Pattern Recognition Letters 87 (2017) 127–138

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Optimizing the class information divergence for transductive classification of texts using propagation in bipartite graphs☆

Thiago de Paulo Faleiros*, Rafael Geraldeli Rossi, Alneu de Andrade Lopes

Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo – Campus de São Carlos, Caixa Postal 668, 13560-970 São Carlos SP, Brazil

ARTICLE INFO

Article history:
Available online 23 April 2016

Keywords:
Text classification
Transductive learning
Graph-based learning
Text mining
Label propagation
Bipartite graphs

ABSTRACT

Transductive classification is an useful way to classify a collection of unlabeled textual documents when only a small fraction of this collection can be manually labeled. Graph-based algorithms have aroused considerable interests in recent years to perform transductive classification since the graph-based representation facilitates label propagation through the graph edges. In a bipartite graph representation, nodes represent objects of two types, here documents and terms, and the edges between documents and terms represent the occurrences of the terms in the documents. In this context, the label propagation is performed from documents to terms and then from terms to documents iteratively. In this paper we propose a new graph-based transductive algorithm that use the bipartite graph structure to associate the available class information of labeled documents and then propagate these class information to assign labels for unlabeled documents. By associating the class information to edges linking documents to terms we guarantee that a single term can propagate different class information to its distinct neighbors. We also demonstrated that the proposed method surpasses the algorithms for transductive classification based on vector space model or graphs when only a small number of labeled documents is available.

1. Introduction

The most common way of storing information is in textual format. In fact, a huge amount of data produced and stored every day are textual documents. In this scenario, automated techniques to help classify textual documents is one of the most important tasks to organize, manage, and extract knowledge from these data and still remain as worthwhile research topics for machine learning and data mining communities.

The task of text classification is usually carried out by inductive learning algorithms which aim to induce a model to classify new or unseen documents. A considerable number of labeled documents is necessary to create an accurate classification model. However, a consistent set of labeled documents to induce a classification model is not available in most of the real applications, since the labeling task be an expensive, time consuming and expert dependent task. Thus, a more practical way of approaching text classification in real applications is to employ methods which make use of a small set of labeled documents with unlabeled documents (a large set is usually available).

Transductive approaches are widely used when labeled training data are insufficient and the goal is to classify a known set of documents. In this case, they make use of unlabeled data to improve classification performance [3,7,11,12,19]. Transductive classification directly estimates the labels of unlabeled instances without creating a model to classify new texts.

In addition to the classifier problem, another issue of the text classification task is the data representation. The traditional representation of textual documents is the vector space model (VSM). Nevertheless, more expressive representations, such as homogeneous or heterogeneous graphs may be employed. In a homogeneous graph, only link between objects of the same type is allowed, therefore it contains only document-document or term-term relations. In a heterogeneous graph, only link between pair of objects of different types is allowed. An intuitive way to represent a collection of documents is by creating a bipartite graph where vertices correspond to documents and terms, and edges represent the occurrence of the word in the document. Eventually, a weight can be assigned to the edge according to the frequency of the word in the document.

To describe algorithms over a graph representation has several advantages, since graph representation: (1) avoids sparsity and ensures low memory consumption; (2) enables easy description of operations for the inclusion of the topological structure of a dataset; (3) enables an optimal description of the topological

☆ This paper has been recommended for acceptance by Cheng-Lin Liu.

* Corresponding author. Tel.: +551633739678.
E-mail address: thiagopf@icmc.usp.br (T. de Paulo Faleiros).

http://dx.doi.org/10.1016/j.patrec.2016.04.006
0167-8655/

structure of a dataset; (4) provides local and global statistics of the datasets structure; and (5) allows extracting patterns which are not extracted by algorithms based on vector-space model [6].

Graph-based algorithms are mostly used in a label propagation schema in which some labeled objects propagate their labels to other objects through the graph structure to perform transductive classification [4,21,32,34]. Appropriate use of the richness of information conveyed by these graphs can lead to label propagation algorithms that using just few labeled examples surpass the classification performance of inductive classifiers using a large number of labeled examples [19].

Here, we propose an algorithm that uses a bipartite heterogeneous graph representation. The rationale behind the proposed graph-based semi-supervised algorithm is to associate class information to each vertex and edge. The class information is a l-dimensional vector, where l is the number of classes. Our algorithm is an iterative propagation procedure, in which the class information related to a vertex influences their neighbors' labels until convergence, by using a label propagation schema. In our proposal, and due to the characteristic of bipartite heterogeneous graph, the documents propagate their class information to edges, and edges propagate their class information back to documents and terms. Traditional label propagation algorithms optimize the class information vectors considering each dimension independently whereas our propagation algorithm optimizes the divergence between closely related vertices considering all dimensions of the class information vectors. The divergence between class information is optimized by maximizing the generalized Kullback–Leibler divergence [17]. The proposed algorithm, named TPBG (Transductive Propagation in Bipartite Graph) obtains better classification performance than state-of-the-art transductive algorithms based on vector space or graphs models when only a small number of labeled documents is available.

The main contributions of this paper are threefold: (1) to bring the advantages of bipartite graph representation and iterative propagation to the semi-supervised transductive learning process; (2) to propose an algorithm which surpasses the classification accuracy of state-of-the-art algorithms based on vector space model or graphs when considering a small number of labeled documents; (3) to carry out a comprehensive comparative evaluation of our proposal.

In the experimental evaluation we also present the behavior of the algorithms for a different range of labeled documents. The results showed that our algorithm returns consistent results, which makes the method a competitive alternative and a new exploratory possibility to state-of-the-art of semi-supervised algorithms.

The remainder of this paper is organized as follows. Section 2 presents related works about transductive classification. Section 3 presents details on the proposed algorithm for transductive classification of texts using bipartite graphs. Section 4 presents details of the experimental evaluation and the results. Finally, Section 5 presents the conclusions and points to future work.

2. Related works

Differently from supervised inductive classification, which aims to learn a model from labeled examples, the goal of transductive learning is to predict the class labels of the given labeled and unlabeled examples. In the context of document classification, transductive learning assigns weights for each dimension of the class information vector of each document and the documents are classified considering these weights.

In general, there are two ways to represent text collections to perform transductive learning: vector-space model and graph based representations. In the vector-space model, documents are represented as vectors and each dimension correspond to term of the document collection. The values in the vectors are based on the frequency of a term, such as binary weights, term frequency (tf) or term frequency-inverse document frequency (tf-idf). In the graph based representation, the objects corresponding to documents or terms are represented as vertices, and the relationship between pairs of objects are represented by edges. Different types of objects and different relations can be used to generate a graph-based representation. Documents can be connected according to "explicit relations" as hyperlinks or citations [15,25], or considering similarity [2]. Terms can be connected by precedence in text [1], if they present syntactic/semantic relationship [24], or if they co-occur in text collection or in pieces of texts as sentences/windows [13,16,27,29]. A combination of different types of objects is also used. In this case, documents and terms generate a bipartite graph where terms are connected to documents in which they occur [8,19,21].

2.1. Transductive learning on vector space model

The first proposals on transductive learning for text classification considered text collections represented by vector space model [5,11,14,30]. Perhaps, the most natural way to perform transductive learning is through Self-Training. Self-Training assumes that the most confident classifications are correct and re-induce the model by adding these new labeled instances to the training set.

Support Vector Machines (SVM) are one of the most popular classification algorithms used in machine learning. Its transductive version, Transductive Support Vector Machine (TSVM), have been used for text classification [11]. TSVM considers labeled and unlabeled documents to obtain a maximal margin hyperplane. The coefficients of a hyperplane correspond to the class information of terms. Based on the assumption that the classes are well-separated, the hyperplane with maximal margin will fall into a low density region. When this assumption does not hold, the TSVM classification algorithm is not accurate.

Transductive learning can also be performed by a probabilistic model. In [14] is presented a probabilistic framework which uses unlabeled data to improve a text classifier. A Expectation Maximization (EM) algorithm based in Multinomial Naive Bayes is used to estimate maximum a posteriori probability. The EM algorithm performs two steps. In the E-Step, the naive Bayes parameters, θ, is used to estimate the component membership of each unlabeled document. In the M-Step, the parameter θ is re-estimated using all the documents and is established the class information for terms. EM classification is not accurate if the generative assumption is violated.

2.2. Transductive learning on graph

Here we define a graph as a triple $G = (\mathcal{V}, \mathcal{E}, f)$, where $\mathcal{V}$ is a set of vertices, $\mathcal{E}$ is a set of edges, and f is a mapping which associate an edge to a real number, *i.e.* $f : \mathcal{E} \rightarrow \mathbb{R}$. To simplify notation we denote $f(e_{j,i})$ as $f_{j,i}$ for $e_{j,i} \in \mathcal{E}$. When $\mathcal{V}$ is compounded by a single type of objects, the graph is called homogeneous graph. When $\mathcal{V}$ is compounded by h different types of objects, *i.e.*, $\mathcal{V} = \{V_1 \cup \ldots \cup V_h\}$, the graph is called heterogeneous graph [10]. To create a graph in a textual context, the vertices can be associated to documents, terms, pieces of a text, sentences or paragraphs, and all these objects can be combined in pairs to describe an edge. Usually, homogeneous networks are created considering explicit relations between pairs of documents [15,25], or considering similarity metric between documents [2]. With respect to heterogeneous graph in textual context, terms can be connected to documents [8,20,21] or sentence [28] in which they occur.

The main algorithms for transductive learning based on graphs aims to maximize a general regularization function. To explain such function, let l be the number of classes and Y_i be a l-dimensional vector associated to labeled object $v_i \in \mathcal{V}$. We denote $Y_{i,k}$ as the kth dimension of vector Y_i. Suppose that c_k $(0 \leq k \leq l)$ is the class label associated to v_i, then $Y_{i,k} = 1$ and $Y_{i,m} = 0$ for all $m \neq k$. Considering a given small subset of labeled objects $\mathcal{V}^l \subset \mathcal{V}$, we describe the general regularization function as

$$Q(G) = \frac{1}{2} \sum_{e_{j,i} \in \mathcal{E}} f_{j,i} \Omega(R_j, R_i) + \mu \sum_{v_i \in \mathcal{V}^l} \Omega'(R_i, Y_i), \quad (1)$$

where R_i is a l-dimensional vector associated to each vertex $v_i \in \mathcal{V}$, it contains the class information of each vertex – each dimension of vector R_i corresponds to the membership level of vertex v_i to a class. The functions $\Omega(\cdot)$ and $\Omega'(\cdot)$ are metrics that returns the similarities between objects represented in graph G.

The objective function in Eq. (1) was based on two assumptions. The first assumption states that the class information values among neighbors must be close. The second assumption requires that the information assigned during the classification process must be close to the real class information. The parameter μ control the influence of the second assumption, *i.e.* how much the labeled objects must keep their class information.

Based on assumptions considered in the optimization problem described in Eq. (1), there are two main algorithms to perform transductive classification on homogeneous graphs: (i) Gaussian Fields and Harmonic Functions (GFHF) [33] and (ii) Learning with Local and Global Consistency (LLGC) [32]. These two algorithms work as a label propagation schema in a homogeneous graphs, where iteratively the labels among the graph are propagate in such a way to minimize Eq. (1).

In a heterogeneous bipartite graph, the documents propagate their labels to terms and the terms propagate their labels to the documents. Considering this approach, the three main algorithms to perform transductive classification on heterogeneous graphs are: (i) Tag-based Model (TB), (ii) GNetMine and (iii) Label Propagation using Bipartite Heterogeneous Networks (LPBHN). TB [31] extends the assumptions of optimization problem in Eq. (1) considering more than one type of object in the graph, and it is used prior knowledge obtained by a domain classifier. GM [10] is based on LLGC algorithm and considers the different types of relations among the different types of objects. The algorithm LPBHN [21] is based on GFHF algorithm and it is a parameter-free algorithm for transductive classification using bipartite graph representation.

3. Proposed algorithm: propagation in bipartite graph for transductive classification

This section we present the problem formulation and general notation, and the mathematical and computational foundations of the proposed semi-supervised algorithm based on a bipartite graph. The TPBG (Transductive Propagation in Bipartite Graph) algorithm is a label propagation algorithm based on the regularization framework described in Eq. (1). Differently from other transductive learning algorithms based on graphs, we consider the KL-divergence as similarity measure to minimize the divergence among class information of documents, terms and and their links in the transductive classification process.

3.1. Problem formulation and general notation

The collection of documents is represented by a bipartite graph. In the bipartite graph representation $G = (\mathcal{V} = \{\mathcal{D} \cup \mathcal{W}\}, \mathcal{E}, f)$, the vertex set $\mathcal{V} = \{\mathcal{D} \cup \mathcal{W}\}$ corresponds to documents (vertices in $\mathcal{D}$) and terms (vertices in $\mathcal{W}$) while edges in $\mathcal{E}$ represent document-term pairs, linking a vertex in $\mathcal{D}$ to a vertex in $\mathcal{W}$. To each edge $e_{j,i}$ it is associated a non-negative value $f_{j,i}$, where $f_{j,i}$ is given by the frequency of occurrence of term w_i in the document d_j. To the vertices $d_j \in \mathcal{D}$ and $w_i \in \mathcal{W}$, and edges $e_{j,i} \in \mathcal{E}$ are associated respectively the class information vectors A_j, B_i and $C_{j,i}$.

Let $\mathbb{C} = \{c_1, \ldots, c_l\}$ represent the set of l class labels, let $\mathcal{W} = \{w_1, \ldots, w_n\}$ be the set of n unique terms, and let $\mathcal{D} = \{d_1, \ldots, d_m\}$ be the set of m documents of a text collection. In a transductive learning scenario, $\mathcal{D} = \mathcal{D}^l \cup \mathcal{D}^u$, in which $\mathcal{D}^l$ represents the set of labeled documents and $\mathcal{D}^u$ represent the set of unlabeled documents.

Finally, let $\mathcal{Y} = \{Y_1, \ldots, Y_m\}$ be the set of l-dimensional vectors, where for each document $d_j \in \mathcal{D}^l$ labeled by a class c_k, the value of kth dimension of Y_j is equal 1 and 0 in other dimensions. We call these vectors as real *class information* vector. The values of Y_j for each $d_j \in \mathcal{D}^u$ are determined after the propagation process. Thus, the goal of transductive learning is to find a function $A : \{\mathcal{D}^l \cup \mathcal{D}^u\} \rightarrow \mathcal{Y}$, in which the unlabeled documents are used to improve the classification.

3.2. Optimizing the divergence between class information vectors

The regularization framework defined in Eq. (1) is used in label propagation algorithms [32–34]. Typically, the Euclidean distance is used to measure the distance between class information of objects represented in the graph. Similarly, we will use this framework to describe our proposed algorithm. However, we use the Kullback–Leibler Divergence as similarity function.

Each class information vectors are closely related to probability distribution, in such a case documents and terms with the same class have low divergence between their probability distribution vectors. Consequently, the Euclidean distance is a poor metric to measure the similarity among neighboring (related) objects. For example, vectors generated by normal overlapping distribution $\mathcal{N}(0, 1000)$ and $\mathcal{N}(10, 10000)$ have an expected Euclidean distance 10. In contrast, the vector generated by distinct (barely overlap) distributions $\mathcal{N}(0, 0.01)$ and $\mathcal{N}(0.1, 0.01)$ will not reflect in the Euclidean distance their distinctions, which is only 0.1. Hence it is expected that a similarity function as Kullback–Leibler is better to distinguish class information.

Label propagation based algorithms for bipartite heterogeneous graphs uses class information of only terms and documents, and assume that the class information of a term w_i in document d_j may be distinct to specify the class in a document $d_j \in \mathcal{D}^U$ [10,21,31]. However, this assumption is not always correct due to terms with multiple meanings in different documents with different classes. Then, in order to overcome this problem, we also associate class information to each pair document-term in vector $C_{j,i}$. This guarantees that the same term w_i can be linked to different documents and propagate different class information to its distinct neighbors.

The assumption of TPBG is that the divergence between class information of documents in $\mathcal{D}^L \cup \mathcal{D}^U$, terms in $\mathcal{W}$ and edges in $\mathcal{E}$ are useful to improve the class information of documents in $\mathcal{D}^U$. Our proposed algorithm propagates the class information of terms and documents to edges, and use the class information of edges to infer the class information of unlabeled documents. Thus, a l-dimensional vector $C_{j,i}$ is used to store the class information of a edge $e_{j,i} \in \mathcal{E}$.

The rationale of our proposed algorithm is that the larger the frequency $f_{j,i}$ the larger should be the agreement between the class information vectors $(A_j \odot B_i)$ and $C_{e_{j,i}}$. Then, basing on Kullback–Leibler divergence, we define the following maximization

function:

$$Q_G(A,B,C) = \sum_{e_{j,i}\in\mathcal{E}} \left(f_{j,i} C_{e_{j,i}} \log \frac{A_j \odot B_i}{C_{e_{j,i}}} \right) + \sum_{d_j \in \mathcal{D}} \mathcal{R}(A_j, \alpha) + \sum_{d_j \in \mathcal{D}^L} Y_j \log \frac{A_j}{Y_j} \quad (2)$$

where $\mathcal{R}(A_j, a)$ are regularization terms for each document d_j, and α is a constant which controls the concentration of class information in the vector.

$$\mathcal{R}(A_j, \alpha) = (\alpha - A_j) \log A_j + A_j (\log A_j - 1). \quad (3)$$

A high α means that each document likely contains a mixture of all classes, and not any single class specifically. A low α value relax such constraints on a document and means that it is more likely that the document may contain a mixture of just a few classes.

The value of $\sum_{d_j \in \mathcal{D}^L} Y_j \log \frac{A_j}{Y_j}$ ensures that the class information assigned during the classification is close to the real class information.

The class information vectors for the whole set of vectors can be obtained by optimizing this equation to each pair of vertices linked by an edge, thus giving rise to the following cost function for the graph G:

$$Q(G) = \arg \max_{A^*, B^*, C^*} \sum_{c_k \in \mathbb{C}} [Q_G(A,B,C)]_k. \quad (4)$$

The induction of the class information of $Q(G)$ is performed using the gradient descent method. The maximum of $Q(G)$ with respect to A_j, B_i, and $C_{e_{j,i}}$, for all document d_j, term w_i and edge $e_{j,i}$ in graph G, are determined by setting the gradient to zero. In order to do so, we first maximize Eq. (2) with respect to $C_{e_{j,i}}$. Here, we constraint the values of vector $C_{e_{j,i}}$ such as $\sum_{c_k \in \mathbb{C}} C_{e_{j,i},k} = 1$. Then, we form the Lagrangian by isolating the terms which contain $C_{e_{j,i}}$ and adding the appropriate Lagrange multipliers.

$$Q_{[C_{e_{j,i}}]} = \left(f_{j,i} C_{e_{j,i}} \log \left(\frac{A_j \odot B_i}{C_{e_{j,i}}} \right) + \lambda \left(\sum_{c_l \in \mathbb{C}} C_{e_{j,i},l} - 1 \right) \right), \quad (5)$$

where we have dropped the arguments of Q for simplicity, and the subscript $[C_{e_{j,i}}]$ denotes that we have retained only those terms in Q that are a function of $C_{e_{j,i}}$. Taking derivatives with respect to $C_{e_{j,i}}$, we obtain:

$$\frac{\partial Q}{\partial C_{e_{j,i}}} = f_{j,i} \left(\log (A_j \odot B_i) - \log (C_{e_{j,i}}) - 1 + \frac{\lambda}{f_{j,i}} \right) \quad (6)$$

Setting this derivative to zero yields the maximizing value of the edges vector $C_{e_{j,i}}$ associated to graph G,

$$C_{e_{j,i}} \propto A_j \odot B_i. \quad (7)$$

As the sum of values in vector $C_{e_{j,i}}$ has to be equal 1, we can normalize it such as

$$C_{e_{j,i}} = \frac{A_j \odot B_i}{\sum_{c_k \in \mathbb{C}} (A_j \odot B_i)_k}. \quad (8)$$

Next, we maximize Eq. (2) with respect to A_j, the vector associated to document $d_j \in \mathcal{D}$. It is not necessary to use Lagrange to constraint the vector A_j because it is constrained by the regularization term. The terms containing A_j are:

$$Q_{[A_j]} = \sum_{w_i \in \mathcal{W}_{d_j}} \left(f_{j,i} C_{e_{j,i}} \log A_j \right) + \mathcal{R}(A_j, \alpha) + \sum_{d_j \in \mathcal{D}^L} Y_j \log \frac{A_j}{Y_j} \quad (9)$$

where the subset $\mathcal{W}_{d_j}$ indicates the set of words linked to document d_j in the bipartite graph G.

We set the values of $A_j = Y_j$ for all d_j in labeled set $\mathcal{D}^L$. On the other hand, for unlabeled documents, we take the derivative with respect to A_j to obtain the following update equation:

$$\frac{\partial Q_{[A_j]}}{\partial A_j} = \frac{1}{A_j} \left(\sum_{w_i \in \mathcal{W}_{d_j}} f_{j,i} C_{e_{j,i}} - A_{j,i} + \alpha \right). \quad (10)$$

Setting this equation to zero yields a maximum at:

$$A_j = \alpha + \sum_{w_i \in \mathcal{W}_{d_j}} f_{j,i} C_{e_{j,i}}. \quad (11)$$

Finally, we maximize Eq. (2) with respect to B_i, the vector associated with words $w_i \in \mathcal{D}$. To maximize with respect to B_i, we isolate terms and add Lagrange multipliers

$$Q_{[B_i]} = \sum_{c_k \in \mathbb{C}} \left(\sum_{d_j \in \mathcal{D}} f_{j,i} C_{e_{j,i}} \log B_i + \lambda_k \left(\sum_{w_p \in \mathcal{W}} B_{v,k} - 1 \right) \right) \quad (12)$$

By taking the derivative $Q_{[B_i]}$, we have

$$\frac{\partial Q_{[B_i]}}{\partial B_{i,k}} = \sum_{d_j \in \mathcal{D}} \sum_{c_k \in \mathbb{C}} \frac{f_{j,i} C_{e_{j,i}}}{B_{j,i}} + \lambda_k \quad (13)$$

Setting this equation to zero, and solving λ_k, such as $\lambda_k = -\sum_{d_j \in \mathcal{D}} \sum_{c_k \in \mathbb{C}} f_{j,i} C_{e_{j,i}}$. Since we have $\sum_{w_i \in \mathcal{W}} B_{i,k} = 1$, we can ignore λ_k to estimate an un-normalized value of $B_{i,k}$

$$\hat{B}_{i,k} \propto \sum_{d_j \in \mathcal{D}} \sum_{c_k \in \mathbb{C}} f_{j,i} C_{e_{j,i}} \quad (14)$$

Normalizing the value of $\hat{B}_{i,k}$ over all word w_p in vocabulary, we have $B_{i,k} = \frac{\hat{B}_{i,k}}{\sum_{p \in \mathcal{W}} \hat{B}_{p,k}}$.

The iterative updates in Eqs. (8), (11) and (14) that minimize Eq. (4) are the basis for the Transductive Propagation in Bipartite Graph (TPBG) algorithm described in the next section.

3.3. The TPBG algorithm

The idea of the TPBG algorithm is to propagate class information throughout vertices' neighborhoods. Assuming that is given a set of labeled documents $\mathcal{D}^L$, i.e. the class information of labeled documents. Assuming also that the class information vectors of terms and unlabeled documents are randomly initialized. The iterative updates are performed in two different manners: (1) local updates, which account for propagations through the neighborhood of each vertex, and (2) global updates, which propagate class information throughout the entire bipartite graph and can be interpreted as a spreading of the information from local to global structures of the bipartite graph. The TPBG algorithm is summarized in Algorithm 1. The local propagation is described in Algorithm 2 and the global propagation is described in Algorithm 3.

The propagation procedure of TPBG algorithm (Algorithm 1) needs as input the set of labeled documents, a bipartite graph G and the concentration parameter α. Initially, for each vertex $d_j \in \mathcal{D}^U$ and $w_i \in \mathcal{W}$ connected by an edge in $\mathcal{E}$, the algorithm randomly initializes the corresponding class information vectors A_j and B_i such that $\sum_{c_k \in \mathbb{C}} A_{j,k} = 1$ for all unlabeled document $d_j \in \mathcal{D}^U$, and $\sum_{w_i \in \mathcal{W}} B_{i,k} = 1$ for all class $c_k \in \mathbb{C}$. The class information vectors of labeled documents $d_j \in \mathcal{D}^L$ are initialized such that $A_j = Y_j$. Then, the local propagation is performed for each edge $e_{j,i}$ incident to the vertex d_j. This procedure creates a l-dimensional vector $C_{e_{j,i}}$ as result of the Hadamard product of A_j and B_i, $C_{e_{j,i}} = A_j \odot B_i$. The class information vector $C_{e_{j,i}}$ is normalized such that $\sum_{c_k \in \mathbb{C}} C_{e_{j,i},k} = 1$. If the document d_j is in labeled set $\mathcal{D}^L$, then $A_j = Y_j$, otherwise,

Algorithm 1: TPBG algorithm.

Input:
bipartite graph G,
$\mathcal{D}^L$ // set of labeled documents
α // concentration parameter
Output:
Y // labels assigned to each document in $\mathcal{D}^U$

begin
  Initialize vector A_j for each document $d_j \in \mathcal{D}$;
  Initialize vector B_i for each word $w_i \in \mathcal{W}$;
  while *convergence* **do**
    foreach $d_j \in \mathcal{D}$ **do**
      repeat
        $A_j \leftarrow$ `localPropag`$(G, d_j, A_j, B, \mathcal{D}^L)$;
      until A_j *convergence*;
    end
    $B \leftarrow$ `globalPropag`(G, A, B);
  end
  for all $d_j \in \mathcal{D}^U$: $\{ Y_{j,k} = 1 \quad | \quad k = \arg\max_{\hat{k}=1}^{l} A_{j,\hat{k}} \}$;
end

Algorithm 2: Local propagation.

function `localPropag`$(G, d_j, A_j, B, \mathcal{D}^L)$
  begin
    foreach *edge* $e_{j,i}$ *incident in* d_j **do**
      $C_{e_{j,i}} \leftarrow \frac{(A_j \odot B_i)}{\sum_{c_k \in \mathbb{C}} (A_j \odot B_i)_k}$;
    end
    if $d_j \in \mathcal{D}^L$ **then**
      $A_j \leftarrow Y_j$;
    else
      $A_j \leftarrow \alpha + \sum_{w_i \in \mathcal{W}_{d_j}} f_{j,i} C_{e_{j,i}}$;
    end
    return A_j;
  end

Algorithm 3: Global propagation.

function `globalPropag`(G, A, B)
  begin
    foreach *vertex* $w_i \in \mathcal{W}$ **do**
      foreach *edge* $e_{j,i}$ *incident in* w_i **do**
        $C_{e_{j,i}} \leftarrow \frac{(A_j \odot B_i)}{\sum_k (A_j \odot B_i)_k}$;
      end
      $B_i \leftarrow \sum_{d_j \in \mathcal{D}} f_{j,i} C_{e_{j,i}}$;
    end
    foreach *vertex* $w_i \in \mathcal{W}$ **do**
      for $c_k \in \mathbb{C}$ **do**
        $B_{i,k} = \frac{B_{i,k}}{\sum_{w_p \in \mathcal{W}} B_{p,k}}$;
      end
    end
    return B;
  end

A_j will receive the class information of vectors $C_{e_{j,i}}$, Eq. 11. The local propagation is repeated for each vertex d_j while entries in A_j are changing. The parameter α was used to control the concentration degree of vector A_j.

The global propagation is performed for all vertex $w_i \in \mathcal{W}$, and for each edge $e_{j,i}$ incident on vertex w_i. This procedure also creates a l-dimensional vector $C_{e_{j,i}}$ given by the Hadamard product of A_j and B_i. Vector $C_{e_{j,i}}$ is normalized such that $\sum_{c_k \in \mathbb{C}} C_{e_{j,i},k} = 1$ and the values propagated back to vectors B_i, as describe in Eq. (14). The class information vectors B_i are normalized over all vertices $w_p \in \mathcal{W}$.

The rationale behind the proposed algorithm is to *locally* concentrate the class information of each word of a document d_j into vector A_j. Then the algorithm *globally* concentrates the influence of all documents into vector B_i. When vectors B_i, for all words w_i, are updated, each entry $k \in \{1..l\}$ of B_i is normalized. This normalization gives the probability of that word assumes the class c_k.

We apply the local and global propagations until a maximum number of iterations is reached or until the class information of unlabeled documents remains the same in two successive iterations.

The complexity of the TPBG algorithm is determined by the maximum number of local propagation T_{local}, the maximum number of interleaving between global and local propagations T, the number of documents m, the number of terms n, the average number of terms per document $\overline{n}$, and the number of classes l. The local propagation is usually fast because it is iterated over the terms in only one document. Thus, the complexity of the algorithm TPBG is $O(T * m * l * ((T_{local} * \overline{n}) + n))$.

4. Experimental evaluation

In the experimental evaluation we have compared TPBG with algorithms presented in Section 2, which consider text collections represented in a vector space model and graphs. We also considered the Multinomial Naïve Bayes and Support Vector Machines, both inductive supervised learning algorithms, to demonstrate if and how much unlabeled documents are useful to improve classification performance. Moreover, our goal is to demonstrate that our proposed algorithm surpasses the classification accuracy obtained by state-of-the-art algorithms for transductive classification of texts. In next sections we present the text collections used in the experimental evaluation, experiment configuration, evaluation criteria, results and discussion. Due to reasons concerning reproducibility, all source codes and text collections used in our experimental evaluation are freely available[1].

4.1. Text collections

We were used 16 textual document collections from different domains: e-mails (EM), medical documents (MD), news articles (NA), scientific documents (SD), sentiment analysis (SA), and web pages (WP). All documents were preprocessed, stopwords were removed, terms were stemmed using the Porter's algorithm [18], HTML tags were removed, and only terms with document frequency greater than 2 were considered. We used term frequency to weight terms in documents to all algorithms except Support Vector Machines, in which we considered term frequency-inverse document frequency (TF-IDF) as term weighting method. Table 1 presents the text collections and the characteristics of these collections: the number of documents ($|\mathcal{D}|$), the number of terms ($|\mathcal{T}|$), the average number of terms per document ($|\overline{\mathcal{T}}|$), the number of

[1] http://sites.labic.icmc.usp.br/thiagopf/prl_2016/

Table 1
Characteristics of the textual document collections.

Collection	$\|\mathcal{D}\|$	$\|\mathcal{T}\|$	$\overline{\|\mathcal{T}\|}$	$\|\mathbb{C}\|$	$\sigma(\mathbb{C})$	$\max(\mathbb{C})$
Classic4 (SD)	7095	7749	35.28	4	1.94	45.16
CSTR (SD)	299	1726	54.27	4	18.89	42.81
Dmoz-Health (WP)	6500	4217	12.40	13	0.00	7.69
Dmoz-Science (WP)	6000	4821	11.52	12	0.00	9.63
Dmoz-Sports (WP)	13500	5682	11.87	27	0.00	3.70
Hitech (NA)	2301	12942	141.93	6	8.25	26.21
La1s (NA)	3204	13196	144.64	6	8.22	29.43
La2s (NA)	3075	12433	144.83	6	8.59	29.43
NFS (CD)	10524	3888	6.65	16	3.82	13.39
Oh0 (MD)	1003	3183	52.50	10	5.33	19.34
Oh10 (MD)	1050	3239	55.64	10	4.25	15.71
Oh15 (MD)	913	3101	59.30	10	4.27	17.20
Oh5 (MD)	918	3013	54.43	10	3.72	16.23
Ohscal (MD)	11162	11466	60.39	10	2.66	14.52
Reviews (NA)	4069	22927	183.10	5	12.80	34.11
WAP (WP)	1560	8461	141.33	20	5.20	21.86

classes ($|\mathbb{C}|$), the standard deviation considering the class percentages in each collection ($\sigma(\mathbb{C})$), and the percentage of the majority class ($\max(\mathbb{C})$)[2].

4.2. Experiment configuration and evaluation criteria

We compared TPBG to traditional and state-of-the-art transductive algorithms based on vector space model and on bipartite graphs. We also ran inductive learning algorithms to verify how much unlabeled documents improve classification performance. The parameters used in the experimental evaluation were based on the values found in the proposal of the algorithms or empirically.

Here we describe the algorithms and its parameters values. The transductive learning algorithms based on vector space model used in this experiments are: **Multinomial Naïve Bayes with Self-Training (ST)**, where the number X of documents with the highest classification was $X \in \{5, 10, 15, 20\}$; **Expectation Maximization (EM)**, in which we considered the EM instantiation for text classification presented in [14], and we used $\lambda \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and 1, 2, 5, 10 components for each class; and **Transductive Support Vector Machines (TSVM)**, in which we considered the proposal presented in [11] and we used $C = 1$ to induce a maximal margin hyperplane and C' parameter varying by a factor of ten from 10^{-5} to 10^{1}. We also run TSVM with and without the function proposed in [11] to maintain the same class proportion of labeled documents in the classification of unlabeled documents.

The transductive learning algorithms based on graphs used in this experimental evaluation were divided into algorithms based on document graph (document as vertex) and algorithms based on bipartite graph (word and document as vertices). These are iterative algorithms and we set the maximum of 1000 iterations for each algorithm. We generated document graphs considering the Mutual k-Nearest Neighbor strategy with $k \in \{7, 17, 37, 57\}$, and *Exp* graphs with $\sigma \in \{0.05, 0.2, 0.35, 0.5\}$ [34]. The algorithms based on document graphs are: **Label Propagation with Gaussian Fields and Harmonic Functions (LP)**, non-parametric algorithm; and **Learning with Local and Global Consistency (LLGC)**, in which we used $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

The algorithms based on bipartite graphs are: **Label Propagation based on Bipartite Heterogeneous Network (LPBHN)**, non-parametric algorithm; **Tab-based Model (TB)**, in which we used $\alpha = 0$, since there are no objects from different domains, $\beta \in \{0.1, 1, 10, 100, 1000\}$, and $\lambda \in \{0.1, 1, 10, 100, 1000\}$; and **GNetMine**, in which we used $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

We also ran the **Multinomial Naive Bayes (MNB)** and **Support Vector Machines (SVM)**, which are inductive supervised learning algorithms and allow us to verify the benefits of using unlabeled documents in the classification. There are no parameters for MNB. For SVM, we considered three types of kernel: *linear*, *polynomial* (exponent $= 2$) and *rbf* (radial basis function). The C values considered for each type of kernel were $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}, 10^{4}, 10^{5}\}$ [20].

For the proposed algorithm, TPBG, we used the concentration parameter $\alpha \in \{0.5, 0.05, 0.005\}$. A more concentrate vector indicate more information to small number of classes, it can be useful to differentiate the documents. As stopping criterion we used the maximum number of 100 iterations for local and global propagations[3].

In this paper we presented the classification performances considering *Micro-Averaging* F^1 (*Micro-*F^1) and *Macro-Averaging* F^1 (*Macro-*F^1) measures [4]. Both are a harmonic mean of Precision and Recall, in which both measures have the same weight. The difference is that Micro-Averaging strategy performs a sum of the terms of the evaluation measures for each class, while Macro-Averaging strategy performs and average over the evaluations measures for all classes. Micro-averaging scores are dominated by the number of correctly classified documents (true positives) and thus large classes dominate small classes in micro-averaging scores. On the other hand, macro-averaging gives equal weight to each class. In this case, the number of correctly classified documents in small classes is emphasized in macro-averaging scores. These two strategies give different scores and are complementary to each other [23].

The classification performance measures were obtained considering the average from 10 runs [33]. In each run we randomly selected N documents from each class as labeled documents. We carried out experiments using $N \in \{1, 10, 20, 30, 40, 50\}$. We started with the minimum number of labeled document per class and varied by a factor of ten from 10 to 50. This variation in the number of labeled documents allowed better understanding of the behavior of the algorithms for different number of labeled documents and a trade-off between the number of labeled documents and classification performance. The remaining $|\mathcal{D}| - (N * |\mathbb{C}|)$ documents were used to evaluate the classification.

4.3. Results

In this section we present the best classification performance values for *Micro-*F^1 and *Macro-*F^1 obtained in the experimental evaluation. Fig. 1 presents the *Micro-*F^1 values and Fig. 2 presents the *Macro-*F^1 values obtained by different algorithms and the different number of labeled documents per class. The *Micro-*F^1 and *Macro-*F^1 values tended to increase as the number of labeled documents grows. There were some decreases in *Macro-*F^1 values as we increased the number of labeled documents for small text collections in which all documents of a class were selected as labeled documents (the values of precision and recall for a class are 0 since there were no documents available for the test).

Our algorithm obtained the highest accuracies for the datasets *Classic4*, *Dmoz-Health*, *Dmoz-Science*, *Dmoz-Sports*, *NFS*, *Oh5*, and

[2] More details about the collections are presented in [22].

[3] A complete analysis about the impact and the behavior of the parameter α in the classification performance are presented at http://sites.labic.icmc.usp.br/thiagopf/prl_2016/parameter_analysis/.

[4] The classification performances considering other classification performance measures, such as *Accuracy*, *Error*, *Micro-Averaging Precision and Recall*, and *Macro-Averaging Precision and Recall*, are available at http://sites.labic.icmc.usp.br/thiagopf/prl_2016/

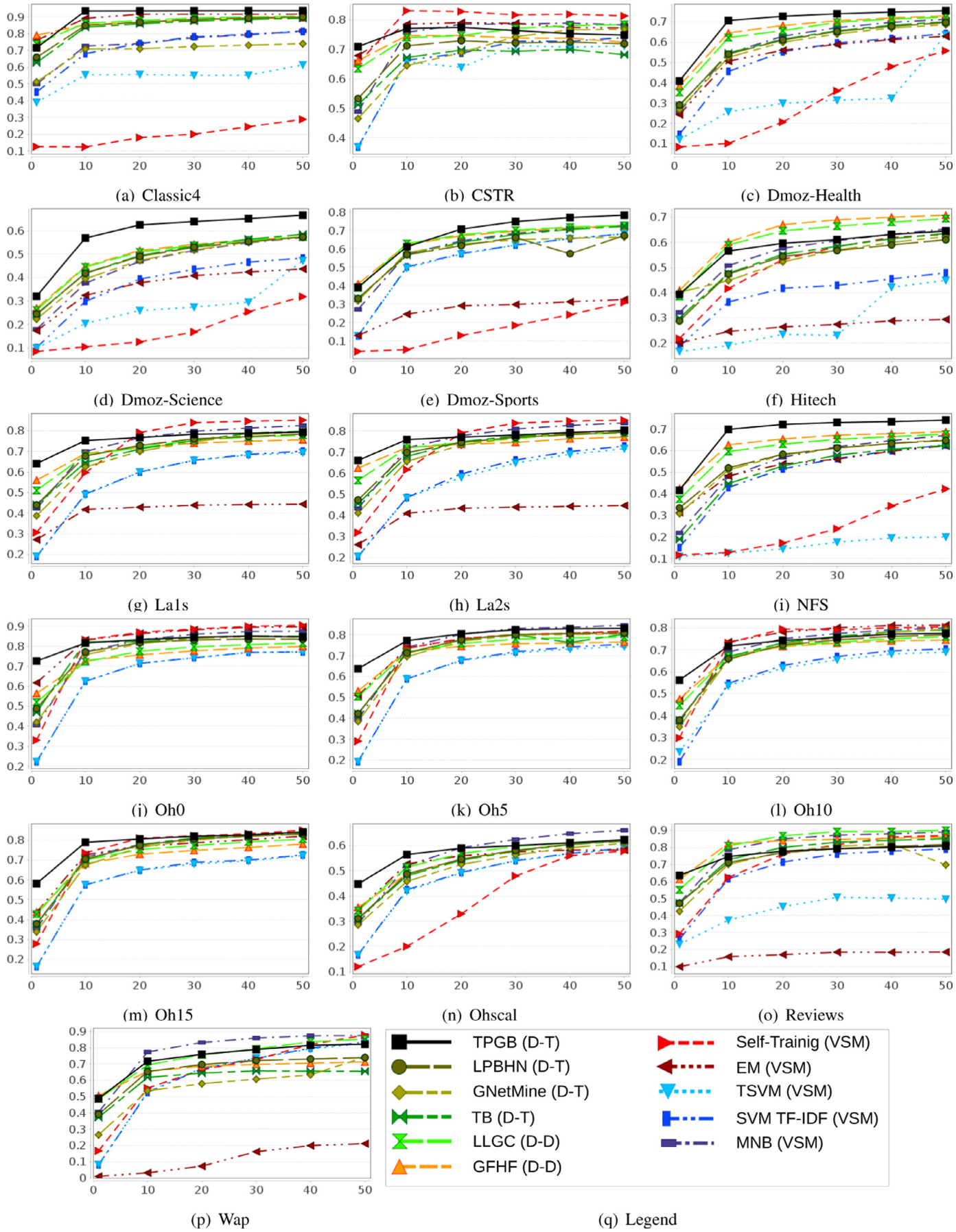

Fig. 1. *Micro-F*[1]: *x*-axis presents the number of labeled documents per class and *y*-axis presents the classification performance.

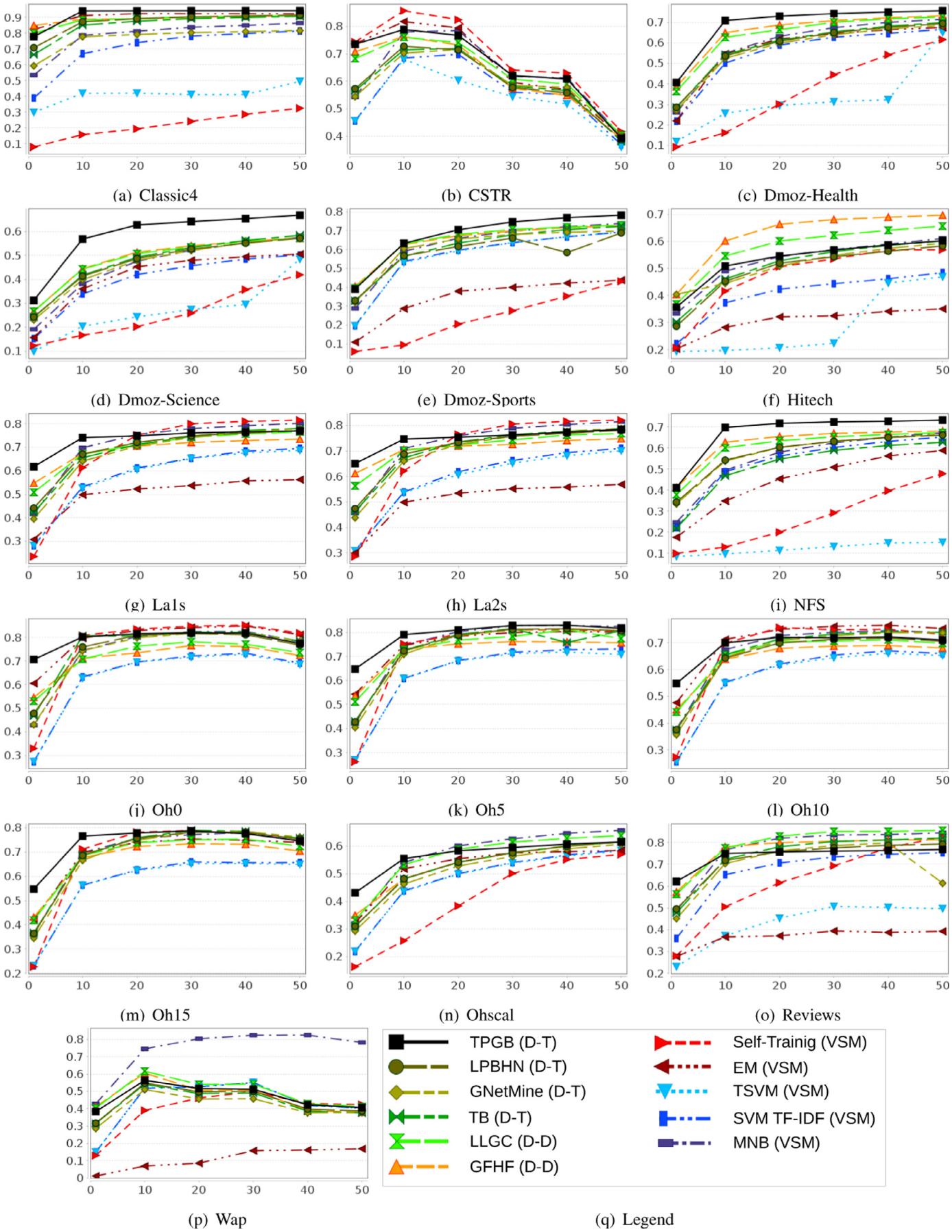

Fig. 2. *Macro-F*[1]: *x*-axis presents the number of labeled documents per class and *y*-axis presents the classification performance.

Table 2
Average ranking (AR), General ranking (GR) and p-value considering classification *Micro-F*[1] values.

	1 labeled doc			10 labeled docs			20 labeled docs			30 labeled docs			40 labeled docs			50 labeled docs		
Alg.	AR	GR	p-value	AR	GR	p-value	AR	GR	p-value	AR	GR	p-value	AR	GR	p-value	AR	GR	p-value
TPBG	1.50	1st	–	2.81	2nd	0.922938	2.25	1st	–	3.83	4th	0.721856	3.21	2nd	0.936277	3.31	2nd	0.83117
LPBHN	5.06	4th	0.002381	5.54	5th	0.243212	5.46	5th	0.006052	6.16	6th	0.204941	6.40	8th	0.005138	6.37	8th	0.004729
GNetMine	6.81	8th	0.000006	7.13	7th	0.073834	7.31	9th	0.000016	7.14	8th	0.098649	6.31	7th	0.006562	6.12	7th	0.009009
TB	5.62	5th	0.000435	5.96	6th	0.183680	5.75	6th	0.002838	6.03	5th	0.223786	5.56	5th	0.037644	5.62	5th	0.028866
LLGC	3.50	3rd	0.088082	2.82	3rd	0.921094	4.56	3rd	0.048597	2.92	1st	–	4.34	3rd	0.29864	4.65	3rd	0.174098
GFHF	1.81	2nd	0.789853	2.56	1st	–	5.09	4th	0.015302	3.81	3rd	0.72824	5.71	6th	0.02697	5.96	6th	0.013195
ST	9.06	9th	0	8.18	9th	0.027854	6.06	7th	0.001149	6.76	7th	0.133076	5.31	4th	0.06211	5.37	4th	0.048597
EM	6.00	6th	0.000124	7.91	8th	0.036321	6.87	8th	0.00008	8.56	10th	0.027309	7.50	9th	0.000191	7.68	9th	0.00008
MNB	6.56	7th	0.000016	4.13	4th	0.539939	3.56	2nd	0.263011	3.51	2nd	0.816413	3.12	1st	–	3.06	1st	–
TSVM	10.21	11th	0	10.09	11th	0.003202	10.03	11th	0	9.32	11th	0.012271	9.90	11th	0	9.62	11th	0
SVM TF-IDF	9.84	10th	0	8.81	10th	0.014499	9.03	10th	0	8.59	10th	0.000003	8.18	10th	0.000012	7.93	9th	0.049881
	SSD $p \leq 0.01106$			SSD $p \leq 0.004056$			SSD $p \leq 0.038789$			SSD $p \leq 0.009662$			SSD $p \leq 0.003354$			SSD $p \leq 0.008886$		

Table 3
Average ranking (AR), General ranking (GR) and p-value considering classification *Macro-F*[1] values.

	1 labeled doc			10 labeled docs			20 labeled docs			30 labeled docs			40 labeled docs			50 labeled docs		
Alg.	AR	GR	p-value	AR	GR	p-value	AR	GR	p-value	AR	GR	p-value	AR	GR	p-value	AR	GR	p-value
TPBG	1.87	1st	–	1.81	1st	–	2.68	1st	–	3.00	1th	–	3.50	2nd	0.68934	3.68	2nd	0.63144
LPBHN	5.06	4th	0.006562	5.78	5th	0.000713	5.68	7th	0.010515	5.65	5th	0.023497	6.15	8th	0.007699	6.03	7th	0.013195
GNetMine	6.65	8th	0.000046	7.31	8th	0.000003	6.81	9th	0.000435	6.31	7th	0.004729	5.53	5th	0.033006	5.18	5th	0.078593
TB	5.81	5th	0.000785	5.81	6th	0.000647	5.43	6th	0.019016	4.87	4th	0.109819	4.93	4th	0.104023	4.87	4th	0.135593
LLGC	3.31	3rd	0.220235	4.18	4th	0.042826	4.31	3rd	0.165807	4.56	3rd	0.182694	4.43	3rd	0.230429	4.62	3rd	0.200825
GFHF	2.09	2nd	0.852013	3.96	2nd	0.065936	5.43	4th	0.019016	5.65	6th	0.023497	5.90	7th	0.014214	6.09	8th	0.011349
ST	9.68	10th	0	7.31	9th	0.000003	6.25	5th	0.002381	6.43	8th	0.003373	5.87	6th	0.015302	6.00	6th	0.014214
EM	6.31	7th	0.000154	6.65	7th	0.000036	7.00	8th	0.000235	7.43	9th	0.000154	7.75	9th	0.000057	7.68	9th	0.0001
MNB	6.18	6th	0.000235	3.96	3rd	0.065936	3.62	2nd	0.423999	3.43	2nd	0.709073	3.03	1st	–	3.12	1st	–
TSVM	10.06	11th	0	10.18	11th	0	10.00	11th	0	9.87	11th	0	10.125	11th	0	9.93	11th	0
SVM TF-IDF	8.93	9th	0	9.00	10th	0	8.75	10th	0	8.75	10th	0.000001	8.75	10th	0.000002	8.75	10th	0.000001
	SSD $p \leq 0.007789$			SSD $p \leq 0.049161$			SSD $p \leq 0.030316$			SSD $p \leq 0.015312$			SSD $p \leq 0.016351$			SSD $p \leq 0.019398$		

Table 4
Average ranking (AR), General ranking (GR) and p-value considering classification *Micro-F*[1] values.

	Micro-F[1]			Macro-F[1]		
Alg.	AR	GR	p-value	AR	GR	p-value
TPBG	2.50	1st	–	2.76	1st	–
LPBHN	5.79	6th	*0*	5.72	6th	*0*
GNetMine	6.81	8th	*0*	6.30	7th	*0*
TB	5.68	5th	*0*	5.29	5th	*0*
LLGC	4.25	3rd	0.000268	4.23	3rd	0.002002
GFHF	4.66	4th	0.000007	4.85	4th	0.000012
ST	6.43	7th	*0*	6.92	8th	*0*
EM	6.96	9th	*0*	7.14	9th	*0*
MNB	3.96	2nd	0.002234	3.89	2nd	0.017701
TSVM	9.96	11th	*0*	10.03	11th	*0*
SVM TF-IDF	8.94	10th	*0*	8.82	10th	*0*
	SSD $p \leq 0.05$			SSD $p \leq 0.05$		

Oh15 for all numbers of labeled documents and for both *Micro-F*[1] and *Macro-F*[1] measures. TPBG also obtained the highest or close to the highest accuracy *Micro-F*[1] and *Macro-F*[1] when the number of labeled documents are less than or equal to 20. This indicates that the proposed approach makes the better use of few labeled documents to improve classification performance.

The algorithms MNB and LLGC reach in some datasets better results than our proposed algorithm when considering more labeled documents. In general, our algorithm was better than all algorithms based on bipartite graphs used in this experimental evaluation. Similarly, graphs-based algorithms obtained highest classification performance than vector space model based algorithms. Label propagation algorithms based on document graphs, such as LLGC and GFHF, were better than algorithms based on bipartite graphs.

We submitted the data presented in Fig. 1, and Fig. 2 to Friedman test and Li's post-hoc test with 95% of confidence level to assess statistically significant differences (SSD) among the classification algorithms[5]. This is an advisable statistically significant difference test to use when there is a control algorithm (usually the proposed one) and results from multiple datasets [9,26]. The null hypothesis states that all the algorithms performed equivalently and therefore their ranks should be equal. In our case we want to determine if the proposed algorithm obtained better results with statistically significant differences in comparison to others algorithms.

In Table 2 we present the results of the statistical test considering *Micro-F*[1] and in Table 3 we present the results of the statistical test considering *Macro-F*[1], both considering each number of labeled documents individually. In Table 4 we present the results of the statistical test considering the results obtained by all number of labeled documents together. In these tables we present the average rank (AR), the general rank (GR), *i.e.*, the ranking of the algorithms considering the average rank, the p-value, and the value of p which produces statistically significant differences (SSD). The results with SSD are highlighted in italic.

The control algorithm is the one with the best average ranking by default. When considering *Micro-F*[1] values, TPBG obtained the 1st position in the average ranking for 1 an 20 labeled documents per class and the 2nd position for 10, 40 e 50 labeled documents per class. We highlight that TPBG obtained better results than all algorithms with SSD when considering 1 labeled document per class, excepting LLGC and GFHF.

[5] Friedman test is a non-parametric test based on average ranking differences. The algorithm with highest performance have the rank of 1, the second best performance 2, and so on. The Li's post test is used to find pairs of algorithms which produce statistically significant differences.

When considering *Macro-F*[1] values, TPBG obtained the 1st position in the average ranking when considering 1, 10, 20, and 30 labeled documents per class and the 2nd position for 40 and 50 labeled documents per class. TPBG presented SSD for all transductive learning algorithms based on vector space model and bipartite graphs when considering 1, 10 and 20 labeled documents per class. Moreover, according to Table 4, TPBG presented better results than all algorihms with SSD when performing the statistical significant test considering the classification performance obtained by all number of labeled documents together.

5. Conclusion and future work

In this paper we presented an algorithm which uses the structure of a bipartite graph to perform transductive classification of texts. The proposed algorithm, named TPBG (Transductive Propagation in Bipartite Graph), propagates the class information vectors associated to vertices and edges in a bipartite graph considering labeled and unlabeled documents. The proposed algorithm also obtains a better classification performance than algorithms based on vector space model and graph representation when only a small set of labeled documents is available.

As future work we intend to: (i) incorporate other types of objects relations as document-document or term-term with document-term relations and analyse the impact in the classification performance and (ii) to develop an online version of TPBG algorithm to address large textual collections.

Acknowledgments

This research was partially supported by grants 2011/23689-9, 2011/12823-6, 2011/22749-8, and 2015/14228-9 from São Paulo Research Foundation (FAPESP) and 302645/2015-2 from National Council of Scientific and Technologic Development (CNPq).

References

[1] C.C. Aggarwal, P. Zhao, Towards graphical models for text processing, Knowl. Inf. Syst. 36 (1) (2013) 1–21.
[2] R. Angelova, G. Weikum, Graph-based text classification: learn from your neighbors, in: Proceedings of the Special Interest Group on Information Retrieval Conference, ACM, 2006, pp. 485–492.
[3] M. Belkin, P. Niyogi, V. Sindhwani, Manifold regularization: a geometric framework for learning from labeled and unlabeled examples, J. Mach. Learn. Res. 7 (2006) 2399–2434.
[4] L. Berton, A.d. A. Lopes, Graph construction based on labeled instances for semi-supervised learning, in: Proceedings of the 22nd International Conference on Pattern Recognition (ICPR), 2014, 2014, pp. 2477–2482, doi:10.1109/ICPR.2014.428.
[5] A. Blum, T. Mitchell, Combining labeled and unlabeled data with co-training, in: Proceedings of the Conference on Computational Learning Theory, ACM, 1998, pp. 92–100.
[6] F.A. Breve, L. Zhao, M.G. Quiles, W. Pedrycz, J. Liu, Particle competition and cooperation in networks for semi-supervised learning, IEEE Trans. Knowl. Data Eng. 24 (9) (2012) 1686–1698.
[7] O. Chapelle, B. Schölkopf, A. Zien (Eds.), Semi-Supervised Learning, MIT Press, 2006.
[8] I.S. Dhillon, Co-clustering documents and words using bipartite spectral graph partitioning, in: Proceedings of the International Conference on Knowledge Discovery and Data Mining, ACM, 2001, pp. 269–274.
[9] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power, Inf. Sci. 180 (10) (2010) 2044–2064.
[10] M. Ji, Y. Sun, M. Danilevsky, J. Han, J. Gao, Graph regularized transductive classification on heterogeneous information networks, in: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2010, pp. 570–586.
[11] T. Joachims, Transductive inference for text classification using support vector machines, in: Proceedings of International Conference on Machine Learning, 1999, pp. 200–209.
[12] X. Kong, M.K. Ng, Z.-H. Zhou, Transductive multilabel learning via label set propagation., IEEE Trans. Knowl. Data Eng. 25 (3) (2013) 704–719.
[13] R. Mihalcea, P. Tarau, TextRank: bringing order into texts, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2004, pp. 404–411.

[14] K. Nigam, A.K. McCallum, S. Thrun, T. Mitchell, Text classification from labeled and unlabeled documents using EM, Mach. Learn. 39 (2/3) (2000) 103–134.
[15] H. Oh, S. Myaeng, M. Lee, A practical hypertext categorization method using links and incrementally available class information, in: Proceedings of the Special Interest Group on Information Retrieval Conference, ACM, 2000, pp. 264–271.
[16] G.K. Palshikar, Keyword extraction from a single document using centrality measures, in: Proceedings of the International Conference on Pattern Recognition and Machine Intelligence, Springer, 2007, pp. 503–510.
[17] L. Pardo, Statistical Inference Based on Divergence Measures, Statistics: A Series of Textbooks and Monographs, CRC Press, 2005.
[18] M.F. Porter, An algorithm for suffix stripping, Read. Inf. Retr. 14 (3) (1980) 130–137.
[19] R.G. Rossi, A. de Andrade Lopes, S.O. Rezende, Optimization and label propagation in bipartite heterogeneous networks to improve transductive classification of texts, Inf. Process. Manag. 52 (2015) 217–257.
[20] R.G. Rossi, T.P. Faleiros, A.A. Lopes, S.O. Rezende, Inductive model generation for text categorization using a bipartite heterogeneous network, in: Proceedings of the International Conference on Data Mining, IEEE, 2012, pp. 1086–1091.
[21] R.G. Rossi, A.A. Lopes, S.O. Rezende, A parameter-free label propagation algorithm using bipartite heterogeneous networks for text classification, in: Proceedings of the Symposium on Applied Computing, ACM, 2014, pp. 79–84.
[22] R.G. Rossi, R.M. Marcacini, S.O. Rezende, Benchmarking Text Collections for Classification and Clustering Tasks, Technical Report 395, Institute of Mathematics and Computer Sciences - University of Sao Paulo, 2013.
[23] M. Sokolova, G. Lapalme, A systematic analysis of performance measures for classification tasks, Inf. Process. Manag. 45 (4) (2009) 427–437.
[24] M. Steyvers, J.B. Tenenbaum, The large-scale structure of semantic networks: Statistical analyses and a model of semantic growth, Cogn. Sci. 29 (2005) 41–78.
[25] Y. Sun, J. Han, J. Gao, Y. Yu, iTopicModel: information network-integrated topic modeling, in: Proceedings of the International Conference on Data Mining, IEEE Computer Society, 2009, pp. 493–502.
[26] B. Trawinski, M. Smetek, Z. Telec, T. Lasota, Nonparametric statistical analysis for multiple comparison of machine learning regression algorithms., Appl. Math. Comput. Sci. 22 (4) (2012) 867–881.
[27] Y.-H. Tseng, Z.-P. Ho, K.-S. Yang, C.-C. Chen, Mining term networks from text collections for crime investigation., Expert Syst. Appl. 39 (11) (2012) 10082–10090.
[28] X. Wan, J. Yang, J. Xiao, Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction., in: Proceedings of the Annual Meeting of the Association for Computational Linguistics, ACM, 2007, pp. 552–559.
[29] W. Wang, D.B. Do, X. Lin, Term graph model for text classification., in: Proceedings of the International Conference on Advanced Data Mining and Applications, Springer, 2005, pp. 19–30.
[30] D. Yarowsky, Unsupervised word sense disambiguation rivaling supervised methods, in: Proceedings of the Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, 1995, pp. 189–196.
[31] Z. Yin, R. Li, Q. Mei, J. Han, Exploring social tagging graph for web object classification, in: Proceedings of the International Conference on Knowledge Discovery and Data Mining, 2009, pp. 957–966.
[32] D. Zhou, O. Bousquet, T.N. Lal, J. Weston, B. Schölkopf, Learning with local and global consistency, in: Proceedings of the Advances in Neural Information Processing Systems, 16, 2004, pp. 321–328.
[33] X. Zhu, Z. Ghahramani, J. Lafferty, Semi-supervised learning using gaussian fields and harmonic functions, in: Proceedings of the International Conference on Machine Learning, AAAI Press, 2003, pp. 912–919.
[34] X. Zhu, A.B. Goldberg, Introduction to Semi-Supervised Learning, Morgan and Claypool Publishers, 2009.