

Semantic labeling of data by using the Web

Leonardo Rigutini, Ernesto Di Iorio, Marco Erlandes, Marco Maggini
Dipartimento di Ingegneria dell'Informazione
Università di Siena
Via Roma 56, I-53100 - Siena - Italy
email: {rigutini, diorio, erlandes, maggini} @dii.unisi.it
WWW home page: <http://airgroup.dii.unisi.it/>

Abstract

The Web consists of a large amount of unstructured information that hardly can be elaborated by automatic agents. In recent years, a considerable number of techniques for information extraction from Web resources have been proposed. In particular, while many different approaches have been devised to automatically identify the structure of the data in a document (f.i. Named Entity Recognition, Part Of Speech tagging), few systems exist to assign the semantics to such data. This fact limits significantly the use of these systems in data integration since a human intervention for labeling the extracted entities is required. Once these data have been semantically labeled, in fact, they can be used to fill databases, to build lexicons and to provide additional attributes for document categorization or clustering tasks. This paper proposes an evolution of the system for automatically categorizing terms or lexical entities presented in [5]. We added a submodule for multi-labels classification and tested the system using a standard benchmark, comparing the performances with the work presented in [1, 2].

1 Introduction

Term categorization is a key task in the Text Mining research area. In fact, while several techniques to identify and extract groups of data from a collection of documents have been proposed, few systems able to assign semantic tags to such data exist. Actually, data extraction systems well capture the structure of the data in the collection, but suffer the lack of semantic information, thus providing a set of objects without descriptive labels. Therefore, these systems require a final human intervention before using the data extracted and a term categorization system aims of automating this final step. In this paper, we extend the approach to term categorization proposed in [5] that exploits the knowledge

available in the Web to classify an unknown entity. We add a Multi-labels submodule which assigns an entity to one or more categories. Moreover, in this paper, we opted to test the system using the same data set proposed in [1, 2] to perform a fair and direct comparison with their results and making a step towards the definition of a benchmark for the term categorization task.

The paper is structured as follows. In the next section the architecture of the system is described in details. Section 3 reports the configuration of the experiments while the section 4 reports the results of the experiments aimed at evaluating the system performance and at comparing the possible design choices. Finally, in section 5 the conclusions and the directions for future research are presented.

2 System description

The system for term categorization is composed by two main modules as sketched in fig. 1. The *training module* is used to train the classifier and the *Multi-labels Selector* using sets of labeled examples, whereas the *entity classification module* is applied to predict the appropriate categories for a given input entity. Both modules exploit the Web to obtain an enriched representation of each entity as described in [5]. Given an entity e , the system builds an *Entity Context Lexicon* ECL_e , a list of terms extracted from a set of "contexts" in which the entity e appears. The snippets returned by the search engine Google when submitting the entity as query are considered to represent the "contexts" of the entity. Therefore, in the training module, a set of labeled $ECLs$ is used to train an automatic classifier, while in the classification module, the trained classifier is used to label an unlabeled ECL_e . Since each entity can be assigned to one or more category (multi-label classification task), we add a *Multi-labels Selector* submodule, which selects the set of labels according to the scores assigned by the classifier to the classified ECL for each class. We propose a trainable submodule based on a neural network and trained using a

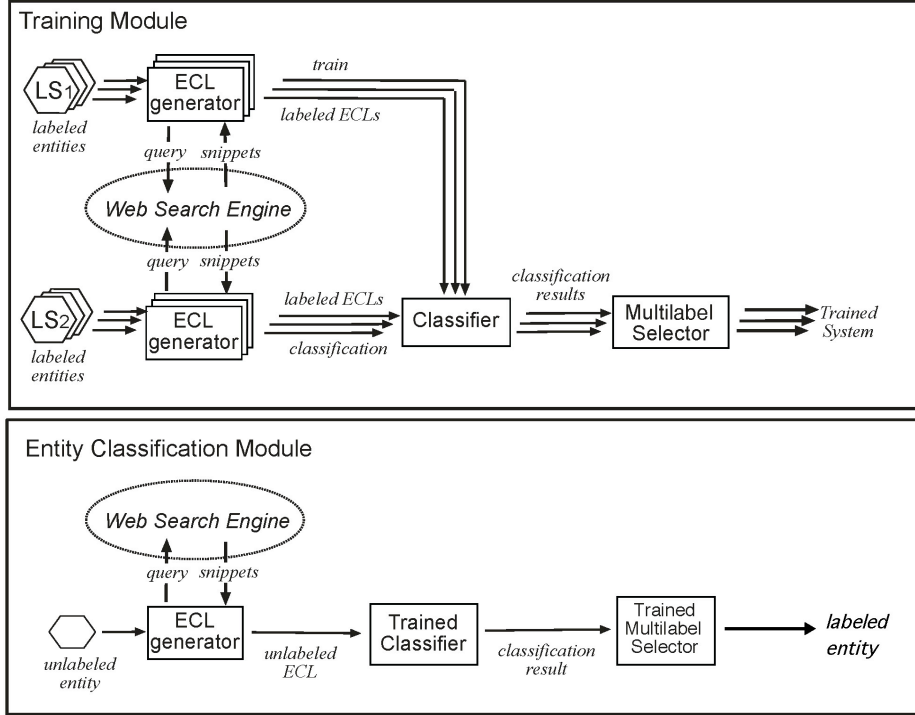


Figure 1. Training module and classification module of the system.

different set of labeled entities.

In the experiment, we tested the system exploiting also the *SVM* classifier [4]. In this case, the *Multi-labels Selector* submodule was not used since *SVMs* are already discriminative classifiers. The *ECL generator sub-module* and the *classifier sub-module* are deeply described in [5]. In this paper, we describe the added *Multi-labels Selector* submodule.

The Multi-labels Selector sub-module

In a term categorization task, each entity can be assigned contemporarily to one or more semantic categories. This kind of problem is known in the literature as *Multi-labels classification* task and most of the commonly used classifiers are not suited for it. In fact, they simply assign a vector of scores to the unlabeled *ECLs* and the classification process need to be completed by a subsequent sub-module called *Multi-labels Selector* which assigns a subset of labels to each example basing on the scores assigned by the classifier. Typically, Multi-labels Selectors are obtained by using a set of thresholds (represented by the vector \mathbf{T}), one for each output class. In this work, a Multi-Layer Perceptron (MLPs) [6] is trained to map the vector of class scores evaluated by the classifier into an output vector of the same dimensionality in order to better derive a multi-labels selec-

tion rule. Then, a threshold τ is then applied to all the MLP output values to select the subset of labels to be assigned to the entity.

3 Experimental Setup

Evaluation Metrics

To evaluate the performances of the proposed system, we adopted the standard evaluation metrics that are used for text categorization, precision Pr and recall Re , combined into the $F1$ score [7]. Since the distribution of the terms can be highly unbalanced among the different domains, (i.e. there could be domains containing thousands of terms and others with very few) usually the results are combined according two ways, microaveraging and macroaveraging [7]. We report the performances using both methods.

The dataset

As pointed in the section 1, we chose to test the system using the dataset described in [1, 2] which we indicated with DS . This is composed by a set of 27048 nouns assigned to one or more of 42 different categories. To compare the results of the proposed system with the results obtained by Avancini et al., we also numerically replicated

the same composition of the training set LS ($\frac{2}{3} \times DS$) and of the test set TS ($\frac{1}{3} \times DS$). Moreover, in our system, half of the learning set (LS_1) was adopted to train the classifiers and the other half (LS_2) was used to train the MLP multi-labels Selector. From LS_2 , 80% of the examples were taken as the neural learning set and the remaining 20% was used to cross-validate the training. When the system uses a SVM model, the Multi-labels Selector is not used and all the data set LS is used as learning set. The authors of [2] performed six different experiments, computing the system performance using only the training and test terms occurring in at least x documents of RCV1, where $x = \{1, 5, 10, 15, 30, 60\}$. In section 4, we report the results obtained in [2] for $x = 1$, when they used all the terms in the intersection between RCV1 and WordNetDomains, and their best result (with $x = 60$).

4 Experimental Results

Experiments setting

We tested the proposed system using a NaiveBayes (NB) classifier [7], a Complement NaiveBayes (CNB) classifier [3], a CCL classifier [5] and a SVM model [4]. For each classifier, we averaged the results with a five-fold validation technique. The Multi-labels Selector submodule was not used when the SVM classifier is used in the system. For this reason, we firstly tested the system using the three models that require the Multi-labels Selector submodule (NB, CNB and CCL) and we evaluated the performances varying the parameter τ . Then, we tested the system using SVM model and evaluated the performances varying the cost ratio factor η ([4]).

Experiments using NB, CNB and CCL classifiers

In these experiments, we realized the Threshold Estimator by using a Multi Layer Perceptron and we trained it using the learning set LS_2 . Moreover, we trained a different network for each classifier. Different architectures of the neural network were tested and we experimentally observed that the best performances were obtained with 50 hidden units and using the hyperbolic tangent activation output function. The threshold estimator output consists of a vector of k components where k is the number of the classes and where each value ranges in $[-1, 1]$. As described in the section 2, a threshold τ applied to the outputs of the MLP defines the final classification result. The table 1, reports the results obtained setting $\tau = 0$ (natural threshold). The results show that, in this case, the *CCL* classifier reports the best performances. To have a clearer view of the system's behavior, we decided to increase τ in the Multi-label selector to obtain high values of precision, also admitting low recall. This allows to compare the performances of the

system when the precision reaches values very similar to the ones reported in [2]. The table 2 shows the results using $\tau = 0.95$. The table 2 shows that the system reaches elevated values of precision, also maintaining the recall and F1 values higher than the ones reported by Avancini et al.

Experiments using the SVM

As hinted in the section 2, the Multi-labels Selector submodule can not be exploited when the SVM model is used in the system. This model, in fact, estimates the hyperplane which minimizes the structural risk of false positives and false negatives and it is a discriminative classifier. Since we aimed of increasing the precision, we evaluated the performances of the SVM classifier varying $\eta \in [1, 0]$ ([4]). That is to say that during the training phase, a false positive costs more than a false negative. In this way, the model is implicitly trained to minimize the false positives, thus increasing the precision.

From the table 3, we note that the system performances reach the breakeven point (in which $P \sim R$) when $\eta = 0.3$. In this case, the system shows better results than the ones reported by Avancini in [1, 2]. When $\eta = 0.1$, the system yet shows very goods performances, reporting a $Pr > 93\%$ and a recall $Re > 16\%$; these values significantly outperform the results reported by the authors in [1, 2]. In this case, the system classifies a larger number of terms ($\simeq 20\%$ with respect to the 4%) reporting a very high precision ($\simeq 95\%$) and without filtering any word (in [2] the authors reported the best results filtering out the terms not appearing in at least $x = 60$ documents).

5 Conclusions

In this paper we addressed the problem of term categorization, whose goal is to assign one or more semantic labels to any lexical entity (e.g. common nouns, proper nouns, verbs, technical terms, ...). In this work, we extends the method proposed in [5] adding a *Multi-labels Selector* submodule that performs multi-labels classification. We test the modified system adopting the benchmark problem proposed by [1]. We directly compare the performances with respect to the results reported in that work using different automatic classification models. The system appears to be more effective than the one described in [1], regardless of the specific classifier used by the system. In particular, the SVM classifiers reported the best performances considerably outperforming the results reported in [1]. We believe that this work helps in defining and stabilizing a benchmark problem for the term categorization task and, at the same time, in providing new state-of-the-art performances.

Classifier	Selector	P^μ	R^μ	F_1^μ	P^M	R^M	F_1^M
NB	Neural Network	55.20%	34.12%	42.17%	39.60%	23.30%	29.33%
CNB	Neural Network	49.66%	38.38%	43.30%	36.24%	27.08 %	30.99%
CCL	Neural Network	56.28%	37.79%	45.22%	43.68%	29.32 %	35.09%
Avancini et al. $x = 1$	-	70.5%	4.3%	8.1%	71.7%	2.3%	4.5%
Avancini et al. $x = 60$	-	75.7%	11.7%	20.3%	83.6%	4.4%	8.4%

Table 1. Comparison of the precision, recall and F_1 of the proposed system with the results reported by Avancini et al. in [2]. The results are performed using NB, CNB and CCL classifiers and setting $\tau = 0$

Classifier	Selector	P^μ	R^μ	F_1^μ	P^M	R^M	F_1^M
NB	Neural Network	80.80%	7.09%	13.03%	69.94%	5.14%	9.58%
CNB	Neural Network	71.73%	9.27%	16.43%	64.69%	7.30%	13.13%
CCL	Neural Network	81.27%	6.7%	12.49%	72.16%	7.45 %	13.51%
Avancini et al. $x = 1$	-	70.5%	4.3%	8.1%	71.7%	2.3%	4.5%
Avancini et al. $x = 60$	-	75.7%	11.7%	20.3%	83.6%	4.4%	8.4%

Table 2. Comparison of the precision, recall and F_1 of the proposed system with the results reported by Avancini et al. in [2]. The results are performed using NB, CNB and CCL classifiers and setting $\tau = 0.95$.

Classifier	η	P^μ	R^μ	F_1^μ	P^M	R^M	F_1^M
SVM	1	20.34%	82.66%	32.65%	25.09%	71.61 %	37.16%
SVM	0.7	29.37%	75.33%	42.26%	36.47%	62.47%	46.05%
SVM	0.5	39.50%	66.18%	49.47%	52.35%	52.10 %	52.22%
SVM	0.3	54%	51.27%	52.6%	67.9%	37.4 %	48.23%
SVM	0.1	93.6%	16.6%	28.26%	96.06%	10.17%	18.4%
Avancini et al. $x = 1$	-	70.5%	4.3%	8.1%	71.7%	2.3%	4.5%
Avancini et al. $x = 60$	-	75.7%	11.7%	20.3%	83.6%	4.4%	8.4%

Table 3. Comparison of the precision, recall and F_1 of our system using SVM classifier with the results reported by Avancini et al. in [2]

References

- [1] H. Avancini, A. Lavelli, B. Magnini, F. Sebastiani, and R. Zanolì. Expanding domain-specific lexicons by term categorization. In *Proceedings of SAC-03, 18th ACM Symposium on Applied Computing*, pages 793–797, Melbourne, US, 2003. ACM Press, New York, US.
- [2] H. Avancini, A. Lavelli, F. Sebastiani, and R. Zanolì. Automatic expansion of domain-specific lexicons by term categorization. *ACM Transactions on Speech and Language Technology*, 2005.
- [3] J.D.Rennie, L.Shih, J.Teevan, and D.Karger. Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, Washington DC, 2003.
- [4] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, DE, 1998.
- [5] L.Rigutini, M.Ernandes, E.DiIorio, and M.Maggini. Automatic term categorization by extracting knowledge from the web. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, 2006.
- [6] J. McClelland, D. Rumelhart, and the PDP research group. *Parallel distributed processing*. The MIT Press, Cambridge, MA, 1987.
- [7] F. Sebastiani. Text categorization. In *Text Mining and its Applications to Intelligence, CRM and Knowledge Management*, pages 109–129. WIT Press, Southampton, UK, 2005.