NAMED ENTITY RECOGNITION IN GREEK TEXTS WITH AN ENSEMBLE OF SVMS AND ACTIVE LEARNING[*]

GIORGIO LUCARELLI, XENOFON VASILAKOS and ION ANDROUTSOPOULOS

Department of Informatics
Athens University of Economics and Business
Patission 76, GR-104 34 Athens, Greece

We present a freely available named-entity recognizer for Greek texts that identifies temporal expressions, person, and organization names. For temporal expressions, it relies on semi-automatically produced patterns. For person and organization names, it employs an ensemble of Support Vector Machines that scan the input text in two passes. The ensemble is trained using active learning, whereby the system itself proposes candidate training instances to be annotated by a human during training. The recognizer was evaluated on both a general collection of newspaper articles and a more focussed, in terms of topics, collection of financial articles.

Keywords: Named-entity recognition; information extraction; Support Vector Machines; machine learning; active learning.

1. Introduction

Named-entity recognizers (NERs) identify occurrences of entity names in texts, and classify them in predefined categories (e.g., names of persons, organizations, dates). Named-entity recognition is important in information extraction,[1] where systems typically identify in documents relationships and events involving named entities, and many other natural language processing and information retrieval applications. In question answering,[2,3] for instance, entity names in the document collection are candidate answers to questions of the corresponding types (requests for person names, organizations, etc.); and in machine translation or cross-language information retrieval, special transliteration processes can be applied to entity names across languages with different alphabets,[4] provided that the names have been identified.

Although NERs that employ mostly hand-crafted rules[5,6] may perform very well, NERs that use statistical and machine learning techniques, including Hidden Markov or Maximum Entropy Models,[7,8,9,10] decision tree learning and/or boosting,[11,12,13] and Support Vector Machines,[14,15] usually outperform them and they are easier to port to new text genres (e.g., biomedical, instead of news articles), where new name categories (e.g., protein names) may also need to be supported. However,

[*] Preprint of an article submitted for consideration to the International Journal on Artificial Intelligence Tools. (c) 2007 copyright World Scientific Publishing Company. See `http://www.worldscinet.com/ijait/ijait.shtml`.

supervised statistical and machine learning-based NERs still require a tedious manual annotation phase, during which humans must tag occurrences of entity names in a training corpus. This problem can be alleviated with *active learning*, also known as *selective sampling*, whereby the learning algorithm itself selects and presents for human annotation only training instances it expects to improve its performance. Active learning, in various flavours, has been exploited in several text processing tasks, such as text classification,[16,17,18,19] part-of-speech tagging,[20] chunking,[21,22] parsing,[23,24] and information extraction.[23]

More directly relevant to the topic of this paper is the work of Becker et al.,[25,26] who employed active learning to train an English NER for astronomy documents. Becker et al. select iteratively from a pool of candidate, non-annotated training examples the new training examples to be presented for human annotation, by measuring for each candidate example the degree of disagreement between two different classifiers. The two classifiers are trained on all of the previously selected (and manually annotated) training examples using the same learning algorithm, a conditional Markov model, but each classifier uses only half of the feature set. Candidate training examples for which the two classifiers disagree most are preferred, and this allows the NER to reach the same accuracy level with fewer training examples, compared to *passive learning*, i.e., random selection of training examples. More generally, one can measure the disagreement of n different classifiers; the classifiers can be created by using different learning algorithms, feature sets, or training data, and they can be thought of as members of a committee.[27,17,20] The committee-based approach, however, can be computationally expensive, because it requires training n classifiers, and then obtaining n opinions on each candidate example.

An alternative active learning approach is to select training examples for which a single classifier is most uncertain.[28,18,19] This scheme is particularly suitable to binary classification with Support Vector Machines (SVMs),[29,30,31] where the classifier is most uncertain for instances that lie close to its separating hyperplane and are, thus, more likely to be support vectors. Vlachos[22] adapted this approach to multi-class English named-entity recognition (and chunking), by using an ensemble of SVMs (one SVM per name category) as a single multi-class learner and by experimenting with different possible ways to combine the distances from the hyperplanes of all the SVMs when selecting training examples. Shen et al.[32] carried out similar work on an English NER, taking into account not only the distances from the hyperplanes of the SVMs, but also measures intended to avoid selecting training examples that are very similar to already selected ones (diversity) and to promote candidate examples that are similar to many other examples that have not been selected (representativeness). However, Shen et al. trained independent binary classifiers for each category of entity names, which does not preclude the possibility of classifying an expression into multiple categories (e.g., as both person and organization) and does not allow active learning to consider the uncertainty of all the classifiers when selecting examples. To the best of our knowledge, no other researchers have examined active learning in NERs to date.

Research on NERs is dominated by work that targets texts written in English and other widely spoken languages. In this paper, we describe a freely available NER for Modern Greek texts, which identifies temporal expressions, person names, and organization names.[a] All other previously published work on Greek NERs that we are aware of relies on hand-crafted rules or patterns,[34,35,36] and/or decision tree induction with C4.5;[37,38] the only exception is the work of Michailidis et al., where SVMs, Maximum Entropy, Onetime[39] and manually crafted post-editing rules were employed. Note that tools such as part-of-speech taggers and chunkers, which are often used in English NERs, are more difficult to obtain in Greek, and this led us to design a NER that does not require them. Also, publicly available Greek training corpora for NERs do not exist, which provided further motivation to pursue active learning. Developing a Greek NER is further complicated by the fact that most Greek person names and some organization names are inflected for case, and some surnames also for gender.

As in English, Greek temporal expressions are relatively easy to identify, and a simple approach that relies on semi-automatically produced regular expression patterns turned out to perform adequately. In contrast, recognizing person and organization names is much more complicated, and we addressed it by using an ensemble of four SVMs.[b] Two of them scan the input text to identify person and organization names, respectively, in a first pass. The other two SVMs then re-scan the text, taking into consideration the decisions of the first pass. As in stacking,[42] the second-pass classifiers can learn to correct mistakes of the first pass, an effect that was demonstrated in earlier named-entity recognition experiments.[43] Additional classifiers, for example trained with different algorithms, could also be used in the first pass, at the expense of additional computational cost, to form a committee presided by the second pass.[44,45,46] Unlike previous stacking for named-entity recognition, however, when our second pass classifies each token of the text, it does not have access only to the tag(s) that the first pass assigned to that token. It also consults information showing whether or not the first pass classified that token or any of its neighbors as persons or organizations *anywhere else* in the same text with high confidence; this allows our NER to identify re-occurrences of names in contexts that make them more difficult to recognize.

Our two-pass approach was inspired by Edinburgh University's MUC-7 NER.[47] In that system, however, the multiple passes were implemented in a radically dif-

[a] The software can be downloaded from `http://www.aueb.gr/users/ion/`. An earlier version of our NER did not support organization names.[33]

[b] We use the LIBSVM[40] implementation of SVMs, including LIBSVM's grid-search parameter-tuning utility. Based on the results of Vlachos[22] in named-entity recognition, we use a radial basis function (RBF) kernel; see Vlachos for related discussion. Using a different SVM per name category is the simplest way to handle multi-category classification with SVMs; it also allows us to use a different feature set per name category. Another possibility is to use one SVM for each pair of name categories, as in the work of Michailidis et al.,[41] but this leads to a very large number of SVMs as the number of name categories increases.

ferent way, using gradually more permissive hand-crafted transduction rules and consulting a Maximum Entropy name-matching component before moving on to a more permissive set of transduction rules. Our two passes are also different from the approach whereby a first phase identifies all entity names and a second one categorizes them.[15] Furthermore, unlike the system of Shen et al., our ensemble acts as a single classifier with non-overlapping categories. In active learning, we select training examples for each pass by considering the distances from the hyperplanes of both SVMs of that pass, much as in Vlachos. Our ensemble, however, differs from that of Vlachos in that it performs two passes, using a very different feature set, and targets a different language. Our two passes are trained on separate data, and we show that this allows us to continue improving the system's performance by adding training data to the second-pass SVMs with active learning, when the training set of the first-pass SVMs has become too large for them to process in reasonable time.

The performance of our NER was evaluated on two corpora: a general collection of Greek newspaper articles, and a more focussed, in terms of topics, collection of Greek financial articles. Section 2 describes further our NER and motivates the design choices that underly it. Section 3 presents our experimental results. Section 4 concludes and highlights directions for further work.

2. System description

At run time, i.e., when using the trained system on new texts, named-entity recognition proceeds in three stages: preprocessing, temporal expression recognition, and recognition of person and organization names. The names of persons and organizations are identified in a single stage, which employs two passes, as discussed above. This section considers in turn the three stages.

2.1. *Preprocessing and classification task*

Both during training and at run time, the system first applies to the texts a simplistic tokenizer, which treats any non-alphanumeric character as a separate token; for example, “Η κ. Γ.Α. Νικολάου-Παπαδάκη δήλωσε στις 13/12/98...” becomes ‘Η’, ‘κ’, ‘.’, ‘Γ’, ‘.’, ‘Α’, ‘.’, ‘Νικολάου’, ‘-’, ‘Παπαδάκη’, ‘δήλωσε’, ‘στις’, ‘13’, ‘/’, ‘12’, ‘/’, ‘98’, ... Words containing both Greek and Latin characters are also split; for instance, “Euroγνώση” becomes ‘Euro’, ‘γνώση’. The texts of our experiments were all initially in HTML; any HTML tags are removed, after marking tokens that are immediately before end-of-paragraph tags as last tokens of sentences. An SVM-based sentence splitter is also applied to locate punctuation symbols that end sentences, as opposed to punctuation that signals, for example, abbreviations. Following Bikel et al.,[8] named-entity recognition is then viewed as the task of assigning each token to one of the name categories (in our case, temporal expression, person name, or organization name) or the not-a-name category. That is, unlike other NERs,[10] we do not use special categories for the first tokens of person names and organizations. This reduces the number of categories and, thus, simplifies the classification task, but it has the disadvantage that we cannot distinguish between adjacent names of the same category. For example, in the Greek translation of "the sister of John Smith Mary Rose said", all four capitalized tokens would be classified as person names, and there would be no indication of the boundaries of the two names. In its final output, our NER marks adjacent tokens of the same name category as single names, and this would produce wrongly: "the sister of `<enamex type='person'>` John Smith Mary Rose`</enamex>` said".[c] Such cases, however, are very rare, to the extent that they can be ignored.

2.2. *Temporal expression recognition*

Temporal expressions are recognized using patterns, which are produced semi-automatically from the training data as follows. First, all the manually tagged temporal expressions are retrieved from the training corpus. The retrieved expressions are then generalized by replacing all numbers with regular expressions and by substituting tokens by pre-defined token types, such as `month`, `sep`(arator), `special`, `article`. For instance, "12 December 2005" becomes "`[0-9]{2} month [0-9]{4}`", "Easter of 68" becomes "`special article [0-9]{2}`", and "12.1.67" becomes "`[0-9]{2} sep [0-9]{1} sep [0-9]{2}`". (In "Easter of 68" the corresponding Greek expression contains an article instead of "of". We show many examples in English for the benefit of readers who do not speak Greek.) We use 13 token types; for each token type, there is a list that specifies which tokens (in all inflected forms) belong to that type. The token types and the lists are created manually and they may have to be modified when moving to texts of a different origin or genre, but otherwise the generation of the patterns is automatic, which is why we call the process semi-automatic.

Generalized expressions that differ only in numeric sub-expressions are then combined by creating disjunctions of the numeric sub-expressions. For instance, "`[0-9]{2} sep [0-9]{1} sep [0-9]{2}`" and "`[0-9]{1} sep [0-9]{2} sep [0-9]{4}`", which may derive from "12.1.67" and "1-11-2005", are replaced by "`([0-9]{2}|[0-9]{1}) sep ([0-9]{1}|[0-9]{2}) sep ([0-9]{2}|[0-9]{4})`". Finally, the resulting patterns are sorted by length. At run time, if multiple temporal patterns apply, we use the longest (more specific) one. It is also possible to impose a frequency threshold, to discard patterns that are too rare in the training corpus, and to add hand-crafted patterns, but these options of the system were not used in our experiments.

The simplistic generalization approach presented above performs very well with temporal expressions, which is why we do not use a more elaborate learning method instead, unlike person and organization names, where initial experiments showed the simplistic approach to be inadequate.

[c] We follow the guidelines of MUC-7, adapted to Greek, as to which expressions should be considered temporal, person names, or organization names, and we use similar XML tags; consult `http://www.itl.nist.gov/iaui/894.02/related_projects/muc/`.

2.3. *Person and organization name recognition*

The names of persons and organizations are identified in a single, third stage, as already mentioned. This stage is invoked after the patterns of the previous section have been applied to identify the temporal expressions. It is assumed that all of the temporal expressions have been identified correctly, an assumption our experiments show to be reasonable. Hence, in this stage the NER classifies all of the tokens that have not been identified as parts of temporal expressions into three non-overlapping categories: person name tokens, organization name tokens, or none.

2.3.1. *Sure-fire rules*

The SVM of the first pass that identifies person names has to classify each token as person or non-person. Similarly, the SVM that identifies organization names has to separate organization tokens from non-organization ones. Person tokens, however, are much fewer than non-person ones, and similarly there are many more non-organization tokens than organization ones. Hence, each SVM of the first pass is faced with a grossly imbalanced binary classification problem, and the same applies to the SVMs of the second pass. This imbalance is problematic, because it leads the SVMs (and most supervised learning algorithms) to learn to classify all tokens in the majority class, in our case non-persons and non-organizations, respectively. To reduce the imbalance, we employ simplistic 'sure-fire' rules; there are separate sure-fire rules for persons and organizations, with minor differences. At run time, tokens that satisfy the sure-fire rules are classified as non-persons or non-organizations, respectively, without consulting the corresponding SVMs; and the SVMs are trained only on examples of tokens that do not satisfy the sure-fire rules (and have not been tagged as temporal expressions). Preliminary experiments indicated that the ratio of person to non-person tokens is initially approximately 1:42. After removing temporal-expressions and tokens satisfying the sure-fire rules of persons, it becomes 1:3.5, and only 0.2% of the removed tokens are person names. In a similar manner, the ratio of organization to non-organization tokens is reduced from 1:40 to 1:4.5 with the same error rate. By "removed tokens" we mean that the SVMs are not invoked to decide upon their categories, and that these tokens do not give rise to training instances of the SVMs. They are not literally removed from the texts, however, and the SVMs may well examine features of the "removed tokens" when classifying other neighboring tokens.

The sure-fire rules classify as non-persons and non-organizations all numeric tokens, all punctuation and other non-alphabetic symbols, tokens that do not start with a capital letter, as well as tokens ending in "–ωνω", "–μαι", "–σαι", and other suffixes that are highly indicative of Greek verb forms. They also classify as non-persons or non-organizations, respectively, stop-words that are rare in names of the corresponding types. The sure-fire rules of persons use more stop-words than those of organizations, because some stop-words that are rare in person names occur frequently in organization names (e.g., "Bank of Greece"). Another difference between the sure-fire rules of persons and organizations is that the latter make some exceptions, and do not classify as non-organizations tokens like "υπουργείο" (ministry) and "χρηματιστήριο" (stock-exchange), even if they do not start with capital letters, because these tokens are often written with a lower-case first letter in organization names. A further complication is that the sure-fire rules of persons are not applied to tokens directly preceded by other tokens that have been hand-tagged (during training) or classified (at run time) as persons, and similarly for the sure-fire rules of organizations. This arrangement is needed in cases like the abbreviated person name "Ευ. Παπανούτσος", where the full stop is part of the name, even though it is a punctuation symbol.

2.3.2. *First pass*

At run time, the two SVMs of the first pass are invoked to classify the tokens of the input text (from left to right) that have not been classified as temporal expressions and do not satisfy the sure-fire rules. The SVM implementation that we use returns a confidence score in $[-1, 1]$ for each instance (in our case, token) being classified, which shows the SVM's certainty that the instance belongs to the positive class (in our case, person or organization, respectively); negative scores indicate that the SVM classifies the instance in the negative class (non-person or non-organization). If the persons SVM classifies a token in its positive class and the organizations SVM classifies it in its negative class, we take the decision of the first pass to be that the token is (part of) a person name, and similarly for organization names. If both SVMs classify a token in their positive classes, we follow the decision of the SVM with the highest confidence. Finally, if both SVMs classify a token in their negative classes, we take the decision of the first pass to be that the token is neither person nor organization.

Each token to be classified by the two SVMs of the first pass is represented as a vector containing features of that token and its context. Actually, the two SVMs use slightly different feature sets, and, hence, there are two different vector representations of each token to be classified, one for each SVM. In the remainder of this sub-section we focus on these vector representations. Henceforth, t_0 denotes the token to be classified, and t_i the $|i|$-th token to the right (positive i) or left (negative i) of t_0.

The first-pass SVM for persons uses 165 features, while the one for organizations uses 170. The features of the two SVMs are summarized in Tables 1 and 2, respectively; a bullet indicates that the corresponding feature is used, and a zero or an empty cell that it is not. For example, both SVMs use seven Boolean features (no. 1–7) that indicate whether or not $t_{-3}, \ldots, t_3$ are commas. They also use 14 features (no. 36–49) that examine if $t_{-3}, \ldots, t_3$ are written in Greek or Latin characters; this information is useful, because foreign person names and organization names are often written in Latin characters in Greek texts. The sizes of the bullets are approximately proportional to the *information gain* scores of the features in use, i.e., to the expected reduction of the decision uncertainty (entropy) that we gain by using the corresponding feature, as opposed to using no features at all.[d] For instance, features 8–14, which check for full stops around t_0, turn out to be more valuable in terms of information gain to the SVM for persons than to the SVM for organizations. Features with near-zero information gain scores ($< 10^{-9}$) are marked with zeros in Tables 1 and 2.

The features of the two SVMs were selected from a manually constructed set of candidate features, which contained a candidate feature for each one of the non-empty cells of Tables 1 and 2. For example, there was initially also a candidate feature (no. 32) of the SVM for persons that checked if t_0 was a number, but it was discarded during the feature selection process. Feature selection was performed using a manually tagged training corpus of approximately 200 randomly selected newspaper articles. (This is part 1 of corpus 1 to be discussed in Section 3.1 below, and it corresponds to approximately 17.000 training instances for each SVM.) We divided that corpus in two equally large parts. The first one was used to compute the information gain scores of Tables 1 or 2, respectively. It was also used to train each SVM using the m features with the highest information gain scores, with m ranging from 60 to 175. For each value of m, we measured the performance of the SVMs on the instances of the second part of the corpus, and eventually kept the m values that led to the best performance. (More precisely, performance was measured in terms of F-measure, to be defined in following sections.) This led to $m = 165$ for the persons SVM, and $m = 170$ for the organizations SVM.

Examining the information gain scores of the discarded candidate features revealed that the feature selection process discarded exactly those candidate features with near-zero information gain scores. We note that selecting directly all of the candidate features whose information gain was above a threshold would have led to the question of how to determine the best threshold value. It would also have provided no indication of whether or not some selected features are redundant given the features with higher scores. Our feature selection process shows that this is not the case: removing any number of the worst (lowest information gain) selected features leads to worse performance on the second part of the corpus.

Returning to Tables 1 and 2, features 64–70, which measure the lengths in characters of t_{-3}, …, t_3 are examples of numeric features. All numeric features are normalized to $[-1, 1]$.[e] In the case of features 64–70, the value -1 corresponds to a length of 1 character, $+0.9$ to 10 characters, and $+1$ to more than ten characters. Features 71–84 show if $t_{-3}, \ldots, t_3$ have prefixes or suffixes that are common in Greek surnames, such as "Παπα–", "Καρα–", "Χατζη–", "–ακης", "–ιδης" and "–οπουλος"; we use about a dozen of each. Features 85–91 employ a list of 350 common Greek first

[d] A formal definition of information gain can be found in the literature.[48] We computed the information gain scores using WEKA,[49] available from http://www.cs.waikato.ac.nz/∼ml/weka/.

[e] See, for example, the documentation of LIBSVM[40] for a discussion of why normalizing all numeric features to the same interval is beneficial.

Table 1. Features of the first-pass SVM for **persons**, with information gain (IG) scores.

no.	feature descriptions	t_{-3}	t_{-2}	t_{-1}	t_0	t_1	t_2	t_3	values
1–7	comma?	•	·	●	●	•	•	•	B
8–14	full stop?	•	·	●	●	●	•	·	B
15–21	dash?	·	·	·	·	·	·	·	B
22–28	slash?	·	0	·	0	·	·	0	B
29–35	number?	·	·	·	0	•	·	·	B
36–42	Greek characters?	•	●	·	•	•	•	·	B
43–49	Latin characters?	·	·	·	•	·	·	·	B
50–56	first character capital?	•	•	•	●	●	•	·	B
57–63	all characters capital?	·	●	●	●	•	·	·	B
64–70	length in characters	●	●	●	●	●	•	•	n
71–77	common surname prefix?	·	•	•	●	•	•	·	B
78–84	common surname suffix?	•	•	•	●	●	•	·	B
85–91	common first name?	●	●	●	●	•	0	•	n
92–98	common last character?	•	·	•	•	•	•	·	B
99–105	common sing. adj. ending?	·	·	•	●	•	•	·	B
106–112	plural noun/adj. ending?	·	•	•	•	•	•	•	B
113–119	common sing. gen. ending?	·	·	·	·	·	·	·	B
120–126	ends in final sigma?	·	·	•	•	·	·	·	B
127–133	distance from “A.E.” etc.	•	·	·	·	0	0	0	n
134–140	starts with “ministry” etc.?	·	·	·	•	·	0	·	B
141–147	is last token of sentence?	·	·	●	•	·	·	·	B
148–154	part of article’s title?	·	·	·	•	•	·	·	B
155–161	distance from start of name				●				B
162	directly preceded by Mr(s)?				●				B
163	preceded by plural Mr(s)?				●				B
164	in P_{1-2}^{t-1} list?			●					n
165	in P_{3-4}^{t-1} list?			•					n
166	in $P_{>4}^{t-1}$ list?			●					n
167	prev. tokens in $P_{1-2}^{t-7,\ldots,t-1}$				●				n
168	prev. tokens in $P_{3-4}^{t-7,\ldots,t-1}$				•				n
169	prev. tokens in $P_{>4}^{t-7,\ldots,t-1}$				•				n
170	in R_{1-2}^{t-1} list?			●					n
171	in R_{3-4}^{t-1} list?			●					n
172	in $R_{>4}^{t-1}$ list?			·					n
173	prev. tokens in $R_{1-2}^{t-7,\ldots,t-1}$				•				n
174	prev. tokens in $R_{3-4}^{t-7,\ldots,t-1}$				•				n
175	prev. tokens in $R_{>4}^{t-7,\ldots,t-1}$				0				n

Note: B: Boolean; n: numeric; 0 : $IG = 0$; · : $0 < IG \leq 0.01$; • : $0.01 < IG \leq 0.1$; ● : $IG > 0.1$.

names that we extracted from a Greek calendar’s name days. For example, the value of feature 88 is 1 if t_0 is present in that list. Otherwise, the value of that feature indicates the length of the longest prefix of t_0 that matches the corresponding prefix of the alphabetically closest entry of the list, normalized to $[-1, 1]$ (-1 for no match, 1 for the longest possible match). For instance, if t_0 is “Μιλτιάδου” and the closest entry of the list is “Μιλτιάδης”, the longest matching prefix is 7 characters long. This partial matching allows us to capture, to a large extent, inflectional variants of Greek person names.

Table 2. Features of the 1st pass SVM for **organizations**, with information gain (IG) scores.

no.	feature descriptions	t_{-3}	t_{-2}	t_{-1}	t_0	t_1	t_2	t_3	values
1–7	comma?	•	•	●	●	•	•	•	B
8–14	full stop?	●	•	•	•	•	●	●	B
15–21	dash?	•	•	•	•	•	•	•	B
22–28	slash?	•	•	•	•	•	•	0	B
29–35	number?	•	•	•	•	•	•	•	B
36–42	Greek characters?	⬤	⬤	⬤	●	•	●	•	B
43–49	Latin characters?	•	●	●	⬤	●	●	●	B
50–56	first character capital?	●	●	●	●	●	⬤	●	B
57–63	all characters capital?	•	•	•	●	•	●	●	B
64–70	length in characters	⬤	⬤	⬤	⬤	0	0	0	n
71–77	common surname prefix?	•	•	•	●	•	•	•	B
78–84	common surname suffix?	•	•	•	●	●	●	•	B
85–91	common first name?	⬤	⬤	⬤	⬤	●	●	0	n
92–98	common last character?	⬤	⬤	⬤	●	•	●	●	B
99–105	common sing. adj. ending?	●	●	●	⬤	•	•	•	B
106–112	plural noun/adj. ending?	•	●	●	•	•	•	•	B
113–119	common sing. gen. ending?	•	•	●	●	●	•	•	B
120–126	ends in final sigma?	●	⬤	⬤	•	•	•	•	B
127–133	distance from “A.E.” etc.	●	●	●	●	●	●	●	n
134–140	starts with “ministry” etc.?	•	•	•	●	●	•	•	B
141–147	is last token of sentence?	●	●	⬤	●	•	•	•	B
148–154	part of article’s title?	•	•	•	●	●	●	●	B
155–161	distance from start of name				⬤				B
162	directly preceded by Mr(s)?				⬤				B
163	preceded by plural Mr(s)?				●				B
164	in P^{t-1}_{1-2} list?			⬤					n
165	in P^{t-1}_{3-4} list?			⬤					n
166	in $P^{t-1}_{>4}$ list?			●					n
167	prev. tokens in $P^{t_{-7},\dots,t_{-1}}_{1-2}$				⬤				n
168	prev. tokens in $P^{t_{-7},\dots,t_{-1}}_{3-4}$				⬤				n
169	prev. tokens in $P^{t_{-7},\dots,t_{-1}}_{>4}$				●				n
170	in R^{t-1}_{1-2} list?			⬤					n
171	in R^{t-1}_{3-4} list?			⬤					n
172	in $R^{t-1}_{>4}$ list?			●					n
173	prev. tokens in $R^{t_{-7},\dots,t_{-1}}_{1-2}$				⬤				n
174	prev. tokens in $R^{t_{-7},\dots,t_{-1}}_{3-4}$				⬤				n
175	prev. tokens in $R^{t_{-7},\dots,t_{-1}}_{>4}$				⬤				n

Note: B: Boolean; n: numeric; 0 : $IG = 0$; • : $0 < IG \leq 0.01$; ● : $0.01 < IG \leq 0.1$; ⬤ : $IG > 0.1$.

Features 92–98 show if t_{-3}, …, t_3 end in “–ς” (final sigma), “–ν”, or a vowel, as most modern Greek words do, or not; if not, this is an indication that the corresponding token may be an abbreviation, as in the person name “Ανδρ. Παπανδρέου” or the company name “Νικ. Ι. Θεοχαράκης Α.Ε.”. There is also a set of features (120–126) that checks if $t_{-3}, \dots, t_3$ end in “–ς” in particular, which is very common in masculine Greek first names. We would have also liked to include part-of-speech information, but we had no Greek part-of-speech tagger available at the time of our experiments. Instead, in earlier work[50] we assessed the information gain (with respect to the SVM for person names) of features corresponding to common endings of Greek nouns, adjectives, etc. Based on that assessment, we included (as candidate features) in the work of this paper features 99–105, which check for some common singular adjective endings (e.g., "–κος", "–κου"), features 106–112, which look for common plural noun and adjective endings (e.g., "–οι", "–ους"), and features 113–119, which look for common singular genitive endings (e.g., "–ου", "–ης").

Features 127–133 check for abbreviations of legal types of companies that are common in Greek organization names (e.g., "Α.Ε.", "Ε.Π.Ε.") in a window of ±10 tokens around each one of $t_{-3}, \ldots, t_3$. The values of these features depend on the distance of the closest legal type abbreviation from t_i: if the abbreviation starts 10 tokens or more before t_i, the value is -1; if it starts 10 tokens or more after t_i, the value is 1; and if it starts in a window ±10 around t_i, the feature takes a value in $(-1, 1)$ proportional to its distance. There are also features (134–140) that look for words like "υπουργείο" (ministry) or "χρηματιστήριο" (stock-exchange), which are often parts of organization names.

Features 141–147 check if $t_{-3}, \ldots, t_3$, respectively, is the last token of a sentence, as determined by the sentence splitter and the end-of-paragraph HTML tags. This information is useful, because it flags, for example, full stops that do not end sentences, which are often parts of abbreviated names. Features 148–154 show if $t_{-3}, \ldots, t_3$ are parts of the title of a news article, which is again useful, because different writing conventions are often used in titles (e.g., all capitals, no full stops); in our experiments, the titles could be identified easily from the HTML tags of the original texts. In features 155–161, we examine the distance of t_0 from the first token of a continuous sequence of person tokens (in the case of the persons SVM) or organization tokens (in the case of the organizations SVM) that directly precede t_0; for example, in the person name "Γεώργιος – Αλέξανδρος Μαγκάκης" the distance of the last token from the first one is 3. These features (155–161) are all Boolean, and they show if the distance is $\leq 1, 2, 3, \ldots, 7$ tokens. They provide information that is useful to estimate how likely it is for t_0 to continue a preceding multi-token person or organization name. The seven features could have been replaced by a single numeric one, but using multiple Boolean features instead allowed us to confirm, by computing their information gain scores, that examining up to seven tokens to the left of t_0 still provides useful information. The information gain scores of all seven features correspond to bullets of the same size, which is why we show a single bullet for all seven features per SVM in Tables 1 and 2.

Features 162 and 163 indicate whether or not t_0 is directly preceded by "κ." ("Mr."/"Mrs." in Greek), and whether or not t_0 is preceded by "κ.κ." or "κκ." (plural of "κ.") in a window of 10 tokens to the left of t_0. Notice that these features turn out to be very useful to the organizations SVM too, because if a token is preceded by an abbreviation of this type, this is strong evidence that the token is part of a person name, which helps the organizations SVM classify it correctly to its negative class. Similar comments can be made for other features, such as 85–91, which look for common first names. Observations of this kind led us to provide initially all candidate features to both SVMs, even though some candidate features appear to be targeting exclusively person or organization names.

We also use 6 numeric features (164–169) that check the degree to which t_0 is preceded by tokens that occur frequently before person tokens in the training corpus of each experiment. During training, we construct 6 lists: $P^{t_{-1}}_{1-2}$, $P^{t_{-1}}_{3-4}$, $P^{t_{-1}}_{>4}$, $P^{t_{-7},\ldots,t_{-1}}_{1-2}$, $P^{t_{-7},\ldots,t_{-1}}_{3-4}$, and $P^{t_{-7},\ldots,t_{-1}}_{>4}$. $P^{t_{-1}}_{1-2}$ stores all the tokens of 1 or 2 characters that occur immediately before person tokens in the training corpus. Similarly, $P^{t_{-1}}_{3-4}$ and $P^{t_{-1}}_{>4}$ store all the tokens of 3-4 characters or more, respectively, that occur immediately before person tokens in the training corpus. The three $P^{t_{-1}}$ lists are mostly used to collect titles like “sir” and “general”, in Greek, that often precede directly person names; the reason for using three seperate $P^{t_{-1}}$ lists, depending on the lengths of the tokens, is explained below. The three $P^{t_{-7},\ldots,t_{-1}}$ lists are similar, but they store tokens of the corresponding lengths that occur anywhere up to 7 positions before person tokens in the training corpus; they are used to collect words like “president” or “director”, which often precede person names, but not necessarily directly, as in “the president of France J. Chirac”. In all six lists, the tokens are stored along with their frequencies in the training corpus. A frequency threshold of 3 is applied to the $P^{t_{-1}}$ lists, i.e., tokens that do not occur at least three times in the training corpus immediately before t_0 are discarded. For the $P^{t_{-7},\ldots,t_{-1}}$ lists, the threshold is 10.

When generating the feature vector of t_0, the three $P^{t_{-1}}$ lists give rise to three numeric features (164–166), whose values are the frequencies of t_{-1}, as stored in the corresponding lists and normalized to $(-1, 1]$, or -1 if t_{-1} is not present in the corresponding list. That is, the values of the three features indicate how frequently the particular t_{-1} token occurs directly before person tokens in the training corpus. We use three separate $P^{t_{-1}}$ lists and features, depending on the lengths of the tokens, because shorter words (e.g., articles) tend to be more frequent in texts than longer words are, and, hence, finding at run time at position t_{-1} a short word that occurs at that position with a high frequency in the training texts may be less informative than finding at the same position a longer word that occurs with the same frequency in the training texts.

Similarly, the three $P^{t_{-7},\ldots,t_{-1}}$ lists, give rise to 3 additional numeric features (167–169). When constructing the feature vector of t_0, the value of each one of these three features is the sum of the frequencies of the particular tokens $t_{-7}, \ldots,$ t_{-1} that precede t_0, as recorded in the corresponding lists, normalized to $[-1, 1]$. A feature value of 1 corresponds to a sum-value equal to, or greater than twice the average sum value, as estimated from the training corpus, -1 corresponds to a zero sum-value, and intermediate sum-values are distributed uniformly in $(-1, 1)$. When computing the values of the three features, if any of $t_{-7}, \ldots, t_{-1}$ does not belong to the same sentence as t_0, that token is ignored, i.e., its frequency does not contribute to the sum. This is because, for example, the occurrence of “president” in a previous sentence provides no indication that t_0 is a person token, even if the two tokens are close. Similar restrictions apply to other features[50].

In a similar manner, we construct 6 more lists, namely $R^{t_{-1}}_{1-2}$, $R^{t_{-1}}_{3-4}$, $R^{t_{-1}}_{>4}$, $R^{t_{-7},\dots,t_{-1}}_{1-2}$, $R^{t_{-7},\dots,t_{-1}}_{3-4}$, and $R^{t_{-7},\dots,t_{-1}}_{>4}$, which store information about the tokens that occur before organization tokens in the training corpus. These lists give rise to features 170–175, as with the P lists of features 164–169. The sizes of the bullets in Tables 1 and 2 show that most of the features that use the P and R lists are particularly valuable in terms of information gain. Note that in Tables 1 and 2 we show, for simplicity, features 167–169 and 173–175 as operating on t_0, whereas in reality they examine $t_{-7}, \dots, t_{-1}$.

By using the P and R lists, we can check the neighborhood of t_0 for tokens that accompany frequently entity names, without increasing excessively the size of the feature set. In contrast, Vlachos[22], for example, uses thousands of Boolean features, each signalling if a particular token is present at a particular position within a window of ±2 tokens around t_0; using such a large feature set, however, increases significantly the training and classification times of the SVMs. Our P and R lists are similar to the lists of "triggers" of Zhou and Su[10], where the triggers appear to be tokens that accompany frequently entity names of a particular category. However, the trigger lists of Zhou and Su do not seem to record the frequencies of the triggers in the training corpus, and they are manually post-processed, whereas the construction of our P and R lists is fully automatic.

2.3.3. *Second pass*

A person or an organization token may occur several times in a text, and context may make some of its occurrences (e.g., "Mr. M. Liapis", "the director of Forthnet S.A.") easier to identify than others ("According to Liapis...", "Forthnet announces..."). Consequently, the first pass may have classified some occurrences of a token as, say, persons and some other occurrences of the same token as non-persons. However, if the first pass has classified an occurrence of a token (e.g., "Liapis") as person with very high confidence, then any other occurrence of the same token in the same text is probably also (part of) a person name. Furthermore, if a token (e.g., "Michalis") is accompanied by another token (e.g., "Liapis" in "Michalis Liapis") that the first pass has classified anywhere in the text as person with very high confidence, then this is an indication that the first token ("Michalis") may also be part of a person name. Similar observations apply to tokens that the first pass has classified as organizations with very high confidence.

To capture these observations, at run-time, once the first pass is complete, we create two sets S_p and S_o of all the tokens that the first pass has classified as person and organization names, respectively, anywhere in the text with confidence greater than 0.9; the reader is reminded that the SVM implementation that we use returns confidence scores in $[-1, 1]$. We then use the second-pass SVMs to re-classify all of the token occurrences of the input text, again from left to right, skipping temporal expressions and tokens that satisfy the corresponding sure-fire rules. For each token occurrence, the decision of the second pass is taken by examining the confidence scores of the two second-pass SVMs, as in the first pass, and this is considered to be the final decision of the NER. The two SVMs of the second pass use the features of the corresponding SVMs of the first pass, and the additional features of Tables 3 and 4. There are 42 and 47 additional features, respectively, for person and organization names. The additional features were selected from an initial set of 56 candidate features, which correspond to the cells of Tables 3 and 4, using the same feature selection process as in the first pass. This time, the feature selection process discarded all the candidate features whose information gain scores were below 0.001; these are marked with zeros in Tables 3 and 4.

Table 3. Additional 2nd pass features for **persons**, with information gain (*IG*) scores.

no.	feature descriptions	t_{-3}	t_{-2}	t_{-1}	t_0	t_1	t_2	t_3	values
1–7	1st pass person confidence	0	●	•	⬤	●	●	•	n
8–14	1st pass org. confidence	•	●	●	●	●	●	•	n
15–21	in S_p?	0	•	0	●	•	•	•	B
22–28	in S_o?	0	●	●	0	●	●	0	B
29–35	distance from prev. person	●	●	0	•	•	•	•	n
36–42	distance from prev. org.	0	0	0	0	●	●	●	n
43–49	distance from next person	●	●	●	●	●	●	●	n
50–56	distance from next org.	0	0	0	●	●	●	●	n

Note: B: Boolean; n: numeric; 0 : $IG < 0.001$; • : $0.001 \leq IG \leq 0.01$; ● : $0.01 < IG \leq 0.1$; ⬤ : $IG > 0.1$.

Table 4. Additional 2nd pass features for **organizations**, with *IG* scores.

no.	feature descriptions	t_{-3}	t_{-2}	t_{-1}	t_0	t_1	t_2	t_3	values
1–7	1st pass person confidence	•	•	•	●	●	•	•	n
8–14	1st pass org. confidence	•	●	●	●	●	●	●	n
15–21	in S_p?	0	0	•	•	•	0	0	B
22–28	in S_o?	●	●	●	●	●	•	•	B
29–35	distance from prev. person	•	•	•	•	•	•	•	n
36–42	distance from prev. org.	•	●	●	●	●	●	●	n
43–49	distance from next person	•	•	•	0	0	0	0	n
50–56	distance from next org.	●	●	●	●	•	•	0	n

Note: All symbols have the same meanings as in Table 3.

Features 1–7 and 8–14 of Tables 3 and 4 are the confidence scores of the two first-pass SVMs, respectively, for $t_{-3}, \ldots, t_3$, normalized to $[-1, 1]$. The second-pass SVMs are trained after the training of the first-pass SVMs has been completed, and they are trained on texts whose tokens have been annotated with the confidence scores of the first-pass SVMs. Features 1–14, allow the second-pass SVMs to learn when and to what degree to trust the decisions of the first-pass SVMs. In that respect, the two passes can be seen as a form of stacking.[42] As already pointed out, however, our two passes differ from previous stacking approaches to named-entity recognition[43,44,45,46] in that when our second pass classifies a token (t_0), it does not have access only to the confidence scores of the first-pass SVMs (features 1–14) for that token. It also consults the S_o and S_p sets, through features 15–28, which show whether or not the first pass classified t_0 or any of its neighbors $(t_{-3}, \ldots, t_{-1}, t_1, \ldots, t_3)$ as persons or organizations *anywhere else* in the same text with high confidence. In that respect, our second pass is similar to a check that Zhou et al.[10] perform during the decoding phase of their Hidden Markov Model, where they examine if a token matches other names that have already been identified. That check, however, appears to be limited to previously identified names that *precede* t_0, and, hence, it cannot help exploit *later* occurrences of the same token in "easier" contexts. Furthermore, Zhou et al. do not seem to examine if the *neighbors* of t_0 have already been classified as entity names.

Some documents contain sequences of person or organization names. For instance, sports news may list the players of a team; and financial news may list the companies the form a consortium. To provide some indication of whether or not t_0 is within a sequence of names, features 29–35 and 36–42 show the distances of $t_{-3}, \ldots, t_3$ from the last token of the closest person or organization name, respectively, as identified by the first pass, that precedes t_i; and similarly features 43–49 and 50–56 show the distances from the first token of the closest person or organization name, respectively, that follows t_i. Tables 3 and 4 show that some of these features turn out to be useful.

Apart from helping our NER increase its effectiveness, the second pass has an additional benefit related to efficiency. As we add training instances to the SVMs of the first pass, their classification accuracy improves, but training them becomes significantly slower, to the extent that adding more training instances eventually becomes impractical, especially in active learning where the SVMs are retrained iteratively. At that point, we stop expanding the training set of the first-pass SVMs, and we start training the SVMs of the second pass. The training set of the second-pass SVMs is initially very small, which allows us to continue to add training instances. Nevertheless, the second-pass SVMs soon exceed the performance of the first pass, because their additional features allow them to exploit the experience of the first-pass SVMs. We discuss these issues further when presenting our experimental results.

2.3.4. *Active learning*

In a binary classification problem, an SVM uses in general non-linear functions to map the feature vectors to a new vector space of higher dimensionality. It then employs optimization techniques to locate a separating hyperplane in the new space, such that the hyperplane separates the training vectors of the two categories with the maximum margin, i.e., with the maximum distance between the closest training vectors of the two categories. The resulting optimal separating hyperplane is always in the middle of two other parallel hyperplanes, which are tangential to the training instances of the positive or negative category, respectively, as shown in the left part of Figure 1. The equation of the optimal hyperplane depends only on training vectors that lie on, and thus define, the two tangential hyperplanes; these vectors are called *support vectors.*

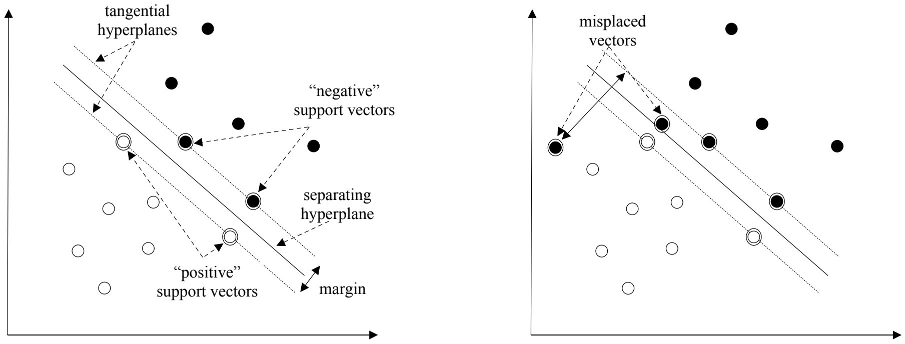

Figure 1. The hyperplane of an SVM with perfect (left) and imperfect (right) separation.

Mapping the original feature vectors to a space of higher dimensionality increases the chances that a separating hyperplane exists, i.e., that the problem becomes linearly separable in the new vector space. The learning problem, however, may remain linearly inseparable in the new space; or requiring perfect separation may lead to a separating hyperplane with a very small margin, which may not classify well new, unseen instances. Hence, the separating hyperplane is usually allowed to misplace a few training instances, i.e., some training instances are allowed to fall either inside the margin or on the wrong side of the hyperplane, as shown in the right part of Figure 1. The optimization process, then, looks for a hyperplane that separates the training vectors of the two categories with the maximum margin and the smallest total misplacement (distances of the misplaced training instances from the corresponding tangential hyperplanes). In that case, the support vectors, i.e., the training vectors that affect the equation of the separating hyperplane, are both those that lie on the tangential hyperplanes and those that are misplaced. The reader may wish to consult, for example, Cristianini and Shawe-Taylor[31] for a formal introduction to SVMs.

As already mentioned, the goal of active learning is to allow the system to select and present for human annotation only those candidate training instances that it expects to improve its performance. In the case of SVMs, adding new training vectors that are not support vectors does not affect the optimal hyperplane, as the new vectors are in effect ignored. Therefore, one should concentrate on adding training vectors that fall inside the margin (or on the tangential hyperplanes) and, hence, close to the separating hyperplane, or on the wrong side of the hyperplane. In the second case, if the SVM has already encountered a large number of training examples and the problem is linearly separable in the new vector space, training vectors that fall on the wrong side will also tend to be close to the separating hyperplane (the SVM will be close to classifying them correctly). Hence, in both cases one should concentrate on adding training instances whose vectors fall close to the separating hyperplane the SVM has learnt so far. Consequently, when selecting among candidate training instances, we prefer the instances that are closest to the current separating hyperplane, an approach that has also been employed in named-entity recognition by Vlachos[22] and Shen et al.[32] A more formal account of why this selection strategy is reasonable in SVMs, along with supporting experimental results from text classification, can be found elsewhere.[28,19]

A further complication is that in each pass we have two SVMs, one for person names and one for organization names, and the two SVMs are trained in parallel. Hence, when we assess each candidate training instance, we need to combine its distances from the current separating hyperplanes of both SVMs. Based on the results of Vlachos,[22] who experimented with several combination functions, we combine the two distances using the *min* function; i.e., we use the shortest of the two distances to assess the usefulness of each candidate training instance. The two distances are comparable, because the SVM implementation that we use normalizes them to $[-1, 1]$. Despite its simplicity, the *min* function was among the top performers in the named-entity recognition experiments of Vlachos.

Note that if a candidate training instance (token) satisfies the sure-fire rules of a name category (persons or organizations), we ignore the distance of the corresponding SVM, and we assess the instance based solely on the distance of the other SVM. Also, when a selected training instance is presented for human annotation, the annotated instance is added to the training data of both SVMs, regardless of which SVM's distance was used to select it. The instance will probably be more useful to the SVM with the shortest distance, but given that it can also be added to the training data of the other SVM without additional annotation effort, it may be a waste of training data not to do so; on the other hand, adding the instance to the training data of both SVMs increases the training time of both, which may not be justified in the case of the SVM with the longest distance. We plan to investigate this issue further in future work.

When training the pair of SVMs of the first or second pass, we assume that there is a large pool of texts (e.g., obtained from the archives of a newspaper). Most learning-based NERs are constructed by picking randomly some of the pool's texts, manually annotating them exhaustively, and then training the NERs on the annotated texts. In our case, this means annotating all of the tokens of the selected texts that do not satisfy the temporal expression patterns nor the sure-fire rules, and training the pair of SVMs on vectors representing the annotated tokens. We call this approach *passive learning*. In contrast, *active learning* in our case assesses repeatedly the remaining candidate training instances of the pool, i.e., the tokens that have not been annotated and do not satisfy the temporal expression patterns nor the sure-fire rules. In each iteration, it presents for human annotation a *batch* consisting of the most useful candidate training instances, and once the instances have been annotated, the pair of SVMs is retrained on all of the instances that have been annotated so far. The remaining candidate training instances are then re-assessed, and a new batch is formed.

Assessing the candidate training instances in each iteration requires computing their distances to the current separating hyperplanes of both SVMs. The distances are, roughly speaking, the confidence scores that the SVMs return during classification; hence, we need to classify all the candidate training instances of the pool, and this has to be repeated whenever the SVMs are retrained, because the hyperplanes change. For large pools, this becomes impractical. As a compromise, we divide the pool in ten parts, and whenever we need a new batch, we select in turn another part of the pool and limit the selection process to the candidate training instances of that part. This allows active learning to consider eventually all of the candidate training instances of the pool, while reducing the distance computations. See also Segal et al.[51] for an alternative approach to speed up the computations.

An important parameter is the *batch size*, i.e., how many new training instances we select before re-training the SVMs. The larger the batch size the fewer iterations and, hence, also re-trainings of the SVMs are necessary to reach the same total number of training instances, which may reduce the overall training time. On the other hand, the larger the batch size the more we risk selecting training instances that will be less useful once the other instances of the batch have been added to the training set. With large batch sizes we also risk including in the batch very similar training vectors, for example deriving from different occurrences of the same tokens in almost identical contexts. To address these issues, we experimented with two additional, more complex selection strategies, which combined the distances from the separating hyperplanes with measures of *diversity*. In the first of these strategies, called *local diversity*, we again filled each batch with the candidate training instances that were ranked most highly in terms of their distances from the hyperplanes, but we ignored candidate training instances that were very similar (in terms of cosine similarity) to any instance already in the batch. The second strategy, *global diversity*, was identical, except that we ignored candidate training instances that were very similar to any instance already in the current batch or any other previous batch. We also experimented with versions of these strategies that incorporated the notion of *representativeness*, i.e., the degree to which each candidate training instance is similar and, thus, representative of many other candidate training instances. Unlike the NER results reported by Shen et al.,[32] however, none of these more complex strategies led to improvements in effectiveness,[52] compared to using only the distances from the hyperplanes. Further discussion related to diversity and representativeness can be found in the literature.[53,17,18,19]

3. Corpora and experiments

This section presents the corpora that we used and our experimental results.

3.1. *Corpora*

We evaluated our NER using three corpora. The first one, called *corpus 0*, contains all the articles (ranging from politics and finance to sports) of the Greek newspapers "To Vima" and "Ta Nea" that were published from July 2000 to October 2001 (12,687 articles) and from March 2001 to July 2002 (9,250 articles), respectively.[f] The second corpus, *corpus 1*, consists of 400 randomly selected articles of corpus 0. It contains 331,000 tokens, 4,815 person and 4,265 organization names (counting once names consisting of multiple tokens), and 1,563 temporal expressions (possibly multi-token). The third corpus, *corpus 2*, consists of 715 financial articles from Greek financial newspapers, and, hence, is more focussed in terms of topics.[g] It contains 205,000 tokens, 1,046 person names, 4,067 organization names, and 1,244 temporal expressions (possibly multi-token). All tokens of corpora 1 and 2 were manually annotated as temporal expressions, person names, organization names, or none.

3.2. *Evaluating temporal expression recognition*

Temporal expression recognition was evaluated separately on corpora 1 and 2, using 10-fold cross-validation. In a 10-fold cross-validation, the corpus is divided in 10 parts, and the experiments are repeated 10 times, each time using a different part for testing and the remaining parts for training; the results are then averaged over the 10 iterations. The results of these experiments are shown in Table 5. Precision is defined as $\frac{TP}{TP+FP}$, recall as $\frac{TP}{TP+FN}$, and F_β as $\frac{(1+\beta^2)\cdot precision\cdot recall}{\beta^2\cdot precision+recall}$, where TP (true positives) and FP (false positives) are the numbers of tokens that are correctly or wrongly, respectively, classified as temporal expressions, and FN (false negatives) are the tokens that are wrongly classified as non-temporal expressions. Precision shows how certain we can be that a token classified as temporal expression is indeed a (part of a) temporal expression, whereas recall shows how many temporal expression tokens we manage to identify correctly. F_β is a combination of precision and recall; we use $\beta = 1$, which gives equal importance to precision and recall.

Table 5. Cross-validation for **temporal expressions**.

corpus	precision (%)	recall (%)	$F_{\beta=1}$ (%)
corpus 1	96.62	92.95	94.75
corpus 2	97.59	95.35	96.46

As can be seen in Table 5, temporal expression recognition performed very well in both cross-validation experiments, with slightly worse results, especially in terms of recall, in the first, more varied corpus. An error analysis in the first cross-validation showed that 49% of the false positives were numbers (e.g., 4-digit numbers that

[f] The articles were downloaded from the on-line archives of the two newspapers; consult `http://tovima.dolnet.gr/` and `http://ta-nea.dolnet.gr/`.

[g] Corpus 2 was created during project MITOS; see `http://iit.demokritos.gr/skel/mitos/`.

were taken to be years), 27% were organizations (e.g., “Athens 2004”), and 6% were locations (e.g., the suburb “Αγία Παρασκευή”, where “Παρασκευή” also means Friday); no person tokens were wrongly classified as temporal expressions. False negatives were difficult to categorize; they included temporal expressions that do not follow any characteristic formats, such as names of periods (e.g., “renaissance”).

3.3. *Evaluating person and organization name recognition*

Person and organization name recognition was evaluated with three sets of experiments. We first compared passive against active learning, using general newspaper articles (corpora 0 and 1) and only one pass. In a second set of experiments, we investigated the effect of adding a second pass, again using general newspaper articles; this time we used only active learning, since the first pass had demonstrated its benefits compared to passive learning. Then, in a third set of experiments, we explored the degree to which the system’s effectiveness improves when using financial news only (corpus 2), i.e., a corpus that is more focussed in terms of topics.

3.3.1. *Active vs. passive learning in the first pass*

In this experiment, we used only the first-pass SVMs to compare passive against active learning. The 400 manually annotated articles of corpus 1 were randomly divided in two parts, approximately 200 articles each, hereafter called *part 1* and *part 2*. First, part 1 was used to induce temporal expression patterns. Then, in passive learning the two SVMs were trained on an increasingly larger set of training vectors, which corresponded to the first n tokens of part 1 that did not satisfy the temporal expression patterns nor the sure-fire rules, with n ranging up to approximately 10,900. (We write “approximately”, because the exact number is slightly different in the two SVMs, due to differences in their sure-fire rules.) In contrast, in active learning the two SVMs were initially trained on the first 1,200 training vectors of passive learning; subsequently, increasingly more training vectors were added, corresponding to tokens that were selected from a large non-annotated pool of texts and were then annotated by a human, again up to a total of approximately 10,900 training instances. The pool consisted of 5,000 randomly selected articles of corpus 0 (2,500 from each newspaper), excluding the articles of corpus 1; we estimate that the 5,000 articles contained approximately 425,000 candidate training instances. In both passive and active learning, the SVMs were evaluated on part 2, which contains approximately 17,400 test instances, again excluding temporal expression tokens and tokens satisfying sure-fire rules. Precision, recall, and F-measure ($F_{\beta=1}$) are now defined as in Section 3.2, except that TP is now the number of correctly classified person or organization tokens, respectively, and similarly for FP and FN.

The results of these experiments are included in Figures 2–4; the second-pass (“SP”) curves are to be ignored for the moment. A first general observation that can be made by looking at the three diagrams is that organization names are more difficult to recognize compared to person names. This agrees with most published results in English named-entity recognition, and can be attributed to the larger variety of organization names (e.g., fewer standard token suffixes, wider variation in the length of multi-token names). A second observation, which can be made by looking at Figure 2, is that active learning outperforms passive learning in terms of F-measure in both name categories, although the difference is smaller in the case of organization names. Consequently, active learning requires fewer training instances to reach the same level of performance as passive learning, which confirms earlier results in named-entity recognition by Shen et al. [32] and Vlachos.[22]

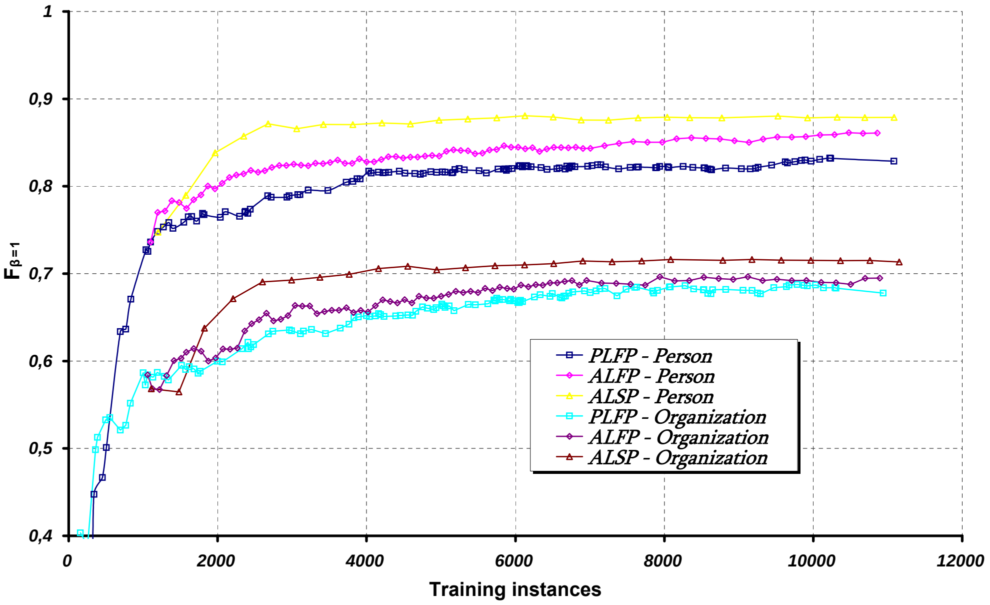

Figure 2. **F-measure** results on general newspaper articles, with active (AL) and passive learning (PL), during the first (FP) and second pass (SP).

Figures 3 and 4 show the corresponding precision and recall results; the error bars correspond to 0.95 confidence intervals. Figure 3 shows that active learning generally outperforms passive learning in terms of precision, both in person and organization names. As the training set becomes larger, though, the difference in precision between active and passive learning diminishes, especially in person names. In the case of person names, however, there is a growing difference in recall between active and passive learning, as can be seen in Figure 4, which is why active learning continues to dominate in terms of F-measure. A possible explanation for the larger recall of active learning, compared to passive learning, in person name recognition is that the training examples of active learning are drawn from a very large pool, which allows active learning to encounter during training, and learn to identify, a larger variety of person names. However, there is no significant difference in the recall of

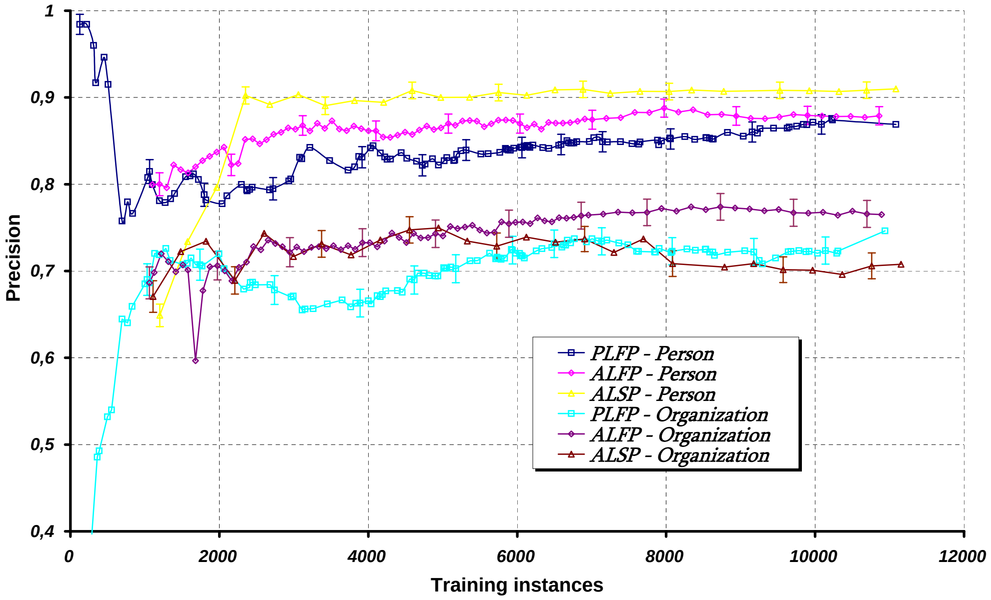

Figure 3. **Precision** results on general newspaper articles, with active (AL) and passive learning (PL), during the first (FP) and second pass (SP).

organization names between active and passive learning. We can only conjecture that the larger variety of organization names that active learning encounters during training has no significant effect, possibly because the organization names that are encountered during testing are still very much different from those in the training set, even when active learning is used. The lack of differentiation in the recall of organization names between active and passive learning is the main reason for the small difference in the corresponding F-measure scores.

Although active learning is overall better than passive learning in terms of effectiveness, as summarized by F-measure, there are efficiency issues that need to be considered. Figure 5 shows the total training time of the two first-pass SVMs in passive and active learning in our experiments with general newspaper articles; again, the second-pass curve is to be ignored for the moment.[h] There is a much steeper increase in the training time of active learning, which is probably due to the fact that the selected training instances are closer to the separating hyperplanes of the SVMs, and, hence, there are fewer training instances that can be ignored, compared to passive learning where the training instances are selected in effect randomly.[i] This has important practical consequences in active learning, where the SVMs have

[h] The experiments were performed on a PC with an Intel Pentium M processor at 1.9 GHz with 1 GB RAM, running Windows XP Professional.
[i] Interestingly, Hachey et al.[26] provide evidence that humans also need more time when annotating training examples selected via active learning.

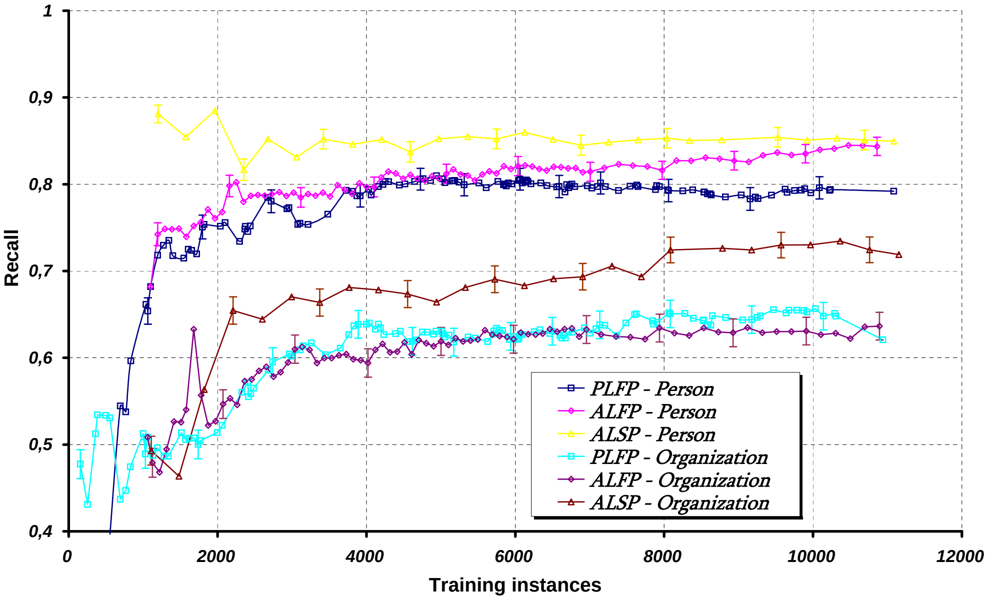

Figure 4. **Recall** results on general newspaper articles, with active (AL) and passive learning (PL), during the first (FP) and second pass (SP).

to be retrained whenever a new batch of training tokens has been manually annotated. In our experiments with active learning, it took approximately 46 minutes to retrain the two first-pass SVMs with 10,900 training instances; and a further 30 minutes were required to reclassify the remaining training candidate instances of the ($\frac{1}{10}$th of the) pool, in order to select the instances of the next batch.

Consequently, the human annotator has to wait for an increasingly longer time for the next batch of tokens to appear for annotation in active learning. Our own experience is that this is acceptable to human annotators, provided that they can carry out other work between annotating batches; and it is more interesting than annotating tokens in passive learning, because active learning proposes for annotation more difficult and, thus, challenging instances. The rapid growth of training time in active learning, however, eventually makes adding more training instances impractical. As a remedy, we stopped expanding the training set of the first-pass SVMs at 10,900 training instances, and thereafter we employed active learning to expand the training set of the second-pass SVMs; this is explained further in the next section. We also increased the size of the batches, from initially 100 to eventually 400 training instances, to reduce the number of iterations (and retrainings) that are required in active learning to reach a particular number of training instances.

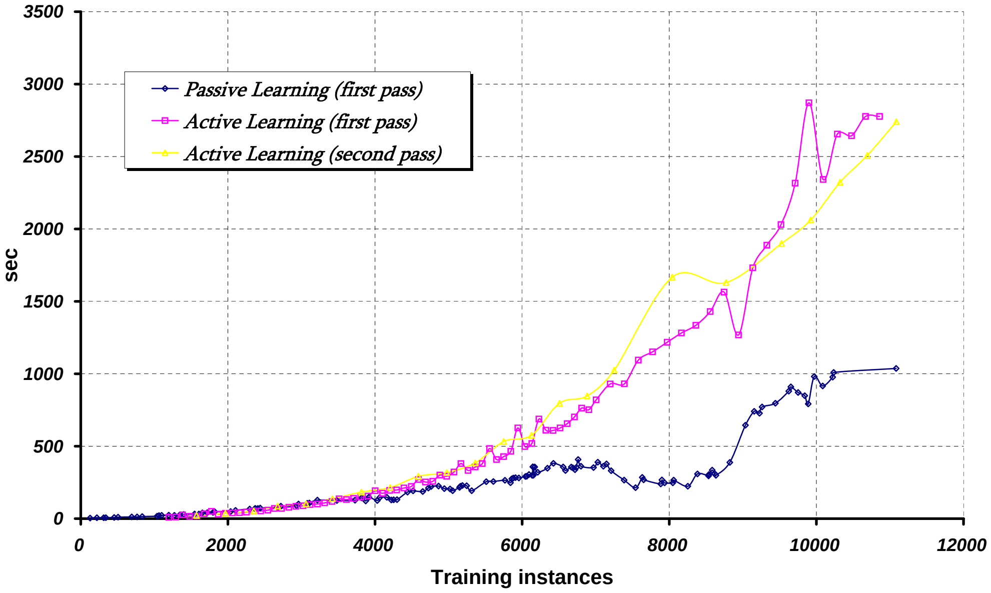

Figure 5. **Training times** of the SVMs on general newspaper articles.

3.3.2. *Second pass experiments with active learning*

In this set of experiments we used two passes, in an attempt to improve further the effectiveness of our NER and bypass the efficiency problems of active learning that we had encountered with large training sets in the previous experiments. We used only active learning, since the results of the previous experiments had indicated that it leads to higher F-measure scores.

For the purposes of these experiments, we randomly selected and manually annotated 10 more newspaper articles (approximately 1,200 training instances) from corpus 0, which were excluded from corpus 1 and the pool of active learning. We then trained the two SVMs of the second pass on the resulting training instances, and gradually increased the training set of the second-pass SVMs using active learning. The selection of training instances was now based on their distances from the separating hyperplanes of the second-pass SVMs, excluding from the pool instances that had been used in the first pass. The reader is reminded that the second-pass SVMs employ additional features, which rely on the confidence scores of the first-pass SVMs. To compute the additional features, we used the first-pass SVMs that we had obtained with 10,900 training instances in the previous experiments.

Since the training set of the second-pass SVMs was initially very small (approx. 1,200 instances), retraining the SVMs of the second pass was initially very fast, which allowed us to expand the training set of the second pass up to again approximately 10,900 training instances, before reaching the same levels of low retraining speed that we had observed at the end of active learning in the first pass; see Figure

5. Figure 2 shows that the second-pass quickly achieved higher F-measure scores, compared to the corresponding scores that we had obtained with the same number of training instances in the first pass, because of the additional features, which in effect summarized the experience that the first-pass SVMs had obtained from their training. Figure 2 also shows that the second pass allowed our NER to eventually reach a higher F-measure score in both person and organization names, compared to the scores we had obtained at the end of active learning in the first pass.

Figures 3 and 4 show that in the case of person names, the increase in F-measure was eventually due to an increase in precision; in contrast, the second pass did not manage to exceed significantly the first pass in recall. In the case of organization names, the increase in F-measure was due to a dramatic increase in recall, which was, however, accompanied by a smaller drop in precision. Overall, then, it appears that in person name recognition the second pass reduced false positives without affecting significantly false negatives. In contrast, in organization names recognition it reduced dramatically false negatives at the expense of a smaller increase in false positives. Table 6 summarizes the results of person and organization name recognition on general newspaper articles, when using approximately 10,900 training instances in each pass.[j]

Table 6. Results of person and organization name recognition on **general newspaper articles**, with approximately 10,900 training instances in each pass.

category	methods	precision (%)	recall (%)	$F_{\beta=1}$ (%)
person	passive, first pass	86.90	79.19	82.87
	active, first pass	87.89	84.36	86.09
	active, second pass	91.00	84.96	87.88
organization	passive, first pass	74.63	62.08	67.78
	active, first pass	76.50	63.65	69.49
	active, second pass	70.77	71.90	71.33

As one would expect, adding a second pass increased the overall average classification time, from approximately 4 msec per test instance at the end of active learning in the first pass to approximately 10 msec per test instance at the end of active learning in the second pass. This is shown in Figure 6, where we divide the total time it takes to classify all of the tokens of the test corpus (part 2) that are not temporal expressions by the number of these tokens. The increase is slightly higher than 100%, presumably because of the additional features of the second pass. Figure 6 also shows that classification time is roughly linear to the number of training instances, unlike training time.[k] Note that training an SVM typically takes $O(mN^2)$

[j] Person name recognition results are slightly better than in our earlier work,[33] because we now use a larger feature set and an additional mutually exclusive category for organization names.
[k] Figure 6 includes the time it takes to classify tokens that satisfy the sure-fire rules, which are classified without consulting the corresponding SVMs, but this time is not affected by the number of training instances.

time, where m is the number of features and N the number of training instances. In contrast, classification time is usually $O(mN)$. Although these are worst case complexity figures and may vary, depending on the kernel and the optimization algorithm of the SVMs, they seem to agree with the active learning curves of Figures 5 and 6. The corresponding curves of passive learning are less stable, with a major drop around 8,000 training instances. The latter may be a sign of a major reorientation of the separating hyperplanes, which may have led to fewer training instances being support vectors.

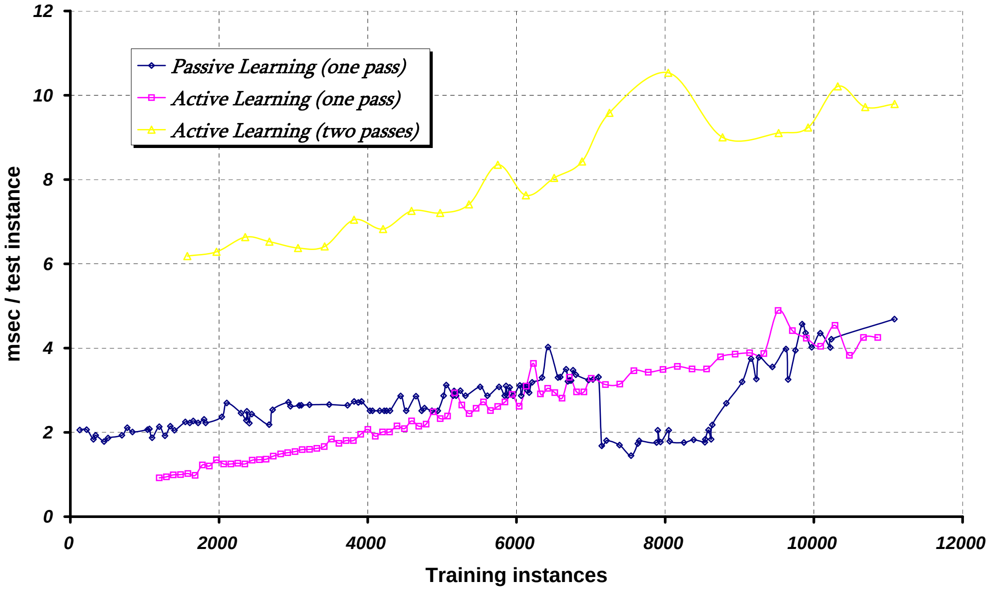

Figure 6. Average **classification time** per token on general newspaper articles.

An error analysis revealed that many of the misclassified tokens at the end of active learning in the second pass were names that even human annotators had trouble classifying, such as the theater "Θέατρο Κάτια Δανδουλάκη", where "Κάτια Δανδουλάκη" is the name of an actress, or "Parliament", which can be used both as an organization name and a location. We expect that adopting more lenient annotation guidelines, which would not penalize, for example, classifying "Parliament" as an organization when used as a location, would allow our NER to reach significantly higher F-measure scores, especially in organizations.

Note that an alternative to using a new, small training set in the second pass and gradually expanding it via active learning is to train the second pass directly on the 10,900 training instances of the first pass, after expanding their vectors with the additional features of the second pass. We experimented with this approach, but it led to much worse results.[52] A possible explanation is that this approach misleads the second-pass SVMs to over-estimate the reliability of the additional features: the first-pass SVMs have already encountered during their training the 10,900 training instances that are reused in the second pass, and hence the additional features, which reflect the decisions of the first-pass SVMs, appear to be more reliable during the training of the second-pass SVMs than they really are in completely new instances.

3.3.3. *Experiments with financial articles*

The previous person and organization experiments used general newspaper articles (corpora 0 and 1). The remaining experiments investigated the degree to which the system's effectiveness improves when using financial news only (corpus 2), i.e., a corpus that is more focussed in terms of topics. These experiments were performed by using 10-fold cross-validation and passive learning only, with one or two passes. In each iteration of the cross-validation, the training data (90% of the dataset) were divided in two equally large parts, which were used to train the two passes, respectively; in experiments with only one pass, the second part was not used.

The results of these experiments are summarized in Table 7. Performance was clearly better in both categories, compared to the results on general newspaper articles (cf. Table 6), even though there were slightly fewer (approximately 10,300 instead of 10,900) training instances available to each pass in each repetition of the cross-validation and no active learning was used. The second pass had a positive effect in both categories, on both precision and recall. As in the experiments on general news articles, the largest improvement was in the recall of organization names, but this time it was not accompanied by a reduction in precision. We attribute the improved results of these experiments to the more standardized expressions of financial news, compared to general news articles.

Note that no valid comparison to the published results of the other Greek NERs of Section 1 can be made, since the results of the other NERs were obtained on different corpora, often of different genres and/or with different annotation guidelines.

4. Conclusions

We presented a freely available named-entity recognizer (NER) for Greek texts, which identifies temporal expressions, person names, and organization names. For temporal expressions, the NER uses manually constructed token lists and automatically generalized regular expression patterns. For person and organization names, it uses an ensemble of SVMs that scan the input text in two passes. The second pass takes into account the decisions of the first one, which allows it to learn how to correct mistakes of the first pass. It also considers whether or not the first pass has classified a token elsewhere in the same text as a person or organization name with high confidence, which allows it to identify re-occurrences of person and organization names in more difficult contexts. Each pass employs two SVMs, for person and organization names, respectively. A set of simplistic sure-fire rules is also used to reduce the class imbalance of the binary decision problem each SVM faces.

Table 7. Cross-validation results of person and organization name recognition on **financial articles**, with approximately 10.300 training instances in each pass.

category	method	precision (%)	recall (%)	$F_{\beta=1}$ (%)
person	passive, first pass	96.33	89.92	93.01
	passive, second pass	97.16	91.13	94.05
organization	passive, first pass	80.25	73.36	76.65
	passive, second pass	82.06	76.89	79.39

Apart from its two-pass architecture, another novelty of our NER is the use of active learning, which allows the system to select by itself candidate training instances to be annotated by a human during training. Our system is one of the very few NERs, regardless of language, that have exploited active learning. The experiments of this paper confirmed that active learning outperforms passive learning, but they also showed that active learning increases significantly the training time of the SVMs, which eventually makes expanding the training set impractical. Our two-stage architecture is also a remedy to this problem, since it allows one to stop expanding the training set of the first pass when it has become too large, and thereafter use active learning to build the training set of the second pass. Despite its small initial training set, the second pass quickly outperforms the first one, because of its additional features that enable it to exploit the experience that the first pass has obtained from its own training set.

The NER was evaluated on both a general collection of articles from two Greek newspapers, and a more focussed collection of Greek financial articles. Both precision and recall were higher on the latter collection, in all three categories of named-entities, because of the more standardized expressions of financial news.

We plan to extend our NER further with additional name categories, such as names of locations. Work is already in progress to use the NER in a Greek question-answering system for document collections, which will initially support questions asking for persons, organizations, and points in time.

Bibliography

1. R. Grishman. Information extraction. In R. Mitkov, editor, *The Oxford Handbook of Computational Linguistics*, chapter 30, pages 545–559. Oxford University Press, 2003.
2. E.M. Voorhees. The TREC QA track. *Natural Languge Engineering*, 7(4):361–378, 2001.
3. S. Harabagiu and D. Moldovan. Question answering. In R. Mitkov, editor, *The Oxford Handbook of Computational Linguistics*, chapter 32, pages 560–582. Oxford University Press, 2003.
4. P. Virga and S. Khudanpur. Transliteration of proper names in cross-lingual information retrieval. In *Proceedings of the ACL Workshop on Multilingual and Mixed-Language Named Entity Recognition*, pages 57–64, Sapporo, Japan, 2003.
5. D.E. Appelt, J.R. Hobbs, J. Bear, D. Israel, M. Kameyama, A. Kehler, D. Martin, K. Myers, and M. Tyson. SRI International FASTUS system MUC-6 test results and analysis. In *Proceedings of the 6th Message Understanding Conference*, Columbia, MD, 1995.

6. B. Mitchell, C. Huyck, H. Cunningham, K. Humphreys, R. Gaizauskas, S. Azzam, and Y. Wilks. University of Sheffield: description of the LaSIE-II system as used for MUC-7. In *Proceedings of the 7th Message Understanding Conference*, Fairfax, VA, 1998.
7. D.M. Bikel, S.Miller, R. Schwartz, and R. Weischedel. Nymble: a high-performance learning name-finder. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, pages 194–201, Washington, D.C., 1997.
8. D.M. Bikel, R.L. Schwartz, and R.M. Weischedel. An algorithm that learns what's in a name. *Machine Learning*, 34(1–3):211–231, 1999.
9. H.L. Chieu and H.T. Ng. Named entity recognition with a maximum entropy approach. In *Proceedings of the 7th Conference on Computational Natural Language Learning*, pages 160–163, Edmonton, Canada, 2003.
10. G. Zhou and J. Su. Machine learning-based named entity recognition via effective integration of various evidences. *Natural Language Engineering*, 11:189–206, 2005.
11. G. Paliouras, V. Karkaletsis, G. Petasis, and C.D. Spyropoulos. Learning decision trees for named-entity recognition and classification. In *Proceedings of the 14th European Conference on Artificial Intelligence*, Berlin, Germany, 2000.
12. D. Wu, G. Ngai, M. Carpuat, J. Larsen, and Y. Yang. Boosting for named entity recognition. In *Proceedings of the 6th Conference on Computational Natural Language Learning*, pages 195–198, Taipei, Taiwan, 2002.
13. X. Carreras, L. Marquez, and L. Padro. A simple named entity extractor using Adaboost. In *Proceedings of the 7th Conference on Computational Natural Language Learning*, pages 152–155, Edmonton, Canada, 2003.
14. J. Kazama, T. Makino, Y. Ohta, and J. Tsujii. Tuning Support Vector Machines for biomedical named entity recognition. In *Proceedings of the ACL Workshop on Natural Language Processing in the Biomedical Domain*, pages 1–8, Philadelphia, PA, 2002.
15. K.J. Lee, Y.S. Hwang, and H.C. Rim. Two-phase biomedical NE recognition based on SVMs. In *Proceedings of the ACL Workshop on Natural Language Processing in the Biomedical Domain*, Philadelphia, PA, 2002.
16. D.D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the 11th International Conference on Machine Learning*, pages 148–156, New Brunswick, NJ, 1994.
17. A. McCallum and K. Nigam. Employing EM and pool-based active learning for text classification. In *Proceedings of the 15th International Conference on Machine Learning*, pages 350–358, Madison, WI, 1998.
18. G. Schohn and D. Cohn. Less is more: active learning with Support Vector Machines. In *Proceedings of the 17th International Conference on Machine Learning*, pages 839–846, Stanford, CA, 2000.
19. S. Tong and D. Koller. Support Vector Machine active learning with applications to text classification. *Machine Learning Research*, 2:45–66, 2002.
20. S. Argamon-Engelson and I. Dagan. Committee-based sample selection for probabilistic classifiers. *Journal of Artificial Intelligence Research*, 11:335–360, 1999.
21. G. Ngai and D. Yarowsky. Rule writing or annotation: cost-efficient resource usage for base noun phrase chunking. In *Proceedings of the 38th Annual Meeting of ACL*, pages 117–125, Hong Kong, 2000.
22. A. Vlachos. Active learning with Support Vector Machines. Master's thesis, School of Informatics, University of Edinburgh, 2004.
23. C.A. Thompson, M.E. Califf, and R.J. Mooney. Active learning for natural language parsing and information extraction. In *Proceedings of the 16th International Conference on Machine Learning*, pages 406–414, Bled, Slovenia, 1999.

24. M. Tang, X. Luo, and S. Roukos. Active learning for statistical natural language parsing. In *Proceedings of the 40th Annual Meeting of the ACL*, pages 120–127, Philadelphia, PA, 2002.
25. M. Becker, B. Hachey, B. Alex, and C. Grover. Optimising selective sampling for bootstrapping named entity recognition. In *Proceedings of the Workshop on Learning with Multiple Views, International Conference on Machine Learning*, Bonn, Germany, 2005.
26. B. Hachey, B. Alex, and M. Becker. Investigating the effects of selective sampling on the annotation task. In *Proceedings of the 9th Conference on Computational Natural Language Learning*, pages 144–151, Ann Arbor, Michigan, 2005.
27. Y. Freund, H.S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28:133–168, 1997.
28. C. Campbell, N. Cristianini, and A.J. Smola. Query learning with large margin classifiers. In *Proceedings of the 17th International Conference on Machine Learning*, pages 111–118, Stanford University, CA, 2000.
29. C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.
30. V. Vapnik. *Statistical learning theory.* John Wiley, 1998.
31. N. Cristianini and J. Shawe-Taylor. *An introduction to Support Vector Machines.* Cambridge University Press, 2000.
32. D. Shen, J. Zhang, J. Su, G. Zhou, and C.-L. Tan. Multi-criteria-based active learning for named entity recognition. In *Proceedings of the 42nd Annual Meeting of ACL*, pages 589–596, Barcelona, Spain, 2004.
33. G. Lucarelli and I. Androutsopoulos. A Greek named-entity recognizer that uses Support Vector Machines and active learning. In *Proceedings of the 4th Hellenic Conference on Artificial Intelligence*, Heraklion, Crete, Greece, 2006.
34. S. Boutsis, I. Demiros, V. Giouli, M. Liakata, H. Papageorgiou, and S. Piperidis. A system for recognition of named entities in Greek. In *Proceedings of the 2nd International Conference on Natural Language Processing*, pages 424–435, Patra, Greece, 2000.
35. D. Farmakiotou, V. Karkaletsis, J. Koutsias, G. Sigletos, C.D. Spyropoulos, and P. Stamatopoulos. Rule-based named entity recognition for Greek financial texts. In *Proceedings of the Workshop on Computational Lexicography and Multimedia Dictionaries*, pages 75–78, Patra, Greece, 2000.
36. D. Farmakiotou, V. Karkaletsis, G. Samaritakis, G. Petasis, and C.D. Spyropoulos. Named entity recognition in Greek Web pages. In *Proceedings of the 2nd Hellenic Conference on Artificial Intelligence, companion volume*, pages 91–102, Thessaloniki, Greece, 2002.
37. V. Karkaletsis, G. Paliouras, G. Petasis, N. Manousopoulou, and C.D. Spyropoulos. Named-entity recognition from Greek and English texts. *Intelligent and Robotic Systems*, 26:123–135, 1999.
38. G. Petasis, F. Vichot, F. Wolinski, G. Paliouras, V. Karkaletsis, and C.D. Spyropoulos. Using machine learning to maintain rule-based named-entity recognition and classification systems. In *Proceedings of the 39th Annual Meeting of ACL and 10th Conference of EACL*, pages 426–433, Toulouse, France, 2001.
39. K. Diamantaras, I. Michailidis, and S. Vasileiadis. A very fast and efficient linear classification algorithm. In *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing*, Mystic, CT, 2005.
40. C.C. Chang and C.-J. Lin. *LIBSVM: a library for Support Vector Machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/∼cjlin/libsvm/`.

41. I. Michailidis, K. Diamantaras, S. Vasileiadis, and Y. Frere. Greek named entity recognition using Support Vector Machines, Maximum Entropy and Onetime. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, pages 45–72, Genova, Italy, 2006.
42. D. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–260, 1992.
43. K. Tsukamoto, Y. Mitsuishi, and M. Sassano. Leaning with multiple stacking for named entity recognition. In *Proceedings of the 6th Conference on Computational Natural Language Learning*, pages 191–194, Taipei, Taiwan, 2002.
44. R. Florian. Named entity recognition as a house of cards: classifier stacking. In *Proceedings of the 6th Conference on Computational Natural Language Learning*, pages 175–178, Taipei, Taiwan, 2002.
45. R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. Named entity recognition through classifier combination. In *Proceedings of the 7th Conference on Computational Natural Language Learning*, pages 168–171, Edmonton, Canada, 2003.
46. D. Wu, G. Ngai, and M. Carpuat. A stacked, voted, stacked model for named entity recognition. In *Proceedings of the 7th Conference on Computational Natural Language Learning*, Edmonton, Canada, 2003.
47. A. Mikheev, C. Grover, and M. Moens. Description of the LTG system used for MUC-7. In *Proceedings of the 7th Message Understanding Conference*, Fairfax, VA, 1998.
48. C.D. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.
49. I.H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2nd edition edition, 2005.
50. G. Lucarelli. Named entity recognition and categorization in Greek texts (in Greek). Master's thesis, Department of Informatics, Athens University of Economics and Business, 2005. `http://www.aueb.gr/users/ion/students.html`.
51. R. Segal, T. Markowitz, and W. Arnold. Fact uncertainty sampling for labeling large e-mail corpora. In *Proceedings of the Third Conference on Email and Anti-Spam*, Mountain View, CA, 2006.
52. X. Vasilakos. Extensions to a named entity recognizer for Greek texts (in Greek). Final year project report, Department of Informatics, Athens University of Economics and Business, 2006. `http://www.aueb.gr/users/ion/students.html`.
53. K. Brinker. Incorporating diversity in active learning with Support Vector Machines. In *Proceedings of the 20th International Conference on Machine Learning*, pages 59–66, Washington, D.C., 2003.