# CS229
# *Supervised Entity and Relation Extraction*

Andrey Gusev and Mason Smith

## Abstract

We present a system for extracting entities and relations from documents: given a natural text document, identify and classify entities mentioned in the document (e.g. people, locations, etc.) and relations between these entities (e.g. person X lives in location Y). We designed separate systems for relation extraction given already-labeled entities, and for entity extraction from plain text, and then combined the two systems in a pipeline. We ran our system on a small set of sports articles and two larger sets containing biomedical and newswire articles. Both entity extraction and relation extraction are trained in a supervised manner using annotations in the datasets. For entity extraction these annotations allow us to train a conditional random field sequence classifier by matching annotated types to part of speech parse trees that are built from the text. For relation extraction we ran logistic regression using a set of syntactic and surface features of the sentence data. We evaluated the utility of these features using forward-search feature selection, and we also separately evaluated the contribution of the syntactic features by running the system using surface features alone. As a result the pipeline achieved an F1 score of 0.40 on the small sports articles dataset and 0.62 on the larger newswire article dataset. We present and discuss results from individual subsystem on these sets. The discussed system is in Java and comprises roughly 30 classes implemented for this project. This system is integrated into Stanford's JavaNLP infrastructure, is fully extensible and can be used as a basis for more advanced entity and relation extraction systems.

## The problem

We worked on the problem of entity and relation extraction for three datasets: the "football", "newswire", and "BioNLP" datasets. Each dataset consists of a set of sentences, a set of entity types, a set of relation types, and, for each sentence, annotations specifying the sentence's entities and relations; the task was to reproduce the correct annotations. Information on each dataset is shown in Table 1.

## Entity Extraction Subsystem

### Overview

There are three major components of the implementation for entity extraction subsystem. The first component is parsing of the dataset to create a logical representation of annotated entities and the relationships they describe. This component is shared with the relation extraction subsystem. The next component is using a lexicalized parser to annotate sentences with part of speech tags. Finally, the last part of the implementation trains the model and runs the classifier on the test set. Depending on the dataset we either perform k-fold cross-validation or use explicit test and training datasets. The parsing is dataset dependent and is not discussed in detail here. As an example we present the format for the Football dataset, where "arg type" refers to annotated entities and "relation type" refers to relations that depends on an entities.

```
<relation type="gameDate" start="186" end="387">
By winning the National Football League (NFL) playoff game, the 49ers will host the winner of Sunday's Dallas-Green Bay game on January 15 to decide a berth in the January 29 championship game at Miami.
    <arg type="NFLPlayoffGame" start="346" end="386">the January 29 championship game at Miami</arg>
</relation>
```

Part of speech annotation is done using a lexicalized parser (from the Stanford JavaNLP library) that attaches part of speech tags to each word in a sentence. These annotations are types such as NP - noun phrase, VP - verb phrase, PP - propositional phrase, etc. As can be observed in the example above a lot of entity annotations are multi term annotations. In order to train our classifier we need annotated single terms - thus for each multi-term annotation we find its head word. We first try to parse the entire sentence and then obtain a parse tree for the argument sub-phrase. This approach

| Dataset name | Football | Newswire | BioNLP |
|---|---|---|---|
| Sample sentence | San Francisco's reign over the National Football League ended here Saturday with a 27 to 17 loss to Green Bay. | However, Franca Chlistovsky, who heads the Brera metereologic institute in Milan, said this winter's dry spell was not exceptional... | Leukotriene B4 stimulates c-fos and c-jun gene transcription and AP-1 binding activity in human monocytes. |
| Entity types | Date, FinalScore, NFLPlayoffGame, NFLGame, NFLTeam | Org, Location, People | Protein |
| Relation types | gameDate(NFLGame,Date)<br>gameLoser(NFLGame,NFLTeam)<br>gameWinner(NFLGame,NFLTeam)<br>teamScoringAll(NFLGame,FinalScore)<br>teamInGame(NFLTeam,NFLPlayoffGame)<br>teamFinalScore(NFLTeam,FinalScore) | Kill(Peop,Peop)<br>WorkFor(Peop,Org)<br>LocatedIn(Loc,Loc)<br>OrgBasedIn(Org,Loc)<br>WorkFor(Peop,Org) | N/A - relations not extracted for this dataset |
| Total number of sentences (testing+training) | 30 | 5925 | 11042 |
| Data source | Use case for DARPA Machine Reading Program; newswire with annotations by Linguistics Data Consortium at U. Penn. | Newswire from TREC corpus, with annotations by Cognitive Computation Group at UIUC, used in (Roth & Yih, 2004) | BioNLP'09 Shared Task (Kim et al 2009) |

Table 1: Datasets used for entity and relation extraction

works a lot of times but in several cases (to be discussed in Error Analysis section) it fails to find the appropriate subtree for given relation argument. Since the goal is to find the head word of the argument we use the fallback method of parsing the argument directly. The drawback of this method is that parsing only the sub-phrase may miss some phrasal context necessary to find the correct head word; however, this method allows us to find the head word for all arguments. Head word finding is done by using an implementation of the head finder found in Michael Collins' 1999 thesis. Below is an example of an annotated sentence after argument head word identification, used for training the conditional random fields sequence classifier. Each position in a sentence is annotated with a word, a part of speech tag and an entity type annotation.

*[ Ron(NNP ) Dixon(NNP) 's(POS) 97(CD) to(TO) yard(CD) kickoff(NN) return(NN) provided(VBD) the(DT) Giants(NNPS* ***NFLTeam****) '(POS) only(JJ) points(NNS) in(IN) Baltimore(NNP* ***NFLTeam****) 's(POS) 34(CD* ***FinalScore****) to(TO) 7(CD* ***FinalScore****) rout(NN* ***NFLGame****) .(.) ]*

After all the sentences have been annotated we proceed with testing. Due to the very small size of the Football dataset we proceed with 10-fold cross validation on that set. On BioNLP and Newswire sets we have explicit training and test datasets. We build the model using the existing implementation of a Conditional Random Field classifier in JavaNLP library. During testing we use the classifier to annotate entity types on sentences that only have part of speech annotations set. The results are presented tables 2a-2c.

## Discussion/Error Analysis

In this section we discuss results for the entity tagging problem. We also briefly consider some reasons why part of speech parsing sometimes failed to find the subtree for an argument. First looking at results we should notice that the football dataset provided is very small - there are about 10 entities tagged per file and there are a total of 12 files. For each k-fold test run we had 27 sentences to train and 3 sentences as a test set - this is usually not enough to train a classifier accurately. The most direct impact of lack of data can be seen in the *NFLPlayoffGame* class which appeared only 9 times in the dataset, too few to correctly learn a hypothesis. On the other hand we can notice very high rates for the *FinalScore* class - this is expected since this class is very easy to learn. All of the arguments are numeric and are usually preceded by either "to" or words such as "loss", "win", "victory", etc. We also introduced a gazetteer feature into the classifier for *NFLTeam* class. More precisely we have enumerated a set of possible NFL teams, and added features to the classifier based on that ontology. This increased the F1 score from 0.653 to 0.700. Finally we used a well-tuned NER classifier from the JavaNLP library to identify dates. The results on BioNLP do not suffer from the small data set problems we observed above. Although we only need to tag one entity type which simplifies the problem, the protein entity type ontology is very rich. Furthermore, it is very content dependent - for example the term "IL-6" tagged as NN part of speech can be either Protein or not Protein depending on the sentence structure. Such subtle differences were not found in Football dataset. Therefore we found the resulting F1 score of 0.924 to be encouraging. We also obtained very high accuracy on all objects in the Newswire dataset. We were surprised by the these results and found that *mutual information* in training and test sets could be the cause of these high results. For example a particular person's name can be present in both training and test sentences because both were sampled from the same domain of articles. We would like to investigate this further. Looking at all datasets another potential source of errors is incorrect identification of the argument head word due to inability to match the argument to some subtree in the parse tree. In these cases we had to parse the argument directly which was more error-prone for identifying the head word. No matching subtree is found for an argument when there is no node in the parse tree which dominates exactly the tree leaves spanned by the argument. Some reasons why this can happen are: *PP attachment error, NP parse error, tokenization, data error, possesive attachment*. PP attachment implies that a prepositional phrase was attached to the wrong node; this is similar to an obvious NP parse error but more subtle. Tokenization means that the tokenizer used by JavaNLP made a mistake in tokenizing the sentence datum. For a particular run of cross validation on the Football dataset we were able to match 239 arguments directly from the full sentence tree and 33 times by parsing arguments.

Results

Table2a: Football dataset

| | **Actual** | **Retrieved** | **Precision** | **Recall** | **F1** |
|---|---|---|---|---|---|
| **FinalScore** | 47 | 46 | 0.891 | 0.891 | 0.891 |
| **Date** | 14 | 20 | 0.555 | 0.786 | 0.647 |
| **NFLPlayoffGame** | 9 | 2 | 0.500 | 0.111 | 0.182 |
| **NFLGame** | 19 | 11 | 0.636 | 0.368 | 0.467 |
| **NFLTeam** | 59 | 41 | 0.854 | 0.593 | 0.700 |
| **Overall*** | **152** | **122** | **0.779** | **0.642** | **0.704** |

* Extremely rate *category "teamFinalScore" omitted from this table*

Table2b: Newswire dataset

| | **Actual** | **Retrieved** | **Precision** | **Recall** | **F1** |
|---|---|---|---|---|---|
| **Loc** | 4765 | 4787 | 0.986 | 0.993 | 0.990 |
| **Org** | 2499 | 2497 | 0.984 | 0.990 | 0.987 |
| **Peop** | 3918 | 3916 | 0.991 | 0.995 | 0.993 |
| **Overall*** | 14175 | 14154 | 0.985 | 0.988 | 0.987 |

* *"Miscellaneous" category omitted from this table*

Table2c: BioNLP dataset

| | **Actual** | **Retrieved** | **Precision** | **Recall** | **F1** |
|---|---|---|---|---|---|
| **Protein** | 11246 | 10747 | 0.945 | 0.903 | 0.924 |
| **Overall** | 11246 | 10747 | 0.945 | 0.903 | 0.924 |

## Relation Extraction Subsystem

We approach relation extraction as a supervised classification problem. Each training datum is a pair of entities from a sentence, labeled with the entities' relation type (which may be "no relation"). For testing, the system produces the labels for pairs of entities from test set sentences. The relation extraction system was tested on entities identified in the annotated datasets (the "gold standard") as well as on entities pipelined from the entity extraction system. Logistic regression with L2 regularization was used for classification.

### Feature generation

We initially generated 24 feature types for the entity pairs, based on the words before, between and after the entities ("surface features"), as well as the syntactic relations between the entities. The syntactic features included features based on each sentence's parse tree (the path between the entities in the tree and the path length) as well as features based on the path between the entities in the graph of syntactic dependency relations in the sentence (relations such as subject->verb, modifier->noun; these features included the relations along the path, the words along the path, relations between the entities and verbs along the path, and path length). Surface features included windows of one, two, and three words before and after the entities, the part-of-speech (POS) tags of words in these windows, the path of words between the entities, the POS tags of these words, the distance between the words in the sentence, the POS tags of the entities, and the entity words themselves. Most of these feature types consisted of binary features, while the path length / distance features were represented both as real-valued features and as binary features with a feature for each integer path length.

### Feature selection and system testing

To evaluate the utility of these features we implemented forward search feature selection, evaluating the features using 10-fold cross-validation on the Newswire training set with gold-standard annotated entities. (Time constraints precluded evaluation using pipelined entities.) Interestingly, we found that only seven features were required to achieve plateau performance. The selected features are listed in Table 3a. The classifier was then trained using these features and tested on a held-out set of Newswire sentences (one-tenth of the dataset). We also implemented forward-search feature selection excluding syntax-based features to generate a set of seven surface features, listed in Table 3b. We used the same two sets of features for the Football dataset. For that dataset we tested using 10-fold cross-validation. Relation extraction was not implemented for the BioNLP dataset (which has a more complicated set of non-binary relations).

Surface and syntactic features

| | |
|---|---|
| entity words | Giants, rout |
| entity types | NFLTeam, NFLGame |
| relations in dependency graph path between entities | poss-> comp-> <-prep_in |
| words in dependency graph path | points-provided |
| surface distance, binary features | surface_distance=8 |
| surface path between entities, part-of-speech tags | POS-JJ-NNS-IN-NNP-POS-CD-TO-CD |
| length of parse tree path between entities, binary features | path_length=5 |

Surface features only

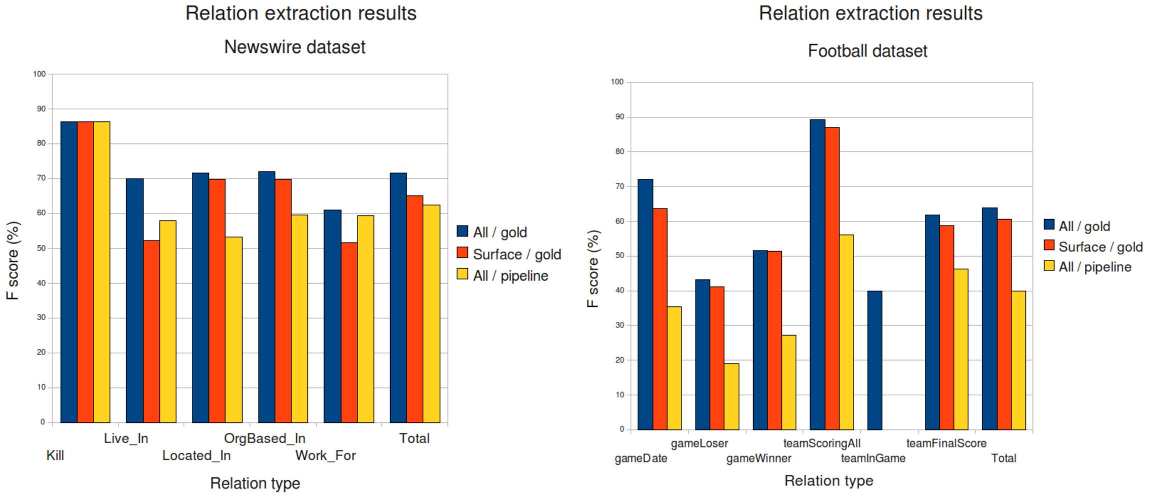
| | |
|---|---|
| entity words | Giants, rout |
| entity types | NFLTeam, NFLGame |
| surface path between entities, part-of-speech tags | POS-JJ-NNS-IN-NNP-POS-CD-TO-CD |
| surface distance, binary features | surface_distance=8 |
| surface windows part-of-speech conjunctive | VBD_DT--TO_CD, POS_JJ--NULL_NULL |
| entity order | entity1_before_entity2 |
| entity part-of-speech tags | NNPS, NN |

Tables 3a and 3b: Features used for relation extraction.
Right column shows features of each type generated for the word-pair ("Giants","rout") in the sentence "Ron Dixon's 97-yard kickoff provided the Giants' only points in Baltimore's 34 to 7 rout," from the Football dataset.

## Results and discussion

Results from testing the system on the football and newswire datasets are displayed in Figure 2. We found that using surface features alone does give worse performance than using syntanctic features, but not much worse. Using entities from the pipeline rather than the gold standard had relatively little impact on performance for the newswire dataset, where entity extraction was nearly perfect, but a large impact on performance for the football dataset, where entity extraction was much more unreliable. On the newswire dataset the combined pipeline system performed as well as the joint entity and relation extraction system of (Roth & Yih, 2004).

One possible area for future improvement in our system is feature sparsity. For example, for 59% of the relations annotated in the Newswire test set, the dependency graph path between the relation's arguments (i.e. the third selected feature in table 2a) appeared no more than once in the training set. The F1 score for relations whose dependency-graph-path feature was sparse in this sense was 0.53, compared to 0.71 for relations with non-sparse dependency-graph-path features. Sparsity was also observed for some potentially useful features which were eliminated in feature selection (e.g. parse tree path). Future work could address feature sparsity by (1) using semi-supervised approaches to increase the quantity of data, or (2) learning higher-level representations of features which could generalize to features not seen before by the system, along the lines of (Bengio et al., 2003).



Relation extraction results

Newswire dataset

Relation extraction results

Football dataset

Figure 2: Performance of relation extraction system.
Different colors show feature set / entity annotation.

## Acknowledgements

We were advised on this project by Chris Manning and Mihai Surdeanu.

## References

Yoshua Bengio, Rejean Ducharme, Pacal Vincent, and Christian Jauvin, "A Neural Probabilistic Language Model." Journal of Machine Learning Research, 3:1137-1155, 2003.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun’ichi Tsujii, "Overview of BioNLP’09 shared task on event extraction." In Proceedings of the NAACL-HLT 2009 Workshop on Natural Language Processing in Biomedicine (BioNLP’09), 2009.

D. Roth and W. Yih, "A Linear Programming Formulation for Global Inference in Natural Language Tasks" CoNLL'04, May 2004.