Inductive Model Generation for Text Categorization Using a Bipartite Heterogeneous Network

Conference Paper · December 2012

DOI: 10.1109/ICDM.2012.130

4 authors:

Rafael Geraldeli Rossi
Federal University of Mato Grosso do Sul, Três …
32 PUBLICATIONS **92** CITATIONS
SEE PROFILE

Thiago Faleiros
University of Brasília
9 PUBLICATIONS **27** CITATIONS
SEE PROFILE

Alneu de Andrade Lopes
University of São Paulo
75 PUBLICATIONS **395** CITATIONS
SEE PROFILE

Solange Oliveira Rezende
University of São Paulo
177 PUBLICATIONS **498** CITATIONS
SEE PROFILE

2012 IEEE 12th International Conference on Data Mining

Inductive Model Generation for Text Categorization using a Bipartite Heterogeneous Network

Rafael Geraldeli Rossi
University of São Paulo
São Carlos, Brazil
ragero@icmc.usp.br

Thiago de Paulo Faleiros
University of São Paulo
São Carlos, Brazil
thiagopf@icmc.usp.br

Alneu de Andrade Lopes
University of São Paulo
São Carlos, Brazil
alneu@icmc.usp.br

Solange Oliveira Rezende
University of São Paulo
São Carlos, Brazil
solange@icmc.usp.br

***Abstract*—Usually, algorithms for categorization of numeric data have been applied for text categorization after a preprocessing phase which assigns weights for textual terms deemed as attributes. However, due to characteristics of textual data, some algorithms for data categorization are not efficient for text categorization. Characteristics of textual data such as sparsity and high dimensionality sometimes impair the quality of general-purpose classifiers. Here, we propose a text classifier based on a bipartite heterogeneous network used to represent textual document collections. Such algorithm induces a classification model assigning weights to objects that represents terms of the textual document collection. The induced weights correspond to the influence of the terms in the classification of documents they appear. The least-mean-square algorithm is used in the inductive process. Empirical evaluation using a large amount of textual document collections shows that the proposed IMBHN algorithm produces significantly better results than the k-NN, C4.5, SVM and Naïve Bayes algorithms.**

***Keywords*-Heterogeneous Network; Text Categorization.**

I. Introduction

A huge amount of the data in the digital world is presented in textual format. Currently, it is already impossible to explore manually all this amount of data. To perform such tasks, techniques for automatic categorization of textual documents have been applied and gained importance in the last decades [1], [2].

Text categorization consists of assigning predefined categories to documents in a document collection. Most of state-of-the art inductive strategies for text categorization considers that texts are represented by a document-term matrix. A graph or network is an alternative for representing relations among documents and terms present in the texts. For instance, a collection of textual documents can be easily represented by a bipartite network consisting of heterogeneous objects representing the documents and terms presented in the collection. In this network, an object of type document is linked to objects of type term when these terms are present in the document. The representation of the textual document collection considering a bipartite heterogeneous network has the advantage of not requiring hyperlinks, citations, or relationships between terms to create the network. The network also avoids the high sparsity of a document-term matrix. Moreover, according to [3], this type of representation has been underexplored for textual data representation and there is much room for further investigation.

Here we propose a textual document categorization algorithm inspired in the structure of a bipartite heterogeneous network to induce a categorization model. The induction of the model consists of assigning weights to terms related to each class in the document collection taking into account the labeled documents in the training data and the bipartite network formed by terms and documents. In the classification phase, the induced weights and the bipartite structure are considered to assign categories to new documents.

We carried out a comprehensive comparison of the proposed classification algorithm to the following inductive algorithms: i) Naïve Bayes (NB), and Multinomial Naïve Bayes (MNB), probabilistic paradigm, ii) C4.5, symbolic paradigm; iii) Support Vector Machine (SVM) statistical paradigm, and iv) k-Nearest Neighbors (k-NN), instance-based paradigm. The results of the statistical significance test showed that the proposed algorithm is superior with significant differences when compared to the NB, C4.5, SVM and k-NN.

The remainder of the paper is organized as follows. Section II presents the basic concepts employed in this paper, and related work. Section III presents details of the proposed classification algorithm. Section IV presents the carried out experiments and the results. Finally, Section V presents the conclusions and future works.

II. Background and Related Work

Two automatic strategies are employed for text categorization: inductive and transductive. The inductive learning strategy induces a model to assign labels (categories) for new documents and has been widely used for the text categorization task [4]. Differently, transductive learning does not create a model to classify new documents as the inductive model does. Instead, it considers a data set of both labeled and unlabeled examples to perform the categorization task, spreading the information from labeled to unlabeled data through the data set. This type of classification is commonly

1550-4786/12 $26.00 © 2012 IEEE
DOI 10.1109/ICDM.2012.130

used when few labeled examples exist and data are represented by networks.

The algorithm proposed in this paper was firstly designed to employ an heterogeneous network to represent a textual document collection, i.e., a network with different types of objects. Formally, an heterogeneous network can be defined as follows [5]. Given m types of objects ($X_1 = \{x_{11}, \ldots, x_{1_{n_1}}\}, \ldots, X_m = \{x_{m1}, \ldots, x_{m_{n_m}}\}$), the network $G = \langle V, E, W \rangle$ is called an heterogeneous network if $V = \bigcup_{i=1}^{m} X_i$ and $m \geq 2$. In this network, E is the set of links between two objects of V and W is the set of link weights. In a bipartite network there is no link between objects of the same type.

One way to classify objects in a heterogeneous network is by creating a weight vector for each network object, in which each position of the vector corresponds to a category of the data set [5], [6]. The vector position corresponding to the object class receives the value 1 and any other position receives the value 0. The weight vector for objects in which there is no information is calculated during the learning process by propagating information from labeled to unlabeled vertices. The weights of the edges could also be considered to improve the learning. Figure 1 shows an heterogeneous network representing a collection of textual documents with L classes modeled for the categorization task.

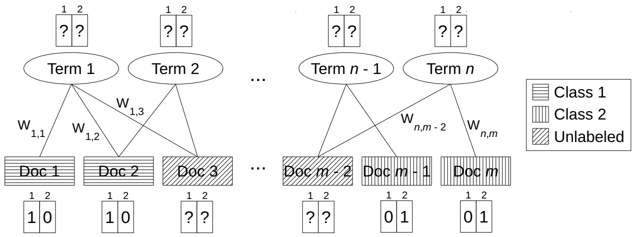

Figure 1. Example of textual document collection representation using a bipartite heterogeneous network.

Some studies have employed heterogeneous bipartite network representation in transductive classification. In [6], the IRC (Iterative Reinforcement Categorization) algorithm uses a bipartite network composed by queries and web pages. The labels of the labeled pages and the estimated labels for the unlabeled pages (inferred by a machine learning algorithm) are propagated to the objects representing the queries. In [7], the TM (Tag-based Classification Model) algorithm considers a bipartite network composed by Web objects and social tags. The labels of the Web objects are propagated using social tags, in order to minimize the differences between the original labels and the propagated labels, keeping the consistence with the neighboring nodes. In [5], GNetMine is proposed as a general framework for object categorization in heterogeneous networks. The information class propagation is similar to the one proposed in [8], however, GNetMine considers the semantics of each type of relationship.

The induction of classification models has been widely and successfully used in text mining. However, for certain scenarios these algorithms are not efficient due to their specificities, or due to the use of a document collection representation with high dimensionality and sparsity. The representation of the textual collection by heterogeneous network is an alternative. However, to the extent of our knowledge, all the algorithms for data classification based on heterogeneous networks are transductive. In the next section we present the proposed inductive algorithm.

III. Inductive Model Based on Bipartite Heterogeneous Network - IMBHN

The algorithm Inductive Model Based on Bipartite Heterogeneous Network (IMBHN), proposed in this paper, aims to induce a classifier by modeling the textual data as a bipartite heterogeneous network.

As illustrated in Figure 1, each object in an heterogeneous network has a corresponding weight vector. This vector size is set as the number of classes in the text collection and each position of the vector corresponds to only one class. Let us represent the weights of each object of the type term by a matrix $W = \{\mathbf{w}_1^T, \ldots, \mathbf{w}_\alpha^T\}^T$, in which α is the number of terms and w_{ij} represents the weight of the object i for the class j. The matrix W has dimension $\alpha \times \omega$ in which ω is the number of classes. The classes are represented by the vector $\mathbf{c} = \{c_1, \ldots, c_\omega\}$ and the terms of the collection are represented by the vector $\mathbf{f} = \{f_1, \ldots, f_\alpha\}$. Each object of the document type has also a weight vector for the classes. Let us represent the weight vectors of the labeled documents by a matrix $Y = \{\mathbf{y_1^T}, \ldots, \mathbf{y}_\theta^T\}^T$, in which θ is the number of training documents in the collection $D = \{\mathbf{d}_1^T, \ldots, \mathbf{d}_\theta^T\}^T$. Each position d_{ki} from matrix D corresponds to the frequency of the term f_i in document $\mathbf{d_k}$. The weight of a document k for a class j is given by Equation 1.

$$y_{kj} = \begin{cases} 1 & \text{if } \mathbf{d_k} \in \text{ class } c_j; \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

A. The Algorithm

The objective of the proposed approach is to compute the influences of the terms present in the collection in the definition of the classes of the documents, i.e., to induce the weights of the influences of each term in the collection with respect to each class. The induction process is guided by the minimization of the cost function presented in Equation 2:

$$\begin{aligned} Q(W) &= \frac{1}{2}\left(\sum_{j=1}^{\omega}\sum_{k=1}^{\theta}(class(\sum_{i=1}^{\alpha} d_{ki}w_{ij}) - y_{kj})^2\right) \\ &= \frac{1}{2}\left(\sum_{j=1}^{\omega}\sum_{k=1}^{\theta} error_{kj}^2\right) \end{aligned} \tag{2}$$

in which

$$class(\sum_{i=1}^{\alpha} d_{ki}w_{ij}) = \begin{cases} 1 & \text{if } c_j = \arg\max_{c_{j*} \in \mathbf{c}} (\sum_{i=1}^{\alpha} d_{ki}w_{ij*}) \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Therefore, the proposed algorithm induces a weight matrix which minimizes the quadratic error ($error^2_{kj}$), i.e., basically the sum of square of the differences between the predicted and real classes of the training documents. The way the problem was modeled allows the use of the gradient descent method to induce the weights of the elements from matrix W. We chose the Least-Mean-Square [9] algorithm to induce the weights due to its simplicity, but any other algorithm for adjusting the weights to minimize the error can be used. The Least-Mean-Square makes successive corrections in the weight vector in the direction of the negative gradient vector leading to the minimum mean squared error.

The weight vector equation using the steepest descent method is presented in Equation 4.

$$\mathbf{w}^{n+1} = \mathbf{w}^{n} + \eta[- \bigtriangledown (Q(W))] \tag{4}$$

Thus, each step goes in the direction that minimizes the cost function $Q(W)$. The direction of the gradient can be estimated by the derivative of $Q(W)$, Equation 5.

$$\frac{\partial Q(W)}{\partial W} = \sum_{j=1}^{\omega}\sum_{k=1}^{\Theta} class(\sum_{i=1}^{\alpha} d_{ki}w_{ij} - y_{kj}) * \sum_{j=1}^{\omega}\sum_{k=1}^{\Theta}\sum_{i=1}^{\alpha} d_{ki} \tag{5}$$

$$\bigtriangledown(Q(W)) = \frac{\partial Q(W)}{\partial W} = \sum_{j=1}^{\omega}\sum_{i=1}^{\Theta} error_{kj} * \sum_{j=1}^{\omega}\sum_{k=1}^{\Theta}\sum_{i=1}^{\alpha} d_{ki}$$

Considering Equations 4 and 5, the weight $w_{ij}^{(n+1)}$ of a term f_i for the class c_j in time $(n+1)$ is given by Equation 6:

$$w_{ij}^{(n+1)} = w_{ij}^{(n)} + (\eta * \sum_{k=1}^{\Theta} d_{ki} * error_{kj}^{(n)}) \tag{6}$$

where η is the correction rate, i.e., the rate in which the error will be considered in the weight updating. We notice that the weight updating is a function of the current weight and the obtained error.

Considering the adopted weight adjustment, the proposed algorithm has 3 main steps: (1) weight vectors initialization, (2) network error calculation, and (3) weights adjustment.

In the weight vector initialization step, the algorithm defines the vectors of weight for each term. The weight values can be 0, randomly chosen, or considered as the likelihood of each term to belong to each class. In this work, the weights of the terms are initially set according to Equation 7. The value given by this equation is closer to 1 for the term that occurs almost exclusively in documents of a specific class.

$$w_{ij}^{(0)} = \sum_{k=1}^{\omega} d_{ki}y_{kj} / \sum_{k=1}^{\omega} d_{ki} \tag{7}$$

In step 2, an output vector $\mathbf{out}_k$ for each document $\mathbf{d}_k$ is computed. Each position of this vector is obtained by the sum of the weights of the document-term connections multiplied by the weight of each term for each class. The weight of the document $\mathbf{d}_k$ for class c_j is given by Equation 3. Then, for each document $\mathbf{d}_k$, the error for each class j is calculated subtracting each position of the output vector out_{kj} and y_{kj}.

In the weight adjustment step, the error of each document influences the weights of the terms linked to the document by adjusting their weights. To update the weight of a term f_i for a class c_j, Equation 6 is applied.

Steps 2 and 3 are repeated for every document until a stopping criterion is reached. We adopted as stopping criterion the `maximum number of epochs`[1] and a `minimum mean squared error`, i.e., when the mean squared error of an epoch is less than a small given value ϵ.

Algorithm 1 summarizes the algorithm IMBHN. Line 2 refers to the weight vectors initialization. The output vector of the document d for each class of the collection is obtained in lines 6–12. The error in the document d for each class is calculated in lines 16 and 17. Finally, the updating of weights of every term connected to d for each class is performed in lines 19–21.

In the classification phase, the induced term weights are employed for the categorization of new documents. This is achieved through the maximum argument of the sum of weights of the document connections with the terms multiplied by the weight of each term for each class (Equation 3).

B. Time Complexity Analysis

The complexity of the IMBHN algorithm is a function of i) the average number of terms ($\overline{|T|}$), since for each document of the collection only the terms connected to the document are updated; ii) the number of documents in the training set ($|D|$), since every document is used for updating the weights, and iii) the number of epochs necessary to achieve the stopping criterion (n). Thus, the complexity of the algorithm IMBHN is $O(n * |D| * \overline{|T|})$.

IV. Experimental Evaluation

This section presents the textual document collections used in the experiments, the algorithms used for comparison, the design of the experimental setup, the evaluation criteria, the results and discussion.

[1] An epoch is one pass for all training documents

Algorithm 1: IMBHN

```
input      :
    $D$ - adjacency matrix composed by objects representing documents, terms and
    the weights of the connections between them; $Y$ - matrix of weights for each
    document for each class of the collection; $W$ - matrix of weights for each
    term for each class of the collection; **f** - vector of terms; **c** - vector of classes;
    $\eta$ - correction rate; $\alpha$ - number of terms; $\omega$ - number of classes; $\Theta$ - number
    of documents.
output     :
    $W$ - term weights induced during the learning process.

$epochs\_number = 0$
weight_initialization($W$)
while the stop criterion is not reached do
    $squared\_error\_acm = 0$
    for $k = 1$ to $\Theta$ do
        /* The output for each training document is
           calculated in this loop                          */
        induced_weights[]
        for $j = 1$ to $\omega$ do
            $class\_weight = 0$
            for $i = 1$ to $\alpha$ do
                $class\_weight = class\_weight + W[i][j] * D[k][i]$
            end
            $induced\_weights[j] = class\_weight$
        end
        $out[] = class(induced\_weights)$   /* set the value 1
        to the highest value and 0 to the others */

        /* Calculating the error                            */
        for $j = 1$ to $\omega$ do
            $error = Y[k][j] - out[j]$
            $squared\_error\_acm = (error * error)/2$
            /* Weight correction for each term
               connected to the document                    */
            for $i = 1$ to $\alpha$ do
                $current\_weight = W[i][j]$
                $new\_weight =$
                $current\_weight + (\eta * D[k][i] * error)$
                $W[i][j] = new\_weight$
            end
        end
    end
    $mean\_squared\_error = squared\_error\_acm/|D|$
    $epochs\_number = epochs\_number + 1$;
    $Stopping\_Analysis(epochs\_number, mean\_squared\_error)$
end
```

35 textual document collections were employed to carried out the evaluation of the IMBHN algorithm. For the 19MClassTextWc collection (Fbis, La1s, La2s, New3s, Oh0, Oh10, Oh15, Oh5, Ohscal, Tr11, Tr12, Tr21, Tr23, Tr31, Tr41, Tr45, Re0, Re1, Wap) [10] no preprocessing was performed since these collections are already in a structured format (document-term matrix). For the other collections, stopwords were removed, terms were stemmed using Porter's algorithm, HTML tags and e-mail headers were removed, and only terms with document frequency ≥ 2 were considered.

The text collections were selected considering different types of texts: web pages (WP), news articles (NA), sentiment analysis (SA), scientific project (SP), medical documents (MD), product reviews (PR), film reviews (FR), abstracts (AB) and TREC Documents (TD). The number of documents range from 204 to 18808, the number of terms from 1726 to 45434, the number of classes from 2 to 51, and the average number of terms from 5.96 to 469.86. Table I presents the number of documents ($|\mathbf{D}|$), number of generated terms ($|\mathbf{T}|$), average number of terms per document ($|\overline{\mathbf{T}}|$), number of classes ($|\mathbf{C}|$) and standard deviation considering the percentage of classes ($\sigma(\mathbf{C})$). The representations of the document collections are in the `ARFF` format [11] and are available at http://sites.labic.icmc.usp.br/ragero/arffs/.

Table I
CHARACTERISTICS OF THE TEXTUAL DOCUMENT COLLECTIONS USED IN THE EXPERIMENTAL EVALUATION.

Collection	$\|\boldsymbol{D}\|$	$\|\boldsymbol{T}\|$	$\overline{\|\boldsymbol{T}\|}$	$\|\boldsymbol{C}\|$	$\sigma(\boldsymbol{C})$
Classic4 (AB)	7095	7749	35.28	4	1.94
CSTR (TR)	299	1726	54.27	4	18.89
Dmoz-Sports-500 (WP)	13500	5682	11.87	27	0.00
Dmoz-Business-500 (WP)	18500	8303	11.93	37	0.00
Dmoz-Health-500 (WP)	6500	4217	12.40	13	0.00
Dmoz-Science-500 (WP)	6000	4821	11.52	12	0.00
Fbis (NA)	2463	2001	159.24	17	5.66
Hitech (NA)	2301	12942	141.93	6	8.25
Irish-Sentiment (SA)	1660	8659	112.65	3	6.83
La1 (NA)	3204	13196	144.64	6	8.22
La2 (NA)	3075	12433	144.83	6	8.59
New3s (NA)	9558	26833	234.53	44	1.32
NFS (SP)	10524	3888	6.65	16	3.82
Oh0 (MD)	1003	3183	52.50	10	5.33
Oh10 (MD)	1050	3239	55.64	10	4.25
Oh15 (MD)	913	3101	59.30	10	4.27
Oh5 (MD)	918	3013	54.43	10	3.72
Ohscal (MD)	11162	11466	60.39	10	2.66
Opinosis (PR)	6457	2693	7.56	51	1.42
Polarity (FR)	2000	15698	205.06	2	0.00
Re0 (NA)	1504	2887	51.73	13	11.56
Re1 (NA)	1657	3759	52.70	25	5.54
Re8 (NA)	7674	8901	35.31	8	18.24
Reviews (NA)	4069	22927	183.10	5	12.80
SyskillWebbert (WP)	334	4340	93.16	4	10.75
Tr11 (TD)	414	6430	281.66	9	9.80
Tr12 (TD)	313	5805	273.60	8	7.98
Tr21 (TD)	336	7903	469.86	6	25.88
Tr23 (TD)	204	5833	385.29	6	15.58
Tr31 (TD)	927	10129	268.50	7	13.37
Tr41 (TD)	878	7455	195.33	10	9.13
Tr45 (TD)	690	8262	280.58	10	6.69
Wap (WP)	1560	8461	141.33	20	5.20
WebACE (WP)	3900	8881	43.15	21	8.44
WebKb (WP)	8282	22892	89.78	7	15.19

The results obtained by the proposed algorithm, IMBHN, were compared with 6 inductive classification algorithms using the Weka tool [11]. The algorithms for comparison were: Naive Bayes (NB), Multinomial Naïve Bayes (MNB), J48 (implementation of C4.5 algorithm), Sequential Minimal Optimization (SMO, which is an algorithm for solving optimization problems during the training of SVMs), and IBk (implementation of the k-NN algorithm).

For the SMO algorithm, we considered three types of kernel: linear, polynomial (exponent = 2) and *rbf* (radial basis function). The C values were considered $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}, 10^{4}, 10^{5}\}$ for each type of kernel. These parameters were based on [12].

The values of k for the IBk algorithm were $\{1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 25, 29, 35, 41, 49, 57, 73, 89\}$. Some studies evaluating the k-NN algorithm selected a set of values ranging from 1 to |*number_of_documents*|. We did not select this variation because we tested the k-NN without and with a weighting scheme, which gives for each of the nearest neighbors a weight vote equal to $1/(1-s)$, where s is a similarity measure between neighbors. Without the weighting scheme, as the number of neighbors becomes closer to |*number_of_documents*|, the documents are classified as the majority class. The cosine distance was used as similarity measure.

For the proposed algorithm, IMBHN, we used the error correction rates $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. The error correction impacts on the induction of the parameters. If the correction rate is very small, the algorithm converges very slowly. If the correction rate is large, a fast convergence is achieved, but it can be unstable around the minimum error. Thus we decided to study small and high values to verify the behavior of the convergence and the accuracy of the algorithm. As stopping criteria we use the minimum mean squared error $\in \{0.01, 0.005\}$. These values control how the weights adjust the model to the data train. Then we chose a commonly value (0.01) [13] and a value that leads to a higher adjustment to the training data (0.005).

The default parameters of the Weka tool was adopted for the algorithms NB, MNB and J48. The measure for evaluation was the classification accuracy obtained by the 10-fold cross validation process. All the algorithms were subjected to the same folds of the cross-validation procedure.

We carried out statistical significance tests using the Friedman test and the Nemenyi post-hoc test with 95% confidence level [14] to compare results. Two types of comparisons were carried out in the experiments: i) comparing all algorithms considering the parameter that obtained the highest accuracy for each collection and ii) comparing all algorithms considering only the parameter that leads the best result for all collections according to the statistical test.

Table II presents the accuracies obtained by the IMBHN algorithm and the algorithms used for comparison taking into account the parameters that obtained the highest accuracy for each collection. The highest accuracy for each collection is in bold. The penultimate line of Table II presents the average ranking of the Friedman statistical test and the last line presents the algorithm position in the ranking. The IMBHN algorithm obtained the highest accuracy in 10 of 35 collections and was the best in the average ranking.

Figure 2 presents the critical difference to illustrate the results of the statistical significance test. The algorithms connected by a line do not present statistically significant differences among them. According to Figure 2, the IMBHN is superior with significant differences to the SMO, J48 and NB algorithms. There was no significant differences with the MNB and IBk algorithms, but the IMBHN algorithm was the first in the ranking of the statistical test.

Table II
BEST ACCURACIES AND AVERAGE RANKING FOR THE ALGORITHMS USED IN EXPERIMENTS.

Dataset	**NB**	**MNB**	**J48**	**SMO**	**IB*k***	**RHLMS**
Classic4	88.48	**95.79**	90.35	94.53	94.24	95.19
CSTR	78.59	**84.64**	66.85	75.26	82.29	80.24
Dmoz-Sports-500	75.85	83.76	83.88	85.48	80.09	**88.95**
Dmoz-Health-500	73.15	**82.07**	73.61	79.90	77.83	81.76
Dmoz-Computers-500	59.67	**70.13**	54.88	66.01	63.70	65.07
Dmoz-Science-500	62.11	**73.81**	57.20	67.38	64.38	70.94
Fbis	61.79	77.18	71.49	78.92	80.99	**81.76**
Hitech	62.92	**72.92**	56.76	66.44	71.79	71.88
IrishEconomic	59.75	**67.65**	51.08	65.54	60.90	64.81
La1s	75.21	**88.17**	76.65	84.30	80.55	88.01
La2s	75.25	**89.91**	76.84	86.76	82.79	89.29
New3s	56.72	79.16	70.85	71.93	79.26	**83.03**
NFS	70.86	**83.84**	70.74	81.87	78.88	82.22
Oh0	79.66	**89.83**	80.95	81.55	81.85	88.14
Oh5	78.76	**86.27**	80.39	77.24	79.41	86.26
Oh10	72.38	**80.66**	72.09	76.00	73.61	77.52
Oh15	75.03	**83.68**	75.78	75.03	75.57	81.16
Ohscal	62.78	74.73	71.30	**76.69**	68.65	76.01
Opinosis	60.74	59.56	60.83	61.03	**62.87**	58.26
Polarity	66.80	80.10	68.25	**83.65**	70.50	82.70
Re0	57.05	79.92	75.26	77.79	83.51	**84.70**
Re1	66.73	83.34	79.60	72.72	81.89	**85.09**
Re8	81.27	95.33	90.73	93.95	94.14	**96.92**
Reviews	85.22	93.33	88.35	91.64	92.30	**94.20**
SyskillWebbert	72.51	90.75	**95.81**	77.85	**95.81**	94.93
Tr11	54.06	85.00	78.98	77.06	**86.95**	85.02
Tr12	57.82	80.15	79.23	69.61	**81.74**	79.87
Tr21	47.95	61.35	81.27	79.77	**88.66**	87.84
Tr23	56.85	70.61	**93.19**	72.54	84.33	88.26
Tr31	79.72	94.38	93.10	90.72	94.60	**95.57**
Tr41	84.95	**94.52**	90.77	87.80	93.04	93.28
Tr45	66.66	82.46	**90.28**	81.30	88.84	89.13
Wap	71.73	81.02	67.05	81.85	74.48	**83.58**
WebKb	41.39	60.38	**69.08**	57.20	67.95	68.47
WebACE	83.17	87.43	81.28	87.23	83.84	**89.74**
Average Ranking	5.60	2.31	4.35	3.65	3.10	**1.97**
Posição	6º	2º	5º	4º	3º	1º

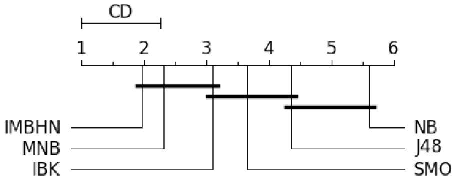

Figure 2. Critical difference diagram considering the best accuracy for each algorithm.

A single parameter for each algorithm was also considered for comparison, considering that in practical situations it is not feasible for the user to carry out tests with a large number of parameters for different algorithms. A statistical test was performed considering the accuracy rates from each parameter of the algorithm and we considered the best parameter in the ranking. The best parameters were linear kernel with $C = 1$ for SMO, $k = 7$ and weighted vote for IBk, error correction rate $= 0.1$ and minimum mean squared error $= 0.01$ for IMBHN. The IMBHN algorithm obtained the highest accuracy in 12 of 35 collections in this situation.

Figure 3 presents the critical difference diagram considering only one parameter for the algorithms. The algorithm IMBHN was superior with statistically significant difference compared to the IBk, SMO, J48 and NB algorithms.

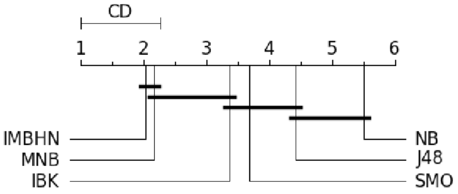

Figure 3. Critical difference diagram considering the accuracies obtained for only one parameter for each algorithm.

The IMBHN proved to be competitive regarding other inductive algorithms used in this work. The results showed that the algorithm IMBHN obtained higher accuracy for a large number of textual document collections. IMBHN algorithm obtained the highest accuracy for balanced class collections, such as Dmoz-Sports-500, New3s and Ohscal, and unbalanced class collections, as Tr11, Tr23, Tr31, Re8 and Reviews.

Furthermore, the IMBHN algorithm always obtained the first position in the statistical test ranking. It presented better results with statistically significant differences compared with the SMO, J48 and NB algorithms when the best accuracies for each algorithm were considered. Statistical significant differences were also observed for IBk algorithm when considering a single parameter for each algorithm.

V. Conclusions and Future Work

In this paper we propose the Inductive Model Based on Heterogeneous Network (IMBHN) algorithm. IMBHN generates a classification model considering a representation of documents modeled as a bipartite heterogeneous network. The algorithm induces weights to objects that represents terms of the collection, which indicates the influence of these terms in the definition of the classes of the documents. After obtaining the weights of the terms for each class, this information is used as a model to classify unseen documents.

Experiments using 35 collections with different characteristics showed that the IMBHN algorithm outperforms, with statistically significant differences, traditional and state-of-art inductive algorithms. As future work, other types of relationships between objects will be addressed to improve the induction process.

Acknowledgments

This research is supported by the Brazilian agency FAPESP (processes 2011/12823-6, 2011/19850-9 and 2011/23689-9).

References

[1] R. Feldman and J. Sanger, *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press, December 2006.

[2] C. C. Aggarwal and C. Zhai, Eds., *Mining Text Data*. Springer, 2012.

[3] M. Newman, *Networks: An Introduction*. New York, NY, USA: Oxford University Press, Inc., 2010.

[4] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys*, vol. 34, no. 1, pp. 1–47, 2002.

[5] M. Ji, Y. Sun, M. Danilevsky, J. Han, and J. Gao, "Graph regularized transductive classification on heterogeneous information networks," in *Principles of Data Mining and Knowledge Discovery*, 2010, pp. 570–586.

[6] G.-R. Xue, D. Shen, Q. Yang, H.-J. Zeng, Z. Chen, Y. Yu, W. Xi, and W.-Y. Ma, "Irc: An iterative reinforcement categorization algorithm for interrelated web objects," in *ICDM'04: Proceeding of the International Conference on Data Mining*, 2004, pp. 273–280.

[7] Z. Yin, R. Li, Q. Mei, and J. Han, "Exploring social tagging graph for web object classification," in *SIGKDD'2009: Proceedings of the International Conference on Knowledge Discovery and Data Mining*, 2009, pp. 957–966.

[8] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *NIPS'2003: Proceeding of the Advances in Neural Information Processing Systems*, 2003.

[9] B. Widrow and M. E. Hoff, "Adaptive switching circuits," in *Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record, Part 4*, 1960, pp. 96–104.

[10] G. Forman, "19MclassTextWc dataset," 2006. [Online]. Available: http://sourceforge.net/projects/weka/files/datasets/text-datasets/19MclassTextWc.zip/download

[11] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*, 2nd ed. Morgan Kaufmann, 2005.

[12] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *ICML 2006: Proceeding of the Internation Conference on Machine Learning*, 2006, pp. 161–168.

[13] T. Kohonen, G. Barna, and R. Chrisley, "Statistical pattern recognition with neural networks: benchmarking studies," in *IEEE International Conference on Neural Networks*, 1988, pp. 61–68.

[14] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.