Automatic adaptation of Proper Noun Dictionaries through cooperation of machine learning and probabilistic methods

Georgios Petasis, Alessandro Cucchiarelli(*), Paola Velardi(§),
Georgios Paliouras, Vangelis Karkaletsis, Constantine D. Spyropoulos

(*) Istituto di Informatica
Università di Ancona
Via Brecce Bianche, Ancona
alex@inform.unian.it

(§) Dip. di Scienze dell'Informazione,
Università di Roma 'La Sapienza'
Via Salaria 113, Roma
velardi@dsi.uniroma1.it

Software and Knowledge Engineering Laboratory
Institute of Informatics and Telecommunications,
National Centre for Scientific Research "Demokritos",
153 10 Ag. Paraskevi, Athens, Greece
e-mail: {petasis, paliourg, vangelis, costass}@iit.demokritos.gr

Abstract

The recognition of Proper Nouns (PNs) is considered an important task in the area of Information Retrieval and Extraction. However the high performance of most existing PN classifiers heavily depends upon the availability of large dictionaries of domain-specific Proper Nouns, and a certain amount of manual work for rule writing or manual tagging. Though it is not a heavy requirement to rely on some existing PN dictionary (often these resources are available on the web), its coverage of a domain corpus may be rather low, in absence of manual updating. In this paper we propose a technique for the automatic updating of a PN Dictionary through the cooperation of an inductive and a probabilistic classifier. In our experiments we show that, whenever an existing PN Dictionary allows the identification of 50% of the proper nouns within a corpus, our technique allows, without additional manual effort, the successful recognition of about 90% of the remaining 50%.

Keywords: information extraction, natural language processing for IR, machine learning and IR, text data mining

1 Proper Noun Classification

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 of the named entities involved in an event and the recognition of the relationships holding between named entities in that event. A named entity (NE) is a proper noun (PN), serving as a name for something or someone.

Named Entity Recognition (NERC) is the task of identifying and semantically tagging proper nouns (PNs) in running texts. In terms of syntactic categories, PNs are lexical noun phrases, consisting of primitive proper nouns (e.g. *Clinton*), groups of proper nouns of different semantic categories (e.g. *Vice Chairman James T. Sherwin)* and also of non-proper nouns (e.g. *Jamaica tourist board)*. In the latter case, capital letters are optional, making the problem of PN identification even more complex. A typical NERC system mainly consists of a dictionary and a grammar. The dictionary is a set of proper nouns that are known beforehand and have been classified into PN types. The grammar is used to recognize PNs that are not in the dictionary and to decide upon the final types of PNs in cases where ambiguity exists in the dictionary. The NERC task

plays an essential role in IE because all proper nouns (named entities) must be located in order to be used in the extracted events. The importance of NERC in IE is such that in the Message Understanding Conferences (MUCs) [9] [10] NERC is evaluated as a separate task.

The special status of the NERC task in information extraction is justified by the fact that in many sublanguages, PNs represent a significant percentage (30% or more) of the words in a corpus. NERC is therefore essential to the effective understanding of language, at least so that PNs can be recognised within their context as locations, products, persons, etc. The semantic information incorporated in a PN dictionary can be of great usefulness to several information retrieval tasks, like term-based information retrieval; i.e. if a user request (or part of) consists of words that form a PN, better results can be obtained if we require that these words appear consecutively in a document rather than returning documents that contain all the words in various (and possibly unrelated) locations. As a result, NERC takes on special significance in many applications, in which names play a key role, e.g. automated telephone call handling and information filtering for financial applications.

The performance of NERC systems has been evaluated in the MUC conferences and was reported to reach performance comparable to humans. Yet, since PNs are mostly domain-specific, there is no evidence that similar performance could be obtained in other languages and domains than those considered in the literature, if not at the price of a similar effort for the manual adaptation of the grammar and for the compilation of a high-coverage PN dictionary. Furthermore, the categories into which PNs are classified constitute semantic information that varies significantly in different thematic domains. For instance the identification of organisation names may be relevant in the domain of financial news, but not in the scientific literature.

The manual adaptation of PN dictionaries and contextual rules to a particular domain is very time-consuming and in some cases impossible, due to the lack of experts. Thus, the automatic acquisition/adaptation of these resources from corpora is highly desirable. In any case, since PNs form an open class, adaptable language engineering systems should provide automatic means to increase system robustness against unknown items.

The exploitation of learning techniques to support the adaptation of linguistic resources to domains and languages has recently attracted the attention of many researchers. Good results have been demonstrated with stochastic classifiers [3] [4] and decision-tree based inductive classifiers [17]. Machine learning techniques are classified 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 [3], Alembic [18] [11], and AutoLearn [6] are examples of systems exploiting supervised learning techniques. On the other hand, the NERC system developed for Italian [7] [8] is an example of a system exploiting unsupervised learning.

In this paper, we present a method that combines an inductive and a probabilistic classifier, in order to achieve high performance in adapting a possibly low-coverage Proper Noun Dictionary to a domain. Both classifiers use as initial knowledge source a list of PNs (dictionary), and cooperate at learning new instances and updating this dictionary.

The two classifiers operate in cascade.

In phase 1 we apply a supervised decision-tree learning algorithm to the task of classifying PNs in predefined categories. The learning algorithm that was used for this task is a general-purpose supervised machine learning algorithm, called C4.5 [16]. The aim of the learning process is to construct a decision tree that will classify PNs based on their structure and the environment they appear in.

In phase 2 an unsupervised corpus-driven statistical technique is employed to classify PNs not recognized in phase 1. Unknown instances are classified on the basis of a syntactic contextual model of PN semantic categories, learned on the basis of seed PN instances that are detected in phase 1.

The important advantage of our integrated classifier is that we achieve both high recall and precision, while posing limited requirements on the initial coverage of the available PN Dictionary.

2 Inductive learning of a PN classifier

In this section we examine the use of the learning algorithm C4.5 for the automated acquisition of PN categorisation rules. C4.5 is a supervised learning algorithm that performs *induction* of *decision trees*, i.e., it constructs decision trees from training data. The algorithm requires the training data to be provided in a *feature-vector* format, which is common in most work in symbolic machine learning. In this representation each PN is represented by a vector of values for a fixed set of features.

For the purpose of this experiment, we have decided to encode two features for each relevant word in the corpus. The first feature represents semantically enriched part-of-speech information, which includes the part of speech (POS) tag (e.g. adjective, possessive determiner, auxiliary verb) extended with a gazetteer[1] tag (e.g. city, country, organisation), when such information is available. In cases where a gazetteer tag is available, then this information supplements the POS information. The second feature represents additional morphological

[1] The gazetteers used in the experiment are general-purpose and should not be confused with the domain-specific PN dictionary to be extended by our method.

information, relative to the value of the first feature. This information includes the number for nouns and adjectives, tense, person and mood for verbs, person for pronouns etc. Note that the actual word form (or its root) is not included in the feature vector.

The learning algorithm demands all feature vectors to be of a fixed length. Thus, except from the choice of what to represent, we have to find a solution about how to represent it, as the length of PNs is not fixed. In order to transform a PN of variable length into a feature vector of fixed length, we have chosen to encode only part of the PN into the representation. This information is augmented with words in the close vicinity of the PN. For the purpose of this experiment, we have chosen to include the first two and the last two words of the PN, as well as two words before and two words after the PN (contextual information). Thus, each PN is represented by a vector of 16 features (8 words times 2 features each).

The way in which C4.5 constructs the decision tree from training data is of limited interest in this study and is only briefly mentioned here. C4.5 uses a greedy hill-climbing search through the space of possible decision trees aiming to construct one that explains well the data. 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, i.e., organisations, persons or locations. The quality of discrimination is assessed by an information-theoretic metric, based on *mutual information*. The same approach 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 PNs in the training set. However, it is expected to capture the most important classification rules.

The training data for C4.5 are constructed with the use of the initial low coverage PN dictionary. Using these data, the algorithm constructs a decision tree, which is applied to the PNs not covered by the dictionary and assigns a semantic class to some of them. The newly classified PNs, together with the unclassified ones are then fed to the probabilistic learning algorithm for further refinement.

3 Probabilistic learning of PN's Contextual Model

In this section we briefly summarize the corpus-based tagging technique for the classification of proper nouns that have not been categorized by the decision-tree classifier.

3.1 Learning contextual sense indicators

This second stage of our method starts by assuming that the decision-tree classifier has detected "some" examples of PNs in each semantic category. Then, through an *unsupervised* probabilistic technique, typical PN syntactic and semantic contexts are learned from a corpus. These contextual models are used to identify new PNs and extend the coverage of the PN dictionary.

The corpus to be used for learning needs to be morphologically processed. Then, a partial parser[2] [1] extracts *elementary syntactic relations* such as Subject-Object, Noun-Preposition-Noun, etc. An *elementary syntactic link* (hereafter *esl*) is denoted by:

$$esl_i(w_j, mod(type_i, w_k))$$

where w_j is the head word, w_k is the modifier, and $type_i$ is the type of syntactic relation (e.g. Prepositional Phrase, Subject-Verb, Verb-DirectObject, etc.).

In our study, the *context* of a word *w* in a sentence S is represented by the esls that include *w* as one of their arguments. The esls including semantically classified PNs as one of their arguments are grouped in a database, called **PN_esl**. This database provides contextual evidence to assign a category to unknown PNs. Another database, **UPN_esl**, includes all the esls with an unknown proper noun. Syntactic contexts can be generalized by replacing words by their hypernyms in WordNet [15].

3.2 Classifying unknown PNs

A corpus-driven algorithm is used to classify unknown proper nouns in **UPN_esl**:

- Let **U_PN** be an unknown proper noun, i.e., a single word or a complex nominal. Let $C_{pn} = (C_{pn1}, C_{pn2}, ..., C_{pnN})$ be the set of semantic categories for proper nouns (e.g. Person, Organisation, Product etc.). Finally, let **ESL** be the set of esls in **UPN_esl** that include **U_PN** as one of its arguments.
- For each esl_i in **ESL** let:

  $$esl_i(w_j, mod(type_i, w_k)) = esl_i(x, U_PN)$$

  where $x=w_j$ or w_k and $U_PN = w_k$ or w_j, $type_i$ is the syntactic type of esl (e.g. N-of-N, N_N, V-for-N etc), and furthermore let:

  $$pl(esl_i(x, U_PN))$$

  be the *plausibility* of a detected esl. The plausibility is a measure of the statistical evidence of a detected syntac-

[2] Shallow, or partial parsers are a well established technique for corpus parsing. Several partial parsers are available in literature, and some are also freely downloadable.

tic relation [2] [13] that depends upon *local* (i.e. at the sentence level) syntactic ambiguity and *global* corpus evidence. Plausibility accounts for the *uncertainty* arising from syntactic ambiguity. Roughly, the *local* plausibility is proportional to the inverse of the number of *colliding* syntactic interpretations in a sentence. At the *global* (corpus) level, identical esls are merged, and their plausibility values are cumulated. In general, correct interpretations cumulate higher evidence, while noise tent to be more sparse.

- Finally, let:
  - $\mathbf{ESL_A}$ be a set of esls in **PN_esl** defined as follows: for each esl_i(x, U_PN) in **ESL** put in $\mathbf{ESL_A}$ the set of esl_j(x,PN_j) with $type_j$=$type_i$, x in the same position as esl_i, and PN_j a known proper noun, in the same position as U_PN in esl_i,
  - $\mathbf{ESL_B}$ be the set of esls in **PN_esl** defined as follows: for each esl_i(x, U_PN) in **ESL** put in $\mathbf{ESL_B}$ the set of esl_j(w,PN_j), with $type_j$=$type_i$, w in the same position as x in esl_i, Sim(w,x)> ε, and PN_j a known proper noun, in the same position as U_PN in esl_i. Sim(w,x) is a similarity measure between x and w. In our experiments, Sim(w,x)> ε holds if w and x have a common hypernym *H* in WordNet. The generality of *H* (i.e. the number of intermediate levels L between x and *H*) is a free parameter to which we assign different values in order to analyze the effect of generalization.
- For each semantic category C_{pnj} compute evidence (C_{pnj}) as:

$$evidence(C_{pnj}) = \alpha \frac{\sum_{esl_i \in ESL_A, C(PNj)=C_{pnj}} weight_{ij}(x) \cdot D(x, C(PN_j))}{\sum_{esl_i \in ESL_A} weight_{ij}(x) \cdot D(x, C(PN_j))} + \beta \frac{\sum_{esl_i \in ESL_B, C(PNj)=C_{pnj}} weight_{ij}(w) \cdot D(w, C(PN_j))}{\sum_{esl_i \in ESLB} weight_{ij}(w) \cdot D(w, C(PN_j))}$$

where:

- $$weight_{ij}(x) = weight_{ij}(esl_i(x, PNj)) = pl(esl_i(x, PNj)) \cdot \left(1 - \frac{amb(x) - 1}{2k - 1}\right)$$
- $$weight_{ij}(w) = weight_{ij}(esl_i(w, PNj)) = pl(esl_i(w, PNj)) \cdot \left(1 - \frac{amb(w) - 1}{k - 1}\right)$$
- $pl(esl_i(x, PN_j))$ is the plausibility and $amb(esl(x, PN_j))$ is the ambiguity (according to WordNet) of x in esl_i.
- k is a constant factor used to incrementally reduce the influence of ambiguous words. Smoothing is tuned to be higher in $\mathbf{ESL_B}$.
- α and β are parametric, and can be used to study the evidence provided by $\mathbf{ESL_A}$ and $\mathbf{ESL_B}$.
- D(x, C_{pnj}) is a discrimination factor used to determine the *saliency* [19] of a context esl_i (x, _) for a category C_{pnj}, i.e., how good a context is at discriminating between C_{pnj} and the other categories[3].

The selected category for U_PN is

$$C = argmax(evidence(C_{pnk}))$$

When grouping all the evidence of a U_PN in the corpus, the underlying hypothesis is that, in a given application, a PN has a *unique sense*. This is a reasonable restriction for PNs, supported by empirical evidence, though we would be more sceptical about the applicability of the one-sense-per-discourse paradigm [12] to generic words. We believe that it is precisely this restriction that makes the use of syntactic and semantic contexts appealing.

In simple terms (details are given in [7], [8]), the *evidence* formula estimates the probability that the syntactic contexts around an U_PN co-occur with PNs belonging to a given category. The first term computes the (weighted) relative frequency, in semantic category $C_{pnj,}$, of contexts *identical* to those occurring with the U_PN, while the second term computes the (weighted) relative frequency of contexts "*similar*" to those occurring with the U_PN. Notice that in the above formula the frequency of contexts in categories is smoothed by several factors: the *plausibility* lessens the weight of syntactically ambiguous contexts; the *discrimination* factor strengthens the weight of contexts that are *typical* of a certain category, i.e. very frequent in that category and rare in others; the *ambiguity* lessens the weight of semantically ambiguous contexts. All these factors are intended to cooperate at reducing the influence of unreliable or uninformative contexts. The formula has also parameters (k, α, β), estimated by running systematic experiments. Standard statistical techniques have been used to balance experimental conditions and analyse the sources of variance. These experiments are discussed in section 4.

Figure 1 shows an example of UP_N tagging.

Notice in figure 1 that we used the 8 MUC-7 semantic categories, with the addition of *product*. However, only the first three categories are considered in the experiment described in the next section. These three categories were generally considered the hardest to recognize in the MUC-7 competition. The other categories are usually captured with full success by domain-independent rules.

[3] For example, a Subject_Verb phrase with the verb *make* (e.g. Ace made a contract.) if found almost with equal probability with Person and Organization names. We used a simple conditional probability model for D(x, C_{pnj}), but other well known statistical measures of "sparseness" could be used (for example, the Dice factor or the Entropy).

```
U_PN:   British_Gas

1.00 G_N_V_Act British_Gas 0 1 nil close
-----------------------------------------
ESLA= 1.00 G_N_V_Act Aetna 1 1 nil close
ESLA= 1.00 G_N_V_Act Alcoa 1 1 nil close
ESLA= 1.00 G_N_V_Act Xerox 1 1 nil close
ESLB= 1.00 G_N_V_Act Eastern 1 1 nil fill
ESLB= 1.00 G_N_V_Act Japanese 9 1 nil fill
ESLB= 1.00 G_N_V_Act Philip_L_._Hall 3 1 nil fill
ESLB= 1.00 G_N_V_Act Brooks_Brothers 1 1 nil shut
ESLB= 1.00 G_N_V_Act Ford 1 1 nil shut

1.00 G_N_V_Act British_Gas 0 1 nil finish
-----------------------------------------
ESLA= 1.00 G_N_V_Act China 2 1 nil finish
ESLA= 1.00 G_N_V_Act Communications 1 1 nil finish
ESLB= 1.00 G_N_V_Act Coniston 1 1 nil complete
ESLB= 2.00 G_N_V_Act Corp_. 1 1 nil complete
ESLB= 1.00 G_N_V_Act Donaldson 1 1 nil complete
ESLB= 1.00 G_N_V_Act Soviets 1 1 nil complete
ESLB= 1.00 G_N_V_Act Sterling 2 1 nil complete
ESLB= 1.00 G_N_V_Act Syms_Corp_. 1 1 nil complete
…<other esls follow >…

α: 0.7000, β: 0.3000

 NUM      CATEGORY      ESLA     ESLB    EVID
 ----   -------------  ------   ------   -----
   1    ORGANISATION    66.80   114.31   0.54
   2    LOCATION         6.98    15.93   0.06
   3    PERSON           5.98    43.27   0.09
   4    DATE             0.00     5.98   0.02
   5    TIME             0.00     0.00   0.00
   6    MONEY            0.00     0.00   0.00
   7    PERCENT          0.00     0.00   0.00
   8    PRODUCT          0.00     0.00   0.00
   9    OTHERS          37.90    31.77   0.26

Max evidence category: ORGANISATION
```

Figure 1: An example of U_PN tagging.

4 Experimental Discussion

4.1 Overview of the experiment

For the purpose of this experiment, we have used part of the Wall Street Journal (WSJ) corpus. The WSJ corpus contains documents that spread over a broad range of thematic domains. The subset of the corpus used for the purpose of this experiment included instances of the 8 types of PN, mentioned above, but we only used 3 PN categories.

The corpus was pre-processed with the help of the VIE information extraction system [14]. In VIE PNs are in part detected using a large gazetteer (PN dictionary) of complex and simple proper nouns. The PNs not detected by the gazetteer are recognised by a specialised grammar.

As a first pre-processing stage, PNs were recognized as a syntactic category by the Brill's POS tagger [5], a now widely used resource. Brill's tagger performance on unknown PNs has been enhanced by adding simple heuristics to detect also complex nominals: for example, a PN can also be a list of adjacent capitalised words, or capitalised words with interleaved prepositions[4]. Then every PN was classified into a semantic category according to the results obtained from VIE. We used one half of the PNs recognised by the Brill tagger to train C4.5 and the remaining half to test the performance of our method. The decision tree classified some of these unknown PNs into the three semantic categories, leaving the others unclassified. The probabilistic learning algorithm was then used to assign categories to as many of the unclassified PNs as possible. The final set of classified PNs was used to update the initial gazetteer and evaluate the performance of our method.

The distribution of PNs in the three semantic categories is shown in Table 1. The numbers correspond to counts of different PNs, rather than their instances in the WSJ corpus. Roughly 50% of the PNs are included in the initial dictionary, while the remaining 50% is used as test data.

PNs covered by the dictionary (training data)		**PNs not covered by the dictionary (test data)**	
Persons	2397	Persons	2390
Organisations	1460	Organisations	1462
Locations	316	Locations	316
Total	4173	Total	4168

Table 1. Distribution of PNs in the three semantic categories for the training and test data.

4.2 Learning the decision-tree classifier

As described above, the initial low coverage dictionary was applied to the corpus in order to create the training set of feature vectors for the decision tree-learning algorithm (C4.5). For every PN in the dictionary, all instances of this particular PN were located in the corpus and for each located instance one (or more, due to ambiguity) feature vector was created. The feature vectors were of fixed length: for every PN the vector contained features from the first and last two words of the PN, as well as features from the two words before and after the PN. Finally the correct category of the PN, according to the dictionary, was included in the feature vector.

The word features encoded in the vector for each word were: semantically enriched part-of-speech information and additional morphological information. Details for these features are given in section 2. In the case of feature absence, usually due to absence of a word at the specific position, a special feature value ('?') was used. This special character is interpreted as missing information by C4.5. In the case of ambiguity, either in the semantically enriched part-of-speech or in the additional morphological information, more than one feature vectors were created in order to cover all possible combinations of the ambiguous values. A typical example of a PN and the resulting vectors are given in Figure 2:

[4] Performance of PN recognition is lower than for common words, but still close to 90% precision.

Sentence: Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.
Identified as PN (pre-processing stage): Pierre Vinken
Dictionary category: person
Features assigned by the pre-processing stage

Word	Semantically Enriched POS Information	Additional Morphological Information
Pierre	proper_noun	singular
	proper_noun_city	invariable
Vinken	?	?
,	punct_comma	invariable
61	adjetive_card-inal_number	invariable
	common_noun	invariable

Resulting Feature Vectors:
Word order: second word to the left (of the PN), first word to the left, first word (of PN), second word, penultimate word, last word, first word to the right, second word to the right.
Two features for each of the 8 words, plus the dictionary category at the end of the vector.]

1) ?,?,?,?,proper_noun_city,invariable,?,?,proper_noun_city,invariable,?,?,punct_comma,invariable, adjective_cardinal_number,invariable,person.
2) ?,?,?,?,proper_noun_city,invariable,?,?,proper_noun_city,invariable,?,?,punct_comma,invariable,common_noun, invariable,person.
3) ?,?,?,?,proper_noun,singular,?,?,proper_noun,singular,?,?,punct_comma,invariable,adjective_cardinal_number, invariable,person.
4) ?,?,?,?,proper_noun,singular,?,?,proper_noun,singular,?,?,punct_comma,invariable,common_noun,invariable,person.

Figure 2. Encoding of a training vector for C4.5.

The learning algorithm was then trained, using all the feature vectors that were created from the dictionary. The algorithm induced a decision tree to be used for the classification of the PNs in the test data. The vectors in the test data contained initially no semantic category and were classified by the decision tree. The classification was accompanied by a confidence figure, in the range of [0..1], with the confidence level increasing as this figure increases. All classifications which had a confidence value below some threshold were removed and considered unclassified by the decision tree. The threshold that we used for this experiment was 0.7. This value was determined empirically, by examining the behaviour of the classifier for different threshold values on the training data. The outcome of the classification was a list of test vectors, each of which was associated either with one of the three semantic categories (person, organisation, location) or with the label "unknown", meaning that it was not classified by the decision tree. In some cases, more than one vector corresponded to the same PN, due to two reasons: (1) each PN may have several instances in the corpus and (2) each PN instance may be represented by several vectors due to ambiguities, as shown above. In those cases the most frequent semantic category was selected for each PN.

Eventually, 3233 of the 4168 PNs in the test set were classified by the decision tree, 3040 of them correctly, while 935 PNs remained unclassified. The use of a high confidence threshold (0.7) provided a bias for precision, leading to high precision (94.3%) in the first phase. At the same time recall was also at an acceptable level (72.94%). The motivation for the precision bias is the fact that the PNs that are classified by the decision tree are then used as training data in the second phase. Thus, classification in the first phase should be very precise, in order to avoid noise in the training data of the second phase. However, the PNs that remain unclassified in the first phase are really the hard cases, making the refinement task in the second phase, especially hard. In particular, 477 out of the 935 unclassified PNs appear only once in the text. Such a low frequency rate makes probabilistic classification very hard.

4.3 Probabilistic classification of unknown proper nouns

The 935 PNs not recognized by the PN classifier in Phase 1 were fed to the untrained probabilistic classifier. A contextual model for the three PN categories was learned by the classifier using all available classified PNs (i.e., the same initial list of PNs used as training data by the inductive classifier, plus the PNs that were classified by the inductive classifier at the end of the first phase). For this experiment, we used the following parameters: α=0.7 β=0.3 and generalization level L=1 (i.e., only one level of generalization)[5].

[5] The selected parameter values have given the best results in previous empirical studies.

Table 2 summarizes the results of the complete experiment[6].

Person								
Phase	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
1	1767	2390	73.93%	1797	98.33%	521	-	-
2	308	521	59.12%	452	68.14%	59	92.26%	86.82%
Organisation								
Phase	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
1	1005	1462	68.74%	1100	91.36%	384	-	-
2	262	384	68.23%	336	77.98%	48	88.23%	86.67%
Location								
Phase	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
1	268	316	84.81%	336	79.76%	30	-	-
2	15	30	50.00%	22	68.18%	8	79.05%	89.56%
Total								
Phase	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
1	3040	4168	72.94%	3233	94.03%	935	-	-
2	585	935	62.57%	810	72.22%	125	89.66%	86.97%

Legends

Phase 1: Decision tree classifier

Phase 2: Probabilistic Contextual classifier

- **A**: PNs correctly tagged at the end of Phase X (X=1,2) in the Test Corpus
- **B**: Total Unknown PNs in the Test Corpus *before* Phase X
- **C**: Local Recall after Phase X (A/B)
- **D**: Total PNs detected *at the end* of Phase X
- **E**: Local Precision after Phase X (A/D)
- **F**: Total PNs still unknown at the end of Phase X
- **G:** Global Precision (Phase 1 + Phase 2)
- **H**: Global Recall (Phase 1 + Phase 2)

Table 2. Experimental results.

The performance of the probabilistic classifier in isolation is somewhat lower than this reported in [8], however in that experiment the test set followed the same distribution of phenomena as in the domain corpus, while in this experiment the statistical classifier is requested to tag the "hardest" cases, those for which C4.5 could not output a decision with a sufficient confidence level.

Despite the difficulty of the task, recall increases substantially in the second phase, reaching 86.97%, while precision remains high at 89.66%. The combined F-measure is 89,13, which would position our method among the highest-performing ones in the MUC competitions. This result is particularly encouraging, given that we are dealing only with the hardest three PN categories and assume the existence of only a low-coverage (50%) PN dictionary.

[6] Clearly the recall is also affected by errors in PN recognition by the POS tagger.

5 Conclusions

Current methods for Proper Noun (PN) recognition and classification perform well, but are admittedly sensitive to domain and language shifts. Though it may be possible to produce (especially using the Web) an initial dictionary of domain-specific PNs, often to achieve an adequate coverage some non-trivial amount of manual work is necessary, for dictionary extension, rule writing and manual tagging of texts.

In this paper we presented an integrated classification method for the automatic extension of a PN dictionary. The method combined two learning approaches: supervised learning of decision-tree classifiers and unsupervised probabilistic learning of syntactic and semantic context. The supervised learning algorithm used the information in the initial PN dictionary and a training corpus to construct a decision tree that assigned semantic categories to PNs, which were not in the initial dictionary. Only high-confidence classifications were accepted, imposing a bias for high precision. Despite this fact, performance in terms of recall was also good. In a second phase, unsupervised learning was used to increase recall and assign a semantic category to those PNs that were still unclassified. In this phase, we investigated the effectiveness of using syntactic contexts and semantic generalization for categorizing unknown PN in running text. Similar techniques have been applied (alone or in combination) to the more general task of word sense disambiguation, with no clear-cut results. In the case of PNs, however, certain favourable conditions (especially the applicability of the one-sense-per-domain hypothesis) favour the good performance of this technique. Remarkably, our combined method achieved both high recall (86.97%) and precision (89.66%), while imposing limited requirements on the coverage of the initial PN dictionary. These results are among the highest reported in the literature and were achieved on three PN categories that are considered hard to recognize: person, organization and location. But the main advantage of our method is the low initial requirement, that is a 50% coverage PN dictionary, while all MUC systems require more or less heavy manual work for rule writing or text tagging.

The results presented here suggest that the combined method that we presented is appropriate for PN recognition. However, we need to evaluate the method further and compare it directly to some of the existing high-performing systems, possibly on the data used for the MUC comparisons. Before doing that, though, we would like to improve on the simple method that we are currently using for the identification of PNs in text. This step is essential, in order to construct a complete NERC system. An alternative direction that we are examining is to apply the method to other problems, such as word sense disambiguation. If the good performance that we achieved on PN recognition carries over to these other problems, we might be able to propose it as a more general method for sense tagging.

6 References

[1] Basili, R., Pazienza M.T., Velardi P., A (not-so) shallow parser for collocational analysis. *Proc. of Coling '94*, Kyoto, Japan, 1994.

[2] Basili, R., Marziali A., Pazienza M.T., Modelling syntax uncertainty in lexical acquisition from texts. *Journal of Quantitative Linguistics*, vol.1, n.1, 1994.

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

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

[5] Brill, E., Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging, *Computational Linguistics*, vol. 21, n. 24, 1995.

[6] Cowie, J. Description of the CRL/NMSU System Used for MUC-6. In [DARPA 1995].

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

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

[9] Defense Advanced Research Projects Agency. *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, Morgan Kaufmann.

[10] Defense Advanced Research Projects Agency. *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, Morgan Kaufmann.

[11] Day, D., Robinson, P., Vilain, M., and Yeh, A. Description of the ALEMBIC system as used for MUC-7. In [DARPA 1998].

[12] Gale, W. K. Church and D. Yarowsky. One sense per discourse. *Proc. of the DARPA speech and Natural Language workshop*, Harriman, NY, February 1992

[13] Grishman, R., J. Sterling, Generalizing Automatically Generated Selectional Patterns. *Proc. of COLING '94*, Kyoto, August 1994.

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

[15] Miller, George A., WordNet: a lexical database for English. *Communications of the ACM* 38 (11), November 1995, pp. 39 - 41

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

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

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

[19] Yarowsky D., Word-Sense disambiguation using statistical models of Roget's categories trained on large corpora. *Proc. of COLING 92*, Nantes, July 1992.