Named Entity Recognition

Vijay Krishnan and Vignesh Ganapathy
{vijayk , vignesh}@cs.stanford.edu

December 16, 2005

Abstract

Named entity recognition (NER) is a subtask of information extraction that seeks to locate and classify atomic elements in text into predefined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc. We use the JavaNLP repository(*http://nlp.stanford.edu/javanlp/*) for its implementation of a Conditional Random Field(CRF) and a Conditional Markov Model(CMM), also called a Maximum Entropy Markov Model. We have obtained results on majority voting with different labeling schemes, with backward and forward parsing of the CMM, and also some results when we trained a decision tree to take a decision based on the outputs of the different labeling schemes. We have also tried to solve the problem of label inconsistency issue by attempting the naive approach of enforcing *hard* label-consistency by choosing the majority entity for a sequence of tokens, in the specific test document, as well as the whole test corpus, and managed to get reasonable gains. We also attempted *soft* label consistency in the following way. We use a portion of the training data to train a CRF to make predictions on the rest of the train data and on the test data. We then train a second CRF with the majority label predictions as additional input features.

1 Introduction

Named entity recognition (NER) is a subtask of information extraction that seeks to locate and classify atomic elements in text into predefined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc.

Existing approaches to NER have explored exploiting:

- Word features of the token and the words in its neighborhood.
- The parts of speech of the word in question and its neighbors.
- Features corresponding to certain prefixes and suffixes of the word and its neighbors.
- Features corresponding to the labels of its neighbors.

Existing approaches have used Maximum Entropy Taggers, SVMs and CRFs for NER. We used the Conference on Computational Natural Language Learning(CoNLL) dataset in order

to train and test our NER tagger. The CoNLL dataset is a standard benchmark used in the literature. Our training data was NER annotated text with about 220, 000 tokens, while the test data contained around 50,000 tokens.

2 Combining results of different labeling schemes

There are two broad representations for Named Entity chunks i.e. Inside/Outside representation and representations encoding Start/End information. The latter is more useful if we have instance of Named Entities of the same kind immediately following each other since it enables us to locate the boundaries.

We have collected the existing training data and modified it to represent them in the forms IOB1, IOB2, IOE1, IOE2, IOBES and IO formats. These are described below:

- IOB1: Here, I is a token inside a chunk, O is a token outside a chunk and B is the beginning of chunk immediately following another chunk of the same Named Entity.
- IOB2: It is same as IOB1, except that a B tag is given for every token, which exists at the beginning of the chunk.
- IOE1: An E tag used to mark the last token of a chunk immediately preceding another chunk of the same named entity.
- IOE2: It is same as IOE1, except that an E tag is given for every token, which exists at the end of the chunk.
- START/END: This consists of the tags B, E, I, S or O where S is used to represent a chunk containing a single token. Chunks of length greater than or equal to two always start with the B tag and end with the E tag.
- IO: Here, only the I and O labels are used. This therefore cannot distinguish between adjacent chunks of the same named entity.

There are existing machine learning techniques such as Hidden Markov Models (HMM), Maximum Entropy(ME) and Support Vector Machines (SVMs) used for NER. Currently we are working with an existing implementation of the Maximum Entropy model from the JavaNLP repository. The features considered here for each word w(i) are the two words on either side of the word(i), their parts of speech(POS), and their labels. In addition a few morphological features like focus word prefix, focus word suffix, previous word prefixes and suffixes.

There are parsing methods considered for the data One is forward parsing which parses from left to right and the other is backward parsing which parses from the right to left. So, with six different chunks representation schemes and two parsing techniques, we have 12 different outputs.

We converted back the outputs of the 12 different models to the IOB1 scheme and then implemented a simple majority vote of the decisions of the different taggers. We managed to get small gains on the raw token tagging accuracy but got mixed results with regard to performance on the F1 score. We also trained a Decision tree on the output of the different taggers, in order to make a decision jointly based on the different outputs, with the objective of combining the outputs in a better fashion than giving each model equal weight.

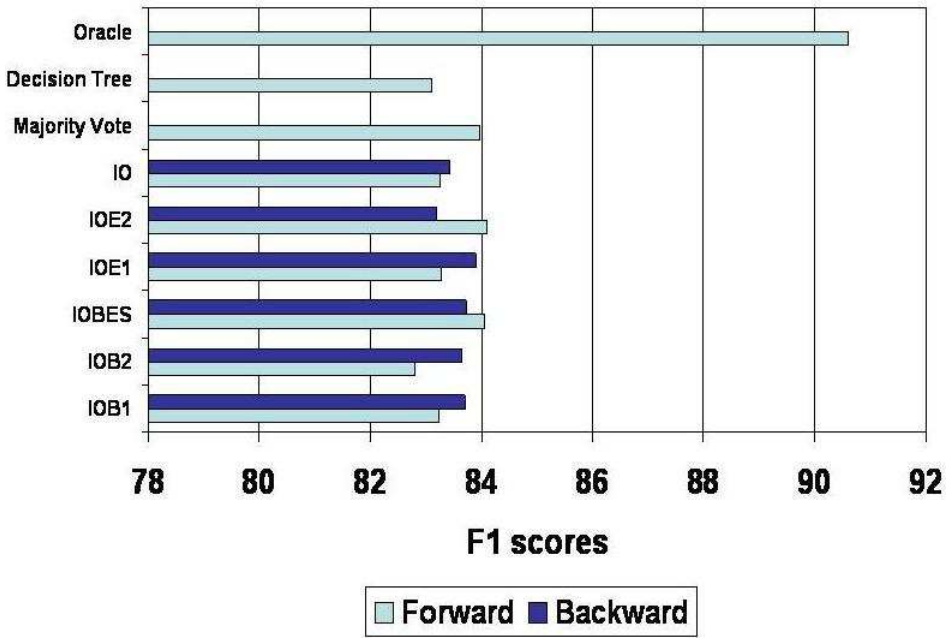

Figure 1: F1 scores obtained with different labeling schemes with the Conditional Markov Model(CMM) run backward and reverse. Also shown are the F1 scores corresponding to a majority voting of the predictions of these different labelings and the F1 scores, for the case where a decision tree gave the output label, taking as input, the output label of the 12 different taggers.

2.1 Results

It can be seen from figure 1 that among the six different labeling schemes, IOE2 and IOBES tagging gave marginally better results as compared to others using the forward parsing technique. The majority vote gave mixed results for different datasets, but typically tended to perform close to the best of the labeling schemes We also calculated the F1 score for an "oracle" tagger which tags the entity correctly if any one of the twelve outputs tagged the entity correctly. This result is a kind of theoretical upper bound estimate for the best F1 score that we can possibly get using this combination of multiple labeling schemes.

3 Label Consistency

We performed manual analysis of the error to find out other ways to improve the F1 score of the results of different labeling schemes. A good number of errors could be fixed by proper application of label consistencies. An example of this is "Melbourne Cricket Ground" which occurred often in the text and was tagged correctly for some instances and incorrectly for other instances.

It is very likely that multiple occurrences of sequences of tokens correspond to the same entity especially if they are close to each other. Since these are non-local dependencies, we need approximate methods to capture them. Finkel et al. capture label-consistency at the document level, by fixing a penalty for disagreement among labelings for the same sequence of tokens, at its different occurrences in a document, and used Gibbs Sampling for tractable inferencing.

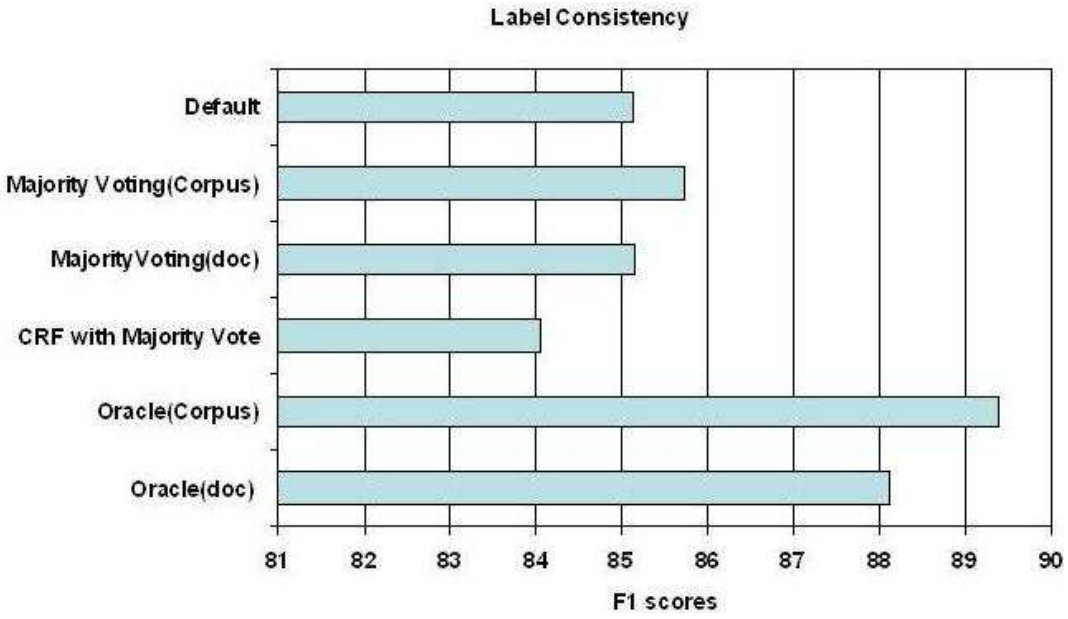

Figure 2: Graph showing the F1 scores with the default CRF, the F1 scores corresponding to enforcement of *hard* label consistency at the corpus and document levels respectively, and the F1 score with *soft* label consistency(with 2 rounds of CRF training). The last two F1 scores correspond to the Oracle scores at the corpus and document level respectively. The Oracle scores are obtained by awarding a point if the entity is labeled correctly in any part of the corpus/document.

We find that there is scope for improvement even with label-consistency at the corpus level. We would intuitively expect that there should be smaller penalty for label disagreement of a token at the corpus level, when compared to disagreement with labelings within a single document. However, there are relatively small number of cases wherein a named entity token appears multiple times in a single document, when compared to its count across the corpus. Thus, label-consistency over the whole corpus is quite a rich source of information, which could be exploited to get good gains on the F1 measure.

We obtained improvements with even naive *hard* label-consistency enforced at the document and corpus levels. We enforced this *hard* consistency by doing post-processing on the output of the CRF. For multiple occurrences of a sequence of tokens in a named entity, we changed the labeling of all occurrences to the majority labeling for that sequence of tokens over the whole corpus. We also attempted a similar experiment, wherein we flipped the labels to the majority labeling within a document.

We also attempted *soft* label consistency in the following way. We use a portion of the training data to train a CRF to make predictions on the rest of the train data and on the test data. We then train a second CRF with the majority label predictions as additional input features.

3.1 Results

It can be seen from figure 2 that the improvement in the F1 score is significant in case of the corpus level majority vote as compared to the document level vote. The result of using these two

majority vote results as additional input features for the CRF did not give any improvement. The results with the Oracle taggers, depict a loose upper bound on the extent to which we could hope to improve performance with label consistency.

4 Conclusions and Future Work

Majority Voting over different labeling schemes on the CMM gave us mixed results with different data sets. In most cases it did reasonably well while the decision tree method performed poorly in comparison.

Capturing of Label consistency is a rich source of information, which we also observed during error analysis. We could get improvements with even naive enforcement of hard-consistency via Majority voting. We could not get improvements using label-consistency Majority vote information obtained from a CRF trained on different data, as additional features. We could look at better ways to encode the *soft* label consistency which is a non-local dependency. One option is to attempt using Markov Random Fields. Another option is to train a CRF without label-consistency incorporated and independently learn a model for label-consistency alone. Both

In addition, we are currently not able to do significantly better with CRFs than with CMMs. This is because, while the CMM can be given a previous context of say five tokens, we cannot give a previous context of more than one or two tokens to the CRF since that exponentially blows up training and testing time. This could be possibly handled by pruning down the search space of the CRF possibly by getting the best three labels for each token without considering a larger context and then finally searching over these labelings alone.

Acknowledgments

We would like to thank Prof. Chris Manning his help and guidance. We would also like to thank Jenny Finkel and Trond Grenager for the highly fruitful and enlightening discussions we shared.

References

[1] Kudo T and Matsumoto Y. 2001. Chunking with Support Vector Machines. In *NAACL-2001*, pages 192-199.

[2] Erik Tjong and Kim Sang. 2002. Memory-Based Named Entity Recognition. In *CoNLL-2002*, pages 203-206.

[3] John Lafferty, Andrew McCallum and Fernando Pererira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML-2001.*

[4] Jenny Finkel, Trond Grenager, and Christopher D. Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In ACL-2005.

[5] JavaNLP code documentation. Online at: *http://nlp.stanford.edu/nlp/javadoc/javanlp/*