Applying Transductive Learning for Automatic Term Extraction: the Case of the Ecology Domain

Merley S. Conrado, Rafael G. Rossi, Thiago A. S. Pardo, and Solange O. Rezende
Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo
Emails: {merleyc,ragero,taspardo,solange}@icmc.usp.br

***Abstract*—Terms are the basis for general text mining and natural language processing applications. However, the manual term extraction is unfeasible due to the huge number of words presented in a domain corpus and also the human effort required to do the extraction. For the term extraction task, machine learning techniques have been used to perform automatic term extraction (ATE). Inductive learning is commonly used, but it requires a large number of words labeled as terms and non-terms to build a classification model and classify unseen words. A better solution is the use of transductive learning, since it requires a small number of labeled examples to classify words as terms and non-terms. In this paper, we propose the use of transductive learning to ATE. The obtained results demonstrate that the application of transductive learning to ATE produces better results than the results obtained by inductive learning.**

***Keywords*—Term Extraction; Transductive Learning.**

I. Introduction

Terms are terminological units from specialised texts [1]. A term may be: (i) simple (a single element), e.g., "*biodiversity*", or (ii) complex (more than one element), e.g., "*natural resource management*". Terms are the basis for general text mining and natural language processing applications such as finding automatically expert/professional profiles [2], word sense disambiguation [3], building of dictionaries, ontologies, and thesauri [4], and summarisation [5]. However, term extraction is a costly task for humans. Therefore, **automatic term extraction** (ATE) [1] is a important task for real world applications.

As stated by Pazienza et al., [6], normally, the ATE methods select candidate terms (CTs) based on some linguistic property. After that, they apply measures or combination of measures (and/or heuristics) to rank the candidates [7], [8], [9], [10]. The higher the candidates are in this rank, the better the chance they are terms. The problem of these approaches is the subjectivity to choose until which position this ranking should be considered good. In most recent researches, this problem has been solved by using an automatic way to choose which candidates should be considered terms. This solution has normally used the machine learning tasks (ML), because they are able to learn by themselves how to recognize a term [11], [12], [13], [14], [15], [16]. Another advantage is that they facilitate the use of a large number of measures and save time extracting terms.

Usually, inductive learning algorithms are applied to build a classification model, which classify unseen words as terms or non-terms. The problem of inductive learning is the need to have a large number of words previously labeled as terms and non-terms in order to obtain good results. Although, it is difficult to have a large number of labelled words since it requires time and human effort to label them.

A solution to avoid the labeling of a large number of words is the use of transductive learning. Transductive learning performs the classification spreading the labels from labeled to unlabeled data in a corpus. This is usually performed considering a small number of labeled examples (words). Consequently, it is feasible for real world applications.

Considering the advantage of transductive learning, in this paper, we evaluated the application of ***T*ransductive *L*earning for *A*utomatic *T*erm *E*xtraction** (called here TLATE). Two traditional algorithms for transductive learning were applied for different sets of candidate terms and considering different attribute selection methods. The different combinations of algorithms, sets of candidates, and attribute selection methods were considered in order to evaluate which of them are feasible for TLATE. We focus on extracting simple terms[1], since this is already a complex task and they are the basis for complex term extraction. TLATE shows to be useful for real world applications since it needs fewer labeled words to extract terms and obtains better results than when using inductive learning.

This paper is presented as follows. Section II introduces the related work about ATE. Section III describes the measures and knowledge used for ATE. Section IV details transductive learning and the algorithms used in this paper. Section V presents the used corpora, experiment configuration, evaluation criteria, results, and discussion. Finally, Section VI details conclusions and offers future research directions.

II. Related work

Usually, the studies in the literature focus on extracting not only simple terms[2], but also complex terms [10], [12], [17], [14], [15], [16]. The most recent research has used machine learning in order to extract terms. Despite the difficulty in comparing the ATE results due to variations (e.g., the size of the used corpora), we mention studies that have highlighted results using ML. When possible, we present the best precision (P) of the related work and its recall (R).

For English, Zhang et al. [12] and Nazar [15] combined different statistical measures based on frequency aiming to extract simple and complex terms together. Zhang et al. achieved

[1]When we refer to *simple terms*, we mean *unigrams*.
[2]It is not specified if [13], [17] extracted simple or complex terms.

P=97% when using a voting scheme. Nazar had P=75% and R=80% when applying a statistical algorithm.

For Spanish, Vivaldi et al. [13] used voting scheme to evaluate different term extraction when using linguistic knowledge (such as lexical semantic knowledge) and statistical measures based on frequency. They achieved P=99.1% and R=32% in a medical domain.

For Brazilian Portuguese (BP), Conrado et al. [11] extracted unigrams using linguistic (such as syntactic knowledge), statistical (such as measures based on frequency), and hybrid (such as corpus of general language) knowledge with ML (inductive classification). They tested their approach in 3 domains: ecology (P=60% and R=6.02%), distance education (P=66.66% and R=8.06%), and nanoscience and nanotechnology (P=61.03% and R=27.73%).

III. Measures and knowledge used for automatic term extraction

In this section, we describe the main measures and knowledge (called here features) that have been used for ATE in the literature. We classified them according to the approach they belong to, which are: linguistic, statistical, and hybrid.

In the **linguistic approach**, terms are identified by their linguistic features. The main linguistic features found in literature are:

1-2) **Phrases**[3] include 2 features: noun phrases (**NP**) and the head of NP (**HNP**).

3) **Part-of-speech Pattern (POS**[3]**)** is a linguistic category of lexical items. In this paper, we consider that terms are nouns.

4) **Indicative Phrases (IP**[3]**)** are usually near of definitions or descriptions, which may be terms. For example, considering *are composed of* as an IP in *All organisms are composed of one or more cells*, the candidates *organisms* and *cells* would have the value "1" for the IP feature.

5-8) **N_Noun, N_Adj., N_Verb, and N_PO** consider the use of some stemming technique to group similar meaning words in a stem. For example, the words *educative, educators, education*, and *educate* are originated from the stem *educ*. Therefore, the term *educ* may have as features **N_Noun**[3] = 2 (*educators* and *education*), **N_Adj.**[3] = 1 *educative*, **N_Verb**[3] = 1 (*educate*), and **N_PO**[3] = 4 (total number of words).

The **statistical approach** for ATE is traditionally based on the word frequency, in which no sentence structure or word sequence is considered. The main statistical features are:

9) **Lenght of Gram (LG**[3]**)** counts the number of characters of a word. For example, the word *education* would have 9 as LG value.

10) **Term Frequency (TF)** counts the word frequency in a corpus.

11) **Document Frequency (DF)** is the number of documents in which a word occurs.

12) **Average Term Frequency (ATF)** corresponds to the term frequency / document frequency.

13) **Residual Inverse Document Frequency (RIDF)** [18] represents the burstiness of a term in the documents in which it occurs.

14) **Term Frequency Inverse Document Frequency (TF-IDF**[3]**)** [19] weights the word frequency according to their distribution along the corpus.

15) **Term Contribution (TCo**[3]**)** [20] considers that the similarity between two documents is given by the product of the TF-IDF of all words in a corpus. The contribution of a word is calculated through the sum of the product of the TF-IDF of that word for all pairs of documents.

16) **Term Variance (TV**[3]**)** [21] weights the words according to their variance in a corpus, i.e., how much the frequency of the words differ from their average frequency.

17) **Term Variance Quality (TVQ**[3]**)** [22] quantifies the quality of the word variance.

The **hybrid approach** combines statistical and linguistic knowledge and is usually language-dependent. The hybrid features found in the literature are:

18) **General Corpus (GC**[3]**)** verifies if a word occurs in a corpus of general language.

19) **General Corpus Frequency (Freq_GC**[3]**)** counts the frequency of a word in a General Corpus.

20) **Term Domain Specificity (TDS)** [23] assumes that a relevant word for a domain has a higher frequency in a corpus of this domain than in other corpora. For that, TDS calculates the ratio between the probability of occurrence (absolute frequency) of a term in a domain corpus and the probability of the same term in a general corpus.

21) **Termhood (THD)** [9] considers that a relevant word is more peculiar to its own domain than to another domain corpus. For that, THD calculates the importance of a word considering the ranking based on word frequency in a domain corpus subtracted by the ranking of the same word in another corpus.

22) **GlossEx**[4] [8] is a method that considers the probability of the word in a domain corpus divided by the probability of the same word in a general corpus. Moreover, the importance of the word is increased according to its frequency in the domain corpus.

23) **Weirdness**[3] [24] considers that the distribution of words in a specific domain corpus is different from the word distribution in a general corpus.

24) **C-value** [7], [25] uses POS in order to extract candidate terms and rank them considering the word frequency in the corpus subtracted by the frequecy of the other candidates that contain this word.

25) **NC-Value** [7], [25] weights a candidate by extending the C-value measure addicting the frequency of the words that occur in the neighborhood of this candidate.

[3] This definition is presented in Conrado et al. [11].

[4] We used the implementation available in https://code.google.com/p/jatetoolkit/

IV. Transductive Learning

Transductive learning was introduced by Vapnik [26] and it considers a data set of both labeled and unlabeled data to perform the classification task, spreading the information from labeled to unlabeled data through the data set. The unlabeled data that was labeled during the process aid the classification of data that is still unlabeled. The transductive learning does not create a model to classify new documents as the inductive model does and is usually applied when few labeled examples are available [27].

Usually the datasets (corpora) are represented by a network to carry out the transductive learning. A network may be characterized as a set of objects and the relationships among them. Formally, a network may be defined as $N = \langle \mathcal{O}, \mathcal{R}, \mathcal{W} \rangle$ in which $\mathcal{O}$ represents the set of objects (instances of the dataset), $\mathcal{R}$ represents the set of relations among the objects, and $\mathcal{W}$ represents the relation weights. For transductive learning, $\mathcal{O}$ is composed by labeled (O^L) and unlabeled (O^U) objects, i.e., $\mathcal{O} = \mathcal{O}^L \cup \mathcal{O}^U$.

Proximity measures are commonly used to generate the networks when there is no information about the relations among the objects. A common way to create a network considering proximity measures is through the Exp-network [28]. In this network, the weight of the relation between the objects o_i and o_j is given by a gaussian function, such as $w_{i,j} = \exp(-\Omega(o_i, o_j)^2/\sigma^2)$, in which Ω is a distance function and σ is the bandwidth of the gaussian function.

Once the network is created, transductive learning algorithms may be applied to perform the classification task. Let $\mathcal{C} = \{c_1, c_2, \ldots, c_l\}$ be the set of class labels of a corpus. To classify objects in a network, each object o_i has a weight vector $\mathbf{f}_i = \{f_1, f_2, \ldots, f_{|\mathcal{C}|}\}$ that contains the weights of the object for each class[5]. All the weight vectors of the objects in a network are stored in a matrix $\mathbf{F} = \{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_{|\mathcal{O}|}\}$. The real class information of a labeled object o_i is stored in a vector $\mathbf{y}_i = \{y_1, y_2, \ldots, y_{|C|}\}^T$, which has the value 1 in the position corresponding to the real class and 0 in the other positions. All the class information of the labeled documents is represented by the matrix $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_{|\mathcal{O}^L|}\}$.

In this paper, we used two traditional algorithms to perform the transductive learning: (i) Gaussian Fields and Harmonic Functions (GFHF) [29] and (ii) Learning with Local and Global Consistency (LLGC) [30]. Both algorithms are based on regularization and have two assumptions: (i) the class information of an object must be similar to the class information of its neighbors and (ii) the class information of an object obtained during the classification process must be equal to its real information class. These two assumptions may be represented in a regularization framework given by the following cost function [31]:

$$Q(\mathbf{F}) = \frac{1}{2}\sum_{o_i,o_j \in \mathcal{O}} w_{i,j}\Omega(\mathbf{f}_i, \mathbf{f}_j) + \mu \sum_{o_i \in \mathcal{O}^L} \Omega'(\mathbf{f}_i, \mathbf{y}_i) \qquad (1)$$

in which μ is the regularization parameter that gives the importance to each assumption and Ω is a distance function.

[5]A weight vector will be also called by class information in this article.

The GFHF algorithm considers the distance between neighbors, as the squared of the subtraction of their weight vectors, and $\mu \to \infty$, i.e., to minimize the cost funtion $\mathbf{f}_i$ must be equal to $\mathbf{y}_i$. Then, the function to be minimized by GFHF is given by:

$$Q(\mathbf{F}) = \frac{1}{2}\sum_{o_i,o_j \in \mathcal{O}} w_{i,j}(\mathbf{f}_i - \mathbf{f}_j)^2 + \infty \sum_{o_i \in \mathcal{O}^L} (\mathbf{f}_i - \mathbf{y}_i)^2 \qquad (2)$$

The final classification of an object o_i ($\lambda(o_i)$) using GFHF is given by the Class Mass Normalization concept:

$$\lambda(o_i) = \text{arg max}_{1 \le m \le |\mathcal{C}|} Pr[c_m] \cdot \frac{f_{i,l}(\mathcal{O}^U)}{\sum\limits_{o_j \in \mathcal{O}} f_{j,l}(\mathcal{O})} \qquad (3)$$

in which $Pr[c_m]$ is the probability to occur the class c_m in the labeled objects.

The LLGC algorithm considers the degree of each node ($d_i = \sum_{o_j \in \mathcal{O}} w_{i,j}$) in the distance function, decreasing the importance of objects with many connections, and μ does not tend to infinity. Therefore, the function to be minimized by LLGC is given by:

$$Q(\mathbf{F}) = \frac{1}{2}\sum_{o_i,o_j \in \mathcal{V}} w_{i;j}\left|\left|\frac{1}{\sqrt{d_i}}\mathbf{f}_i - \frac{1}{\sqrt{d_j}}\mathbf{f}_j\right|\right|^2 + \mu \sum_{o_i \in \mathcal{O}^L} ||\mathbf{f}_i - \mathbf{y}_i||^2 \qquad (4)$$

The final classification of a network object using LLGC is given through the maximum argument:

$$\lambda(o_i) = \text{arg max}_{1 \le m \le |\mathcal{C}|} f_{i,m} \qquad (5)$$

V. Experimental evaluation

In order to model the ATE task as a transductive learning solution, we consider each word in the input texts of a specific domain (except the stopwords) and their respective calculated features as an instance (candidate term). The classification predicts which words (unigrams) are terms. We also tested different sets of features (called here attribute selection), presented in Section III, in order to identify which of them provide better results for TLATE. Different sets of instances (called candidate selection) were also tested to verify if the pruning of candidates might improve the classification quality. Experiments were carried out to evaluate which algorithm for transductive learning and which number of labeled instances are more adequate for TLATE.

A. Text Preprocessing and Feature and Candidate Selections

For the experiments, we used a specific domain corpus in the Portuguese language that was the ECO[6] corpus [32]. ECO contains 390 texts of the ecology domain and its gold standard is composed of 322 unigrams that correspond to the intersection of terms extracted from 2 books, 2 glossaries, and 1 online dictionary, about the same domain. We also annotated the ECO corpus using the PALAVRAS[7] parser [33].

[6]ECO corpus - http://www.nilc.icmc.usp.br/nilc/projects/bloc-eco.htm

[7]As all NLP tool for general domains, PALAVRAS is not excellent for specific domains. However, as it would be expensive (time and manual work) to customize it for the ecology domain, we used it even though there are error tagging.

The preprocessing of the ECO corpus consisted of normalizing its words using the stemming[8] technique and removing[9] 531 stopwords in BP. Although our test corpus is in BP, we also removed 601 stopwords in English, because there were, e.g., many book names in English that contain stopwords, such as "the" and "and".

Taking into account that the terms are normally nouns, we considered all nouns, except stopwords, as candidate terms of the domain. For each candidate, we implemented and calculated 25 features, detailed in Section III. For the NP, HNP, POS, C-value, and NC-value features, we used the annotated ECO corpus. For the GC, Freq_GC, TDS, THD, GlossEx, and Weirdness features, we used the NILC corpus[10] as a general corpus in the Portuguese language. This general corpus contains 40 million words. For the IP feature, we considered the 40 indicative phrases created in [11]. NP, HNP, POS, IP, and GC are discrete features (0 or 1). For example, if the POS pattern of a word is noun, the POS feature will be 1; otherwise, it will be 0. All other features are continuous (a real value).

In order to identify which features[11] are more adequate for ATE, we carried out 8 different ***attribute selection methods***. The attribute set composed of all 25 calculated features was called **All_Feat**. According to our classifying about which approach each feature belongs to, described in Section III, the set that has only linguistic features was called **L_Feat.**, the attribute set composed of only statistical features was called **S_Feat.**, and the set composed of just hybrid features was called **H_Feat**. Moreover, over the All_Feat set, we applied 2 attribute selection methods, available in WEKA [34]. Their selection criteria is based on consistency (CBF) and correlation (CFS). We combined these methods with 2 search methods: Random Search Filter (R) and BestFirst (BF). These methods return feature sets that are considered the most representative to characterize the terms. The attribute sets selected by each of these methods are called **CFS_R, CFS_BF, CBF_R**, and **CBF_BF**.

Regarding the candidate terms, when we mention **All_CTs** we refer to all the candidates. Since there is a huge number of candidates, for each one of these selected attribute sets, we also tested 2 different ***candidate selection methods***. The first candidate set has only the nouns that occur in at least two documents in the corpus (called here **DF**). In the second one (called **DF_NP**), the candidates must occur in at least two documents in the corpus and also appear in at least one noun phrase.

When selecting the candidates, i.e., decreasing the candidate number, the ATE process will require less time to run it. Nevertheless, when removing candidates, it might also remove true terms. Figure 1 presents the number of candidates in each set and the number of true terms that were removed from each set. The true terms are the words of the gold standard of the ECO corpus. After stemming the 322 true terms from the gold standard, we obtained 300 stems of true terms. We observe in Figure 1 that, when selecting candidates using DF and DF_NP, we decreased, respectively, 45.35% and 72.64% in relation to the number of all the obtained candidates and removed only 9.67% and 15.33% of the true terms, respectively.

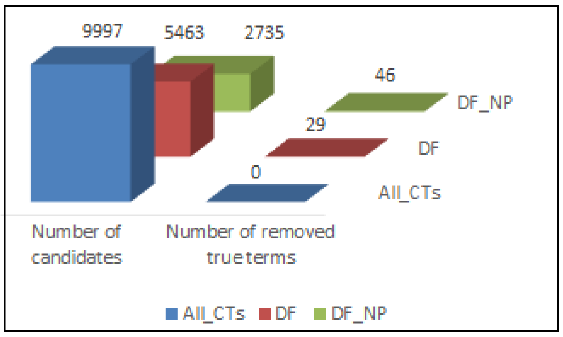

Fig. 1. Number of candidates and true terms in each set. All of them were normalized with a stemming technique.

B. Experiment Configuration and Evaluation Criteria

Some of the parameter values used in the experiments were based on values found in description of the algorithms [29], [30]. Other parameter values were defined based on empirical evaluations. Exp and mutual KNN networks were used to generate the networks. The Euclidian distance was used to compute the similarity among the instances of a corpus.

Exp networks were generated using $\sigma = \{0.05, 0.20, 0.35, 0.50, 0.75, 0.90, 1.05, 1.20, 1.35, 1.50\}$ and KNN networks were generated using $K = \{1, 7, 17, 37, 57\}$. In the literature, the exp networks were used with the GFHF and LLGC algorithms [29], [30]. Because of that, we also used the exp networks with these algorithms. However, KNN networks were used only with the LLGC algorithm, since we empirically verified that the exp networks obtain better results. For LLGC, we used the parameter $\alpha = \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

We used the iterative solutions presented in [29] for GFHF and in [30] for LLGC. The maximum number of iterations was set to 1000 since this is a common threshold for iterative solutions. We carried out experiments using 1, 10, 20, 30, 40, and 50 labeled instances for terms and non-terms.

The evaluation measures used in this paper are based on a contingency matrix presented in Table I. The meaning of the cells of this matrix are: number of non-terms predicted as non-terms (TN - true non-terms), number of non-terms predicted as terms (FT - false terms), number of terms predicted as non-terms (FN - false non-terms), and number of terms predicted as terms (TT - true terms).

TABLE I. CONTINGENCY MATRIX.

Predicted			
Non-Term	Term		
TN	FT	Non-Term	Real
FN	TT	Term	

Considering the cells of the contingency matrix, we used as evaluation measure the precision ($P = \frac{TT}{TT+FT}$), recall ($R = \frac{TT}{TT+FN}$), f-measure ($FM = \frac{2*P*R}{P+R}$), and accuracy ($Acc = \frac{TT+TN}{TT+TN+FT+FN}$). Precision, recall, and f-measure

[8]We used PTStemmer, which is a stemming toolkit for the Portuguese language. - http://code.google.com/p/ptstemmer/

[9]These stoplists include the verb "to be", punctuation, numbers, accents, the words composed of only one character, and no significant words for the domain corpus, such as "that", "figure", and "introduction". They are available in our repository: http://sites.labic.icmc.usp.br/merleyc/stoplist/stopPortIngl.txt

[10]NILC Corpus - http://www.nilc.icmc.usp.br/nilc/tools/corpora.htm

[11]When we refer to *features*, we mean *attributes* from the ML area.

were calculated considering just the classification of terms as target, i.e., we used just the TT as correct classification. On the other hand, TT and TN are considered in the accuracy measure. The items of the contingency matrix were obtained considering the sum of 10 runs with different labeled instances in each run. The algorithms used in the experiments were applied to the same labeled and unlabeled instances.

C. Results and Discussion

For some applications, the user may be interested in maximizing some evaluation measure. For instance, the user may be interested in extracting all the true terms, even that the extracted word set contains non-terms (recall maximization). On the other hand, the user may be interested in extracting just the true terms, even though the extracted word set does not contain all true terms (precision maximization). The user may desire a balance between the recall and precision maximization (f-measure maximization) or to maximize the number of terms and non-terms correctly extracted (accuracy maximization). Tables II, III, IV, and V present the results of the Top-10 precision (P), recall (R), f-measure (FM), and accuracy (Acc), respectively. We also present the attribute selection, candidate selection, algorithm (Alg.), number of labeled words ($|\mathcal{O}^L|$), network parameter (σ/k), and algorithm parameter (μ).

TABLE II. PRECISION MAXIMIZATION.

Attribute Selection	Candidate Selection	Alg.	$\|\mathcal{O}^L\|$	σ/k	μ	P	R	FM	Acc
S_Feat.	DF_NP	LLGC	50	1.50	0.90	**0.64**	0.01	0.02	0.92
S_Feat.	DF_NP	LLGC	50	1.35	0.90	**0.58**	0.01	0.02	0.92
S_Feat.	DF_NP	LLGC	50	1.05	0.90	**0.55**	0.01	0.02	0.92
All_Feat.	DF	LLGC	40	0.90	0.90	**0.42**	0.06	0.10	0.95
S_Feat.	DF_NP	LLGC	30	1.20	0.90	**0.40**	0.02	0.03	0.92
S_Feat.	DF_NP	LLGC	30	1.50	0.90	**0.40**	0.01	0.03	0.92
S_Feat.	DF_NP	LLGC	30	0.90	0.90	**0.40**	0.02	0.04	0.92
All_Feat.	DF	LLGC	40	0.75	0.90	**0.40**	0.13	0.19	0.94
All_Feat.	DF	LLGC	30	0.90	0.90	**0.39**	0.07	0.11	0.94
All_Feat.	DF	LLGC	50	0.90	0.90	**0.38**	0.07	0.12	0.95

TABLE III. RECALL MAXIMIZATION.

Attribute Selection	Candidate Selection	Alg.	$\|\mathcal{O}^L\|$	σ/k	μ	P	R	FM	Acc
L_Feat.	DF	LLGC	40	0.50	0.90	0.05	**1.00**	0.10	0.07
L_Feat.	DF	LLGC	30	0.50	0.90	0.05	**1.00**	0.10	0.07
L_Feat.	DF	LLGC	30	0.75	0.90	0.05	**1.00**	0.10	0.06
L_Feat.	All_CTs	LLGC	20	0.90	0.90	0.03	**1.00**	0.06	0.07
L_Feat.	All_CTs	LLGC	20	0.90	0.90	0.03	**1.00**	0.06	0.07
L_Feat.	All_CTs	LLGC	10	1.35	0.90	0.03	**1.00**	0.06	0.07
L_Feat.	All_CTs	LLGC	10	1.35	0.90	0.03	**1.00**	0.06	0.07
L_Feat.	All_CTs	LLGC	10	1.20	0.90	0.03	**1.00**	0.06	0.07
L_Feat.	All_CTs	LLGC	10	1.20	0.90	0.03	**1.00**	0.06	0.07
L_Feat.	All_CTs	LLGC	20	1.05	0.90	0.03	**1.00**	0.05	0.03

TABLE IV. F-MEASURE MAXIMIZATION.

Attribute Selection	Candidate Selection	Alg.	$\|\mathcal{O}^L\|$	σ/k	μ	P	R	FM	Acc
CBF_R	DF	LLGC	30	0.50	0.90	0.23	0.33	**0.27**	0.90
CBF_R	DF	LLGC	40	0.50	0.90	0.22	0.33	**0.27**	0.91
All_Feat.	DF	LLGC	30	0.50	0.90	0.29	0.24	**0.26**	0.93
S_Feat.	DF_NP	LLGC	20	0.20	0.90	0.23	0.30	**0.26**	0.85
S_Feat.	DF_NP	GFHF	20	∞	0.35	0.21	0.33	**0.25**	0.83
S_Feat.	DF_NP	GFHF	20	∞	0.50	0.21	0.33	**0.25**	0.83
S_Feat.	DF_NP	GFHF	20	∞	0.20	0.20	0.33	**0.25**	0.83
S_Feat.	DF_NP	LLGC	20	1.50	0.50	0.20	0.33	**0.25**	0.83
S_Feat.	DF_NP	LLGC	20	1.35	0.50	0.20	0.33	**0.25**	0.83
S_Feat.	DF_NP	LLGC	20	1.20	0.50	0.20	0.33	**0.25**	0.83

TABLE V. ACCURACY MAXIMIZATION.

Attribute Selection	Candidate Selection	Alg.	$\|\mathcal{O}^L\|$	σ/k	μ	P	R	FM	Acc
All_Feat.	DF	LLGC	50	1.35	0.9	0.33	0.03	0.05	**0.95**
All_Feat.	DF	LLGC	50	0.9	0.9	0.38	0.07	0.12	**0.95**
All_Feat.	DF	LLGC	40	1.5	0.9	0.33	0.02	0.03	**0.95**
All_Feat.	DF	LLGC	40	0.9	0.9	0.42	0.06	0.10	**0.95**
All_Feat.	DF	LLGC	50	0.75	0.9	0.35	0.12	0.18	**0.95**
All_Feat.	DF	LLGC	30	1.35	0.9	0.35	0.03	0.05	**0.94**
All_Feat.	DF	LLGC	40	0.75	0.9	0.40	0.13	0.19	**0.94**
All_Feat.	DF	LLGC	30	1.05	0.9	0.37	0.04	0.08	**0.94**
All_Feat.	DF	LLGC	30	0.9	0.9	0.39	0.07	0.11	**0.94**
All_Feat.	DF	LLGC	30	0.75	0.9	0.38	0.11	0.17	**0.94**

Considering attribute selection, when using features of different levels of knowledge (linguistic, statistical, and hybrid) together (All_Feat.), we maximized the accuracy measure. This result was expected because the joining of these features were also relevant when extracting terms using inductive learning [11]. The use of L_Feat. maximized the recall values. The top f-measure values were similar when using CBF_R, All_Feat., and S_Feat. The use of S_Feat. was useful for maximizing precision.

Taking into account the candidate selection, the use of All_CTs maximized the recall values and DF-NP maximized the precision and f-measure values. DF was the unique candidate selection method that maximized all the evaluation measures. This fact might be due to the fact that CTs probably better represent the corpus than the ones that are present in a single document. We highlighted it is better to use DF or DF_NP instead of All_CTs, since DF and DF_NP achieved the same results using, respectively, 45.35% and 72.64% less words than the number of all the obtained candidates (All_CTs.).

LLGC provided the maximization of all evaluated measures, while GFHF provided good results to maximize f-measure. We believe that LLGC obtained better results than GFHF because of two reasons: i) LLGC does not allow objects with high degree to dominate the label spreading and ii) LLGC allows to change the class information of labeled objects during the classification process.

Considering the number of words labeled as terms and non-terms for TLATE, 10 words labeled as terms and 10 as non-terms were sufficient to maximize recall, 20 for f-measure, 30 or 40 for accuracy, and 50 for precision.

Finally, we compared our results with ATE from the ECO corpus using inductive learning. For that, we compared the best results of each evaluation measure obtained by Conrado et al. [11] with ours. TLATE (our results) achieved better values of precision (64% - 1^{st} line of Table II), recall (100% - 1^{st} line of Table III), and f-measure (27% - 1^{st} line of Table IV) when comparing with precision (60%), recall (21%), and f-measure (24%) obtained using inductive learning in [11].

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we applied transductive learning for automatic term extraction (TLATE). We used features of different levels of knowledge (linguistic, statistical, hybrid, and all of them) to characterize terms and non-terms. For the ecology domain, linguistic features should be used to maximize recall and all of the features should be used to maximize accuracy. Statistical features are better to maximize precision. F-measure is maximized using all or only statistical features. In this case, we advise the use of statistical features since they spend less time to be calculated and are language independent.

We also applied different candidate selection methods and traditional transductive learning algorithms to perform automatic term extraction (ATE). TLATE is worthwhile for ECO corpus when compared with the use of ATE using inductive learning [11], the use of TLATE is worthwhile. We achieved better results of precision, recall, and f-measure using TLATE when comparing with results of Conrado et al. [11]. Futhermore, TLATE needs only a little number of labeled words to extract terms, instead of ATE with inductive learning. Considering We highlight all the discussions presented in this paper concern about the ecology domain. As future work, we intend to apply TLATE to other corpora and domains. Moreover, as the number of terms is usually much smaller than the number of non-terms in a corpus, we intend to adapt the transductive learning algorithm to consider this fact.

Acknowledgment

Grants 2009/16142-3, 2011/12823-6, 2011/19850-9, 2012/03071-3, and 2012/09375-4 Sao Paulo Research Foundation (FAPESP).

References

[1] M. T. Cabré, R. Estopà, and J. Vivaldi, "Automatic term detection: a review of current systems," in *Recent Advances in Computational Terminology*, D. Bourigault, C. Jacquemin, and M.-C. L'Homme, Eds. Amsterdam/Philadelphia: John Benjamins, 2001, pp. 53–88.

[2] K.-H. Yang, C.-Y. Chen, H.-M. Lee, and J.-M. Ho, "EFS: Expert finding system based on wikipedia link pattern analysis," in *SMC*, 2008, pp. 631–635.

[3] A. Jimeno-Yepes and A. Aronson, "Query expansion for UMLS metathesaurus disambiguation based on automatic corpus extraction," in *ICMLA*, 2010, pp. 965–968.

[4] N. Ni, K. Liu, and Y. Li, "An automatic multi-domain thesauri construction method based on lda," in *ICMLA*, vol. 2, 2011, pp. 235–240.

[5] D. Wang, S. Zhu, and T. Li, "SumView: A web-based engine for summarizing product reviews and customer opinions," *Expert Systems with Applications*, vol. 40, no. 1, pp. 27–33, 2013.

[6] M. T. Pazienza, M. Pennacchiotti, and F. M. Zanzotto, "Terminology extraction: An analysis of linguistic and statistical approaches," in *Knowledge Mining Series: Studies in Fuzziness and Soft Computing*, S. Sirmakessis, Ed. Springer Verlag, 2005, pp. 255–279.

[7] K. T. Frantzi, S. Ananiadou, and J. I. Tsujii, "The C-value/NC-value method of automatic recognition for multi-word terms," in *Proc 2nd ECDL*. London, UK: Springer-Verlag, 1998, pp. 585–604.

[8] L. Kozakov, Y. Park, T. Fin, Y. Drissi, Y. Doganata, and T. Cofino, "Glossary extraction and utilization in the information search and delivery system for IBM technical support," *IBM Systems Journal*, vol. 43, no. 3, pp. 546–563, Jul. 2004.

[9] C. Kit and X. Liu, "Measuring mono-word termhood by rank difference via corpus comparison," *Terminology*, vol. 14, no. 2, pp. 204–229, 2008.

[10] J. Vivaldi, L. A. Cabrera-Diego, G. Sierra, and M. Pozzi, "Using wikipedia to validate the terminology found in a corpus of basic textbooks," in *Proc 8th Int. CNF on LREC*, N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, and D. Tapias, Eds. Istanbul, Turkey: ELRA, 2012, pp. 3820–3827.

[11] M. S. Conrado, T. A. S. Pardo, and S. O. Rezende, "A machine learning approach to automatic term extraction using a rich feature set," in *Proc 2013 NAACL HLT Student Research Wksp*, Atlanta, USA, 2013, pp. 16–23.

[12] Z. Zhang, J. Iria, C. Brewster, and F. Ciravegna, "A comparative evaluation of term recognition algorithms," in *Proc 6th on LREC*, N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, and D. Tapias, Eds. Marrakech, Morocco: ELRA, 2008, pp. 2108–2113.

[13] J. Vivaldi, L. Màrquez, and H. Rodríguez, "Improving term extraction by system combination using boosting," in *Proc 12th European CNF on Machine Learning*. London, UK: Springer-Verlag, 2001, pp. 515–526.

[14] N. Loukachevitch, "Automatic term recognition needs multiple evidence," in *Proc 8th on LREC*, N. Calzolari, K. Choukri, T. Declerck, M. Dogan, B. Maegaard, J. Mariani, Odijk, and S. Piperidis, Eds. Istanbul, Turkey: ELRA, 2012, pp. 2401–2407.

[15] R. Nazar, "A statistical approach to term extraction," *Int. Journal of English Studies*, vol. 11, no. 2, pp. 159–182, 2011.

[16] J. Foo and M. Merkel, "Using machine learning to perform automatic term recognition," in *Proc 7th LREC - Wksp on Methods for automatic acquisition of Language Resources and their Evaluation Methods*, N. Bel, B. Daille, and A. Vasiljevs, Eds., 2010, pp. 49–54.

[17] X. Zhang, Y. Song, and A. Fang, "Term recognition using conditional random fields," in *Proc of IEEE NLP-KE*, 2010, pp. 333–336.

[18] K. W. Church, "One term or two?" in *Proc 18th Annual Int. ACM SIGIR CNF on Research and Development in Information Retrieval*, ser. SIGIR. New York, NY, USA: ACM, 1995, pp. 310–318.

[19] G. Salton and C. Buckley, "Term weighting approaches in automatic text retrieval," Ithaca, NY, USA, Tech. Rep., 1987.

[20] T. Liu, S. Liu, and Z. Chen, "An evaluation on feature selection for text clustering," in *Proceedings of the 10th Int. CNF on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann, 2003, pp. 488–495.

[21] L. Liu, J. Kang, J. Yu, and Z. Wang, "A comparative study on unsupervised feature selection methods for text clustering," in *Proc of IEEE NLP-KE*, 2005, pp. 597–601.

[22] I. Dhillon, J. Kogan, and C. Nicholas, "Feature selection and document clustering," in *Survey of Text Mining*, M. W. Berry, Ed. Springer, 2003, pp. 73–100.

[23] Y. Park, S. Patwardhan, K. Visweswariah, and S. C. Gates, "An empirical analysis of word error rate and keyword error rate," in *INTERSPEECH*. ISCA, 2008, pp. 2070–2073.

[24] K. Ahmad, L. Gillam, and L. Tostevin, "University of surrey participation in trec8: Weirdness indexing for logical document extrapolation and retrieval (WILDER)," in *TREC*, 1999, pp. 1–8.

[25] A. Barrón-Cedeño, G. Sierra, P. Drouin, and S. Ananiadou, "An improved automatic term recognition method for spanish," in *Proc 10th Int. CICLing*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 125–136.

[26] V. Vapnik, *Statistical learning theory*. Wiley, 1998.

[27] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*. MIT Press, 2006.

[28] X. Zhu, "Semi-supervised learning with graphs," Ph.D. dissertation, Carnegie Mellon University, 2005.

[29] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proc Int. CNF on Machine Learning*. AAAI Press, 2003, pp. 912–919.

[30] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in Neural Information Processing Systems*, vol. 16, 2004, pp. 321–328.

[31] X. Zhu, "Semi-supervised learning literature survey," Computer Sciences, University of Wisconsin-Madison, Tech. Rep. 1530, 2005.

[32] C. Zavaglia, L. H. M. Oliveira, M. G. V. Nunes, and S. M. Aluísio, "Estrutura ontológica e unidades lexicais: uma aplicação computacional no domínio da ecologia," in *Proc 5th TIL Wksp*. RJ, Brazil: SBC, 2007, pp. 1575–1584.

[33] E. Bick, *The Parsing System "PALAVRAS". Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework*. Aarhus Universitetsforlag, 2000.

[34] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," in *SIGKDD-ACM*, vol. 11, 2009, pp. 10–18.