Automatic Generation of Fine-Grained Named Entity Classifications

Aspasia Beneti Woiyl Hammoumi Eric Hielscher
Martin Müller David Persons

March 22, 2006

Contents

Abstract

The purpose of this work is to explore possible approaches to generating fine-grained classifications of named entities. Three such approaches are discussed: the use of the categories in the online encyclopedia Wikipedia; the use of a predefined set of classification rules to apply to words in a text; and the use of a machine learner to identify fine-grained location classifications. Results are promising with all three systems performing well.

Such fine-grained classifications are extremely useful in QA systems since they can be used to mark up source texts such that the QA system can more easily scan for candidate answers to questions.

1 Introduction

We explore three approaches to automatically generating fine-grained classifications of named entities (NEs): the use of the categories in the online encyclopedia Wikipedia; the use of a set of rules to classify words according to their context and structure; and the use of a machine learner to identify fine-grained location classifications based on context and other NE features.

1.1 Motivation

In QA system competitions, questions appear that resemble the following [1]:

- The submarine Kursk was a part of which Russian fleet?
- What country did the winner of the Miss Universe 2000 competition represent?
- What was the nickname for the 1998 French soccer team?

In this case the system is looking for candidate answers to the questions that fall into three specific classes: fleets, countries, and nicknames. It would thus be very useful for the system to have a component that could mark up NEs in source texts with such fine-grained classifications. Then it could simply scan the sources for NEs marked with the sought classification and use context to decide on the appropriate NE to return as an answer. Previous work has been done on this problem with respect to classifications of persons [3] while we aim to extend this to include subcategories of both organizations and locations.

1.2 Problem Description

As mentioned above, our work consists of three separate approaches to fine-grained classification: an encyclopedia, a rules, and a machine learning approach. In the Wikipedia approach the categories into which all articles are grouped are extracted and then are ranked according to certain features that allow assessment of their presumed usefulness. In the rules approach certain rules that capture the structure of NEs and their contexts are used to automatically classify NEs. In the machine learning approach features of the NE's context and the NE itself are used to train a machine learner to classify locations into a small subset of finer categories.

1.3 Use of LingPipe for Initial Processing of Text

All three of our approaches pre-processed the input texts with an open-source Named Entity Recognizer (NER) called LingPipe. LingPipe scans the texts for NEs and tags them with three broad classificiations: *Person*, *Organization*, and *Location*. Any errors generated at this stage were simply ignored. It was then our task to further refine these classifications.

1.4 Goals

The goal of this project is to mark up NEs in texts with as many relevant categories as possible while minimizing uninteresting categories or ones that aren't useful in a QA setting.

1.5 Previous Work

Much work has been done previously in this field with both rules-based systems [4],[5] as well as with machine learners [3],[9]. Wikipedia, however, is only starting to be recognized as the rich information source that it is. Ahn, et al [1] have used it in their QA system for TREC 2004 as an importance model for answering open-ended ”other” questions.

2 Generating Classifications using Wikipedia

Wikipedia is rapidly maturing into an invaluable free information resource. As of this writing the English version alone has over 900,000 articles on a wide range of subjects. One part of the process of creating a Wikipedia article involves placing it into one or more categories (e.g. the article on Germany is placed into the following four categories: *Europe*, *European Union Member States*, *Germany*, and *Republics*). We explore the use of this classification hierarchy as an information source for the classification of named entities in a text.

2.1 Method

Our system uses as input the output generated by LingPipe (see section 1.3), a system that automatically tags NEs into three broad classifications: *People*, *Locations*, and *Organizations*. In our approach we scan the file generated by LingPipe and for each tagged NE we look up the corresponding Wikipedia entry via the internet. Only those NEs that LingPipe recognizes are further classified and no attempt is made to correct classification errors made by LingPipe.

2.1.1 Extracting Wikipedia Categories

The next step in the process is to parse the returned page and to extract the categories listed at the bottom of the article. While it would be possible to simply return all the categories as the final result, many of them end up not being useful. For instance, in the above example list of the categories to which the article on Germany belongs we find the subject of the article itself. This is completely uninteresting and adds to cluttering of the data. In addition to this specific problem, there are many other categories which are meant to be Wikipedia-internal classifications and are not meaningful outside of that context (as an example consider the category *Cleanup from December 2005* which indicates that an article was tagged as needing cleanup during the month of December, 2005). Filtering out categories that are most likely not going to be useful can help to eliminate confusion downstream in the information pipeline of a QA system and increase the speed with which the system would be able to answer questions.

2.1.2 Addition of Separate Nationality Categories

Often Wikipedia would return complex categories such as *Presidents of the United States*, etc. While this is interesting in its own right, it can be very useful in cases like this to have as a separate category the person's nationality. To accomplish thus we first built a list of most nationality adjectives. Then, whenever a category was returned that contained a hyphen-separated sequence of these adjectives (such as *Irish-American*), we would add that sequence and all nationality words contained within it as separate categories. Thus for this example *Irish*, *American*, and *Irish-American* would all be added to the NE's category list.

2.2 Ranking System

We thus have devised a ranking system for the categories extracted from a Wikipedia article on a given subject. A category receives or loses points based on whether it meets certain criteria that indicate its likelihood of being a useful or useless category, respectively. The nationality categories added by our system are scored in the same fashion as the ones returned by Wikipedia. The point values assigned to these features were created by hand and thus are somewhat ad-hoc but the performance of the system justified their use.

2.2.1 Filters Used to Remove Categories

The following are the filters our system used to remove categories likely to be uninteresting:

- Categories with the same name as the NE in question are given large negative scores to ensure their placement toward the bottom of the list in the final result.
- If the article belongs to the category *Disambiguation,* then the article doesn't have actual content but rather is meant as a placeholder with links to articles that match the search string. For now we naïvely assign all categories on such a page a large negative score to signify that they can't be reliably trusted.
- Large negative scores to Wikipedia's *Cleanup* or *Stubs* categories. These are categories used to indicate that articles require work or are in a very incomplete state. In order to do this we built lists of these categories.
- A small negative value is added to the score of any category which contains the word Wikipedia as it is likely that this is also an internal usage category.
- Wikipedia assigns to each person categories of all the people who were born and died in the same year as that person (e.g. *1946 births*). We give these very specific categories large negative scores.
- In addition, we add a small negative value to all categories that contain a date as these are often far too specific to be interesting.

2.2.2 Other Scoring Mechanisms

In addition to the above-mentioned filters, we also increased the score of a category if it had four or fewer words and decreased it otherwise. This reflects the fact that many categories with long names are too specific to be interesting.

2.2.3 Scoring Example

The following is an example of the final category list and the corresponding scores produced by our system for the NE "Javier Solana". The category *Spanish* was added through our nationality extraction and the category *1942 births* was filtered to the bottom due to its specificity.

Category	Score
NATO Secretaries General	0.20
European Union security	0.20
Spanish politicians	0.15
Spanish	0.10
1942 births	-10.05

2.3 Results

Wikipedia's category hierarchy proved to be a very rich information source and the results of the system were in general quite good. In order to benchmark performance we added two configurable parameters to control the final category output. First, we added a parameter that controlled the maximum number of categories to return for any given NE. An optimal setting of this would allow most of the relevant categories to be kept while culling the less interesting ones. Second, we added a parameter that sets the minimum score a category can receive in order to be kept in the final list.

2.3.1 Benchmarks on Test Text

In order to test the three systems we developed a test text and a list of categories that we would expect any perfectly functioning system to be able to provide for all the NEs occurring in this text.[1] The following illustrates our performance with two different settings of the above-mentioned parameters:

- Results with maximum categories set to 5 and minimum score set to 0.01
  - Recall: 54%
  - Precision: 57%
- Results with no maximum number of categories and no minimum score
  - Recall: 58%
  - Precision: 43%

These numbers were calculated by ignoring NEs in the test text which were not generated by LingPipe since the Wikipedia system would never be able to recognize them.

It is interesting to note that the precision score is significantly better with the more strict paramater settings while the recall score doesn't suffer very much. This leads us to conclude that our settings in this case are reasonable. In addition, we believe that the low precision scores may be misleading in respect to the performance of our system. Often the extra categories that were returned that weren't in the gold standard lists for the test text were in fact interesting and relevant – we'd simply not thought of them when creating the gold standard. Another possible metric would be to run the system on the text and of all the categories returned decide by hand afterward which should contribute negatively to the precision score.

2.3.2 Other Results

We also performed tests of the system on a suite of 10 other test files which totalled 247 NEs. Of these, 220 were assigned at least one category, for a score of 89%. Some of the failure can be explained by noting errors in the LingPipe tagging step while others illustrate holes in the Wikipedia knowledge base.

Within the 220 NEs that were assigned categories, 20 of these (10%) were assigned a *Disambiguation* category and thus had all their categories fully discounted. This shows that while we are getting a high percentage of relevant information, we would benefit greatly from work to provide an intelligent decision procedure for following links given on these disambiguation pages.

[1]See Appendix A for the test text and its gold standard. See Appendix B for more detailed results of the Wikipedia system.

2.4 Conclusion

We are quite pleased with how our system performed and in addition believe this work to show conclusively that Wikipedia is a valuable information source for this and other information retrieval tasks. There is much more that could be done but it is our hope that this work can provide some ideas for future research.

2.5 Future Work

There are many open-ended avenues available for future research in this area. First, a much more thorough analysis of the categories present in Wikipedia could be done so that a more comprehensive set of filters could be developed in order to remove uninteresting categories.

Also, as mentioned, 10% of all NEs were assigned a *Disambiguation* category, rendering their results useless. It would be very interesting to develop a procedure for using the context surrounding a given NE and that category assigned to it by LingPipe in order to automatically choose one of the candidate articles listed on the disambiguation page and extract its categories.

Further, more lists of common categories like the nationalities list could be used to automatically extract frequently occurring useful classifications. Possibilities include things like occupations, countries, and religions.

Finally, a machine learner could be used to more accurately set the scoring weights for all the features of categories used to produce scores. It would be interesting to see how more accurate rankings of the returned categories would look.

3 Generating Classifications using Predefined Rulesets

3.1 Introduction

The rule module uses hand-coded rules that assign types to NEs based on their internal format (e.g. GeorgeTOWN is a city) or their context (e.g. in RHODESIAN PRESIDENT Smith, Smith is the Rhodesian president).

Types of Rules There are two main rule based approaches to extract information on the type a NE belongs to: we can either focus on it's *internal structure*. So, for instance, a recognized location like *Capetown* could be further classified as a city since the compound contains *town* as part. Or an e-mail address could be identified by its specific structure containing an @-sign and .domain at the end. The other possibility is to investigate the *context* of NEs. By doing this, we could extract information like the title of a person (e.g. President Bush) or the type of a location (e.g. Sherwood Forest). More elaborated methods allow extraction of more than one category as in *German Minister of Foreign Affairs Frank-Walter Steinmeier*, where nationality and title could be assigned to a person.

In both, the internal and the context approach, morphologic, orthographic and syntactic features can be used as indicators for certain categories or for disambiguation (cf. [5]). Words ending with *-land*, for instance, are most probably countries; capitalized NEs are organizations in most of the cases; constructions like *President of the U.S. G. W. Bush* containing certain prepositions, or appositions often separated by commas (e.g. Steinmeier, Minister of Foreign Affairs, ...) can be syntactically exploited for classification of NEs. In addition, lists can be helpful for classifying items further. In our module, for example, we use lists of countries for a subclassification of NEs tagged as locations.

Ambiguities A major task in classification of NEs is the resolution of ambiguities. We can distinguish *semantic* and *structural* ambiguities. In order to decide, whether the type *Washington* refers to the person George Washington or to the city, we can check the preceding context for names or initials – which suggest the type PERSON –, or the succeeding context for a state abbreviation, *DC*, in order to assign it the category CITY. A type like *May* could refer to a person or to the month. If there are no trigger words in the immediate context, as a preceding title for the class PERSON or the prepositions *in* or *at* or succeeding digits for DATE, a deeper syntactical analysis of the context is necessary, although a statistic-based assignment might be a more appropriate method in this case. Examples for structural ambiguities are *Cable and Wireless* vs. *Microsoft* or *Dell* or *Center for Computational Linguistics* vs. message from *City Hospital* for *John Smith* ([5]). In theses cases multi word NEs are involved and it has to be decided, how many types they consist of.

Why Using Rules? Rules are an important constituent of many NE recognition and classification systems (e.g. MUSE,[6]; DRAMNERI [8]). On the one hand, by the use of regular expressions for pattern matching they provide an easy method for extracting information based on the internal structure of NEs. Therefore NER systems use them in order to discover e-mails addresses, dates etc., but also orthographic characteristics like case, abbreviations etc.

On the other hand, some tasks are just difficult to be solved with other approaches: there are constantly new NEs built especially company names but also e-mail addresses or urls, which cannot appear in any dictionary or list. (cf. [4]). These could be identified, though, by their internal structure and by a specific context they appear in. The same holds for entities like money amounts or dates: it does not make sense or is even impossible to create adequate lists, but with the use of regular expressions or recursive rules these entities become tractable.

3.2 Method

3.2.1 Developing Rules

The development of our rules was based on considerations mentioned above and on ideas we got by browsing some literature ([6], [7]) and lecture slides on NER and NEC ([4], [5]). The range reached from relatively simple structure based regular expressions over rules using gazetteers up to rules sensible for bigger context and syntactic constructions including punctuation. The order of development was determined by the complexity of the string to match, but also by the fact, that in order to refer to NEs tagged by LingPipe and to the assigned category we depended on a functioning parser which first had to be developed. Most rules were developed and tested separately (and in a different programming language) and were intended to be converted and implemented in the system later on.

Regular Expressions We started writing regular expressions for entities not tagged by LingPipe as e-mails, urls and dates. Whereas the former have a rigidly constrained structure, there exists a variety of different possible date formats: the order of its components, day, month, and year is flexible, year and month often appear on its own, and the months can be given as proper names or as digits. So, instead of writing one complex expression to cover all possible formats, we developed several more simple ones. We also had to include a list of months in order to match single appearances of months, i.e. in case no disambiguating context as preceding digits (for days) or succeeding digits (for years) is given.

Syntactic and Context Based Approach using short lists In a second step, we experimented with more small lists trying to match class-identifying parts of compounds such as *-town*, *-land*, *-street* and – which is more relevant since compounds are rare in English – started matching components of multi word NEs with lists containing, for instance, location types like *City, Street, Avenue, Museum, Wood* etc. we could assign as type to the NE as a whole.

We also started exploring the context of NEs. The first approach was a combination of using lists and ”guessing” relevant types in the context of a classified NE by orthographical characteristics of words (as starting with upper case) and characteristics of the position they had to the NE. Since the parser was not finished at that state, we assumed that the original structure of a phrase would be preserved by the parsing process or would be completely reconstructible such that the rule could use information like word order and punctuation. So, for instance, if we had a NE of the class PERSON we applied to the preceding context a regular expression containing a list with nationalities and an expression to match words starting with capitals and optionally separated by the preposition *of*. By this we matched often appearing phrases like *British Minister of Defense M. Cook* and could assign a nationality as well as an occupation to the name.

Similarly, by looking to the succeeding context, we could match the variation *M. Cook, the British Minister of Defense,* where the commas are important indicators for the description being related to the preceding name, not to a following. Due to constraints of our parser, though, which did not preserve all relevant information, and because of time constraints, these more syntactic based rules could not be integrated. We had to come up with a different approach which fit better to the parser architecture and allowed a more immediate implementation.

Gazetteers We created lengthy lists (mostly retrieved from different web resources and preprocessed) for different kinds of categories: lists for occupations, countries, different kinds of locations, titles etc. Some of them like the country or location list could immediately be applied to the content of tagged NEs. Others like the occupation list were used to examine the closer context of a NE. Rough heuristics determined the wide of the context and the order of list applications. A more detailed description of the gazetteers based rules is given in the following section.

3.2.2 Implementing the System

After the input and output part of our system was done, it was time to get to business; implementing the rules we came up with.

LingPipe doesn't recognize some types with easy to recognize structures. E-mail addresses, urls and time/date for example all got very clear structures. Regular expressions can be used to easily classify this types. An e-mail address for example will contain some characters followed by a @, some characters again followed by a point and then the country code. This gives us the following regular expression:
c+@
c+
.
c{2,3} The current version of the system uses 10 regular expressions. Most of them are used for recognizing time and date patterns. In the future more lists can be added to higher our recall.

The other rules we implemented are based on the classification done by LingPipe. We'll try to further categorizes LingPipes Persons/Locations and Companies. A good way to do this is searching the context of these objects using lists. When LingPipe classifies some words as a person for example, there is a high change to find an occupation or a nationality in the context. Our systems now uses the following lists: occupations countries nationalities In the future more lists can be added to higher our recall.

Persons When a word sequence is classified as a person by LingPipe, we look at the words close to that person. We first search the occupation list, when a word in the context is found in this list, our system adds this occupation to the results. This method works for sentences like the following: French Minister Philippe Douste-Blazy

In our first version we looked at the words before and after the person. This worked properly most of the time but also gave some wrong results. An example is the following sentence: Foreign Minister Jack Straw and his French counterpart Philippe Douste-Blazy

What our first version of the system recognized was that Jack Straw was a minister but thought Jack Straw had a French nationality. So, only looking at the words before and don't pay attention to the words after would have been better in this sentence. What the final version of the system does is only looking at the words after in special cases. One of this cases is given below: Philippe Douste-Blazy, French Minister, will . . .

When a couple of words surrounded by commas are behind a person, this words will be extra information about that person so the lists will be searched for these words.

Locations LingPipe recognizes country names, nationalities and well known towns. So this is another good purpose for our country/nationality lists. When testing our first implementation we again found some things we could improve. Our first system tagged locations as country/nationality if found in the list and town when not found. This structure of locations can be stretched even further. On the internet we found a list containing countries and their different provinces, of all of these different provinces their major towns were mentioned. It was out of our scope to implement this structure but it will be a nice thing to implement in the future.

Companies Companies are handled in the same way as persons. The only difference is that the occupation list is not searched because it makes no sense for companies.

3.3 Results

We tested our system on a couple of test files. After we felt our system did a good job, we calculated its recall and precision on the evaluation file. The precision of our system was 72%. The most of the mistakes it made was because some of the states of the USA were mentioned in the file and our system recognized them all as towns.

The recall of our system was 20%. The reason the recall is not that high is because the system does not that well on some of the things LingPipe tags as location. In our evaluation text, the *Panathenaic Stadium* is mentioned for example. In the context of this location there is no information which helps our system to label the word. So our system doesn't label it and at the same time there were 3 categories which could have been applied to the word. If we calculate the recall only for words which we made rules and expressions for then the recall will be about 80%.

3.4 Conclusion

There were difficulties in coordinating the parallel development of rules sets and parser at the beginning. This was partly due to the use of different programming languages and differing prepositions the developments were based on. Therefore some of the earlier developed context-based rules are not implemented and so some examples mentioned in section 3.1 and 3.2.1 are not covered by the current system yet.

Rules written for recognition of NEs not tagged by LingPipe work fine in our system and show the strength of this approach in finding entities with characteristic structural properties.

Our later approach in classifying LingPipe NEs further which was more adequate to the parser architecture by using gazetteers and not relying so much on syntactic information yielded good results (see 3.3). This approach makes the system more simple and transparent. It can now easily be tuned by expanding given lists or adding new list which can be done by non-programmers and non-linguists. So the performance can be improved or the system can be adapted to certain domains. This could make the system attractive for users not experienced in NL processing.

3.4.1 Future Work

More expression and more lists can be added to the system, this will higher the recall. To higher the precision it will be a good idea to use lists with more hierarchical information. For locations for example, a list can start with the earth. The earth is subdivided in continents and they are divided in countries. The list ends with towns. Using and searching in a list like this will provide more accurate tags for locations.

4 Generating Location Classifications using a Machine Learner

4.1 Introduction

The learning module aims in developing a system that automatically enriches plain text with entity class information using a learning algorithm. Learning could be described as a way to make a machine improve its performance through experience; enable a machine to classify new instances from a number of already given training samples. For this purpose supervised learning methods are used and in particular the TiMBL software, which is a program implementing several Memory-Based Learning techniques with k-nearest neighbor classification kernel set as default.

4.2 Previous Work

It seems that not a lot of research has been done so far concerning fine-grained entity classification using learning methods. The most relevant works will be mentioned and shortly described in this section.

Fleischman et al. (2002) [3] having the same goal, tried few machine learning algorithms that also consider the context around the entity. They also included semantic information derived from topic signatures and WordNet as well as they took into acount the presence of entities in multiple contexts. The main difference lays on the fact that they based their classification on features that have to do with the frequency of words appearing together in text. On the other hand, our approach considers each document independently and it does not take into account freaquencies at all.

Li et al. (2002) [9] used supervised learning methods to perform fine-grained classification of questions. In their work they combine a coarse and a fine classifier to discover what kind of entity should be expected as an answer to a question. As the goal in this case is slightly different, different features were selected to be used for classification including relational features.

4.3 Method

The following sub-sections contain some background information together with a description of the most important methods used for the implementation of this module.

4.3.1 About Supervised learning

In a supervised learning model the data are split into two groups: the training and the test data. The training data are usually tuples of the form data features-class. On the other hand the test data consist of the data features alone without any given classification. So the system ’learns’ through already given classified examples and creates some kind of description for each of the classes, often the class distribution. Then it compares every test instance to each of the classes and decides in which class it should be classified. Supervised algorithms often perform really well but this highly depends on the number and the nature of the training samples.

4.3.2 The TiMBL learner

TiMBL [2] is a program implementing several memory-based learning algorithms designed with linguistic classification tasks in mind. Despite their differences, all of these algorithms have in common that they store some representation of the training set explicitly in memory. During testing, new cases are classified by extrapolation from the most similar stored cases.

TiMBL accepts the data in several pre-specified formats. They all share one common characteristic: the data themselves are not included in the input files, but they are described by a list of symbolic and/or numeric features and a number of discrete classes as defined by the user. The reason for this is that this way the learner does not get stuck to specific examples but uses their characteristics in order to generalize the model and to be able to classify unseen instances.

The TiMBL learner interacts with the user through command line arguments. This way the user can specify which are the data files, the algorithm to be used, the desired metrics and the feature weighting. If one does not specify one of the above options they will be automatically set to their default value.

4.3.3 Selected Features

One very important task is to specify the features to describe the entities in a free text. A lot of these kind of features exist but not all of them would be meaningful for the particular goal. In this case, we deal with the problem of assigning meaningful tags to words that appear in free text. Taking this into account and after several experiments, the features are decided to be the following:

1. Are in the word entity any of the class words and if yes which?
2. Which is the ending of the word entity (the last three characters of the final word)?
3. Is the word entity an abbreviation? (+ for yes and - for no)
4. Which is the previous word?
5. Which is the word before the previous?
6. Which is the next word?
7. Do any of the class words appear in the previous or next four (4) words and if yes which?

4.3.4 Example

The following example will demonstrate a line of an input file in column format. Each line of the data file refers to a single instance, followed by its class. The class can be replaced by any symbol in the case of test data when it is not known yet. However this will not allow TiMBL to calculate the systems performance.

Text: The UK, France and Germany are holding urgent talks with Iran on the stand-off over its nuclear program.

Classes: country, city, continent

Word Entity: France

Features: ??? nce – UK The and ??? COUNTRY

Note: In the case that no value can be assigned to a feature (e.g. there is no previous word in the sentence, because the word we are looking at is at the beginning of the sentence) it is filled up with the characters???.

4.3.5 Feature evaluation

Based on the measures provided by the TiMBL learner and in particular the information gain ratio/values, the system's features are ranked as follows:

1. Feature 1 – gain ratio: 0.335/2
2. Feature 3 – gain ratio: 0.220/2
3. Feature 7 – gain ratio: 0.163/6
4. Feature 2 – gain ratio: 0.265/146
5. Feature 4 – gain ratio: 0.140/95
6. Feature 6 – gain ratio: 0.154/146
7. Feature 5 – gain ratio: 0.195/213

4.3.6 Classes

As in all three modules, the input is in XML format, the direct output of the LingPipe software. The categories assigned to word entities by the LingPipe are LOCATION, PERSON and ORGANISATION. With this further classification the aim is to assign to the already tagged words more specific subcategories. In the specific case of the learning module and because of the fact that the bigger the number of the classes are the more training data are required, we had to focus in one category. The category selected is the location and a candidate division to subcategories grouped by map types is shown on the following table:

Political	Topographic	City
Continent	Lake	Street
Country	River	Park
State	Ocean	Museum
Region	Mountain	Garden
City	Desert	University
Capital	Forest	Airport
	Island	Port
	Sea	
	Volcano	
	Canyon	
	Gulf	
	Basin	

However, because of the restricted amount of time, further limitations took place. From all these subcategories only the following five (5) are actually consider as classes in the current software. However, we strongly believe that given more time to collect the required amount of data, there would not be any restrictions on the number of classes.

- Continent
- Country
- City
- Capital
- General (for all the LOCATION word entities that cannot be classified as any of the above classes)

4.3.7 Collecting the data

All the data are collected from the Internet and particularly from news, history and geography websites. This is because this specific module focuses on the LOCATION category and the text taken from this kind of web sites seems to contain useful for the domain data.

Initially, the collected data were transformed using the LingPipe Interactive Demo into the input XML file and then manually converted to a TiMBL format. Also the training instances were assigned to classes manually. Later, once the implementation of the automatic conversion was ready, the data in XML form were automatically converted to a TiMBL format and classified using the learner. The given from the learner classification is then manually checked and corrected so the data are ready to used for training. This way the gathering of training data becomes simpler and faster.

4.3.8 Implementation

In this section one can find the most important points of the system's implementation. The overall system can be broken down into its three main parts:

- Conversion from the LingPipe XML output to the column format required by the TiMBL learner
- Execute the learner and collect results
- Use the results to create the output XML file

Implementing such a program in Java is relatively straightforward. The decision that had to be taken is what data structure is more suitable to store the XML document itself so it makes it simple to extract the features. The text was stored in the form of sentences and words, with pointers pointing at the places where the words that we are interested in tagging are. This made really simple the feature extraction, as most of the features involve with the entity's context.

The system automatically produces the following three files:

- data.test : where the features describing the entities are placed in the required format

- data.test.IB1.O.gr.k1.out : the file returned from the TiMBL learner, which is the same as the data.test but with an additional column for the predicted classes
- output.xml : the file where the output is written in XML format

Finally, the software if fully functionable and it is really time efficient. Also it is quite reusable and extensible. It can be downloaded at the Language Technology Project 2006 website.

4.3.9 Changes to the original design

This section points out the most important changes that took place from the initial design to the completion of the implementation.

Initially the features were nine in number; the existing ones plus two more features checking whether the entity is in the beginning or at the end of the sentence. Later, through experiments and using some of the TiMBL learner measures, it was obvious that those two features did not contribute much to the learning task. In particular, their gain info ratio was relatively low and in the same time they seemed to overlap with the other features. These are the main reasons why those two features were removed from the final set.

Moreover, two additional classes were supposed to be inlcuded; Region and State. However, soon it was clear that such an amount of classes would require even more training data. Due to restricted amount of time this was impossible so it was decided that two classes had to be eliminated. The decision to eliminate the specific two classes did not seem to be the best later. This is because the remaining classes, city and capital, overlap and it is difficult to distinguish with result to reduce the system performance quite a bit. However, there was not time for such a radical change as most of the training data had already been collected and manually classified. Given more time, those two extra classes could have been added and the required number of training data could have been collected. This should have led to much clearer classification as now a lot of the entities have to be classified to the general class.

Finally, the first training examples were created manually. At this point, it was not clear whether the punctuation marks of the text should be considered as entities or not. Later, I decided to include punctuation as it can play its role in the sentence context. This decision also led to alter the TiMBL data format from C4.5 which uses ',' as a separator to the column format which uses single spaces as separators, so it would not cause confusion if read by a human.

4.4 Results

In this section the results of the system given a gold text will be presented (see appendix A).

Only the LOCATION entities are mentioned here because of the learner's restricted domain. The learner produced the following results:

Iran: country +, city - (it should be country)

Athens: country - (it should be capital)

Panathenaic Stadium: country - (it should be stadium)

Chicago: country - (it should be city)

Illinois: capital - (it should be state)

Pittsburgh: country - (it should be city)

Pennsylvania: country - (it should be state)

Monessen: capital - (it should be city)

According to the above results and without taking into account the current limitations of the system the performance measures could be calculated as follows:

1 correct, 9 classified: recall = 1/9 = 11%
1 correct, 79 entities in gold data: precision = 1/79 = 1%

However, it does not seem fair to ask the learner to predict classes that it has no training data for. So this would exclude anything that should be tagged as a state or a stadium. From the tagged entities, only 7 are from a known class, so this would increase the recall to 1/7=14%.

Also it should be mentioned that one other limitation of the learner in the current system is that it can predict only one class per entity. This could be changed at least for the class 'capitcal' to 'city'+'capital'. The predictions would remain the same but the performance would be calculated as follows (taking into account that only the location entities were expected to be tagged):

precision = 3/12 = 25%
recall = 3/10 = 30%

This could be characterised as a reasonable performance given the small amount of 333 training samples. We believe that for better results at least few thousands of training data would be needed but this was impossible given the restricted amount of time. More details about the results will be discussed in the following section.

4.5 Discussion

In this section the overall system's performance will be discussed together with its advantages and disadvantages.

As mentioned in the previous section, the overall system's performance was tested given a specific test text. One can notice that a lot of misclassifications took place. There are two main reasons for this fact. First of all, the learner's performance depends a lot on the amount of training data. Not much could be expected from a machine learning algorithm that only has a few hundreds of training data. The main disadvantage of the learning approach for such a system is that it requires a big amount of training samples in order to achieve better accuracy. In other words, such an approach requires a lot of human effort, as all the training samples have to be classified manually by a human expert. Another solution would be to let the system classify new training data, but still a human expert is required to check the classification and correct possible mistakes.

Also, the performance of the system highly depends on the nature of the training data. When the system was tested given some other text (see Appendix D) it performed much better in some cases. This has to do with the source of the training data and the nature of the individual examples. For the learner to be able to guarantee correct classification for an entity, we have to ensure that it has seen before examples with similar features. This means that in order to boost the performance of such a classifier, the data should be collected carefully from all possible domains.

Another problem of the learner in its current state is the limited number of classes. As mentioned in previous section, the class selection was not the most suitable. There are only four actual classes, from which the two (capital and city) overlap. A possible solution for this problem could be to allow the learner to predict two classes per entity.

Generally speaking, we strongly believe that the system could perform really well given more time to collect the required amount of data and expand the classes in a way that overlaps would be avoided.

4.6 Future work

The current system could be extended in several ways. First of all it should include more classes from all three main domains (location, organization and person). This would of course require a sufficient amount of training data. It would probably be a good idea to use one of the other two modules, maybe the rule module, to classify data. This way we could retrieve cheap training data. Of course human effort would also be required to inspect and correct possible errors.

Also I believe that once the system is complete, further research should take place with main purpose to discover weather the current set of features is optimal for this task or it should be revised. In order to discover which features contribute more to the learning task, one of the measures implemented by TiMBL could be used. Finally, to improve accuracy features weighting could be applied. This would not alter the classification results so much, but it would use more intelligent criteria to resolve possible ties.

4.7 Conclusion

The outcome of this project is a fully functional system that performs fine-grained entity classification using a learning method. Despite the system's limitations, it gives a clear picture of its functionality. With further improvement and expansion it should be a stable and accurate method to be used for the specific problem domain with only disadvantage its need for a large amount of training data.

References

[1] David Ahn, Valentin Jijkoun, Gilad Mishne, Karin Müller, Maarten de Rijke, and Stefan Schlobach. Using wikipedia at the trec qa track, 2004.

[2] Van der Sloot K. Daelemans W., Zavrel J. and Van den Bosch A. Timbl: Tilburg memory based learner, version 5.1, reference guide., 2004.

[3] Michael Fleischman and Eduard Hovy. Fine grained classification of named entities, 2002.

[4] Stephan Lesch. Maschinelle lernverfahren fr informationsextraktion und text mining: Named entity recognition. Slides, 2000.

[5] Diana Maynard. Named entity recognition. Slides, 2001.

[6] Diana Maynard, Valentin Tablan, Cristian Ursu, Hamish Cunningham, and Yorick Wilks. Named entity recognition from diverse text types. Technical report, Dept. of Computer Science University of Sheffield, 2001.

[7] Georgios Petasis, Frantz Vichot, Francis Wolinski, Georgios Paliouras, Vangelis Karkaletsis, and Constantine D. Spyropoulos. Using machine learning to maintain rule-based named-entity recognition and classification systems, 2001.

[8] Antonio Toral. DRAMNERI: a free knowledge based tool to Named Entity Recognition. In *Proceedings of the 1st Free Software Technologies Conference*, pages 27–32, 2005.

[9] Li X. and Roth D. Learning question classifiers, 2002.

5 Appendix A - Example Test Text and Associated LingPipe Output

The following is the test text that we used to have standard test for each different system:

Tuesday, 31 January 2006, 21:03 GMT.
UN confirms Iran enrichment move.
Iran has started preparations to resume uranium enrichment, says a report by the UN nuclear watchdog.
German takes over top Bosnia job.
Christian Schwarz-Schilling, a German diplomat, becomes the powerful top envoy to Bosnia.
Bernanke approved as new Fed head.
Ben Bernanke is approved as the new head of the Federal Reserve, taking over from the outgoing Alan Greenspan.
Stark warning over climate change.
Greenhouse gas emissions are rising at an ”unsustainable” rate and the Greenland ice cap is at risk, scientists say.
Actor Baker becomes voice of text.
Former Doctor Who Tom Baker is lending his voice to a service which transfers text messages to landlines.
October 6, Athens.
Explore the Acropolis temples, including the masterpiece of Periclean Athens, the Parthenon. Stop for lunch in the picturesque neighborhood of the plaka. Visit the Panathenaic Stadium where the first Olympics of the modern era were held in 1896, and continue to the National Archaeological Museum, reopened after extensive renovation.
Frances McDormand.
McDormand was born in Chicago, Illinois, the youngest of three children adopted by Canadian parents Vernon McDormand (a Disciples of Christ minister) and Noreen. She spent much of her youth in the Pittsburgh, Pennsylvania, suburb of Monessen, where she graduated from high school. She attended Bethany College, Bethany, West Virginia, and earned a B.A. in Theater in 1979.
Bill, who owns www.microsoft.com, can be reached at billg@microsoft.com.

The following is the output of LingPipe on the above input file:

```
<?xml version=”1.0” encoding=”UTF-8”?>
<DOCUMENT>
<P>
<sent>Tuesday, 31 January 2006, 21:03 GMT.</sent>
<sent>
<ENAMEX id=”0” type=”ORGANIZATION”>UN</ENAMEX> confirms
<ENAMEX id=”1” type=”LOCATION”>Iran</ENAMEX> enrichment move.</sent>
<sent>
<ENAMEX id=”1” type=”LOCATION”>Iran</ENAMEX> has started preparations to resume uranium enrichment, says
a report by the
<ENAMEX id=”0” type=”ORGANIZATION”>UN</ENAMEX> nuclear watchdog.</sent>
<sent>German takes over top
<ENAMEX id=”2” type=”LOCATION”>Bosnia</ENAMEX> job.</sent>
<sent>
<ENAMEX id=”3” type=”PERSON”>Christian Schwarz-Schilling</ENAMEX>, a German diplomat, becomes the pow-
```

erful top envoy to
<ENAMEX id=”2” type=”LOCATION”>Bosnia</ENAMEX>.</sent>
<sent>Bernanke approved as new
<ENAMEX id=”4” type=”ORGANIZATION”>Fed</ENAMEX> head.</sent>
<sent>
<ENAMEX id=”5” type=”PERSON”>Ben Bernanke</ENAMEX> is approved as the new head of the
<ENAMEX id=”6” type=”ORGANIZATION”>Federal Reserve</ENAMEX>, taking over from the outgoing
<ENAMEX id=”7” type=”PERSON”>Alan Greenspan</ENAMEX>.</sent>
<sent>Stark warning over climate change.</sent>
<sent>Greenhouse gas emissions are rising at an “unsustainable” rate and the
<ENAMEX id=”8” type=”LOCATION”>Greenland</ENAMEX> ice cap is at risk, scientists say.</sent>
<sent>
<ENAMEX id=”9” type=”PERSON”>Actor Baker</ENAMEX> becomes voice of text.</sent>
<sent>Former Doctor Who
<ENAMEX id=”10” type=”PERSON”>Tom Baker</ENAMEX> is lending
<ENAMEX id=”10” type=”MALE_PRONOUN”>his</ENAMEX> voice to a service which transfers text messages to landlines.</sent>
<sent>October 6,
<ENAMEX id=”11” type=”LOCATION”>Athens</ENAMEX>.</sent>
<sent>Explore the Acropolis temples, including the masterpiece of
<ENAMEX id=”12” type=”LOCATION”>Periclean</ENAMEX>
<ENAMEX id=”11” type=”LOCATION”>Athens</ENAMEX>, the
<ENAMEX id=”13” type=”ORGANIZATION”>Parthenon.</ENAMEX> </sent>
<sent>Stop for lunch in the picturesque neighborhood of the plaka.</sent>
<sent>Visit the
<ENAMEX id=”14” type=”LOCATION”>Panathenaic Stadium</ENAMEX> where the first Olympics of the modern era were held in 1896, and continue to the
<ENAMEX id=”15” type=”ORGANIZATION”>National Archaeological Museum</ENAMEX>, reopened after extensive renovation.</sent>
<sent>
<ENAMEX id=”16” type=”PERSON”>Frances McDormand</ENAMEX>.</sent>
<sent>
<ENAMEX id=”16” type=”ORGANIZATION”>McDormand</ENAMEX> was born in
<ENAMEX id=”17” type=”LOCATION”>Chicago</ENAMEX>,
<ENAMEX id=”18” type=”LOCATION”>Illinois</ENAMEX>, the youngest of three children adopted by Canadian parents
<ENAMEX id=”16” type=”PERSON”>Vernon McDormand</ENAMEX> (a
<ENAMEX id=”19” type=”ORGANIZATION”>Disciples of Christ</ENAMEX> minister) and
<ENAMEX id=”20” type=”PERSON”>Noreen.</ENAMEX> </sent>
<sent>
<ENAMEX id=”20” type=”FEMALE_PRONOUN”>She</ENAMEX> spent much of
<ENAMEX id=”20” type=”FEMALE_PRONOUN”>her</ENAMEX> youth in the
<ENAMEX id=”21” type=”LOCATION”>Pittsburgh</ENAMEX>, Pe nnsylvania, suburb of
<ENAMEX id=”22” type=”LOCATION”>Monessen</ENAMEX>, where
<ENAMEX id=”20” type=”FEMALE_PRONOUN”>she</ENAMEX> graduated from high school.</sent>
<sent>
<ENAMEX id=”20” type=”FEMALE_PRONOUN”>She</ENAMEX> attended
<ENAMEX id=”23” type=”ORGANIZATION”>Bethany College</ENAMEX>,
<ENAMEX id=”23” type=”LOCATION”>Bethany</ENAMEX>,
<ENAMEX id=”24” type=”LOCATION”>West</ENAMEX>

```
<ENAMEX id="25" type="LOCATION">Virginia</ENAMEX>, and earned a
<ENAMEX id="26" type="ORGANIZATION">B.A.</ENAMEX> in
<ENAMEX id="27" type="LOCATION">Theater</ENAMEX> in 1979.</sent>
<sent>
<ENAMEX id="28" type="PERSON">Bill</ENAMEX>, who owns www.microsoft.com, can be reached at billg@microsoft.com.</sent>
</P> </DOCUMENT>
```

6 Appendix B - Example Output of Wikipedia System

The following is the output of the Wikipedia system on the test text provided in Appendix A.

```
    <?xml version=”1.0” encoding=”UTF-8”?>
<DOCUMENT>
<CATEGORY id=”0” lp-type=”ORGANIZATION” text=”UN” type=”Limited geographic scope” score=”0.2”/>
<CATEGORY id=”0” lp-type=”ORGANIZATION” text=”UN” type=”Nobel Peace Prize winners” score=”0.18”/>
<CATEGORY id=”0” lp-type=”ORGANIZATION” text=”UN” type=”International organizations” score=”0.15”/>
<CATEGORY id=”0” lp-type=”ORGANIZATION” text=”UN” type=”United Nations” score=”0.15”/>
<CATEGORY id=”0” lp-type=”ORGANIZATION” text=”UN” type=”Wikipedia style guidelines” score=”0.0”/>
<CATEGORY id=”0” lp-type=”ORGANIZATION” text=”UN” type=”1945 establishments” score=”-0.049999997”/>
<CATEGORY id=”0” lp-type=”ORGANIZATION” text=”UN” type=”Law-related articles lacking sources” score=”-9.82”/>
<CATEGORY id=”1” lp-type=”LOCATION” text=”Iran” type=”Persian” score=”0.3”/>
<CATEGORY id=”1” lp-type=”LOCATION” text=”Iran” type=”Middle Eastern countries” score=”0.2”/>
<CATEGORY id=”1” lp-type=”LOCATION” text=”Iran” type=”Near Eastern countries” score=”0.2”/>
<CATEGORY id=”1” lp-type=”LOCATION” text=”Iran” type=”Persian Gulf states” score=”0.2”/>
<CATEGORY id=”1” lp-type=”LOCATION” text=”Iran” type=”Southwest Asian countries” score=”0.2”/>
<CATEGORY id=”1” lp-type=”LOCATION” text=”Iran” type=”Cities in Iran” score=”0.2”/>
<CATEGORY id=”1” lp-type=”LOCATION” text=”Iran” type=”Special territories” score=”0.15”/>
<CATEGORY id=”1” lp-type=”LOCATION” text=”Iran” type=”Articles lacking sources” score=”-9.8”/>
<CATEGORY id=”1” lp-type=”LOCATION” text=”Iran” type=”Iran” score=”-9.9”/>
<CATEGORY id=”2” lp-type=”LOCATION” text=”Bosnia” type=”Disambiguation” score=”-9.9”/>
<CATEGORY id=”3” lp-type=”PERSON” text=”Christian Schwarz-Schilling” type=”German” score=”0.3”/>
<CATEGORY id=”3” lp-type=”PERSON” text=”Christian Schwarz-Schilling” type=”Living people” score=”0.15”/>
<CATEGORY id=”3” lp-type=”PERSON” text=”Christian Schwarz-Schilling” type=”German politician stubs” score=”-
9.8”/>
<CATEGORY id=”4” lp-type=”ORGANIZATION” text=”Fed” type=”Federal Reserve” score=”0.35000002”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”American” score=”0.3”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”Phi Beta Kappa members” score=”0.18”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”Living people” score=”0.15”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”Guggenheim Fellowships” score=”0.15”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”MIT alumni” score=”0.15”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”Stanford alumni” score=”0.15”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”Princeton alumni” score=”0.15”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”Jewish Americans” score=”0.15”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”Harvard alumni” score=”0.15”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”American professors” score=”0.15”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”American economists” score=”0.15”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”Current events” score=”0.15”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”U.S. Council of Economic Advisors” score=”0.12”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”People from Georgia (U.S. state)” score=”0.12”/>
<CATEGORY id=”5” lp-type=”PERSON” text=”Ben Bernanke” type=”1953 births” score=”-10.05”/>
<CATEGORY id=”6” lp-type=”ORGANIZATION” text=”Federal Reserve” type=”Federal Reserve” score=”-9.650001”/>
<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”Federal Reserve economists” score=”0.2”/>
<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”Living people” score=”0.15”/>
<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”Jewish Americans” score=”0.15”/>
<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”Columbia alumni” score=”0.15”/>
<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”Business leaders” score=”0.15”/>
<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”U.S. Council of Economic Advisors” score=”0.12”/>
```

<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”Presidential Medal of Freedom recipients” score=”0.12”/>
<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”Chairmen of the Federal Reserve” score=”0.12”/>
<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”British” score=”0.1”/>
<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”Knights Commander of the British Empire” score=”0.1”/>
<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”Economists” score=”0.1”/>
<CATEGORY id=”7” lp-type=”PERSON” text=”Alan Greenspan” type=”1926 births” score=”-10.05”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”Danish” score=”0.3”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”North Atlantic Islands” score=”0.2”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”Islands of Denmark” score=”0.2”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”Former Norwegian colonies” score=”0.2”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”North American countries” score=”0.2”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”World records” score=”0.15”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”Island nations” score=”0.15”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”Danish dependencies” score=”0.15”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”Norwegian” score=”0.1”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”American” score=”0.1”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”Special territories of the European Union” score=”0.1”/>
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”Greenland” score=”-9.9”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”Roman Catholic actors” score=”0.2”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”Little Britain actors” score=”0.2”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”Doctor Who actors” score=”0.2”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”English television actors” score=”0.2”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”English film actors” score=”0.2”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”Voice actors” score=”0.15”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”Premature obituaries” score=”0.15”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”Living people” score=”0.15”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”Blackadder actors” score=”0.15”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”Irish” score=”0.1”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”People of Irish descent in Great Britain” score=”0.1”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”Liverpudlians” score=”0.1”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”1934 births” score=”-10.05”/>
<CATEGORY id=”11” lp-type=”LOCATION” text=”Athens” type=”Greek” score=”0.3”/>
<CATEGORY id=”11” lp-type=”LOCATION” text=”Athens” type=”Greek prefectural capitals” score=”0.2”/>
<CATEGORY id=”11” lp-type=”LOCATION” text=”Athens” type=”Capitals in Europe” score=”0.2”/>
<CATEGORY id=”11” lp-type=”LOCATION” text=”Athens” type=”Places on Placeopedia” score=”0.2”/>
<CATEGORY id=”11” lp-type=”LOCATION” text=”Athens” type=”Coastal cities” score=”0.15”/>
<CATEGORY id=”11” lp-type=”LOCATION” text=”Athens” type=”Cities and towns in Greece” score=”0.12”/>
<CATEGORY id=”11” lp-type=”LOCATION” text=”Athens” type=”Host cities of the Summer Olympic Games” score=”0.1”/>
<CATEGORY id=”11” lp-type=”LOCATION” text=”Athens” type=”Athens” score=”-9.9”/>
<CATEGORY id=”13” lp-type=”ORGANIZATION” text=”Parthenon” type=”Greek” score=”0.3”/>
<CATEGORY id=”13” lp-type=”ORGANIZATION” text=”Parthenon” type=”Ancient Greek structures” score=”0.2”/>
<CATEGORY id=”13” lp-type=”ORGANIZATION” text=”Parthenon” type=”Buildings and structures in Athens” score=”0.12”/>
<CATEGORY id=”14” lp-type=”LOCATION” text=”Panathenaic Stadium” type=”Athens Olympic venues” score=”0.2”/>
<CATEGORY id=”14” lp-type=”LOCATION” text=”Panathenaic Stadium” type=”Sports venues in Greece” score=”0.18”/>
<CATEGORY id=”14” lp-type=”LOCATION” text=”Panathenaic Stadium” type=”Olympic stadiums” score=”0.15”/>
<CATEGORY id=”14” lp-type=”LOCATION” text=”Panathenaic Stadium” type=”1896 Summer Olympics” score=”0.0”/>
<CATEGORY id=”15” lp-type=”ORGANIZATION” text=”National Archaeological Museum” type=”Naples” score=”0.3”/>
<CATEGORY id=”15” lp-type=”ORGANIZATION” text=”National Archaeological Museum” type=”Museums in Italy” score=”0.2”/>

<CATEGORY id=”15” lp-type=”ORGANIZATION” text=”National Archaeological Museum” type=”National museums” score=”0.15”/>
<CATEGORY id=”15” lp-type=”ORGANIZATION” text=”National Archaeological Museum” type=”Archaeology museums” score=”0.15”/>
<CATEGORY id=”15” lp-type=”ORGANIZATION” text=”National Archaeological Museum” type=”Archaeology stubs” score=”-9.85”/>
<CATEGORY id=”15” lp-type=”ORGANIZATION” text=”National Archaeological Museum” type=”Museum stubs” score=”-9.85”/>
<CATEGORY id=”17” lp-type=”LOCATION” text=”Chicago” type=”Chicago, Illinois” score=”0.15”/>
<CATEGORY id=”17” lp-type=”LOCATION” text=”Chicago” type=”Spoken articles” score=”0.15”/>
<CATEGORY id=”17” lp-type=”LOCATION” text=”Chicago” type=”Communities on U.S. Highway 66” score=”0.12”/>
<CATEGORY id=”18” lp-type=”LOCATION” text=”Illinois” type=”States of the United States” score=”0.12”/>
<CATEGORY id=”18” lp-type=”LOCATION” text=”Illinois” type=”Illinois” score=”-9.9”/>
<CATEGORY id=”19” lp-type=”ORGANIZATION” text=”Disciples of Christ” type=”Restoration Movement” score=”0.35000002”/>
<CATEGORY id=”19” lp-type=”ORGANIZATION” text=”Disciples of Christ” type=”Christian denominations” score=”0.15”/>
<CATEGORY id=”20” lp-type=”PERSON” text=”Noreen” type=”Cryptography stubs” score=”-9.85”/>
<CATEGORY id=”21” lp-type=”LOCATION” text=”Pittsburgh” type=”Cities in Pennsylvania” score=”0.2”/>
<CATEGORY id=”21” lp-type=”LOCATION” text=”Pittsburgh” type=”Allegheny County, Pennsylvania” score=”0.2”/>
<CATEGORY id=”21” lp-type=”LOCATION” text=”Pittsburgh” type=”Pittsburgh, Pennsylvania” score=”0.15”/>
<CATEGORY id=”21” lp-type=”LOCATION” text=”Pittsburgh” type=”All-America City” score=”0.15”/>
<CATEGORY id=”22” lp-type=”LOCATION” text=”Monessen” type=”Westmoreland County, Pennsylvania” score=”0.4”/>
<CATEGORY id=”22” lp-type=”LOCATION” text=”Monessen” type=”Cities in Pennsylvania” score=”0.2”/>
<CATEGORY id=”23” lp-type=”ORGANIZATION” text=”Bethany College” type=”Disambiguation” score=”-9.9”/>
<CATEGORY id=”23” lp-type=”LOCATION” text=”Bethany” type=”Disambiguation” score=”-9.9”/>
<CATEGORY id=”24” lp-type=”LOCATION” text=”West” type=”Orientation” score=”0.1”/>
<CATEGORY id=”25” lp-type=”LOCATION” text=”Virginia” type=”States of the United States” score=”0.12”/>
<CATEGORY id=”25” lp-type=”LOCATION” text=”Virginia” type=”Virginia” score=”-9.9”/>
<CATEGORY id=”26” lp-type=”ORGANIZATION” text=”B.A.” type=”Bachelor’s degrees” score=”0.15”/>
<CATEGORY id=”27” lp-type=”LOCATION” text=”Theater” type=”Stage terminology” score=”0.15”/>
<CATEGORY id=”27” lp-type=”LOCATION” text=”Theater” type=”Theatre” score=”0.1”/>
<CATEGORY id=”27” lp-type=”LOCATION” text=”Theater” type=”Articles lacking sources” score=”-9.8”/>
<CATEGORY id=”28” lp-type=”PERSON” text=”Bill” type=”Disambiguation” score=”-9.9”/>
</DOCUMENT>

7 Appendix C - Example Outputs of Rules Module

The Rules Module generates two outputs: it modifies the LingPipe output by adding tags for newly discovered NEs and it generates a stand-off annotation for all NEs it has found a type for.

7.1 Modified LingPipe Output of Example Text

```
<DOCUMENT>
<ENAMEX id=”29” type=”none”>
21:03</ENAMEX>
<ENAMEX id=”30” type=”none”>
2006,</ENAMEX>
<ENAMEX id=”31” type=”none”>
31 January</ENAMEX>
<ENAMEX id=”0” type=”ORGANIZATION”>
UN</ENAMEX>
confirms <ENAMEX id=”1” type=”LOCATION”>
Iran</ENAMEX>
enrichment move. has started preparations to resume uranium enrichment, says a report by the nuclear watchdog. German
takes over top <ENAMEX id=”2” type=”LOCATION”>
Bosnia</ENAMEX>
job. <ENAMEX id=”3” type=”PERSON”>
Christian Schwarz-Schilling</ENAMEX>
, a German diplomat, becomes the powerful top envoy to Bernanke approved as new <ENAMEX id=”4” type=”ORGANIZATION”>
Fed</ENAMEX>
head. <ENAMEX id=”5” type=”PERSON”>
Ben Bernanke</ENAMEX>
is approved as the new head of the <ENAMEX id=”6” type=”ORGANIZATION”>
Federal Reserve</ENAMEX>
, taking over from the outgoing <ENAMEX id=”7” type=”PERSON”>
Alan Greenspan</ENAMEX>
Stark warning over climate change. Greenhouse gas emissions are rising at an unsustainable rate and the <ENAMEX
id=”8” type=”LOCATION”>
Greenland</ENAMEX>
ice cap is at risk, scientists say. <ENAMEX id=”9” type=”PERSON”>
Actor Baker</ENAMEX>
becomes voice of text. Former Doctor Who <ENAMEX id=”10” type=”PERSON”>
Tom Baker</ENAMEX>
is lending his voice to a service which transfers text messages to landlines. <ENAMEX id=”32” type=”none”>
October 6,</ENAMEX>
<ENAMEX id=”11” type=”LOCATION”>
Athens</ENAMEX>
Explore the Acropolis temples, including the masterpiece of <ENAMEX id=”12” type=”LOCATION”>
Periclean</ENAMEX>
, the <ENAMEX id=”13” type=”ORGANIZATION”>
Parthenon.</ENAMEX>
Stop for lunch in the picturesque neighborhood of the plaka. Visit the <ENAMEX id=”14” type=”LOCATION”>
Panathenaic Stadium</ENAMEX>
```

<ENAMEX id=”33” type=”none”>
1896,</ENAMEX>
<ENAMEX id=”15” type=”ORGANIZATION”>
National Archaeological Museum</ENAMEX>
, reopened after extensive renovation. <ENAMEX id=”16” type=”PERSON”>
Vernon McDormand</ENAMEX>
was born in <ENAMEX id=”17” type=”LOCATION”>
Chicago</ENAMEX>
<ENAMEX id=”18” type=”LOCATION”>
Illinois</ENAMEX>
, the youngest of three children adopted by Canadian parents <ENAMEX id=”19” type=”ORGANIZATION”>
Disciples of Christ</ENAMEX>
minister) and <ENAMEX id=”20” type=”PERSON”>
Noreen.</ENAMEX>
She spent much of her youth in the <ENAMEX id=”21” type=”LOCATION”>
Pittsburgh</ENAMEX>
, Pe nnsylvania, suburb of <ENAMEX id=”22” type=”LOCATION”>
Monessen</ENAMEX>
, where she graduated from high school. She attended <ENAMEX id=”23” type=”LOCATION”>
Bethany</ENAMEX>
<ENAMEX id=”24” type=”LOCATION”>
West</ENAMEX>
<ENAMEX id=”25” type=”LOCATION”>
Virginia</ENAMEX>
, and earned a <ENAMEX id=”26” type=”ORGANIZATION”>
B.A.</ENAMEX>
<ENAMEX id=”27” type=”LOCATION”>
Theater</ENAMEX>
<ENAMEX id=”34” type=”none”>
1979.</ENAMEX>
<ENAMEX id=”28” type=”PERSON”>
Bill</ENAMEX>
<ENAMEX id=”35” type=”none”>
billg@microsoft.com.</ENAMEX>
<ENAMEX id=”36” type=”none”>
www.microsoft.com,</ENAMEX>
</DOCUMENT>

7.2 Stand-Off Annotation for Example Text

<?xml version=”1.0” encoding=”UTF-8”?>
<DOCUMENT>
<CATEGORY id=”29” lp-type=”none” text=”21:03” type=”time”/>
<CATEGORY id=”30” lp-type=”none” text=”2006,” type=”year”/>
<CATEGORY id=”31” lp-type=”none” text=”31 January” type=”date”/>
<CATEGORY id=”1” lp-type=”LOCATION” text=”Iran” type=”country”/>
<CATEGORY id=”2” lp-type=”LOCATION” text=”Bosnia” type=”town”/>
<CATEGORY id=”3” lp-type=”PERSON” text=”Christian Schwarz-Schilling” type=”diplomat”/>
<CATEGORY id=”3” lp-type=”PERSON” text=”Christian Schwarz-Schilling” type=”German”/>

```
<CATEGORY id=”8” lp-type=”LOCATION” text=”Greenland” type=”country”/>
<CATEGORY id=”10” lp-type=”PERSON” text=”Tom Baker” type=”Doctor”/>
<CATEGORY id=”32” lp-type=”none” text=”October 6,” type=”date”/>
<CATEGORY id=”11” lp-type=”LOCATION” text=”Athens” type=”town”/>
<CATEGORY id=”12” lp-type=”LOCATION” text=”Periclean” type=”town”/>
<CATEGORY id=”14” lp-type=”LOCATION” text=”Panathenaic Stadium” type=”town”/>
<CATEGORY id=”33” lp-type=”none” text=”1896,” type=”year”/>
<CATEGORY id=”17” lp-type=”LOCATION” text=”Chicago” type=”town”/>
<CATEGORY id=”18” lp-type=”LOCATION” text=”Illinois” type=”town”/>
<CATEGORY id=”19” lp-type=”ORGANIZATION” text=”Disciples of Christ” type=”Canadian”/>
<CATEGORY id=”21” lp-type=”LOCATION” text=”Pittsburgh” type=”town”/>
<CATEGORY id=”22” lp-type=”LOCATION” text=”Monessen” type=”town”/>
<CATEGORY id=”23” lp-type=”LOCATION” text=”Bethany” type=”town”/>
<CATEGORY id=”24” lp-type=”LOCATION” text=”West” type=”town”/>
<CATEGORY id=”25” lp-type=”LOCATION” text=”Virginia” type=”town”/>
<CATEGORY id=”27” lp-type=”LOCATION” text=”Theater” type=”town”/>
<CATEGORY id=”34” lp-type=”none” text=”1979.” type=”year”/>
<CATEGORY id=”35” lp-type=”none” text=”billg@microsoft.com.” type=”email”/>
<CATEGORY id=”36” lp-type=”none” text=”www.microsoft.com,” type=”url”/>
</DOCUMENT>
```

8 Appendix D - Example Output of Machine Learning Module

Text: Eastern and western churches reconciled (519). Justinian I, the Great (483565), becomes Byzantine emperor (527), issues his first code of civil laws (529), conquers North Africa, Italy, and part of Spain. Plague spreads through Europe (542 et seq.). Arthur, semi-legendary king of the Britons (killed, c. 537). Bothius, Roman scholar (executed, 524). Vandals destroy Rome (A.D. 455). Western Roman empire ends as Odoacer, German chieftain, overthrows last Roman emperor, Romulus Augustulus, and becomes king of Italy (A.D. 476). Ostrogothic kingdom of Italy established by Theodoric the Great (A.D. 493). Clovis, ruler of the Franks, is converted to Christianity (A.D. 496). First schism between western and eastern churches (A.D. 484). Claudius poisoned (A.D. 54), succeeded by Nero (commits suicide, A.D. 68). Missionary journeys of Paul the Apostle (A.D. 3460). Jews revolt against Rome; Jerusalem destroyed (A.D. 70). Roman persecutions of Christians begin (A.D. 64). Colosseum built in Rome (A.D. 7180). Trajan (rules A.D. 98116); Roman empire extends to Mesopotamia, Arabia, Balkans. Hungary and Scandinavia converted to Christianity. Viking raider Leif Eriksson discovers North America, calls it Vinland. Beowulf, Old English epic. Danes control England. Canute takes throne (1016), conquers Norway (1028), dies (1035); kingdom divided among his sons: Harold Harefoot (England), Sweyn (Norway), Hardecanute (Denmark).At Council of Clermont, Pope Urban II calls for a holy war to wrest control of Jerusalem from Muslims, which launches the First Crusade (1096), one of at least 8 European military campaigns between 1095 and 1291 to regain the Holy Land. (For detailed chronology, see The Crusades.)

Learning approach classification:

```
<?xml version=”1.0” encoding=”utf-8”?>
<!– Text Learning approach –>
<!DOCTYPE ANNOTATION SYSTEM ”annotation.dtd”>
<ANNOTATION>
<DOCUMENT id=”doc-1”>
<CATEGORY id=”97807” lp-type=”LOCATION” text=”North Africa” type=”CONTINENT” score=””/>
<CATEGORY id=”97808” lp-type=”LOCATION” text=”Italy” type=”CAPITAL” score=””/>
<CATEGORY id=”97809” lp-type=”LOCATION” text=”Spain” type=”COUNTRY” score=””/>
<CATEGORY id=”97810” lp-type=”LOCATION” text=”Plague” type=”COUNTRY” score=””/>
<CATEGORY id=”97811” lp-type=”LOCATION” text=”Europe” type=”CONTINENT” score=””/>
<CATEGORY id=”97814” lp-type=”LOCATION” text=”Bothius” type=”COUNTRY” score=””/>
<CATEGORY id=”97815” lp-type=”LOCATION” text=”Roman” type=”CAPITAL” score=””/>
<CATEGORY id=”97816” lp-type=”LOCATION” text=”Rome” type=”CAPITAL” score=””/>
<CATEGORY id=”97808” lp-type=”LOCATION” text=”Italy” type=”COUNTRY” score=””/>
<CATEGORY id=”97808” lp-type=”LOCATION” text=”Italy” type=”COUNTRY” score=””/>
<CATEGORY id=”97806” lp-type=”LOCATION” text=”Great” type=”GENERAL” score=””/>
<CATEGORY id=”97819” lp-type=”LOCATION” text=”Clovis” type=”COUNTRY” score=””/>
<CATEGORY id=”97820” lp-type=”LOCATION” text=”Franks” type=”COUNTRY” score=””/>
<CATEGORY id=”97816” lp-type=”LOCATION” text=”Rome” type=”CAPITAL” score=””/>
<CATEGORY id=”97824” lp-type=”LOCATION” text=”Jerusalem” type=”CAPITAL” score=””/>
<CATEGORY id=”97816” lp-type=”LOCATION” text=”Rome” type=”CAPITAL” score=””/>
<CATEGORY id=”97825” lp-type=”LOCATION” text=”Mesopotamia” type=”GENERAL” score=””/>
<CATEGORY id=”97826” lp-type=”LOCATION” text=”Arabia” type=”CAPITAL” score=””/>
<CATEGORY id=”97827” lp-type=”LOCATION” text=”Balkans” type=”COUNTRY” score=””/>
<CATEGORY id=”97830” lp-type=”LOCATION” text=”Christianity” type=”COUNTRY” score=””/>
<CATEGORY id=”97832” lp-type=”LOCATION” text=”North America” type=”CONTINENT” score=””/>
```

<CATEGORY id=”97833” lp-type=”LOCATION” text=”Beowulf” type=”COUNTRY” score=””/>
<CATEGORY id=”97834” lp-type=”LOCATION” text=”England” type=”COUNTRY” score=””/>
<CATEGORY id=”97835” lp-type=”LOCATION” text=”Norway” type=”COUNTRY” score=””/>
<CATEGORY id=”97834” lp-type=”LOCATION” text=”England” type=”COUNTRY” score=””/>
<CATEGORY id=”97835” lp-type=”LOCATION” text=”Norway” type=”CAPITAL” score=””/>
<CATEGORY id=”97838” lp-type=”LOCATION” text=”Hardecanute” type=”CITY” score=””/>
<CATEGORY id=”97839” lp-type=”LOCATION” text=”Denmark” type=”COUNTRY” score=””/>
<CATEGORY id=”97840” lp-type=”LOCATION” text=”Clermont” type=”CAPITAL” score=””/>
<CATEGORY id=”97824” lp-type=”LOCATION” text=”Jerusalem” type=”CAPITAL” score=””/>
<CATEGORY id=”97842” lp-type=”LOCATION” text=”Muslims” type=”CONTINENT” score=””/>
</DOCUMENT>
</ANNOTATION>