# **Detection of Entity Mentions Occurring in English and Chinese Text**

Kadri Hacioglu, Benjamin Douglas and Ying Chen Center for Spoken Language Research University of Colorado at Boulder {hacioglu,benjamin.douglas,yc}@colorado.edu

## Abstract

In this paper, we describe an integrated approach to entity mention detection that yields a monolithic, almost language independent system. It is optimal in the sense that all categorical constraints are simultaneously considered. The system is compact and easy to develop and maintain, since only a single set of features and classifiers are needed to be designed and optimized. It is implemented using oneversus-all support vector machine (SVM) classifiers and a number of feature extractors at several linguistic levels. SVMs are well known for their ability to handle a large set of overlapping features with theoretically sound generalization properties. Data sparsity might be an important issue as a result of a large number of classes and relatively moderate training data size. However, we report results that the integrated system performs as good as a pipelined system that decomposes the problem into a few smaller subtasks. We conduct all our experiments using ACE 2004 data, evaluate the systems using ACE metrics and report competitive performance.

## 1 Introduction

The entity-relation (ER) model (Chen, 1976) views the physical world as a collection of entities with complex relationships. Automatic extraction of this model from raw text is important for creating a knowledge base (such as relational databases, marked-up text etc.) that can be used to achieve better end-to-end performances in several natural language processing (NLP) applications including information retrieval, question answering and machine translation. For example, in a typical QA system this knowledge base can be used to facilitate extraction of answers and retrieval of relevant documents.

Entities and relations in a document can be mentioned in several different ways. For example, a person entity, e.g. **Bill Clinton**, can be expressed in many different ways such as *The President*, *President Clinton*, *Mr. Clinton*, *he*, *him* etc. Similarly, one can express a geo-political entity, e.g. **United States**, as *his country* or another person entity, e.g. **Hillary Clinton**, as *his wife*, and their relation to the entity **Bill Clinton** as "president-of" and "family", respectively. It is clear that the detection of these mentions is the first crucial step for the extraction of the ER model to populate a database or an ontology.

Extraction of entities and their relationships is usually done in a pipelined system that first identifies entity mentions, next resolves mentions into unique entities (co-reference) and finally finds relations among them (Florian et al., 2004; Kambhatla, 2004). In that architecture, the errors in the first stage propagate and reduce the performance of subsequent stages; namely, co-reference resolver, that clusters all different mentions of an entity into a unique entity, and relation finder, that links entities according to their relationships. In fact, the subtask of entity mention detection itself is a very challeng-

Table 1: Categorical structure of entities in ACE program

Entity Mention				
Entity			Mention	
Type Sub-Type Class			Туре	Role

ing subtask since respective expressions can have relatively complex syntactic and categorical ("semantic") structures. That is, entity mentions in a body of text can occur in relatively complex embedded constructs with many attributes. Table 1 illustrates the categorical structure of an entity mention as specified in the Automatic Content Extraction (ACE) program run by NIST (ACE, 2004). Compared to the previous years the number of entity types and subtypes is greater.

The following segment of a sentence provides a typical example of the annotation:

[The [[Jordanian] military] spokesman] added ...

For simplicity, the entity mention attributes are excluded. The annotation clearly shows the embedded structure of entity mentions. We identify three entity mentions as *The Jordanian military spokesman, Jordanian military* and *Jordanian*.

Due to its complex nature, it is not uncommon that the mention detection task itself is also divided into a number of smaller sub-tasks. However, in this paper, we adopt an integrated classification approach to this problem that yields a monolithic structure. This allows all attributes, which define the categorical ("semantic") structure of a mention, to be jointly considered. The system has the ability to achieve better performance in principle provided that there is "enough" data to train, is easier to maintain and develop, and has a single set of features and classifiers to be engineered. All possible class labels are obtained by filling in the values of each attribute in the label etype\_subtype\_class\_mtype\_role, where, to avoid confusion, etype and mtype are used to denote entity and mention types, respectively.

Our data representation requires segmenting documents into sentences and then tokenizing sentences into words and punctuation. Each word is then assigned a label depending on its role in the mention. This data representation reduces the problem to a tagging task. For each token in focus, we create a number of features at lexical, syntactic and semantic levels. Additionally, we augment those features using features from external resources (e.g. named entity taggers, gazetteers, wordnet). We train a number of one-versus-all classifiers (Allwein et. al, 2000) using SVMs (Vapnik, 1995; Burges, 1998). During testing, classification of each token is performed in a greedy left-to-right manner using a finite-size sliding context window centered at the token in focus (Kudo and Matsumato, 2000).

This approach yields a large number of classes and a large number of overlapping features. We used a machine learning framework based on SVM classification since a large number of classes (in a oneversus-all set-up) and a large number of overlapping features can be easily handled with good generalization properties. We argue that data sparsity and computational complexity is not as severe as it might be expected in the other machine learning methods that are based on maximum likelihood parameter estimation. In other words, we claim that the large set of classification labels and training data sparseness are not major drawbacks. To provide evidence for this we also consider an approach that divides the task into relatively simpler tasks with considerably smaller numbers of labels. The approach yields a pipelined structure in which the decisions in earlier stages are used in later stages. We report results that the integrated approach performs similar to, and in some cases, even slightly better than the pipelined structure.

We also implement a novel post-processing scheme based on an entity base (EB) created from the tagged test data. This is motivated by the fact that an entity is identically referenced several times in a document. However, depending on the capitalization information of the entity mention and context in which it occurs, the entity can be missed at several positions in the document. A simple postprocessing algorithm that checks untagged tokens with low confidence against the EB is implemented. In doing so, it is highly likely that some of those missed entities could be identified. This is expected to reduce misses at the expense of false alarms. We report results that support our expectation.

The paper is organized as follows. Section 2 describes the ACE 2004 data used for training and evaluation. In Section 3, the problem is explained

 Table 2: ACE 2004 corpus statistics for English and Chinese text.

	Language	Train	Test
ĺ	English	~ 150K words	~ 50K words
	Chinese	~ 150K words	~ 50K words

and its data representation is introduced. Section 4 describes the general system architecture, that consists of a number of feature extractors, a (machine-learned) classifier and a simple post processor. In section 5, the features used for both English and Chinese systems are described. In section 6, we describe an alternative pipelined system. A novel post processing algorithm is introduced in section 7. Section 8 reports experimental results. Concluding remarks are made in the final section.

## 2 ACE Data

The ACE 2004 corpus consists of various text annotated for entities and relations. This corpus was created by the Linguistic Data Consortiom (LDC) in three languages: English, Chinese and Arabic (with support from the ACE program that began in 1999). Resources for data are newswire reports and broadcast news programs. Table 2 gives train and test statistics of this corpus for English and Chinese languages. Both languages have almost the same amount of data for both training and evaluation.

## **3** Problem Description and Data Representation

As shown in Table 1, an entity mention is characterized along 5 dimensions; namely *etype, sub-type class, mtype* and *role*. The ACE program specifies seven entity types; *person, organization, geopolitical, location, facility, vehicle, weapon.* All entity types except *person* are further divided into several sub-types. For example, *organization* has *government, commercial, educational, non-profit* and *other* as its sub-types. The *class* attribute describes the kind of reference the entity mention makes to the entity in the world by taking one of the values {*generic, specific, negative, under-specified*}. Entity mentions are further characterized according to linguistic types of references as name (proper noun), nominal (common noun), pronominal (pronoun) and premodifier. The *role* of entity mention applies only to *geo-political* entities indicating the role of the entity in the context of the mention as one of *person*, *location*, *organization* and *geo-political*. For further details the reader is referred to (ACE, 2004)

All entity mentions in the original data are XML tagged with their respective attributes. In addition to the full extent of mentions, mention heads are also tagged. Referring to the previous example, the entity mention "The Jordanian military spokesman" which refers to a PERSON has the word "spokesman" as its head. Similarly, the entity mention "Jordanian military" which refers to an ORGANIZATION has the word "military" as its head. If one reduces the problem of entity mention detection to the detection of its head, the nature of the problem changes and the annotation of data becomes flat;

The [GPE Jordanian] [ORG military] [PER spokesman] .....

This allows us to consider the problem as a tagging/chunking problem and describe each word as beginning (B) an entity mention, inside (I) an entity mention or outside (O) an entity mention (Ramhsaw and Marcus, 1995; Sang and Veenstra, 1999). However, we believe that the information regarding the embedded structure in which the heads of entities occur is also useful for subsequent stages of an IE system including inference of relations among heads occurring in the same embedded construct. So, in addition to the IOB tags we introduce bracketing tags that might partially recover the embedded structure surrounding the heads. We refer to the following simple example

[Javier Trevino] was [the campaign manager for [the [ruling party] candidate [Fox] beat ]].

to illustrate our tokenwise vertical representation:

#SNT_BEG#	
Javier	B-PER_NAM
Trevino	I-PER_NAM
was	0



Figure 1: System Architecture

(\* the campaign B-PER\_NOM manager \* for the (\* \* ruling B-ORG\_NOM party candidate B-PER\_NOM Fox B-PER\_NAM \*)) beat 0 #SNT\_END#

If one does not use the bracketing representation, all non-head tokens will be labeled as "Outside". We believe that it is useful to discriminate the tokens that take part in mentions from those that do not occur in mentions.

## 4 General System Architecture

The general system block diagram is illustrated in Figure 1. It consists of a pre-processor, several feature extractors, a classifier and a post-processing module. Although the architecture is language independent, there are some minor language specific differences in some modules depending on the nature of the language and availability of resources for that language. In the following, we briefly describe both English and Chinese systems and indicate differences between them.

In the English system, the pre-processor segments the documents into sentences. It also includes a caser that restores the capitalization information of text without case (e.g. broadcast news) and a tokenizer that separates contractions and punctuation from words. Tokenized sentences are then processed at different linguistic levels to create features. At this stage, we employ a lexical pattern analyzer, part-of-speech tagger, a base phrase chunker, a syntactic parser, a dependency analyzer, look-up interfaces to external knowledge sources, and external small scale named entity taggers trained on different genres of text with different machine learning algorithms. All features are combined and then input to a classifier based on one-versus-all SVM classifiers. Finally, we perform simple post-processing to make sure that the final bracketing information is consistent.

The POS tagger and BP chunker are trained in-house using the Penn TreeBank. The syntactic parser is the Charniak parser which has models trained on the Penn TreeBank. The dependency analyzer performs dependency analysis using a set of head rules. The software was generously made available to us by the University of Maryland. The look-up interface to external knowledge sources such as WordNet or gazetteers is implemented using simple pattern matching.

In the Chinese system, the pre-processor is slightly different from that of the English system. It (obviously) does not need a caser and considers single Chinese characters as the minimal units of processing. It jointly segments a document into sentences and words. Then, it passes both word and sentence segmentation information to the subsequent stages along with Chinese characters. The SVM-based joint sentence/word segmenter is trained using the Chinese TreeBank (CTB). Linguistic analysis at different levels is performed in a manner similar to the analysis in the English system. In the Chinese system, the CTB is used to train a SVM-based POS tagger and BP chunker. The syntactic parser is trained on the CTB using Dan Bikel's parser. Dependency analysis is performed as in the English system using a set of Chinese head rules. Several in-house external taggers are trained using SVMs and different corpora. We have used only gazetteers for chinese as external knowledge sources.

## **5** Features

The following features are used in the English system:

- tokens: words in their original and all lowercased forms
- **n-grams**: token prefixes and suffixes of length less than and equal to four
- **lexical patterns**: indicate case information (all lower-case, mixed case, first letter capital, all upper-case), is\_hyphen, type (numeral, al-phanumeral, alpha, other)
- Part of Speech tags
- **BP Positions**: The position of a token in a BP using the IOB representation (e.g. B-NP, I-NP, O etc.)
- Clause tags: The tags that mark token positions in a sentence with respect to clauses. (e.g \*S)\*S) marks a position that two clauses end)
- Named entities-1: The IOB tags of named entities. There are four categories; LOC, ORG, PERSON and MISC. A SVM-based tagger which is trained on CoNLL 2003 shared task data is used.
- Named entities-2: IOB tags of named entities found by the Identifinder (Bikel et. al, 1999); a HMM-based named entity tagger with 29 classes
- Named entities-3: IOB tags from a named entity tagger trained on MUC-6 and MUC-7 data using only the entity classes PERSON, LOCA-TION and ORGANIZATION.

- **Gazetteer labels**: indicate the name of the list to which the token belongs. Simple pattern matching is employed here.
- WordNet categories: concepts or class names in the WordNet 2.0 hypernym hierarchy rooted at "entity" concept. We trace hypernym hierarchies of the two most frequent senses of tokens that are tagged as nouns (NN, NNS, NNP etc.) to the top concepts. We count the number of concepts (that match to ACE entity types and subtypes) that occur in the hypernym hierarchy indicating that *token* is a (kind of) *concept*. The concepts (i.e entity/types/subtypes) with the maximum counts in the top two senses are selected as features (can also be considered as "maybe" labels)
- **Syntactic tags**: patterns of non-terminals and brackets that indicate the position of tokens in syntactic trees.
- Head words: words that the tokens depend
- POS of Head words:
- **main verb**: the verb at which the dependency parse tree is rooted.
- **Relations**: the grammatical and semantic relations between tokens and their heads.
- Head word flag: indicates whether the token plays a role of head in the sentence.

The features used in the Chinese system are

- tokens: Chinese characters
- token positions: IOB tags that indicate position of characters in words
- **Part of Speech tags**: POS tags of words to which tokens (characters) belong
- **BP Positions**: The position of a token in a BP using the IOB representation (e.g. B-NP, I-NP, O etc.)
- Named entities-1: IOB tags of two type of entities; *location* and *person*. A SVM based tagger trained on part of the Sinica corpus from Taiwan is used to generate these features.

- Named entities-2: IOB tags of named entities: person, location, organization etc. Another SVM based tagger trained on the People Daily data from mainland of China.
- Gazetteer labels: indicate the name of the list to which the token belongs. Simple pattern matching is employed here. Examples are labels that indicate Chinese last name, foreign person last name, first name etc.
- **Syntactic labels**: base phrase chunk labels and paths in syntactic trees
- Head words: as determined by Chinese dependency analysis
- POS of Head words:
- **Relations**: the grammatical and semantic relations between tokens and their heads.

## 6 A Pipelined System

As mentioned earlier the structure of entity mention categories is very complex. Considering all attributes together yields a large number of classes. One can argue that the large number of classes and data sparsity is an important issue here that it might have significant effect on performance. However, several attempts to divide the task into simpler subtasks have failed to yield a system with a better performance than that of the integrated system. In this section, we describe one such system.

The system consists of three stages in cascade: (i) entity mention extent detector, , (ii) mention type detector and (iii) entity type, subtype and mention role detector. Referring to the earlier example, the data representation in terms of class labels at each level is as follows:

#### #SNT\_BEG#

Javier	(*	B-NAM	PER
Trevino	*)	I-NAM	PER
was	0	0	0
the	(*	0	0
campaign	*	0	0
manager	*	B-NOM	PER
for	*	0	0
the	(*	0	0
ruling	(*	0	0

party	*)	<b>B-NOM</b>	ORG
candidate	*)	<b>B-NOM</b>	PER
Fox	*	B-NAM	PER
beat	*))	0	0
	0	0	0
#SNT_EN	D#		

where the second column is for the extent labels of mentions in bracketed representation, the third column is for the mention type labels in IOB representation and the last column is for the type labels (subtype and role labels are omitted for the sake of simplicity) of entity mentions in plain representation.

The pipelined system operates as follows. First it detects embedding structure of mention extents. Using that information the second stage identifies the type of mentions. In the final stage, the system identifies entity types, subtypes and mention roles using information (as features within context) from previous stages. Finally we combine all information into entity mention attributes and resolve inconsistencies by simple postprocessing.

Here, we have not done any feature selection specific to each stage. Instead we used the same features in all stages. One can argue that this is not the optimal set up for a cascaded system; separate feature design and selection should be made for each stage. Also we acknowledge that there are several other ways of dividing the task into smaller, simpler subtasks. Although we have not explored all possible pipelined architectures with all possible feature selections, we conjecture that the data sparsity is not as big an issue in SVMs as expected to be in the other machine learning algorithms based on maximum likelihood parameter estimation such as those based on maximum entropy (ME) or conditional random fields (CRF) frameworks.

## 7 A Novel Post-Processing Method

In our experiments, we have consistently observed that the identical mentions of a unique entity are missed depending on the missing capitalization information, unseen context and errors in feature extraction. For example, although the *name* mention of *person* "Eminem" is captured at several positions in the document, the entity mention "eminem" is missed, probably, due to its missing capitalization.

Table 3: Statistics on ACE 2004 data.

Language	Train Samples	Test Samples	# Joint Classes	# Pipelined Classes		
				Extent MType EType-SubTypey-Ro		EType-SubTypey-Role
English	~ 167K	~ 61K	384	24	9	93
Chinese	~ 307K	~ 105K	374	15	7	95

As a solution we propose a post-processing method that is based on an entity base (EB) created from the tagged text. We populate the EB with all entity mentions (particularly with those that have *name* values) identified in the text. After we create the EB, we tag the text again by case insensitive pattern matching. We determine all tagged tokens that were initially left untagged or tagged with a different label by the SVM classifier. Using the SVM output (distance from separating hyperplane) as a confidence measure, we accept or reject the new tag based on a preselected threshold.

## 8 Experiments and Results

In this section, we describe the experiments conducted and results obtained using the ACE 2004 data. The number of training and test examples, which are words/punctuations in English and characters in Chinese, are summarized in Table 3. The number of classes in the joint task and in each pipelined subtask are also included.

In the first set of experiments we evaluated our integrated system and investigated the performance with respect to broad classes of features introduced in section 5, by adding one group of features at a time. Grouping of features into broad classes were done as follows:

- baseline features: tokens
- lexical features: POS, lexical patterns
- syntactic features: base phrase chunks, syntactic tree features
- "semantic" features: heads and grammatical relations
- external features: features from external resources; e.g. wordnet, gazetteers, other entity taggers etc.

Table 4: English system performance with respect to broad classes of features; lex: lexical features, syn: syntactic features, sem: "semantic" features, ext: external features,  $F_{uw}$ : unweighted F-score,  $F_w$ : weighted F-score, ACE: ACE value.

Feature class	$F_{uw}$	$F_w$	ACE
baseline (tokens)	56.5	54.8	36.1
baseline+lex	76.8	86.7	75.6
baseline+lex+syn	76.9	87.4	76.8
baseline+lex+syn+sem	77.1	87.8	77.6
baseline+lex+syn+sem+ext	82.0	90.7	82.9

The results are summarized in Table 4 and Table 5 for both English and Chinese systems. Both unweighted and weighted F-scores, and also ACE values are reported. It is interesting to note that significant gains were achieved by simple lexical and external features when they are added. The degree of improvement by using computationally intensive syntactic and dependency analysis is marginal. This might partly be due to the type of features derived from parse trees and partly due to the mismatch of the genre of text to the text on which the syntactic chunker and parser is trained. Since the dependency analysis is based on the syntactic analysis using a set of head rules, the extracted dependency based features might also be inaccurate. Although we observed moderate improvement for English, those features slightly hurt the performance of the Chinese system. This is because of the fact that the Chinese syntactic parser performs relatively worse than the English syntactic parser.

Table 6 presents the integrated and pipelined system performances using all features extracted for English and Chinese. Post-processing results are also included. It shows notable performance improvement with the recovery of many misses by post-processing. It should be noted that, in the pipelined architecture the post-processing is performed twice; at both mention and entity levels. Table 5: Chinese system performance with respect to broad classes of features; lex: lexical features, syn: syntactic features, sem: "semantic" features, ext: external features,  $F_{uw}$ : unweighted F-score,  $F_w$ : weighted F-score, ACE: ACE value.

Feature class	$F_{uw}$	$F_w$	ACE
baseline (tokens)	77.6	83.5	70.8
baseline+lex	78.3	85.2	73.4
baseline+lex+syn	76.1	83.7	70.8
baseline+lex+syn+sem	74.8	83.6	70.8
baseline+lex+syn+sem+ext	78.4	86.8	76.1

#### **9** Conclusions

We have discussed the significance of the entity mention detection in ER model extraction from raw text and presented the complex syntactic and categorical structure of the entity mentions specified in the ACE program. We have explored different ways of representing the problem and implemented two architecturally different (supervised) machine-learning based systems to accomplish the task; namely, a monolithic system and a cascaded system. We have described those systems in detail and empirically compared them. Both systems have achieved comparable performances on English text. However, the integrated system has achived moderately better performance on Chinese text. We have argued that it is easier to develop and maintain the monolithic system since it has a single set of features and classifiers to be tuned. We believe that the performance levels achieved at mid 80s (in ACE values) for English and at upper 70s for Chinese, using only the ACE data, are competitive. We have introduced a post-processing algorithm based on an entity base created during the testing. It has worked very well for both languages to recover several missed entity mentions and considerably improved the performance.

#### 10 Acknowledgement

We extend our special thanks to Wayne Ward, Steven Bethard, James H. Martin and Dan Jurafsky for their useful feedback during this work. This work is supported by the ARDA Aquaint II Program via contract NBCHC040040.

Table 6: English and Chinese system performances with all features and post-processing:  $F_{uw}$ : unweighted F-score,  $F_w$ : weighted F-score, ACE: ACE value.

English System	$F_{uw}$	$F_w$	ACE
Integrated	82.0	90.7	82.9
Pipelined	82.1	90.8	83.1
Integrated+Post	82.2	91.5	84.3
Pipelined+Post	82.3	91.3	84.0
Chinese System			
Integrated	78.4	86.8	76.1
Pipelined	76.9	85.7	74.1
Integrated+Post	79.6	87.7	77.5
-			

#### References

- E. L. Allwein, R. E Schapire and Y. Singer. 2000. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113-141,
- Dan M. Bikel, Robert L. Schwartz, and Ralph M. Weischedel. 1999 An algorithm that learns what's in a name. *Machine Learning*, Vol. 34, pp. 211-231.
- Chiristopher J. C. Burges 1998. Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2), pages 1-47.
- Peter P. Chen 1976. The Entity-Relationship Model: Toward a Unified View of Data. *ACM Trans. on Database Systems*, Vol. 1, No. 1, pages 1-36.
- 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. *Proceedings of HLT-2004*.
- Nanda Kambhatla. 2004. Combining Lexical Syntactic and Semantic Features with Maximum Entropy Models for Extracting Relations. *Proceedings of ACL-04*.
- Taku Kudo and Yuji Matsumato. 2000. Use of support vector learning for chunk identification. *Proc. of the 4th Conference on Very Large corpora*, pages 142-144.
- Lance E. Ramhsaw and Mitchel P. Marcus. 1995. Text Chunking Using Transformation Based Learning. *Proceedings of the 3rd ACL Workshop on Very Large Corpora*, pages 82-94.
- Erik F. T. J. Sangand and Jorn Veenstra 1999. Representing text chunks. *Proceedings of EACL'99*, pages 173-179.
- The Automatic Content Extraction (ACE) Evaluation Plan. 2004. www.nist.gov/speech/tests/ace/
- Vladamir Vapnik. 1995. The Nature of Statistical Learning Theory. Springer Verlag, New York, USA.