Named Entity Recognition for South Asian Languages

First Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
e-mail@domain

Second Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
e-mail@domain

Abstract

Much work has already been done on building named entity recognition systems. However most of this work has been concentrated on English and other European languages. Hence, building a named entity recognition (NER) system for South Asian Languages (SAL) is still an open problem. This paper makes an attempt to build a named entity recognizer which also identifies nested name entities for Hindi language using machine learning algorithm, trained on an annotated corpus. However, algorithm is designed in such a manner that it can easily be ported to other South Asian Languages. I also compare results of Hindi data with English data of CONLL shared task of 2003.

1 Introduction

Identifying and classifying named-entities into person, location, organization or other names in a text is an important task for numerous applications. I focus here on building a named entity recognition system that will automatically mark the boundaries and labels of the named entities (NEs) in the running text. The system identifies nested named entities too. The nested entities are a superset of the maximal entities. E.g. Lal Bahadur Shastri National Academy of Administration is an organization name and is referred as maximal entity. However it also contains Lal Bahadur Shastri as a person name present inside an organization name and which is referred as a part of nested entity along with Lal Bahadur Shastri National Academy of Administration as an organization name which is a maximal entity.

To make the problem simpler, I split the problem into three sub tasks. The first (NER module) of which identifies whether an entity is a NE or not; the second (NEC module) identifies the type of label associated with each entity; the third (NNE module) identifies the nested name entities (NNE). Since, I focus here on considering all specific 'things' as NE. Hence, I need to choose a larger tag (label) set to classify NEs. Labels considered for this task are: person, organization and location names, measure, time, number, domain specific terms, abbreviation, title and designation.

Conditional random fields (CRFs) (Lafferty et al. 2001) with a variety of novel and traditional features have been used as a classifier for above three modules. CRFs are undirected graphical models, a special case of which is linear chains which are well suited to sequence labeling tasks. They have shown to be useful in part of speech tagging (Lafferty et al. 2001), shallow parsing (Sha and Pereira 2003), and named entity recognition for newswire data (McCallum and Li 2003).

The remainder of the paper is organized as follows: Section 2 describes the related work, Section 3 describes contributions, Section 4 describes the features used by the algorithm, Section 5 describes the various modules, Section 6 describes the experiments and results and Section 7 concludes the paper.

2 Related Work

Named Entity Recognition (NER) has been considered as subtask of Information Extraction. Different NER systems were evaluated as a part of the

Sixth Message Understanding Conference in 1995 (MUC6). The target language was English. Palmer and Day (1997) have worked on Chinese, English, French, Japanese, Portuguese and Spanish and found that the difficulty of the NER task was different for the six languages but that a large part of the task could be performed with simple methods. Cucerzan et al. (1999) used both morphological and contextual clues for identifying named entities in English, Greek, Hindi, Rumanian and Turkish. With minimal supervision, they obtained overall F measures between 40 and 70, depending on the languages used. Collins (1999) showed that use of unlabelled data for NER can reduce the requirements for supervision to just 7 simple seed rules. The CoNLL shared task of 2002 and 2003 focused on language independent NER and has performed evaluations on English, Spanish, Dutch and German and participating systems have performed well. McCallum and Li (2003) used CRFs and feature induction (McCallum 2003) to get an F-score of 71.50 for Hindi language on test-set. May et al. (2003) used HMM to create NER for Hindi and Cebuano. Ekbal et al. (2007) used lexical pattern learning from corpus data for NER for Bangla language.

3 My Contributions

I focus here on building a NER system for Hindi language using conditional random fields (CRFs) using NLPAI Machine Learning Contest 2007 data. The system is built in such a manner that it could be easily ported to other languages. This method was evaluated on test set 1 and test set 2 and attains a maximal F1 measure around 49.2 and nested F1 measure around 50.1 for test-set 1; maximal F1 measure around 44.97 and nested F1 measure 43.70 around for test-set 2. These numbers are pretty bad but I was able to get an F-measure of 58.85 on development set. The great difference in the numbers could be due to some difference in test and development set. I have also compared my results on Hindi data with English data of CONLL shared task of 2003 by introducing interesting phenomena's which are not present in English. Everyone says one cannot compare SAL results with English or any other language which have orthographic features like capitalization as a very important clue whereas Hindi or any other SAL doesn't have capitalization. So to make it even, I perform experiments on English after removing capitalization. Also there is another interesting phenomenon in Hindi or any other SAL i.e. a word can be a common noun as well as a proper noun. For example sambhab sinha is a name of a person but when I use 'sambhab' in a sentence "yaha kaam mujse sambhab nahi" It act as a common noun meaning 'possible' in English. Hindi language is full of such cases making the task more difficult. Hence it becomes very difficult for NER system to classify it as person or not. To do something interesting of that sort, we can do For example: 'Cheese' goes to market. In the above sentence we have replaced 'John' with 'Cheese' to introduce the common noun phenomena of SAL or Hindi. After introducing this as well we make Hindi even to English, now I hope people shouldn't have any problem in comparing results across these languages. Even though later in the paper, I will show that it is still not right to compare results across these languages.

4 Features

The success of any machine learning algorithm depends on finding an appropriate combination of features. This section outlines three types of features.

4.1 Contextual features

- Word Window: A word window of size n centered in position i_w is the sequence of words in the sentence placed at $i_w + j_w$ positions, with $j_w \in [-n , +n]$. For each word in the window, word and it's POS + its relative position j_w forms a feature
- Chunk window: A chunk window of context size n centered in position i_c is the sequence of chunks in the sentence placed $i_c + j_c$ positions, with $j_c \in [-n , +n]$. The tags (labels) of the chunks in the window + its relative position j_c form a feature.

4.2 Statistical features

- Binary features: As name suggests these features have value 0 or 1. These features are not mutually exclusive features that test whether the following predicates hold in the word: all digits, 4 digit number, contains hyphen, punctuation mark, acronym, alpha-

numeric etc. I also modeled whether a particular word is a noun or not using the POS information.

- Trigger words: Using the annotated training data I find all those words which have a high probability of being a number, measure, abbreviation and time. I model 4 binary features giving value 1 to high probable words and 0 to the rest. For example, high probable words for number would be eka, xo, wIna, cAra etc. (words here are in wx-notation) and will get a value as 1.

4.3 Word Internal Feature

- Affixes: Some prefixes and suffixes are good indicators for identifying certain classes of entities. Suffixes are typically even more informative. For example, suffixes like -bad , -pur, -pally are good indicators of a name of a location.

- Words are also assigned a generalized 'word class (WC)' similar to Collins (2002), which replaces all letters with 'a', digits with '0', punctuation marks with 'p', and other characters with '-'. There is a similar 'brief class (BWC) (Settles 2004)' which collapses consecutive characters into one. Thus the words "D.D.T." and "AB-1946" would both be given the features WC=apapap, BWC=apapap and WC=aap0000, BWC=ap0 respectively, in above example hyphen forms the part of punctuation marks. This feature has been modeled since this feature can be useful for both unseen words as well as solving the data sparsity problem.

- Stem of the Word was also obtained using a morph analyzer.

We have tried to use the different combination of all these features for all three modules which I am going to discuss in the next section. But before ending there are few features which I haven't used and would like to use in future. Bag of words i.e. form of the words in the window without considering their position. Gazetteer Features can also be useful. These features couldn't be used due to computational reasons, lack of resources and time.

5 Modules

5.1 NER module

This module identifies whether an entity is a NE or not. I use well-known BIO model. B denotes begin of an entity, I denotes inside an entity; O denotes outside and is not part of any entity. Here I have only one label i.e. NE. Hence it becomes a three class problem with B-NE, I-NE and O as output labels. Here I am identifying NEs as it's an easier task as compare to classifying them among named-entity tag-set. It is also done with a hope that this information can be useful for NEC module. For example in entity like Raja Ram Mohun Roy tags would be Raja/B-NE Ram/I-NE Mohun/I-NE Roy/I-NE. Similarly for Microsoft Corp. tags would be Microsoft/B-NE Corp./I-NE. Words like tiger, eat, happy etc which are not NEs are tagged as O.

5.2 NEC module

Here I try to classify the NEs among various classes/labels like person (like Mahatma Gandhi), location(like Delhi) and organization(like Microsoft Corp.) names, number (like one, two etc), time (like one day), measure (like 5 kg), domain specific terms (Botany, zoology etc), title (Mr., The Seven Year Itch), abbreviation (D.D.T.) and designation (Emperor). For the contest I was supposed to use some sort of heuristics to split the title tag into title object (like The Seven Year Itch) and title person (like Mr.) tag. But due to shortage of time, no annotated data and very less accuracy on title class (shown later in Table 2). I was unable to classify them into two separate classes. Also, there were only 4 or 5 instances of brand label in training data and hence I didn't consider it as part of my system. Hence it becomes a 10(labels/classes) * 2(B+I) = 20 + 1 (O which denotes remaining words) =21 class problem. This module is independent from the previous module. For example in entity like Raja Ram Mohun Roy tags would be Raja/B-NEP Ram/I-NEP Mohun/I-NEP Roy/I-NEP. Similarly for Microsoft Corp. tags would be Microsoft/B-NEO Corp./I-NEO.

We could have tried labeling the identified named-entities from NER module but it has been found that whenever I try to make one module dependent on other, the accuracy of the system couldn't get beyond a particular threshold. Hence I

tried to use the output of the NER module as one of the features for NEC.

5.3 NNE module

Since the length of a nested entity can be anything but as the length of an entity increases, the number of instances corresponding to that decreases. Hence it's better to choose nested entities up to length of 3 for which I have enough training data. I try to train three classifiers to learn entities of length 1, 2 and 3 independently. This allows us to learn nested entities since the bigger entities can have different tags when compared to smaller entities. For example, Srinivas Bangalore will be tagged as a name of a person by a classifier who is trained to classify NEs of length 2. However, Srinivas and Bangalore will be tagged as a name of a person and location respectively by a classifier which is trained to classify entities of length 1.

In this module also I use the same BIO model and there will be 21 classes for each of the three classifiers.

6 Experiments and Discussion

In this section I describe the experiments I performed to evaluate presented algorithm with its variations.

NLPAI 2007 NER contest Corpus, I was provided annotated training and development data comprising of 19825 and 4812 sentences respectively for Hindi. The average sentence length of the corpus is 24.5. The first step was to enrich the data with POS, chunk information and root of the word using POS tagger, Chunker (Avinesh et al. 2007) and IIIT-Hyderabad morph analyzer. Hence, I am assuming that if you want to port this algorithm to any other SAL you should have these existing tools for that particular language.

In the training data, in about 50% sentences i.e.10524 sentences there was not even a single NE. Hence, I removed all such sentences and kept only those sentences which contain NEs i.e. 9301 sentences. But as I thought the remaining 50% of the sentences could be useful as noise which always helps in discriminating whether a word occurs as NE in a particular context or not. Hence I trained a basic classifier using only context information using both types of training data and it was found that the there was not much in a difference in performance of the classifier on the development set. Also to check whether if that noise was comparatively less say 20% of 10524 sentences. I randomly choose 20% sentences out of those 10524 sentences which doesn't contain NEs and add it to remaining 9301 sentences to form a new train set and found that still there was not much effect of adding noise on the overall system. Then I mixed 40%, 60% and 80% still there was no success. The only reason that can justify the above phenomena is that sentences containing NEs already contain sufficient noise and there is no need to add more. Hence I carried all the remaining experiments with sentences containing NEs. The reason for choosing it over the others is it takes less time to train and more experiments could be performed given the time constraints.

Then I tried to find an appropriate set of features for NER and NEC module. For NNE I used the same features as used in NEC module since I don't have explicitly labeled data for nested entities. Also tweaking and tuning of feature doesn't affect the accuracy in a big way.

For NER module, where I am trying to identify name entities; context information seems to be more informative than statistical features. I use a window of -1 to +1 for words, -2 to +2 POS and also use features which are combinations of consecutive POS tags and words. For example Ram/NNP eat/VB mangoes/NNS. Combination features for word 'eat' would be NNP/VB, VB/NNS, Ram/eat, eat/mangoes, NNP/VB/NNS, Ram/eat/mangoes. The stem of the word and chunk information also doesn't affect the accuracy. The prefixes and suffixes of length 3 and 4 are found to improve the accuracy of the classifier. For example Hyderabad will have Hyd, Hyde, bad, abad as prefixes and suffixes of length 3 and 4 respectively. The word class (WC) and Brief word class (BWC) features are also very useful features for recognizing named-entities. I have achieved an F-measure of 64.28 by combination of all these features for identifying name-entities on development set. Table 1 shows the detailed results of named entity recognition (NER) module.

For NEC module, the contextual features as well as statistical features are helpful in deciding to which class a name-entity belongs. I use word and POS window of -1 to +1 as context. No combination features are being used as introduction of such features degrades the accuracy rather than improving it. However the statistical features are found to

Features	Precision	Recall	F-measure
Contextual	64.19	60.53	62.31
Contextual+ Word Internal	64.84	63.73	64.28

Table1: Detailed performance of NER module using only contextual features and combining word internal features.

Entity	Precision	Recall	F-measure
Abbreviation	43.21	36.46	39.55
Designation	69.61	46.84	56.00
Location	67.51	63.08	65.22
Measure	73.98	72.84	73.41
Number	70.41	87.74	78.13
organization	49.71	39.73	44.16
Person	61.18	47.37	53.40
Title	31.82	14.00	**19.44**
Terms	30.81	16.72	**21.67**
Time	67.30	58.53	62.61
Overall	62.60	55.52	**58.85**

Table2: Detailed performance of the best feature set on development set for maximal/nested named entities.

be more useful in this case as compared to NER. Here also prefixes and suffixes of length 3 and 4 are found to be useful. BWC feature alone is sufficient for classification, we don't need to use WC feature for improving the accuracy. Chunk information and stem of the word doesn't improve the accuracy.

I have modeled NER module so that the output of that module can be used as feature for NEC. But using it as a feature doesn't improve the classification accuracy. Also, I tried using the boundary information from the NER module and combining it with labels learned from NEC module. It also seems to be a futile attempt.

I have used unlabelled data i.e. 24630 sentences provided during the contest and used bootstrapping to make use of it. I have doubled the data i.e. 50% manually annotated data and rest is system output on unlabelled data i.e. 12323 sentences; we have used only those sentences which contains at least one NE. With this data I almost get the same accuracy as I got with only manually annotated data. Table 2 shows the detailed performance of the best feature set on development set for maximal/nested named entities using evaluation script of CONLL shared task of 2003. I have used the evaluation script of NLPAI contest to report results on Test set-1 and Test set-2 (which contains 1091 and 744 sentences) for two systems in Table 3 and 4. One trained using only annotated data and the other trained on annotated and bootstrapped data for the same feature set which performed best on development set. For test-set 2, system trained using annotated and bootstrapped data performs better than the system trained using only annotated data. However, for test set1 both the systems perform almost same. One of the reasons for less results as compared to development set is I haven't further classified title tag into title object and title person tag and Test sets contain lot of instances of them.

I have trained a single classifier for all the entities but we can use more classifiers and divide the tags in such a fashion that those which are closer to one another fall in one group. For example we can club number, time and measure in one group and call them as number group since these are closer to each other and train a classifier to automatically annotate these entities in running text. Similarly, we can club person, number, and location and call them as name group. I have attempted a similar experiment using the same features of NEC module for number and name group but still there is no improvement.

For NNE module, I have used the same set of features which I have used in NEC module and I am handling nested entities up to length of 3. Since the development set is not enriched with nested entities, hence it is difficult to optimize the features for this module and the results would be same as NER module since nested entities are superset of maximal entities. For Test set-1 and Test set-2 Table 3 and 4 are used to report results.

May be the accuracy seems to be less but since I have included domain specific terms and some other entities whose training data is very less. Also if some one wants to learn domain specific terms (like Computer, Botany, Umpire, Referee etc.). He needs to train the system for a particular domain. Learning it with rest of the tags is pretty difficult, since a term can be anything depending upon the domain. For NEs like title there are less instances in training data which is a reason for its low F-measure i.e. 19.44 on development set which is even less than terms (i.e. 21.67) which are most difficult to learn. Hence, the results are not as bad

as they look. Also here I have focused on a large tag set but it would be interesting to concentrate only on person, location and organization names. Since most of the systems report accuracy for these entities. Hence I did some experiments with Hindi data concentrating only on person, location and Organization but there is not so much increase in the performance.

Entity	Test set1	Test set 2
Maximal Precision	70.78	55.24
Maximal Recall	37.69	35.75
Maximal F-Measure	**49.19**	43.41
Nested Precision	74.28	58.62
Nested Recall	37.73	33.07
Nested F-Measure	50.04	42.29

Table3: System trained using only annotated data

Entity	Test set1	Test set 2
Maximal Precision	70.28	57.60
Maximal Recall	37.62	36.88
Maximal F-Measure	49.00	**44.97**
Nested Precision	73.90	60.98
Nested Recall	37.93	34.05
Nested F-Measure	**50.13**	**43.70**

Table 4: System trained using annotated and bootstrapped data

When I trained my system on English data (which I have made mono case) of Conll-2003 shared task, with only contextual features, system gets an overall F-measure of 84.09 on development set and 75.81 on test set which is far better than Hindi. I have just used contextual features with window size of -1 to +1 for words, POS and chunk to achieve the results reported in Table 5 for test set. The reason for using only contextual information is that these features give the maximum accuracy and the rest of the features don't increase the accuracy by such a great amount. Also the aim over here is to compare results with Hindi language and not to make the best NER system for English language.

Entity	Precision	Recall	F-measure
Person	82.05	79.16	80.58
Location	84.16	79.32	81.67
Organization	70.76	67.01	68.83
Misc.	73.71	61.11	66.82
Overall	78.40	73.39	75.81

Table 5: System trained on English mono case data using contextual features

Also to include common noun phenomena in English I have taken 10 random person names from the data and replaced them with common nouns and the results are really surprising. By introducing this, system achieves an F-measure of 84.32 on development set and 76.19 on test set which is better than the results on normal system. The number of tokens corresponding to these names in training data is 500. Table 6 contains the detailed results.

Entity	Precision	Recall	F-measure
Person	81.92	79.84	80.86
Location	84.18	80.10	82.09
Organization	71.98	67.13	69.47
Misc.	73.04	60.97	66.46
Overall	78.71	73.83	76.19

Table 6: System trained on English mono case data with common noun phenomena using contextual features

The results for English are far better than Hindi language. The reason is English already has tools like POS tagger and chunker which achieves an F measure around 95 whereas for Hindi we only have an F-measure of 85 for tagger and 80 for chunker. This is the reason why the accuracy of English system didn't fall when I removed capitalization and introduced common noun phenomena since POS context and chunk context helps a lot. Since CONLL 2003 data is already POS tagged and chunked, hence POS and chunks correspond to capitalized data. I hope now we can un-

derstand why it is not still correct to compare results with English. Hence to make it more even, I ran Stanford POS tagger (Toutanova et al. 2003) on the same mono case CONLL 2003 data and then train the model using only word and POS context. The numbers drop on test set by more than 15% as shown in Table 7. One more thing we didn't include chunk context but it only increases the performance by 1 or 2 %. For development set the overall F-measure is around 74%.

Entity	Precision	Recall	F-measure
Person	66.97	53.93	59.75
Location	68.57	56.54	61.98
Organization	71.64	53.55	61.29
Misc.	74.71	55.98	64.01
Overall	69.69	54.84	61.38

Table7: System trained on POS tagger ran on mono-case data

Now these numbers are comparable to Hindi data. The reason is POS tagger performs really bad after removing capitalization. Now the POS tagged data marks proper noun i.e. NNP as common noun i.e. NN or foreign word as FW. The reason is it uses capitalization to mark NNP tag. We still haven't included common noun phenomena. So to so that, I take the common noun phenomenon English data and train the model using the same features as used above. Here also the system performs in the same way. There is just a decrease of 1% in F-measure of person class. Table 8 contains the detailed results. The introduction of common noun phenomena doesn't seem to affect the performance too much. The reason can be context helps in disambiguating between the real 'cheese' and the 'cheese' which has been made up by replacing it with 'John'.

Entity	Precision	Recall	F-measure
Person	65.48	53.37	58.81
Location	68.23	56.18	61.62
Organization	73.95	53.01	61.75
Misc.	74.81	56.27	64.23
Overall	69.74	54.45	61.16

Table8: System trained on POS tagger ran on mono case data which contains common noun phenomenon

After looking at these results, we can easily say that if we can improve the performance of POS tagger, we can do very well on the NER task. Without that it's even difficult for English to give good numbers. It is correct that Hindi and SAL don't have capitalization but we could make use of morphological features since most of SAL are morphologically rich. A hybrid approach involving rules along with machine learning approach could help us to improve POS tagger and NER systems.

Now after seeing results on English let's see what are the actual reasons for less numbers on Hindi data? Inconsistency of annotated data is one of the big problems but it's very difficult to create 100% correct manual data since we have chosen a finely grained tagset. Also the data used for Hindi is from different domains. Hence due to which the lot of terms doesn't occur in corpus more than once. One of the plausible reasons for bad results on test set for Hindi compared to development set could be difference in domain of test set. Also due to lack of resources like gazetteer for SAL the task becomes more challenging to create everything from scratch. Also the accuracy of tagger, chunker and morph analyzer are not as good as when we compare results with English. Also from the perspective of contest the task becomes very difficult for those teams which don't have tools like POS tagger and chunker. So, to make the contest more even, it would be better if the all teams could be provided with the same POS tagged and chunked data. To summarize, I believe it is possible to get better numbers for this task, since lot of people are working on improving POS tagger, chunker and NER systems. The only important thing is to make these tools public, so that people can use them and contribute either by improving accuracy of existing systems or building new NLP tools.

7 Conclusion

In conclusion, I have shown that use of machine learning algorithm on annotated data for Hindi language can be useful and the same algorithm can be useful for other languages. I only need to tune and tweak the features for a particular language. I have described some traditional and novel features for Hindi language. I have also shown that it's better to directly classify name-entities into various labels or classes rather than first recognizing them.

Also the attempt to make use of unlabelled data didn't help much.

Also I have showed that capitalization is one of the important clues for high performance of English on various NLP applications. But we could also recognize some other important clues in SAL and can hope to do better than English without having capitalization.

Directions for future work include concentrating on a smaller tag set and trying to improve accuracy for each of the label. Since still we don't have enough labeled data for other SAL, it would be interesting to try out some unsupervised or semi-supervised approaches. Also I haven't tried rule based approach which could be very handy when combined with some machine learning approach. Hence adopting a hybrid approach should help in improving the accuracy of the system but still it's an open question.

References

Andrew McCallum. 2003. *Efficiently Inducing Features of Conditional Random Fields.* In Proceedings of the 19th Conference in UAI.

Andrew McCallum and Wei Li. 2003. *Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons*. In Proceedings CoNLL 2003.

Avinesh.PVS. and Karthik G. 2007. *Part-Of-Speech Tagging and Chunking using Conditional Random Fields and Transformation Based Learning.* In Proceedings of SPSAL2007

Asif Ekbal and Sivaji Bandyopadhyay. 2007. *Lexical Pattern Learning from Corpus Data for Named entity recognition*. In Proceedings of ICON 2007.

Burr Settles. 2004. *Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets*. In Proceedings of the International Joint Workshop on NLPBA.

David D. Palmer and David S. Day. 1997. *A Statistical Profile of the Named Entity Task*. In Proceedings of Fifth ACL Conference for ANLP.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 Shared Task: *Language-Independent Named Entity Recognition*. In Proceedings of the CoNLL 2002.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. *Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition*. In Proceedings of the CoNLL 2003.

Fei Sha and Fernando Pereira. 2003. *Shallow parsing with conditional random fields*. In Proceedings of the HLT and NAACL 2003.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*. In Proceedings of ICML 2001.

Kristina Toutanova and Christopher D. Manning. 2000. *Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger.* Proceedings of the Joint SIGDAT Conference on (EMNLP/VLC-2000), Hong Kong.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. *Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network.* In Proceedings of HLT-NAACL 2003.

Michael Collins. 2002. *Ranking algorithms for named-entity extraction: Boosting and the voted perceptron*. In Proceedings of ACL 2002.

Michael Collins and Yoram Singer. 1999. *Unsupervised models for named entity classification.* In Proceedings of the Joint SIGDAT Conference on EMNLP *and Very Large Corpora*.

Silviu Cucerzan and David Yarowsky. 1999. *Language independent named entity recognition combining morphological and contextual evidence*. In Proceedings of 1999 Joint SIGDAT Conference on EMNLP and VLC.

Wei Li and Andrew McCallum. 2003. *Rapid Development of Hindi Named Entity Recognition Using Conditional Random Fields and Feature Induction*. In Proceedings of ACM TALIP.