolution tasks, which have been the focus of several recent investigations (Bikel et al., 1997; Miller et al., 1998; Borthwick, 1999; Mikheev et al., 1999; Soon et al., 2001; Ng and Cardie, 2002; Florian et al., 2004), and have been at the center of evaluations such as: MUC-6, MUC-7, and the CoNLL'02 and CoNLL'03 shared tasks. Usually, in computational linguistics literature, a *named entity* is an instance of a location, a person, or an organization, and the NER task consists of identifying each of these occurrences. Instead, we will adopt the nomenclature of the Automatic Content Extraction program (NIST, 2004): we will call the instances of textual references to objects/abstractions *mentions*, which can be either named (e.g. John Mayor), nominal (the president) or pronominal (she, it). An entity is the aggregate of all the mentions (of any level) which refer to one conceptual entity. For instance, in the sentence

> President John Smith said he has no comments

there are two mentions (named and pronomial) but only one entity, formed by the set {John Smith, he}.

We separate the EDR task into two parts: a mention detection step, which identifies and classifies all the mentions in a text – and a coreference resolution step, which combinines the detected mentions into groups that refer to the same object. In its entirety, the EDR task is arguably harder than traditional named entity recognition, because of the additional complexity involved in extracting non-named mentions (nominal and pronominal) and the requirement of grouping mentions into entities. This is particularly true for Arabic where nominals and pronouns are also attached to the word they modify. In fact, most Arabic words are morphologically derived from a list of base forms or *stems*, to which prefixes and suffixes can be attached to form Arabic surface forms (blank-delimited words). In addition to the different forms of the Arabic word that result from the

Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages, pages 63–70,
Ann Arbor, June 2005.

derivational and inflectional process, most prepositions, conjunctions, pronouns, and possessive forms are attached to the Arabic surface word. It is these orthographic variations and complex morphological structure that make Arabic language processing challenging (Xu et al., 2001; Xu et al., 2002).

Both tasks are performed with a statistical framework: the mention detection system is similar to the one presented in (Florian et al., 2004) and the coreference resolution system is similar to the one described in (Luo et al., 2004). Both systems are built around from the maximum-entropy technique (Berger et al., 1996). We formulate the mention detection task as a sequence classification problem. While this approach is language independent, it must be modified to accomodate the particulars of the Arabic language. The Arabic words may be composed of zero or more prefixes, followed by a stem and zero or more suffixes. We begin with a segmentation of the written text before starting the classification. This segmentation process consists of separating the normal whitespace delimited words into (hypothesized) prefixes, stems, and suffixes, which become the subject of analysis (tokens). The resulting granularity of breaking words into prefixes and suffixes allows different mention type labels beyond the stem label (for instance, in the case of nominal and pronominal mentions). Additionally, because the prefixes and suffixes are quite frequent, directly processing unsegmented words results in significant data sparseness.

We present in Section 2 the relevant particularities of the Arabic language for natural language processing, especially for the EDR task. We then describe the segmentation system we employed for this task in Section 3. Section 4 briefly describes our mention detection system, explaining the different feature types we use. We focus in particular on the stem n-gram, prefix n-gram, and suffix n-gram features that are specific to a morphologically rich language such as Arabic. We describe in Section 5 our coreference resolution system where we also describe the advantage of using stem based features. Section 6 shows and discusses the different experimental results and Section 7 concludes the paper.

2 Why is Arabic Information Extraction difficult?

The Arabic language, which is the mother tongue of more than 300 million people (Center, 2000), present significant challenges to many natural language processing applications. Arabic is a highly inflected and derived language. In Arabic morphology, most morphemes are comprised of a basic word form (the root or stem), to which many affixes can be attached to form Arabic words. The Arabic alphabet consists of 28 letters that can be extended to ninety by additional shapes, marks, and vowels (Tayli and Al-Salamah, 1990). Unlike Latin-based alphabets, the orientation of writing in Arabic is from right to left. In written Arabic, short vowels are often omitted. Also, because variety in expression is appreciated as part of a good writing style, the synonyms are widespread. Arabic nouns encode information about gender, number, and grammatical cases. There are two genders (masculine and feminine), three numbers (singular, dual, and plural), and three grammatical cases (nominative, genitive, and accusative).

A noun has a nominative case when it is a subject, accusative case when it is the object of a verb, and genitive case when it is the object of a preposition. The form of an Arabic noun is consequently determined by its gender, number, and grammatical case. The definitive nouns are formed by attaching the Arabic article ال to the immediate front of the nouns, such as in the word الشركة (the company). Also, prepositions such as ب (by), and ل (to) can be attached as a prefix as in للشركة (to the company). A noun may carry a possessive pronoun as a suffix, such as in شركتهم (their company). For the EDR task, in this previous example, the Arabic blank-delimited word شركتهم should be split into two tokens: شركة and هم. The first token شركة is a mention that refers to an organization, whereas the second token هم is also a mention, but one that may refer to a person. Also, the prepositions (i.e., ب and ل) not be considered a part of the mention.

Arabic has two kinds of plurals: broken plurals and sound plurals (Wightwick and Gaafar, 1998; Chen and Gey, 2002). The formation of broken plurals is common, more complex and often irregular. As an example, the plural form of the noun رجل (man) is رِجَال (men), which is formed by inserting the infix ا. The plural form of the noun كتَاب (book) is كتب (books), which is formed by deleting the infix ا. The plural form and the singular form may also be completely different (e.g. إمرأة for woman, but نسَاء for women). The sound plurals are formed by adding plural suffixes to singular nouns (e.g., بَاحث meaning researcher): the plural suffix is ات for feminine nouns in grammatical cases (e.g., بَاحثَات), ون for masculine nouns in the nominative case (e.g., بَاحثون), and ين for masculine nouns in the genitive and accusative cases (e.g., بَاحثين). The dual suffix is ان for the nominative case (e.g., بَاحثَان), and ين for the genitive or accusative (e.g., بَاحثين).

Because we consider pronouns and nominals as mentions, it is essential to segment Arabic words into these subword tokens. We also believe that the in-

formation denoted by these affixes can help with the coreference resolution task[1].

Arabic verbs have perfect and imperfect tenses (Abbou and McCarus, 1983). Perfect tense denotes completed actions, while imperfect denotes ongoing actions. Arabic verbs in the perfect tense consist of a stem followed by a subject marker, denoted as a suffix. The subject marker indicates the person, gender, and number of the subject. As an example, the verb قَابل (to meet) has a perfect tense قَابَلت for the third person feminine singular, and قَابَلُوا for the third person masculine plural. We notice also that a verb with a subject marker and a pronoun suffix can be by itself a complete sentence, such us in the word قَابلتهم: it has a third-person feminine singular subject-marker ت (she) and a pronoun suffix هم (them). It is also a complete sentence meaning "she met them." The subject markers are often suffixes, but we may find a subject marker as a combination of a prefix and a suffix as in تقَابلهم (she meets them). In this example, the EDR system should be able to separate تقَابلهم, to create two mentions (ت and هم). Because the two mentions belong to different entities, the EDR system should not chain them together. An Arabic word can potentially have a large number of variants, and some of the variants can be quite complex. As an example, consider the word ولبَاحثيهَا (and to her researchers) which contains two prefixes and one suffix (و + ل + بَاحثي + هَا).

3 Arabic Segmentation

Lee et al. (2003) demonstrates a technique for segmenting Arabic text and uses it as a morphological processing step in machine translation. A trigram language model was used to score and select among hypothesized segmentations determined by a set of prefix and suffix expansion rules.

In our latest implementation of this algorithm, we have recast this segmentation strategy as the composition of three distinct finite state machines. The first machine, illustrated in Figure 1 encodes the prefix and suffix expansion rules, producing a lattice of possible segmentations. The second machine is a dictionary that accepts characters and produces identifiers corresponding to dictionary entries. The final machine is a trigram language model, specifically a Kneser-Ney (Chen and Goodman, 1998) based backoff language model. Differing from (Lee et al., 2003), we have also introduced an explicit model for unknown words based upon a character unigram model, although this model is dominated by an empirically chosen unknown word penalty. Using 0.5M words from the combined Arabic Treebanks 1V2, 2V2 and 3V1, the dictionary based segmenter achieves a exact word match 97.8% correct segmentation.

[1]As an example, we do not chain mentions with different gender, number, etc.

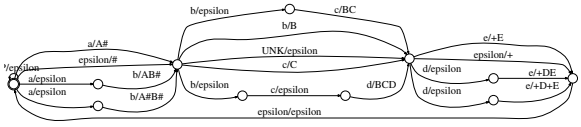

Figure 1: Illustration of dictionary based segmentation finite state transducer

3.1 Bootstrapping

In addition to the model based upon a dictionary of stems and words, we also experimented with models based upon character n-grams, similar to those used for Chinese segmentation (Sproat et al., 1996). For these models, both arabic characters and spaces, and the inserted prefix and suffix markers appear on the arcs of the finite state machine. Here, the language model is conditioned to insert prefix and suffix markers based upon the frequency of their appearance in n-gram character contexts that appear in the training data. The character based model alone achieves a 94.5% exact match segmentation accuracy, considerably less accurate then the dictionary based model.

However, an analysis of the errors indicated that the character based model is more effective at segmenting words that do not appear in the training data. We seeked to exploit this ability to generalize to improve the dictionary based model. As in (Lee et al., 2003), we used unsupervised training data which is automatically segmented to discover previously unseen stems. In our case, the character n-gram model is used to segment a portion of the Arabic Gigaword corpus. From this, we create a vocabulary of stems and affixes by requiring that tokens appear more than twice in the supervised training data or more than ten times in the unsupervised, segmented corpus.

The resulting vocabulary, predominately of word stems, is 53K words, or about six times the vocabulary observed in the supervised training data. This represents about only 18% of the total number of unique tokens observed in the aggregate training data. With the addition of the automatically acquired vocabulary, the segmentation accuracy achieves 98.1% exact match.

3.2 Preprocessing of Arabic Treebank Data

Because the Arabic treebank and the gigaword corpora are based upon news data, we apply some small amount of regular expression based preprocessing. Arabic specific processing include removal of the characters *tatweel* (ـ), and vowels. Also, the following characters are treated as an equivalence class during all lookups and processing: (1) ى ,ي, and (2) آ، أ، إ، ا. We define a token and introduce whitespace boundaries between every span of one or more alphabetic or numeric characters. Each punctuation symbol is considered a separate token. Character classes, such as punctuation, are defined according to the Unicode Standard (Aliprand et al., 2004).

4 Mention Detection

The mention detection task we investigate identifies, for each mention, four pieces of information:

1. the mention type: person (PER), organization (ORG), location (LOC), geopolitical entity (GPE), facility (FAC), vehicle (VEH), and weapon (WEA)
2. the mention level (named, nominal, pronominal, or premodifier)
3. the mention class (generic, specific, negatively quantified, etc.)
4. the mention sub-type, which is a sub-category of the mention type (ACE, 2004) (e.g. OrgGovernmental, FacilityPath, etc.).

4.1 System Description

We formulate the mention detection problem as a classification problem, which takes as input segmented Arabic text. We assign to each token in the text a label indicating whether it starts a specific mention, is inside a specific mention, or is outside any mentions. We use a maximum entropy Markov model (MEMM) classifier. The principle of maximum entropy states that when one searches among probability distributions that model the observed data (evidence), the preferred one is the one that maximizes the entropy (a measure of the uncertainty of the model) (Berger et al., 1996). One big advantage of this approach is that it can combine arbitrary and diverse types of information in making a classification decision.

Our mention detection system predicts the four labels types associated with a mention through a cascade approach. It first predicts the boundary and the main entity type for each mention. Then, it uses the information regarding the type and boundary in different second-stage classifiers to predict the subtype, the mention level, and the mention class. After the first stage, when the boundary (starting, inside, or outside a mention) has been determined, the other classifiers can use this information to analyze a larger context, capturing the patterns around the entire mentions, rather than words. As an example, the token sequence that refers to a mention will become a single recognized unit and, consequently, lexical and syntactic features occuring inside or outside of the entire mention span can be used in prediction.

In the first stage (entity type detection and classification), Arabic blank-delimited words, after segmenting, become a series of tokens representing prefixes, stems, and suffixes (cf. section 2). We allow any contiguous sequence of tokens can represent a mention. Thus, prefixes and suffixes can be, and often are, labeled with a different mention type than the stem of the word that contains them as constituents.

4.2 Stem n-gram Features

We use a large set of features to improve the prediction of mentions. This set can be partitioned into 4 categories: lexical, syntactic, gazetteer-based, and those obtained by running other named-entity classifiers (with different tag sets). We use features such as the shallow parsing information associated with the tokens in a window of 3 tokens, POS, etc.

The context of a current token t_i is clearly one of the most important features in predicting whether t_i is a mention or not (Florian et al., 2004). We denote these features as backward token tri-grams and forward token tri-grams for the previous and next context of t_i respectively. For a token t_i, the backward token n-gram feature will contains the previous $n-1$ tokens in the history $(t_{i-n+1}, \ldots t_{i-1})$ and the forward token n-gram feature will contains the next $n-1$ tokens $(t_{i+1}, \ldots t_{i+n-1})$.

Because we are segmenting arabic words into multiple tokens, there is some concern that trigram contexts will no longer convey as much contextual information. Consider the following sentence extracted from the development set: هذَا يمثل المَقر لِلمَكتب السيَاسي لِلحزب (translation "This represents the location for Political Party Office"). The "Political Party Office" is tagged as an organization and, as a word-for-word translation, is expressed as "to the Office of the political to the party". It is clear in this example that the word مَقر (location for) contains crucial information in distinguishing between a location and an organization when tagging the token مَكتب

(office). After segmentation, the sentence becomes: هذَا + ي + مثل + اَل + مَقر + ل + اَل + مَكتب + اَل + سيَاسي + ل + اَل + حزب.

When predicting if the token مَكتب (office) is the beginning of an organization or not, backward and forward token n-gram features contain only ل + اَل (for the) and سيَاسي + اَل (the political). This is most likely not enough context, and addressing the problem by increasing the size of the n-gram context quickly leads to a data sparseness problem.

We propose in this paper the *stem n-gram* features as additional features to the lexical set. If the current token t_i is a stem, the backward stem n-gram feature contains the previous $n-1$ stems and the forward stem n-gram feature will contain the following $n-1$ stems. We proceed similarly for prefixes and suffixes: if t_i is a prefix (or suffix, respectively) we take the previous and following prefixes (or suffixes)[2]. In the sentence shown above, when the system is predicting if the token مَكتب (office) is the beginning of an organization or not, the backward and forward stem n-gram features contain مثل مَقر (represent location of) and سيَاسي حزب (political office). The stem features contain enough information in this example to make a decision that مَكتب (office) is the beginning of an organization. In our experiments, n is 3, therefore we use stem trigram features.

5 Coreference Resolution

Coreference resolution (or entity recognition) is defined as grouping together mentions referring to the same object or *entity*. For example, in the following text,

(I) "John believes Mary to be the best student"

three mentions "John", "Mary", "student" are underlined. "Mary" and "student" are in the same entity since both refer to the same person.

The coreference system system is similar to the Bell tree algorithm as described by (Luo et al., 2004).

In our implementation, the link model between a candidate entity e and the current mention m is computed as

$$P_L(L=1|e,m) \approx \max_{m_k \in e} \hat{P}_L(L=1|e,m_k,m), \quad (1)$$

where m_k is one mention in entity e, and the basic model building block $\hat{P}_L(L=1|e,m_k,m)$ is an exponential or maximum entropy model (Berger et al., 1996).

For the start model, we use the following approximation:

$$P_S(S=1|e_1,e_2,\cdots,e_t,m) \approx 1 - \max_{1\le i\le t} P_L(L=1|e_i,m) \quad (2)$$

The start model (cf. equation 2) says that the probability of starting a new entity, given the current mention m and the previous entities $e_1,e_2,\cdots,e_t$, is simply 1 minus the maximum link probability between the current mention and one of the previous entities.

The maximum-entropy model provides us with a flexible framework to encode features into the the system. Our Arabic entity recognition system uses many language-indepedent features such as strict and partial string match, and distance features (Luo et al., 2004). In this paper, however, we focus on the addition of Arabic stem-based features.

5.1 Arabic Stem Match Feature

Features using the word context (left and right tokens) have been shown to be very helpful in coreference resolution (Luo et al., 2004). For Arabic, since words are morphologically derived from a list of roots (stems), we expected that a feature based on the right and left stems would lead to improvement in system accuracy.

Let m_1 and m_2 be two candidate mentions where a mention is a string of tokens (prefixes, stems, and suffixes) extracted from the segmented text. In order to make a decision in either linking the two mentions or not we use additional features such as: do the stems in m_1 and m_2 match, do stems in m_1 match *all* stems in m_2, do stems in m_1 *partially* match stems in m_2. We proceed similarly for prefixes and suffixes. Since prefixes and suffixes can belong to different mention types, we build a parse tree on the segmented text and we can explore features dealing with the gender and number of the token. In the following example, between parentheses we make a word-for-word translations in order to better explain our stemming feature. Let us take the two mentions لِلمَكتب السيَاسي لِلحزب (to-the-office the-politic to-the-party) and مَكتب الحزبي (office the-party's) segmented as ل + اَل + مَكتب + اَل + سيَاسي + ل + اَل + حزب and مَكتب + اَل + حزب + ي respectively. In our

[2]Thus, the difference to token n-grams is that the tokens of different type are removed from the streams, before the features are created.

development corpus, these two mentions are chained to the same entity. The stemming match feature in this case will contain information such us *all stems* of m_2 match, which is a strong indicator that these mentions should be chained together. Features based on the words alone would not help this specific example, because the two strings m_1 and m_2 do not match.

6 Experiments

6.1 Data

The system is trained on the Arabic ACE 2003 and part of the 2004 data. We introduce here a clearly defined and replicable split of the ACE 2004 data, so that future investigations can accurately and correctly compare against the results presented here.

There are 689 Arabic documents in LDC's 2004 release (version 1.4) of ACE data from three sources: the Arabic Treebank, a subset of the broadcast (bnews) and newswire (nwire) TDT-4 documents. The 178-document devtest is created by taking the last (in chronological order) 25% of documents in each of three sources: 38 Arabic treebank documents dating from "20000715" (i.e., July 15, 2000) to "20000815," 76 bnews documents from "20001205.1100.0489" (i.e., Dec. 05 of 2000 from 11:00pm to 04:89am) to "20001230.1100.1216," and 64 nwire documents from "20001206.1000.0050" to "20001230.0700.0061." The time span of the test set is intentionally non-overlapping with that of the training set within each data source, as this models how the system will perform in the real world.

6.2 Mention Detection

We want to investigate the usefulness of stem n-gram features in the mention detection system. As stated before, the experiments are run in the ACE'04 framework (NIST, 2004) where the system will identify mentions and will label them (cf. Section 4) with a type (person, organization, etc), a sub-type (OrgCommercial, OrgGovernmental, etc), a mention level (named, nominal, etc), and a class (specific, generic, etc). Detecting the mention boundaries (set of consecutive tokens) and their main type is one of the important steps of our mention detection system. The score that the ACE community uses (ACE value) attributes a higher importance (outlined by its weight) to the main type compared to other subtasks, such as the mention level and the class. Hence, to build our mention detection system we spent a lot of effort in improving the first step: detecting the mention boundary and their main type. In this paper, we report the results in terms of precision, recall, and F-measure[3].

Lexical features			
	Precision (%)	Recall (%)	F-measure (%)
Total	73.3	58.0	**64.7**
FAC	76.0	24.0	36.5
GPE	79.4	65.6	71.8
LOC	57.7	29.9	39.4
ORG	63.1	46.6	53.6
PER	73.2	63.5	68.0
VEH	83.5	29.7	43.8
WEA	77.3	25.4	38.2
Lexical features + Stem			
	Precision (%)	Recall (%)	F-measure (%)
Total	73.6	59.4	**65.8**
FAC	72.7	29.0	41.4
GPE	79.9	67.2	73.0
LOC	58.6	31.9	41.4
ORG	62.6	47.2	53.8
PER	73.8	64.6	68.9
VEH	81.7	35.9	49.9
WEA	78.4	29.9	43.2

Table 1: Performance of the mention detection system using lexical features only.

To assess the impact of stemming n-gram features on the system under different conditions, we consider two cases: one where the system only has access to lexical features (the tokens and direct derivatives including standard n-gram features), and one where the system has access to a richer set of information, including lexical features, POS tags, text chunks, parse tree, and gazetteer information. The former framework has the advantage of being fast (making it more appropriate for deployment in commercial systems). The number of parameters to optimize in the MaxEnt framework we use when only lexical features are explored is around 280K parameters. This number increases to 443K approximately when all information is used except the stemming feature. The number of parameters introduced by the use of stemming is around 130K parameters. Table 1 reports experimental results using lexical features only; we observe that the stemming n-gram features boost the performance by one point (64.7 vs. 65.8). It is important to notice the stemming n-gram features improved the performance of each category of the main type.

In the second case, the systems have access to a large amount of feature types, including lexical, syntactic, gazetteer, and those obtained by running other

[3]The ACE value is an important factor for us, but its relative complexity, due to different weights associated with the subparts, makes for a hard comparison, while the F-measure is relatively easy to interpret.

AllFeatures			
	Precision (%)	Recall (%)	F-measure (%)
Total	74.3	64.0	**68.8**
FAC	72.3	36.8	48.8
GPE	80.5	70.8	75.4
LOC	61.1	35.4	44.8
ORG	61.4	50.3	55.3
PER	75.3	70.2	72.7
VEH	83.2	38.1	52.3
WEA	69.0	36.6	47.8
All-Features + Stem			
	Precision (%)	Recall (%)	F-measure (%)
Total	74.4	64.6	**69.2**
FAC	68.8	38.5	49.4
GPE	80.8	71.9	76.1
LOC	60.2	36.8	45.7
ORG	62.2	51.0	56.1
PER	75.3	70.2	72.7
VEH	81.4	41.8	55.2
WEA	70.3	38.8	50.0

Table 2: Performance of the mention detection system using lexical, syntactic, gazetteer features as well as features obtained by running other named-entity classifiers

named-entity classifiers (with different semantic tag sets). Features are also extracted from the shallow parsing information associated with the tokens in window of 3, POS, etc. The *All-features* system incorporates all the features except for the stem n-grams. Table 2 shows the experimental results with and without the stem n-grams features. Again, Table 2 shows that using stem n-grams features gave a small boost to the whole main-type classification system[4]. This is true for all types. It is interesting to note that the increase in performance in both cases (Tables 1 and 2) is obtained from increased recall, with little change in precision. When the prefix and suffix n-gram features are removed from the feature set, we notice in both cases (Tables 1 and 2) a insignificant decrease of the overall performance, which is expected: what should a feature of preceeding (or following) prepositions or finite articles captures?

As stated in Section 4.1, the mention detection system uses a cascade approach. However, we were curious to see if the gain we obtained at the first level was successfully transfered into the overall performance of the mention detection system. Table 3 presents the performance in terms of precision, recall, and F-measure of the whole system. Despite the fact that the improvement was small in terms of F-measure (59.4 vs. 59.7), the stemming n-gram features gave interesting improvement in terms of ACE value to the hole EDR system as showed in section 6.3.

[4]The difference in performance is not statistically significant

	Precision (%)	Recall (%)	F-measure (%)
All-Features	64.2	55.3	59.4
All-Features+Stem	64.4	55.7	59.7
Lexical	64.4	50.8	56.8
Lexical+Stem	64.6	52.0	57.6

Table 3: Performance of the mention detection system including all ACE'04 subtasks

6.3 Coreference Resolution

In this section, we present the coreference results on the devtest defined earlier. First, to see the effect of stem matching features, we compare two coreference systems: one with the stem features, the other without. We test the two systems on both "true" and system mentions of the devtest set. "True" mentions mean that input to the coreference system are mentions marked by human, while system mentions are output from the mention detection system. We report results with two metrics: ECM-F and ACE-Value. ECM-F is an entity-constrained mention F-measure (cf. (Luo et al., 2004) for how ECM-F is computed), and ACE-Value is the official ACE evaluation metric. The result is shown in Table 4: the baseline numbers without stem features are listed under "Base," and the results of the coreference system with stem features are listed under "Base+Stem."

On true mention, the stem matching features improve ECM-F from 77.7% to 80.0%, and ACE-value from 86.9% to 88.2%. The similar improvement is also observed on system mentions.The overall ECM-F improves from 62.3% to 64.2% and the ACE value improves from 61.9 to 63.1%. Note that the increase on the ACE value is smaller than ECM-F. This is because ACE-value is a weighted metric which emphasizes on NAME mentions and heavily discounts PRONOUN mentions. Overall the stem features give rise to consistent gain to the coreference system.

7 Conclusion

In this paper, we present a fully fledged Entity Detection and Tracking system for Arabic. At its base, the system fundamentally depends on a finite state segmenter and makes good use of the relationships that occur between word stems, by introducing features which take into account the type of each segment. In mention detection, the features are represented as stem n-grams, while in coreference resolution they are captured through stem-tailored match features.

	Base		Base+Stem	
	ECM-F	ACEVal	ECM-F	ACEVal
Truth	77.7	86.9	80.0	88.2
System	62.3	61.9	64.2	63.1

Table 4: Effect of Arabic stemming features on coreference resolution. The row marked with "Truth" represents the results with "true" mentions while the row marked with "System" represents that mentions are detected by the system. Numbers under "ECM-F" are Entity-Constrained-Mention F-measure and numbers under "ACE-Val" are ACE-values.

These types of features result in an improvement in both the mention detection and coreference resolution performance, as shown through experiments on the ACE 2004 Arabic data. The experiments are performed on a clearly specified partition of the data, so comparisons against the presented work can be correctly and accurately made in the future. In addition, we also report results on the official test data.

The presented system has obtained competitive results in the ACE 2004 evaluation, being ranked amongst the top competitors.

8 Acknowledgements

This work was partially supported by the Defense Advanced Research Projects Agency and monitored by SPAWAR under contract No. N66001-99-2-8916. The views and findings contained in this material are those of the authors and do not necessarily reflect the position of policy of the U.S. government and no official endorsement should be inferred.

References

Peter F. Abbou and Ernest N. McCarus, editors. 1983. *Elementary modern standard Arabic.* Cambridge University Press.

ACE. 2004. Automatic content extraction. http://www.ldc.upenn.edu/Projects/ACE/.

Joan Aliprand, Julie Allen, Joe Becker, Mark Davis, Michael Everson, Asmus Freytag, John Jenkins, Mike Ksar, Rick McGowan, Eric Muller, Lisa Moore, Michel Suignard, and Ken Whistler. 2004. The unicode standard. http://www.unicode.org/.

A. Berger, S. Della Pietra, and V. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel. 1997. Nymble: a high-performance learning name-finder. In *Proceedings of ANLP-97*, pages 194–201.

A. Borthwick. 1999. *A Maximum Entropy Approach to Named Entity Recognition.* Ph.D. thesis, New York University.

Egyptian Demographic Center. 2000. http://www.frcu.eun.eg/www/homepage/cdc/cdc.htm.

Aitao Chen and Fredic Gey. 2002. Building an arabic stemmer for information retrieval. In *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002)*, National Institute of Standards and Technology, November.

S. F. Chen and J. Goodman. 1998. An empirical study of smoothing techinques for language modeling. Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University, Cambridge, Massachusettes, August.

R. Florian, H. Hassan, A. Ittycheriah, H. Jing, N. Kambhatla, X. Luo, N Nicolov, and S Roukos. 2004. A statistical model for multilingual entity detection and tracking. In *Proceedings of HLT-NAACL 2004*, pages 1–8.

Y.-S. Lee, K. Papineni, S. Roukos, O. Emam, and H. Hassan. 2003. Language model based Arabic word segmentation. In *Proceedings of the ACL'03*, pages 399–406.

Xiaoqiang Luo, Abe Ittycheriah, Hongyan Jing, Nanda Kambhatla, and Salim Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the bell tree. In *Proc. of ACL'04*.

A. Mikheev, M. Moens, and C. Grover. 1999. Named entity recognition without gazetteers. In *Proceedings of EACL'99*.

S. Miller, M. Crystal, H. Fox, L. Ramshaw, R. Schwarz, R. Stone, and R. Weischedel. 1998. Bbn: Description of the SIFT system as used for MUC-7. In *MUC-7*.

V. Ng and C. Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proceedings of the ACL'02*, pages 104–111.

NIST. 2004. Proceedings of ace evaluation and pi meeting 2004 workshop. Alexandria, VA, September. NIST.

W. M. Soon, H. T. Ng, and C. Y. Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

R. Sproat, C. Shih, W. Gale, and N. Chang. 1996. A stochastic finite-state word-segmentation algorithm for Chinese. *Computational Linguistics*, 22(3).

M. Tayli and A. Al-Salamah. 1990. Building bilingual microcomputer systems. *Communications of the ACM*, 33(5):495–505.

J. Wightwick and M. Gaafar. 1998. *Arabic Verbs and Essentials of Grammar.* Passport Books.

J. Xu, A. Fraser, and R. Weischedel. 2001. Trec2001 cross-lingual retrieval at bbn. In *TREC 2001*, Gaithersburg: NIST.

J. Xu, A. Fraser, and R. Weischedel. 2002. Empirical studies in strategies for arabic information retrieval. In *SIGIR 2002*, Tampere, Finland.