Arabic Cross-Document Coreference Detection

Asad Sayeed,[1,2] **Tamer Elsayed,**[1,2] **Nikesh Garera,**[1,6] **David Alexander,**[1,3]
Tan Xu,[1,4] **Douglas W. Oard,**[1,4,5] **David Yarowsky,**[1,6] **Christine Piatko**[1]
[1]Human Language Technology Center of Excellence, Johns Hopkins University, Baltimore, MD, USA—[2]Dept. of Computer Science, University of Maryland, College Park, MD, USA—[3]BBN Technologies, Cambridge, MA, USA—[4]College of Information Studies, University of Maryland, College Park, MD, USA—[5]UMIACS, University of Maryland, College Park, MD, USA—[6]Dept. of Computer Science, Johns Hopkins University, Baltimore, MD, USA
{asayeed,telsayed}@cs.umd.edu, ngarera@cs.jhu.edu, dalexand@bbn.com,
{tanx,oard}@umd.edu, yarowsky@cs.jhu.edu, Christine.Piatko@jhuapl.edu

Abstract

We describe a set of techniques for Arabic cross-document coreference resolution. We compare a baseline system of exact mention string-matching to ones that include local mention context information as well as information from an existing machine translation system. It turns out that the machine translation-based technique outperforms the baseline, but local entity context similarity does not. This helps to point the way for future cross-document coreference work in languages with few existing resources for the task.

1 Introduction

Our world contains at least two noteworthy George Bushes: President George H. W. Bush and President George W. Bush. They are both frequently referred to as "George Bush." If we wish to use a search engine to find documents about one of them, we are likely also to find documents about the other. Improving our ability to find *all* documents referring to one and *none* referring to the other in a targeted search is a goal of cross-document entity coreference detection. Here we describe some results from a system we built to perform this task on Arabic documents. We base our work partly on previous work done by Bagga and Baldwin (Bagga and Baldwin, 1998), which has also been used in later work (Chen and Martin, 2007). Other work such as Lloyd et al. (Lloyd, 2006) focus on techniques specific to English.

The main contribution of this work to cross-document coreference lies in the conditions under which it was done. Even now, there is no large-scale resource—in terms of annotated data—for cross-document coreference in Arabic as there is in English (e.g. WebPeople (Artiles, 2008)). Thus, we employed techniques for high-performance processing in a resource-poor environment. We provide early steps in cross-document coreference detection for resource-poor languages.

2 Approach

We treat cross-document entities as a set of graphs consisting of links between within-document entities. The graphs are disjoint. Each of our systems produces a list of such links as within-document entity pairs (A, B). We obtain within-document entities by running the corpus through a within-document coreference resolver—in this case, Serif from BBN Technologies.

To create the entity clusters, we use a union-find algorithm over the pairs. If links (A, B) and (C, B) appear in the system output, then $\{A, B, C\}$ are one entity. Similarly, if (X, Y) and (Z, Y) appear in the output, then it will find that $\{X, Y, Z\}$ are one entity. If the algorithm later discovers link (B, Z) in the system output, it will decide that $\{A, B, C, X, Y, Z\}$ are an entity. This is efficiently implemented via a hash table whose keys and values are both within-document entity IDs, allowing the implementation of easily-searched linked lists.

2.1 The baseline system

The baseline system uses a string matching criterion to determine whether two within-document entities are similar enough to be considered as part of the same cross-document entity. Given within-document entities A and B, the criterion is implemented as follows:

1. Find the mention strings $\{a_1, a_2, \ldots\}$ and

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 357–360,
Suntec, Singapore, 4 August 2009.

$\{b_1, b_2, \ldots\}$ of A and B, respectively that are the longest for that within-document entity in the given document. (There may be more than one longest mention of equal length for a given entity.)

2. If *any* longest mention strings a_n and b_m exist such that $a_n = b_m$ (exact string match), then A and B are considered to be part of the same cross-document entity. Otherwise, they are considered to be different entities.

When the system decides that two within-document entities are connected as a single cross-document entity, it emits a link between within-document entities A and B represented as the pair (A, B). We maintain a list of such links, but we omit all links between within-document entities in the same document.

The output of the system is a list of pairwise links. The following two experimental systems also produce lists of pairwise links. Union is performed between the baseline system's list and the lists produced by the other systems to create lists of pairs that include the information in the baseline. However, each of the following systems' outputs are merged *separately* with the baseline. By including the baseline results in each system, we are able to clarify the potential of each additional technique to improve performance over a technique that is cheap to run under any circumstances, especially given that our experiments are focused on *increasing* the number of links in an Arabic context where links are likely to be disrupted by spelling variations.

2.2 Translingual projection

We implement a novel cross-language approach for Arabic coreference resolution by expanding the space of exact match comparisons to approximate matches of English translations of the Arabic strings. The intuition for this approach is that often the Arabic strings of the same named entity may differ due to misspellings, titles, or aliases that can be corrected in the English space. The English translations were obtained using a standard statistical machine translation system (Chiang, 2007; Li, 2008) and then compared using an alias match.

The algorithm below describes the approach, applied to any Arabic named entities that fail the baseline string-match test:

1. For a given candidate Arabic named entity pair (A, B), we project them into English by translating the mentions using a standard statistical machine translation toolkit. Using the projected English pair, say, (A', B') we perform the following tests to determine whether A and B are co-referent:
   (a) We do an exact string-match test in the English space using the projected entities (A', B'). The exact string match test is done exactly as in the baseline system, using the set of longest named entities in their respective co-reference chains.
   (b) If (A', B') fail in the exact string-match test as in the baseline, then we test whether they belong to a list of high confidence co-referent named-entity pairs[1] precomputed for English using alias-lists derived from Wikipedia.
   (c) If (A', B') fails (a) and (b) then (A, B) is deemed as non-coreferent.

While we hypothesize that translingual projection via English should help in increasing recall since it can work with non-exact string matches, it may also help in increasing precision based on the assumption that a name of American or English origin might have different variants in Arabic and that translating to English can help in merging those variants, as shown in figure 1.

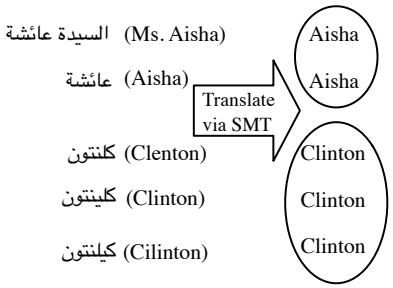

Figure 1: Illustration of translingual projection method for resolving Arabic named entity strings via English space. The English strings in parentheses indicate the literal glosses of the Arabic strings prior to translation.

2.3 Entity context similarity

The context of mentions can play an important role in merging or splitting potential coreferent men-

[1]For example: (Sean Michael Waltman, Sean Waltman) are high confidence-matches even though they are not an exact-string match.

tions. We hypothesize that two mentions in two different documents have a good chance of referring to the same entity if they are mentioned in contexts that are topically very similar. A way of representing a mention context is to consider the words in the mention's neighborhood. The context of a mention can be defined as the words that surround the mention in a window of *n* (50 in our experiments) tokens centered by the mention. In our experiments, we used highly similar contexts to link mentions that might be coreferent.

Computing context similarity between every pair of large number of mentions requires a highly scalable and efficient mechanism. This can be achieved using MapReduce, a distributed computing framework (Dean, 2004)

Elsayed et al. (Elsayed, 2008) proposed an efficient MapReduce solution for the problem of computing the pairwise similarity matrix in large collections. They considered a "bag-of-words" model where similarity of two documents d_i and d_j is measured as follows: $sim(d_i, d_j) = \sum_{t \in d_i \cap d_j} w_{t,d_i} \cdot w_{t,d_j}$, where $w(t, d)$ is the weight of term t in document d. A term contributes to *each* pair that contains it. The list of documents that contain a term is what is contained in the postings of an inverted index. Thus, by processing all postings, the pairwise similarity matrix can be computed by summing term contributions. We use the MapReduce framework for two jobs, inverted indexing and pairwise similarity.

Elsayed et al. suggested an efficient df-cut strategy that eliminates terms that appear in many documents (having high df) and thus contribute less in similarity but cost in computation (e.g., a 99% df-cut means that the most frequent 1% of the terms were discarded). We adopted that approach for computing similarities between the contexts of two mentions. The processing unit was represented as a bag of n words in a window surrounding each mention of a within-document entity. Given a relatively small mention context, we used a high df-cut value of 99.9%.

3 Experiments

We performed our experiments in the context of the Automatic Content Extraction (ACE) evaluation of 2008, run by the National Institute of Standards and Technology (NIST). The evaluation corpus contained approximately 10,000 documents from the following domains: broadcast conversation transcripts, broadcast news transcripts, conversational telephone speech transcripts, newswire, Usenet Newsgroup/Discussion Groups, and weblogs. Systems were required to process the large source sets completely. For performance measurement after the evaluation, NIST selected 412 of the Arabic source documents out of the larger set (NIST, 2008).

For development purposes we used the NIST ACE 2005 Arabic data with within-document ground truth. This consisted of 1,245 documents. We also used exactly 12,000 randomly selected documents from the LDC Arabic Gigaword Third Edition corpus, processed through Serif. The Arabic Gigaword corpus was used to select a threshold of 0.4956 for the context similarity technique via inspection of (A, B) link scores by a native speaker of Arabic.

It must be emphasized that there was no ground truth available for this task in Arabic. Performing this task in the absence of significant training or evaluation data is one emphasis of this work.

3.1 Evaluation measures

We used NIST's scoring techniques to evaluate the performance of our systems. Scoring for the ACE evaluation is done using an scoring script provided by NIST which produces many kinds of statistics. NIST mainly uses a measure called the ACE value, but it also computes B-cubed.

B-Cubed represents the task of finding cross-document entities in the following way: if a user of the system is searching for a particular Bush and finds document D, he or she should be able to find all of the other documents with the same Bush in them as links from D—that is, cross-document entities represent graphs connecting documents. Bagga and Baldwin are able to define precision, recall, and F-measure over a collection of documents in this way.

The ACE Value represents a score similar to B-Cubed, except that every mention and within-document entity is weighted in NIST's specification by a number of factors. Every entity is worth 1 point, a missing entity worth 0, and attribute errors are discounted by multiplying by a factor (0.75 for *CLASS*, 0.5 for *TYPE*, and 0.9 for *SUBTYPE*).

Before scoring can be accomplished, the entities found by the system must be mapped onto those found in the reference provided by NIST. The ACE scorer does this document-by-document,

selecting the mapping that produces the highest score. A description of the evaluation method and entity categorization is available at (NIST, 2008).

3.2 Results and discussion

The results of running the ACE evaluation script on the system output are shown in table 1. The translingual projection system achieves higher scores than all other systems on all measures. Although it achieves only a 2 point improvement over the baseline ACE value, it should be noted that this represents a substantial number of attributes per cross-document entity that it is getting right.

System	Threshold	B-Cubed			ACE Val.
		Prec	Rec	F	
Baseline		37.5	44.1	40.6	19.2
TrnsProj		38.4	44.8	41.3	21.2
CtxtSim	0.2	37.6	35.2	36.4	15.9
CtxtSim	0.3	37.4	43.8	40.3	18.9
CtxtSim	0.4	37.5	44.1	40.6	19.3
CtxtSim	0.4956	37.5	44.1	40.6	19.3
CtxtSim	0.6	37.5	44.1	40.6	19.2

Table 1: Scores from ACE evaluation script.

On the other hand, as the context similarity threshold increases, we notice that the B-Cubed measures reach identical values with the baseline but never exceed it. But as it decreases, it loses B-Cubed recall and ACE value.

While two within-document entities whose longest mention strings match exactly and are legitimately coreferent are likely to be mentioned in the same contexts, it seems that a lower (more liberal) threshold introduces spurious links and creates a different entity clustering.

Translingual projection appears to include links that exact string matching in Arabic does not—part of its purpose is to add close matches to those found by exact string matching. It is able to include these links partly because it allows access to resources in English that are not available for Arabic such as Wikipedia alias lists.

4 Conclusions and Future Work

We have evaluated and discussed a set of techniques for cross-document coreference in Arabic that can be applied in the absence of significant training and evaluation data. As it turns out, an approach based on machine translation is slightly better than a string-matching baseline, across all measures. It worked by using translations from Arabic to English in order to liberalize the string-matching criterion, suggesting that using further techniques via English to discover links may be a fruitful future research path. This also seems to suggest that a Bagga and Baldwin-style vector-space model may not be the first approach to pursue in future work on Arabic.

However, varying other parameters in the context similarity approach should be tried in order to gain a fuller picture of performance. One of them is the df-cut of the MapReduce-based similarity computation. Another is the width of the word token window we used—we may have used one that is too tight to be better than exact Arabic string-matching.

References

Javier Artiles and Satoshi Sekine and Julio Gonzalo 2008. Web People Search—Results of the first evaluation and the plan for the second. *WWW 2008*.

A. Bagga and B. Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. *COLING-ACL 1998*.

Y. Chen and J. Martin. 2007. Towards robust unsupervised personal name disambiguation. *EMNLP*.

D. Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics, 33(2)*.

J. Dean and S. Ghemawat. 2004. MapReduce: Simplified Data Processing on Large Clusters. *OSDI*.

T. Elsayed and J. Lin and D. W. Oard. 2008. Pairwise Document Similarity in Large Collections with MapReduce. *ACL/HLT*.

Z. Li and S. Khudanpur. 2008. A Scalable Decoder for Parsing-based Machine Translation with Equivalent Language Model State Maintenance. *ACL SSST*.

L. Lloyd and Andrew Mehler and Steven Skiena. 2006. Identifying Co-referential Names Across Large Corpora. *Combinatorial Pattern Matching*.

NIST. 2008. Automatic Content Extraction 2008 Evaluation Plan (ACE08).