SYMBOLIC AND NEURAL LEARNING OF NAMED-ENTITY RECOGNITION AND CLASSIFICATION SYSTEMS IN TWO LANGUAGES

G. Petasis, S. Petridis, G. Paliouras, V. Karkaletsis, S.J. Perantonis and C.D. Spyropoulos
Institute of Informatics and Telecommunications,
National Centre for Scientific Research "Demokritos",
153 10 Ag. Paraskevi, Athens, Greece
e-mail: {petasis, petridis, paliourg, vangelis, sper, costass}@iit.demokritos.gr

Abstract This paper compares two alternative approaches to the problem of acquiring named-entity recognition and classification systems from training corpora, in two different languages. The process of named-entity recognition and classification is an important subtask in most language engineering applications, in particular information extraction, where different types of named entity are associated with specific roles in events. The manual construction of rules for the recognition of named entities is a tedious and time-consuming task. For this reason, effective methods to acquire such systems automatically from data are very desirable. In this paper we compare two popular learning methods on this task: a decision-tree induction method and a multi-layered feed-forward neural network. Particular emphasis is paid on the selection of the appropriate data representation for each method and the extraction of training examples from unstructured textual data. We compare the performance of the two methods on large corpora of English and Greek texts and present the results. In addition to the good performance of both methods, one very interesting result is the fact that a simple representation of the data, which ignores the order of the words within a named entity, leads to improved results over a more complex approach that preserves word order.

Keywords Named entity recognition, tree induction, neural networks.

1 INTRODUCTION

Named Entity Recognition and Classification (NERC) is the task of identifying and semantically classifying named entities (NEs) in text. In several practical problems, NEs tend to represent a significant percentage of the words in a corpus. As a result, NERC constitutes an essential subtask in most language engineering applications where the effective understanding of language is required, such as information extraction. Information Extraction (IE) is the task of automatically extracting information of interest from unconstrained text and creating a structured representation of this information. In IE we are mainly interested in extracting *events*. Every event involves a number of named entities, which belong in different semantic classes (e.g. persons, organisations, locations, dates), and some relationships that hold among these named entities (e.g. personnel joining and leaving companies in management succession events). As a result, an IE task involves two main subtasks: the recognition and classification of named entities

(NERC) and the identification of relationships (the events) holding between named entities.

A typical NERC system consists of a lexicon and a grammar. The lexicon is a set of named entities that are known beforehand and have been classified into semantic classes. The classes into which the named entities are classified constitute semantic information that varies significantly among different thematic domains. For instance, the identification of organisation names makes sense in financial news, but not in the scientific literature. The grammar is used to recognize and classify NEs that are not in the lexicon and to decide upon the final classes of NEs in cases where ambiguity exists in the lexicon. Due to the semantic nature of these two resources, domain-specific systems are needed for successful NERC. Furthermore, the NERC task is considerably different when moving from one language to another. Thus, a global system for NERC is not meaningful and a new system usually needs to be constructed for each separate problem.

The manual adaptation of NERC systems to a particular domain or to a new language is a very time-consuming process and in some cases impossible, due to the lack of experts. Thus, the automatic acquisition/adaptation of the needed resources from corpora is highly desirable. Automated knowledge acquisition, with the use of machine learning techniques, has recently been proposed as a promising solution to this and other similar problems in language engineering. Machine learning techniques are divided into two broad categories: supervised and unsupervised. Supervised learning techniques require the existence of training examples that have been hand-tagged with the correct class. On the other hand, unsupervised techniques assume that the correct classification of the training examples is not known and classify the examples according to a similarity metric. Supervised methods are more expensive than unsupervised ones, in terms of the time spent to pre-process the training data. However, the additional information included in supervised data leads usually to a better classification system. Nymble [Bikel *et al.* 1997], Alembic [Vilain *et al.* 1996, Day *et al.* 1998], AutoLearn [Cowie 1995], and RoboTag [Bennet *et al*. 1997, Sekine 1998] are examples of systems exploiting supervised learning techniques. On the other hand, the NERC system developed for Italian [Cucchiarelli & Velardi 1998a, Cucchiarelli & Velardi 1998b] is an example of a system exploiting unsupervised learning. Machine learning algorithms can be further classified according to the model representation that they use into symbolic and subsymbolic (or numeric). Symbolic methods use discrete symbolic representations, such as sets of rules or decision trees, while subsymbolic methods use numeric representations, such as regression functions or neural networks.

This article deals with one half of the NERC problem, namely the acquisition of the NERC grammar, assuming the existence of the corresponding lexicon of known names. We present an evaluation of two different NERC methods based on supervised machine learning. The first method uses the decision-tree induction algorithm C4.5 [Quinlan 1993], while the second uses a standard feed-forward multi-layer perceptron with one hidden layer. Decision trees are a typical symbolic representation, while neural networks are typical representatives of subsymbolic models.

In addition to the representation of the classification models, particular emphasis is given on the representation of the training data for the two learning methods. The symbolic nature of decision trees makes them more suitable to language engineering tasks with symbolic textual data. On the other hand, neural networks require the encoding of the data into numeric feature vectors, which can be problematic if we want to preserve all of the original information. In this paper, we use a simple symbolic representation, as well as a numeric one, which ignores the order of the words within a named entity, in order to reduce the di-

mensionality of the problem to manageable levels. The decision-tree learning method can handle both data representations. For this reason, it is tested with both representations, leading to conclusions about the effect of ignoring word order.

The data for the evaluation comes from two corpora containing financial news articles. The two corpora are in two different languages: Greek and English. In order to use the supervised learning methods, the named entities have been identified and tagged manually in the two evaluation corpora.

Details about the two learning algorithms that we used are given in section 2. Section 3 describes the evaluation data, as well as the experimental setting. The results of the experiments are presented in section 4. Finally, section 5 presents conclusions on the usability of machine learning to the named-entity recognition task, as well as interesting directions for further work.

2 MACHINE LEARNING IN NAMED-ENTITY RECOGNITION AND CLASSIFICATION

As mentioned above, we apply two different supervised learning algorithms to the task of constructing a NERC automatically from tagged training data. In terms of syntactic categories, NEs are lexical noun phrases (NPs). Both recognisers accept as input noun phrases augmented with contextual information. Their task is to recognise the noun phrases that are named entities and classify them into the appropriate semantic classes.

2.1 Decision Tree NERC system

The first NERC system is constructed by a general-purpose symbolic machine learning algorithm, called C4.5. C4.5 is a supervised learning algorithm that performs induction of decision trees, i.e., it constructs decision trees from training data. C4.5 uses a greedy hill-climbing search through the space of possible decision trees aiming to construct one that explains the data well. It performs this search by the method of recursive partitioning of the training data. It starts with the complete dataset and chooses one feature that discriminates best between examples (feature vectors) of different types, e.g. organisations, persons or locations. The quality of discrimination is assessed by an information-theoretic metric, based on mutual information. The same process is applied recursively on each subset, choosing other features for discrimination and partitioning the training set further. This continuous partitioning leads to increasingly “purer” subsets, i.e., sets which contain many examples of one class, e.g. person, and few of all other classes. The process ends when a stopping criterion is satisfied. In the simplest case, this criterion requires completely pure subsets, i.e., each training subset associated with a leaf node should contain only one type of example. This criterion is unrealistic for real-world problems and leads to overtraining of the decision tree to the data. In order to avoid this problem, C4.5 incorporates a pruning method, which constructs a more robust decision tree, allowing a small amount of impurity on the final subsets generated by the recursive partitioning. Thus, each of the leaves in the tree may classify incorrectly a few of the NEs in the training set. However, it is expected to capture the most important classification patterns.

Being a symbolic learning method, decision-tree induction was originally designed to handle symbolic data [Quinlan 1983]. More recent versions of the method are able to handle numeric features, by transforming them into an equivalent set of binary ones. Thus, C4.5 can handle both data representations used here.

2.2 Neural Network NERC System

The second NERC system is based on a neural network. Artificial neural networks (ANNs) are computational systems whose architecture and operation are based on the present knowledge about biological nervous systems. By analogy to these systems, ANNs consist of a set of suitably positioned simple processing elements (nodes) representing the neurons. Each node receives signals from a fixed set of other nodes by links called synapses and determines its activation as a function of these signals and the strengths (weights) of the synapses. The response of each node is a function of its activation. Different ANN models can be constructed by suggesting different ways of connecting processing elements. The paradigm used in this work is the multi-layered feed-forward network (FNN) which consists of an input layer, intermediate or "hidden" layers of nodes and a layer of output nodes, with each node receiving input only from nodes in the previous layer.

An important result of the studies related to FNN is the rigorous theoretical establishment that these networks are universal approximators. Given that a sufficient number of hidden nodes is included in the network architecture, there exists a set of synaptic weights, whereby a FNN can realize any arbitrarily complicated, generically non-linear functional relationship between its input and its output by superposition of the elementary node functions. A major theme in neural network research is training the network in order to find the appropriate set of synaptic weights. This is achieved using training algorithms, such as the popular *back-propagation*, which minimise by gradient descent the error of the output nodes with respect to the desired function. Recently, a family of Algorithms for Learning Efficiently using Constrained Optimisation (ALECO) has been proposed for training FNNs [Perantonis *et al.* in print]. These algorithms are based on principles of non-linear programming theory and incorporate in their formalism additional information about the learning properties of FNNs in the form of non-linear constraints. The variant used in this work is described in detail in [Karras & Perantonis 1995]. The algorithm is faster than back-propagation, exhibits good convergence properties in difficult benchmark problems, is well suited for solving large scale problems and has been shown to achieve good generalization performance in classification tasks.

3 EXPERIMENTAL SETTING

3.1 Corpus Preprocessing

In order to evaluate the performance of the two learning approaches presented here, we conducted experiments in two different languages, Greek and English. Both corpora used for evaluation covered similar thematic domains, involving several types of named entity, such as person, organisation, location, date, time, money, etc. The general consensus is that person and organisation types are more difficult to identify and classify. For this reason, our study focuses only on these two types of entity. The corpus used for the Greek experiments was provided by the Greek company Kapa-TEL and contained short financial news articles from 1998 until 2000. This corpus contained 5837 organisation and 695 person instances. For the purposes of the English experiments, we used part of the corpus that was used for the evaluation of the systems in the 6th Message Understanding Conference (MUC-6) conference [DARPA 1995]. The thematic domain in MUC-

6 was *management succession events* and the part used for the experiments contained 461 organisation and 373 person instances.

The performance of the two systems was compared to the performance of two manually constructed systems, the MITOS NERC system [Farmakiotou *et al.* 2000] for the Greek language and the VIE NERC system [Humphreys *et al.* 1997] for the English language. The MITOS NERC system was developed in the context of the R&D project MITOS[1] and mainly consists of three processing stages: linguistic pre-processing, named-entity identification and named-entity classification. The linguistic pre-processing stage involves some basic tasks, like tokenisation, sentence splitting, part-of-speech tagging and stemming. As Greek is an inflectional language, the stemmer is particularly useful for reducing the size of the NERC lexicon and the complexity of the NERC grammar. The NERC lexicon contains 1550 organisation names, 134 organisation specifiers, i.e., words that indicate the presence of an organisation name, 503 person names, 78 person specifiers and 1006 location names. The named-entity identification stage involves the detection of NE boundaries, i.e., the start and the end of all the possible text ranges that are likely to belong in a named entity. Once the possible named entities have been identified, the system uses the NERC lexicon, as well as internal and external evidence [McDonald 1996], i.e., the words and symbols that comprise the possible NE and the context in which it occurs, in order to classify the NEs into the desired categories.

The VIE system was developed at Sheffield University and consists of several modules: a tokeniser, a sentence splitter, a part-of-speech tagger, a gazetteer-list lookup module (NERC lexicon), and a named-entity parser. This system makes use of a set of gazetteer lists, consisting of person names, organisation names, company designators (such as Ltd. and Co.), person titles (such as Mr. and MD), etc., and a grammar, incorporating internal and external information about a phrase. The gazetteer lists that were used for our experiments contained 2599 organisation names, 229 organisation specifiers, 476 person names, 163 person specifiers, 2114 locations and some monetary and time expressions. The information used in the grammar consists of tags assigned by looking up the gazetteer lists, part-of-speech and syntactic properties of the words in a phrase. A simple bottom-up chart parser uses this grammar to identify phrases of interest.

3.2 Noun Phrase Representation

A crucial matter for machine learning in general is the choice of data representation. Both of the algorithms used here require the data to be provided in a *feature-vector* format, which is common in most work in machine learning. This representation requires the data (in our case NP instances) to be encoded in a fixed-length vector of values for a specified set of features.

In the work presented here, we focused on two issues:

- How to represent a single word, in a way significant for our recognition and classification tasks.
- How to combine single word representations so as to form a fixed-length feature vector representing the whole NP.

For the purpose of the experiments presented here, we have decided to encode two features for each relevant word in the corpus. The first feature represents part-of-speech information (POS tag) (e.g. adjective, possessive determiner, auxiliary verb) while the second feature is a gazetteer tag (e.g. city, country, organisation), when such information is available from the lexicon. Note that the actual word form (or its root) is not included in the feature vector.

The procedure that was followed in creating the vectors was the following:

Stage 1: *Identification of noun phrase instances.* Regarding the NP identification for Greek we used a phrase chunker developed in the context of the MITOS project.[2] The chunker identifies four types of phrase: noun phrases (NPs), prepositional phrases (PPs), verb phrases (VPs) and adverbial phrases (APs). However, it is not able to identify overlapping phrases, i.e., a NP which is part of a larger phrase. This is a problem in several cases, e.g. NPs included in larger PPs. For the purposes of our experiment, the results of the chunker, were post-processed in order to identify NPs included in larger phrases. For the English corpus we used the NP recogniser of the VIE NERC system. In addition to the NP itself, each instance was augmented with information about the close vicinity of the NP. This contextual information included two words before and two words after the NP.

Stage 2: *Semantic tagging of the noun phrases.* Once all NPs have been identified, every NP is compared against the manually annotated NEs in both corpora. The annotation includes the semantic category of the NEs allowing the classification of all NPs into three classes that are of interest in this study: persons, organisations and non-NEs. The third class, which corresponds to NPs that are not NEs, was used to capture the dual nature of the NERC task, namely the identification *and* classification of NE phrases. By providing a learning algorithm simply with person and organisation phrases, the algorithm would only learn how to distinguish between person and organisation names. By adding negative examples through the non-NE class, a NERC system is able to learn ways to also distinguish between NE and non-NE phrases.

The number of non-NE noun phrases in the Greek corpus is 37493 while in the MUC-6 data is 4333. It should be noted here that some of the non-NE phrases may subsume NE phrases. For instance, the NP *George Black's garden* is not a NE, but subsumes the person name *George Black*. In the English corpus we may also have non-NEs subsumed by NEs. For instance, the phrase *Greek Society for the Protection* is not a NE, but part of the organisation name *Greek Society for the Protection of Forests.* This case poses an important problem for a learning algorithm, which needs to identify what is missing from the phrase, i.e., the words *of Forests*, rather than what should be in it, in order for it to be a NE.

Stage 3: *Encoding of the instances into feature vectors.* The solutions that we adopted differ to a certain extent for the two learning algorithms, due to the different types of input that they require, as explained in section 2 above. The following two subsections describe the two different representations that were used. However, one common characteristic of both representations is that the feature vectors need to be of a fixed length. In order to achieve that, we limited the length of NPs to 9 words, i.e. NPs longer than 9 words were ignored. The occurrence of such NPs in the corpus was very rare.

3.3 Symbolic Feature Vector

For the symbolic learning algorithm, the encoding of the feature vectors was rather straightforward. The feature vector consisted of 26 features (two features for each of the 9 words in a NP, plus the words before and after the NP). In the case where a NP was shorter than 9 words, the remaining features were assigned a special value ("?"), treated as a label for missing information by C4.5. Missing gazetteer tags were treated differently. They were given a special tag called `NOTAG` instead. As an example of the way in which NPs were coded into feature vectors, consider the following phrase:

… of the *Securities and Exchange Commission* in the …

where the organisation phrase is shown in italics. The vector corresponding to this phrase is the following:

```
[IN, NOTAG, DT, NOTAG, NNP, org_key+organisation, CC, organi-
sation, NNP, organisation, NNP, org_base+organisation, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, IN, NOTAG, DT, NOTAG]
```

where the part-of-speech tags are to be interpreted as follows:

`IN`: preposition, `DT`: determiner, `NNP`: proper noun, `CC`: conjunct.

The gazetteer tags appearing in the above example are: `organisation`, `org_key`, `org_base` and `NOTAG`. The phrase *Securities and Exchange Commission* appears in the list of organisations and as a result all of its component words are assigned the tag `organisation`. Note that more than one gazetteer tag may be given to a word, meaning that the word exists in more than one gazetteer list, as in the case of the word *Securities*, which is both an `org_key` and part of an `organisation` (*Securities and Exchange Commission*). Multiple tags are joined by the plus sign in the symbolism that we use. Finally, the question marks in the vector symbolise missing words, as explained above. Three sets of vectors are constructed for the three classes of interest.

3.4 Numeric Feature Vector

Linguistic information is naturally encoded in symbolic form. Indeed, both the part-of-speech and the gazetteer tags have no established numerical type properties. For instance, there is no simple way to order a verb and an adjective. On the other hand, neural networks take as input a one-dimensional, fixed-length vector of real values. It follows that a suitable coding of tag information in a vector is not straightforward. However, it is crucial for the performance of the neural network and thus, emphasis was given to form the most meaningful representation.

To preserve symbolic information as much as possible, without making any assumptions about tag relations, we assume that tags are *orthogonal to each other*. Thus, to encode a word in a vector form, we proceeded as follows:

- Let N_{pos} be the number of possible different part-of-speech (POS) tags. Assuming that the tags are independent, we can form a "POS" space of N_{pos} dimensions, one dimension for each POS tag. Hence, every tag can be coded as a N_{pos}-length vector with all dimensions set to 0, except one, particular to the POS tag, which is set to 1. Moreover, the set of N_{pos} POS tag coding vectors formed may be considered as an orthogonal base for the "POS" space.
- The same reasoning can be applied to a "gazetteer tag" space. Hence, another N_{gaz} coding vectors of N_{gaz} dimensions, orthogonal to each other, can represent N_{gaz} independent gazetteer tags.
- We may now represent a single word by combining the two coding vectors. The complete vector is formed by the concatenation of one part-of-speech vector and one gazetteer vector. Hence, each word is represented by a vector of N dimensions, where $N = N_{pos} + N_{gaz}$ and may be considered as a point in a *N-dimensional* space.

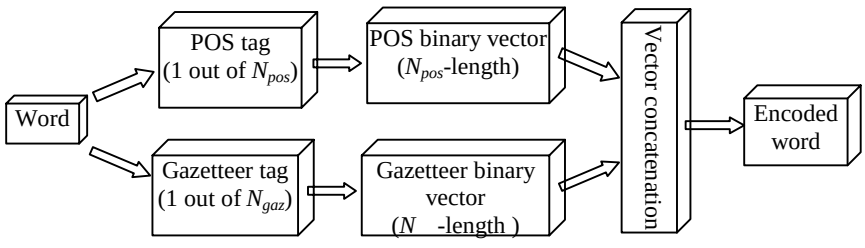
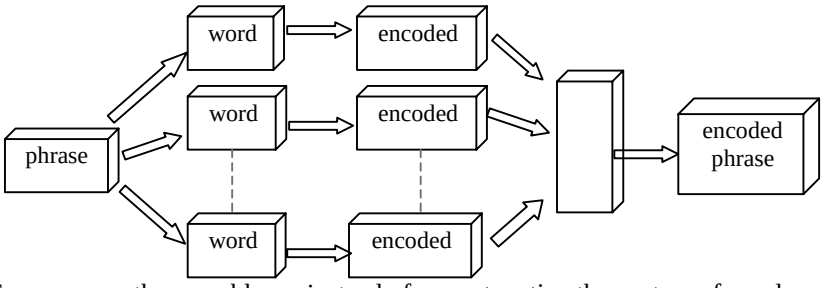


Figure 1. Word encoding for neural network input.

For the specific classification task, more than one word is given as input to the classifier. To represent a group of W words, one could think of concatenating W encoded words, forming a vector of WxN dimensions. However, such a representation has two drawbacks:

- The total length of vector tends to be very large. In our case, it would be about 650 features.
- In most cases, not all of the words are given. Indeed, noun phrases may have a varying number of words. Since, neural networks have no inherent mechanism for dealing with missing values, this would result, in our case, in vectors of varying length, which is inappropriate as input.

Figure 2. Noun phrase encoding for the neural network input.



To overcome these problems, instead of *concatenating* the vectors of words, we *add* them. Hence, each group of *W* words is still represented by a *N-length* vector. Adding vectors results at vectors with values greater than 1. For instance, if a noun phrase has, among others, exactly two words that are tagged as adjectives, the *adjective* dimension will take the value 2. Notice that, by adding the vectors, rather than concatenating them, *word order information is lost*. This is an important difference between the two representations. The information loss might be acceptable within the NP itself, but it is still imperative to distinguish between words that are *inside* the NP and words that come before and after it. Thus, the final input vector is a concatenation of three *N-length* vectors, one for each group of words. Using this method, the vector has a constant length of ~150 dimensions. The vector dimensionality is still quite big, which is undeniably a drawback when training a neural network. However, it is a reasonable compromise between a meaningful representation and a fixed-length numerical vector form.

3.5 Overview of the Experiment

Once the identified noun phrases were encoded in the two different representations, we conducted experiments evaluating the performance of the two methods in the two corpora. As explained above, the NERC task in our experiments was formulated as a 3-class classification problem. The three classes were: *person*, *organisation* and *non-NE*. Thus, each algorithm was required to perform both parts of the NERC task simultaneously, i.e., identification and classification of NE phrases. For every experiment, 10-fold cross-validation was used in order to gain an unbiased estimate of the performance of the system under examination on unseen data. According to this evaluation method, the dataset is split into ten, equally-sized subsets and the final result is the average over ten runs. In each run, nine of the ten subsets of the data are used to train the learning algorithm while the tenth is held out for evaluation.

The measures that were chosen for the evaluation were those typically used in the language engineering literature: *recall* and *precision*. Recall measures the number of items of a certain type (e.g. organisation) correctly identified, divided by the total number of items of this type in the training data. Precision is the ratio of the number of items of a certain type correctly identified to all items that were assigned that particular type (e.g. organisation) by the system. In total, four measures are used: recall of organisations, recall of persons, precision of organisations and precision of persons.

Finally, as a basis for comparing the results in the experiments we can use the performance of the manually constructed systems on the same data. The results of the MITOS and VIE NERC systems, measured on the Kapa-TEL and MUC-6 corpora respectively, are shown in Table 1.

Table 1. Performance of the manually constructed NERC systems.

	MITOS NERC System		VIE NERC SYSTEM	
	Persons	***Organisations***	***Persons***	***Organisations***
Recall	76.5 %	84.2 %	84.97 %	69.25 %
Precision	87.5 %	89.8 %	92.50 %	83.42 %

Note that the results for the VIE system are significantly lower than the *aggregate* results presented for the various systems participating in MUC-6. This is due to the difficulty of identifying person and organisation names. The results are better for persons than for organisations. Person names are shorter and are usually either included in the gazetteers, or preceded by a person title. This fact makes their identification easier than for organisation phrases, which can be lengthy and may consist of words of various parts of speech and gazetteer types.

On the other hand, the performance figures of the MITOS system for persons are lower, compared to the performance for organisations. This is mainly attributed to the disproportionate distribution of the two classes in the corpus. As mentioned in section 3.1, the Kapa-TEL corpus contains 5837 organisation and only 695 person instances. As a result, the emphasis in the design of the NERC system was in the recognition and classification of organisation, rather than person names.

4 EXPERIMENTAL RESULTS

In our first experiment, we applied both methods on the English corpus. The results are shown in table 2.

Table 2. Results for the English corpus. **(*DT*:** decision tree with symbolic representation, ***NN*:** neural network, ***DT-N***: decision tree with numeric representation**)**

	English corpus	
	Persons	***Organisations***
DT Recall	90.40 %	87.85 %
NN Recall	90.61 %	86.21 %
DT-N Recall	89.35 %	86.98 %
DT Precision	93.23 %	80.43 %
NN Precision	94.73 %	83.33 %
DT-N Precision	93.05 %	86.43 %

Comparing the results in table 2 with the results of the VIE NERC system shown in table 1, we can conclude that both learned systems perform considerably better than the manually constructed system. Both recall and precision for persons and organisations are at higher levels than the VIE system. Improvements in recall for both persons and organisations are greater than the corresponding increases in precision, showing that the learned system was able to identify and classify correctly more NEs than the manually constructed system.

These results are very encouraging for the use of machine learning in the construction of NERC systems, especially if we consider that some of the VIE domain-specific resources, such as the NERC grammar, were specifically designed for the MUC-6 conference. This is an unexpected but interesting result, considering the amount of resources (especially in human effort) needed for the manual construction of a NERC grammar. The algorithms presented here require substantially less effort in the creation of the required resources. Thus, our results suggest that machine learning may be a suitable alternative for the automated construction of NERC systems.

If we compare the performance of the decision tree, using the symbolic data representation, to the performance of the neural network, we can conclude that both systems achieve comparable performance, with the neural network performing slightly better than the decision tree. However, the most interesting property of the neural NERC system is that it achieved higher performance than the decision-tree system having *less information* at its input, i.e., by ignoring the order of words in the noun phrase. This is an unexpected result, as our initial belief was that word order is important for NERC. In order to further examine whether this improved performance is due to the learning algorithm or to the data representation used, we contacted additional experiments. In these experiments, we retrained C4.5 with the same vectors used in the training of the neural network (numeric representation). From the results (also presented in table 2) we can see that the overall performance of C4.5 has improved, reaching a similar level as that of the neural network. The improvement is mainly in terms of the precision for organisations (~ 6 %), while all other measures remain practically unchanged. This is an interesting result, as decision trees are primarily designed to be used with symbolic features, but also unexpected, as the numeric representation ignores word order. These results indicate that the information reduction simplifies the learning task, thus improving classification performance.

In the second experiment, we applied the two methods on the Greek corpus. The results are presented in table 3.

Table 3. Results for the Greek corpus. (***DT*****:** decision tree with symbolic representation, ***NN*****:** neural network, ***DT-N*****:** decision tree with numeric representation)

	Greek corpus	
	Persons	***Organisations***
DT Recall	59.68 %	72.69 %
NN Recall	60.29 %	75.67 %
DT-N Recall	59.86 %	80.30 %
DT Precision	80.45 %	77.60 %
NN Precision	79.36 %	79.00 %
DT-N Precision	83.79 %	79.10 %

Comparing the results in table 3 with the results of the MITOS NERC system shown in table 1, we can conclude that the learned systems perform worse than the manually constructed system. This failure is primarily attributed to the linguistic pre-processing phase and mainly to the phrase chunker that is used in the recognition of noun phrases and which is unable to identify overlapping noun

phrases, as explained in section 3.2. As a result, the training data include only the longest possible noun phrases and not the noun phrases included in them, as is the case with the English noun phrase recogniser. Note that the MITOS NERC system does not use the chunker, but uses an alternative procedure, based on a regular grammar, for identifying possible NEs.

Regarding the comparison of the decision tree and the neural network, we observe a similar pattern as with the English corpus. Both systems achieve comparable performance, with the neural network doing slightly better than the decision tree, when the latter uses symbolic features. However, the use of numeric features improves the performance of C4.5, making it slightly better than the neural network. More specifically, when using the numeric representation, we observe an improvement in the performance of decision trees, according to all measures. The largest increase is in the recall of organisations (~ 7,5 %). These results are a further indication that the simpler numeric representation leads to the construction of better NERC systems.

5 CONCLUDING REMARKS AND FUTURE WORK

In this article we evaluated the behaviour of two supervised machine learning methods, decision-tree induction and multi-layered feed-forward neural networks, on the task of automatically constructing NERC systems from pre-tagged corpora. We have chosen these particular methods as representatives of the symbolic and subsymbolic families. The main advantage of symbolic methods over the subsymbolic ones is their ability to produce results that are easier for humans to understand and transform into more "traditional" formats, such as grammars. The fact that linguistic information is primarily symbolic is a further reason for using symbolic algorithms. On the other hand, the ability to understand and transform the results into other representations is not always important for practical applications, where classification performance might be of greater importance. By comparing a symbolic and a subsymbolic approach, we wanted to test whether the loss in comprehensibility for neural networks is balanced by a corresponding gain in classification performance. Our results do not support this hypothesis, since both methods achieved very similar performance in all of our experiments. In particular, when using a numeric representation of named-entity phrases, it is very hard to distinguish between the two methods, in terms of their performance.

However, what is most surprising is that the performance of the symbolic method improves with the use of the numeric representation. This is unexpected, not only because decision trees were originally designed to be used with symbolic features, but also because the particular representation that we have used ignores the order of the words within a named-entity phrase. Our results indicate that this information reduction simplifies the learning task, thus improving classification performance. This seems to be true, at least for the linguistic features (part-of-speech and gazetteer tags) that we have included in the representation.

In our experiments, we used two different evaluation corpora, containing Greek and English texts respectively. The performance of both learning methods was better for English than for Greek texts. This difference was primarily due to problems in the pre-processing language engineering tools that were used for Greek. Especially for the experiment with English texts, both of the automatically constructed NERC systems outperformed manually constructed grammars that were designed especially for the examined NERC task. This is an interesting result, which suggests that the NERC systems constructed with the use of machine

learning may be able to replace systems that require considerable manual effort in their construction.

An interesting issue for further investigation is the comparative evaluation of alternative symbolic and subsymbolic learning methods. The first set of candidates among the symbolic approach could be learning methods that use the same feature-vector representations as C4.5, e.g. AQ15 [Michalski *et al.* 1986] and CN2 [Clark & Niblett 1989]. An alternative family of methods could be those performing explicitly grammar induction [Langley 1987, Langley & Stromsten 2000, Lari & Young 1990]. These methods are able to construct grammars of different complexity from data. This is particularly interesting for NERC, which has traditionally been performed by parsers, using grammars. On the other hand, other numerical approaches could also be used. For instance, local type statistical classifiers, such K-nn and its variants, have proven effective in classifying multi-dimensional data.

Another equally interesting research direction would be to try to reduce or completely remove the need for manual tagging of the training data, through the use of unsupervised machine learning techniques [Mannes 1993, Kohonen 1989]. This could be of great benefit to the designer of a NERC system, as it would greatly improve the ability to adapt the system to different domains or even languages. Another way to reduce human involvement in the construction of a NERC system is by removing the need for gazetteer lists or by exploring techniques that acquire/update gazetteers out of raw corpora [Petasis *et al*. 2000].

ACKNOWLEDGMENTS

This work has been partially supported by the R&D projects MITOS “Document filtering, information extraction and data mining applied to financial news” and AYTONOMA “Automatic Acquisition of Named-Entity Recognition Grammars for Greek”. Both projects are funded by the Greek General Secretariat of Research & Technology.

REFERENCES

Bennett, S.W., Aone, C. and Lovell, C. “Learning to Tag Multilingual Texts Through Observation.” In *Proceedings of the Second Conference on Empirical Methods in NLP*, pp. 109-116, 1997.

Bikel D., Miller S., Schwartz R. and Weischedel R., 1997, Nymble: a High-Performance Learning Name-finder. *Proc. of 5th Conference on Applied Natural Language Processing*, Washington.

Borthwick A., Sterling J., Agichten E. and Grishman R., 1998, NYU: Description of the MENE named Entity system as Used in MUC-7. *Proc. of MUC-7*.

Clark P. and Niblett T., 1989, The CN2 algorithm. *Machine Learning*, 3(4), pp. 261-283.

Cowie J., 1995, Description of the CRL/NMSU System Used for MUC-6. *Proc. of MUC-6*.

Cucchiarelli A. and Velardi P., 1998, Finding a Domain-Appropriate Sense Inventory for Semantically Tagging a Corpus. *Int. Journal on Natural Language Engineering*.

Cucchiarelli A. and Velardi P, 1998, Using Corpus Evidence for Automatic Gazetteer Extension. *Proc. of Conf. on Language Resources and Evaluation*, Granada, Spain, 28-30 May 1998.

DARPA (Defense Advanced Research Projects Agency), 1995. Proceedings of the Sixth Message Understanding Conference (MUC-6), Morgan Kaufmann.

Day, D., Robinson, P., Vilain, M., and Yeh, A, 1998, Description of the ALEMBIC system as used for MUC-7. *Proc. of MUC-7*.

Farmakiotou D., Karkaletsis V., Koutsias J., Sigletos G., Spyropoulos C.D and Stamatopoulos P., 2000, Rule-based Named Entity Recognition for Greek Financial Texts. In *Proceedings of the Workshop on Computational lexicography and Multimedia Dictionaries (COMLEX 2000)*, pp. 75-78, Greece, September 22-23.

Hornik K., Stinchcombe M., and White H., 1989, Multilayer feedforward networks are universal approximators, *Neural Networks*, vol.2, pp.359-366.

Humphreys K., Gaizauskas R., Cunningham H., and Azzam S., 1997, VIE Technical Specifications. Department of Computer Science, University of Sheffield.

Karras D. A. and Perantonis S. J., 1995, An efficient constrained training algorithm for feedforward networks, *IEEE Transactions on Neural Networks*, **6**, 1420-1434.

Kohonen T., 1989, *Self-organisation and associative memory*. 3rd edition, Springer-Verlag, Berlin.

Langley P., 1987, Machine learning and Grammar induction. *Machine Learning*, v. **2**, pp. 5-8.

Langley P. & Stromsten, S., 2000, Learning context-free grammars with a simplicity bias. *In Proceedings of the Eleventh European Conference on Machine Learning*. Barcelona: Springer-Verlag.

Lari K. and Young S. J., 1990, The estimation of stochastic context-free grammars using the Inside-Outside algorithm. *Computer Speech and Language*, **4**.

Mannes C., 1993, Self-organising grammar induction using a neural network model. *In New trends in neural computation*, Lecture Notes in Computer Science, **686**, eds. J. Mira *et al*, pp. 198-203.

McDonald D., 1996, Internal and External Evidence in the Identification and Semantic Categorization of Proper Names. In *B. Boguraev & J. Pustejovski (eds.) Corpus Processing for Lexical Acquisition*, MIT Press, pp 21–39.

Michalski R. S., Mozetic I., Hong J. and Lavrac N., 1986, The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. *In Proceedings of the National Conference on Artificial Intelligence*, pp. 1041-1045.

Petasis G., Cucchiarelli A., Velardi P., Paliouras G., Karkaletsis V., Spyropoulos C.D., 2000, Automatic adaptation of Proper Noun Dictionaries through cooperation of machine learning and probabilistic methods. *In Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in information Retrieval*, July 24-28, Athens.

Perantonis S. J., Ampazis N. and Virvilis V., A Constrained Optimization Framework for Feedforward Neural Networks, *Annals of Operations Research*, in print.

Quinlan, J. R., 1983, Learning Efficient Classification Procedures and Their Application to Chess End Games, *In Machine learning: an artificial intelligence approach*, eds. Michalski, R.S., Carbonell, J.G. and Mitchell, T.M., Kaufmann, Palo Alto, CA, pp. 463-482.

Quinlan, J. R., 1993, C4.5: Programs for machine learning, Morgan-Kaufmann, San Mateo, CA.

Sekine S., 1998, NYU System for Japanese NE-MET2. *Proc. of MUC-7*.

Vilain, M., and Day, D., 1996, Finite-state phrase parsing by rule sequences. *Proceedings of COLING-96*, vol. 1, pp. 274-279.

---

[1] MITOS (EPET II – 1.3 – 102) is an R&D project on information filtering, extraction and data mining, funded partially by the Greek government. MITOS partners include NCSR "Demokritos" (coordinator), Athens Univ. of Economics & Business, Univ. of Piraeus, Univ. of Patras, KNOWLEDGE S.A., SENA and Kapa-TEL.

[2] The phrase chunker has been developed by the Wireless Communications Laboratory (WCL), Department of Electric and Computer Engineering, University of Patras.