Centrality Measures from Complex Networks in Active Learning

Robson Motta, Alneu de Andrade Lopes, and Maria Cristina F. de Oliveira

Instituto de Ciências Matemáticas e de Computação (ICMC)
University of São Paulo, P.O. Box 668, 13560-970, São Carlos, SP, Brazil

Abstract. In this paper, we present some preliminary results indicating that Complex Network properties may be useful to improve performance of Active Learning algorithms. In fact, centrality measures derived from networks generated from the data allow ranking the instances to find out the best ones to be presented to a human expert for manual classification. We discuss how to rank the instances based on the network vertex properties of closeness and betweenness. Such measures, used in isolation or combined, enable identifying regions in the data space that characterize prototypical or critical examples in terms of the classification task. Results obtained on different data sets indicate that, as compared to random selection of training instances, the approach reduces error rate and variance, as well as the number of instances required to reach representatives of all classes.

Keywords: Complex networks, Active learning, Text mining.

1 Introduction

Text Mining [1] addresses the development of techniques and tools to help humans in tasks that require discriminating potentially useful content from irrelevant material. It encompasses a wide range of techniques in information retrieval, information and topic extraction from texts, automatic text clustering and classification and also strategies supported by visual interfaces [2]. Yet, identifying and selecting relevant information in large repositories of textual documents may still be very difficult, despite the wide availability of text mining techniques.

The problem of automatic text classification requires a set of examples, or instances from the problem domain. A set of labeled (i.e., already classified) instances is input to train a classifier algorithm that will (hopefully) be able to predict the label of new (non-classified) examples, within certain precision. Obtaining training sets for text classification tasks is particularly critical, as labeling an even moderately large set of textual documents demands considerable human effort and time [3].

Active Learning handles this problem departing from the assumption that even when faced with a reduced training set, a learning algorithm may still achieve good precision rates as long as training instances are carefully selected [3]. An active learner may pose to an 'oracle' (e.g., a human expert) queries relative

J. Gama et al. (Eds.): DS 2009, LNAI 5808, pp. 184–196, 2009.

to critical examples, such as those located in class borders. This approach is strongly motivated by the scarcity of labeled instances, particularly severe in the case of text data.

We present some preliminary results indicating that Complex Network properties may be useful to improve performance of active learning algorithms. In fact, centrality measures derived from networks generated from the data allow ranking the training instances to find out the best examples to be presented to a human expert for manual classification.

The rest of the paper is organized as follows. In Section 2 we briefly discuss related work on active learning and also introduce a few concepts in complex networks. In Section 3 we describe how to generate a similarity network from examples on a particular domain – a corpus of scientific papers. We also discuss how centrality measures obtained from such network derived from a corpus of scientific papers may help to select training instances for a paper classification task. In Section 4 we evaluate the proposed approach on different data sets, and finally Section 5 includes some final remarks and a brief comment on further work.

2 Background

2.1 Active Learning

The amount of labeled examples available for training is an important parameter for inductive learning algorithms. They may adopt a supervised learning process, if a large enough set of labeled examples exists, or a semi-supervised learning approach, if otherwise few labeled examples are available. The low number of training examples poses additional challenges in semi-supervised learning.

A popular algorithm for semi-supervised learning is *Co-training* [4], which employs two independent views of data to induce two different hypotheses, adopting either one or two distinct supervised learning algorithms. The presence of two views of each example suggests iterative strategies in which models are induced separately on each view. Then, predictions of one algorithm on new unlabeled instances are employed to expand the training set of the other.

Labeled examples are hardly available in many practical situations involving real data. Nonetheless, a classifier may still be trained with the assistance of a human expert. This is the underlying principle of the active learning paradigm, which addresses the construction of reduced training sets capable of ensuring good precision performance [5]. The rationale is to carefully select the training instances, so that the training set includes those examples that are most likely to strongly impact classifier precision.

Given a set of non-labeled instances, highly representative examples located in the decision boundaries are selected and presented to the expert for labeling. The resulting labeled set is then employed to train the classifier. The problem, of course, is how to identify the representative examples.

Cohn [5] proposes a statistical approach, considering that three factors affect classification error: (a) noise, which is inherent to the data and independent of

the classifier; (b) bias, due to the overall strategy and choices of the induction algorithm; and (c) variance, which measures the variation of the correctness rates obtained by the induced models. Representativeness of a particular example is therefore measured by how much it reduces the errors due to bias and classifier variance. The author formulates techniques to select examples that reduce variance working with Gaussian mixtures and locally weighted regression.

Two strategies may be considered to reduce error due to variance. Committee-based methods [6] employ different classifiers to predict the class of an example, querying the expert whenever there is a conflict. Uncertainty-based methods [7] require expert intervention if the classifier prediction has low confidence.

Addressing the problem of text classification, Tong and Koller [8] introduced an active learning algorithm with *Support Vector Machines*. They attempt to select the instance that comes closest to the hyperplanes separating the data. Also handling texts, Hoi et al. [9] employ the *Fisher information* to select a subset of non-labeled examples at each iteration, while reducing redundancy among selected examples. The Fisher information represents a global uncertainty of each example in the classification model. The authors report experiments showing high efficiency in text classification tasks.

We suggest using vertex centrality and community measures from complex networks to assist example selection in active learning. The goal is to minimize the manual classification effort and, simultaneously, improve precision of automatic classifiers. In the following we introduce the relevant concepts in complex networks.

2.2 Complex Networks

Complex Networks [10] are large scale graphs that model phenomena described by a large number of interacting objects. Objects are represented as graph vertices, and relationships are indicated by edges – which may be directed and/or weighted, depending on the nature of the problem. Objects and relationships are usually dynamic and determine the network behavior.

Several models of network behavior have been identified and extensively discussed in the literature – a detailed description and discussion may be found elsewhere [11]. We shall restrict ourselves to briefly introducing a few properties of networks and their vertices that are directly related to the approach described in Section 3.

Vertex Degree: As in ordinary graphs, the degree of a vertex is given by the number of its adjacent edges.

Connectivity Distribution: Defined as the probability of a randomly selected vertex having degree k. The connectivity distribution of a network is defined by a histogram of the degrees of its vertices.

Closeness: Central vertices are strategic elements in network topology. A measure of centrality referred to as vertex *closeness* [12] is obtained by computing the inverse of the average shortest-path length from the vertex to all the other vertices in the network. The higher its closeness, the closer the vertex is, in average, to the remaining network vertices.

Betweenness: This is another centrality measure. Equation 1 describes the *betweenness* of a vertex v_i, contained in a set of vertices V. $amt\_shortest\_paths_{ab}$ is the number of shortest paths between an arbitrary pair of vertices v_a e v_b and $amt\_shortest\_paths_{ab}(i)$ is the number of such paths that include vertex v_i. A high value of betweenness is typical of vertices that link groups of highly connected vertices.

$$b_i = \sum_{a \neq b \neq i \in V} \frac{amt\_shortest\_paths_{ab}(i)}{amt\_shortest\_paths_{ab}} \tag{1}$$

Community Structure: Newman [10] defines community structure as a property of networks that are organized in groups of vertices that are strongly connected amongst themselves, (i.e., internally, within the group), and weakly connected with elements in other (external) groups. This is typical of networks organized into a modular structure. Several strategies and algorithms have been proposed to identify community structures in networks. Newman [13], for example, introduced an agglomerative hierarchical algorithm that has the advantage of not requiring the number of communities as an input parameter [13].

3 Centrality Measures in Active Learning

Given an unlabeled data set and a measure of similarity between any two examples, it is possible to derive a network from the examples. Data examples are represented as network vertices, which will be connected by edges based on a certain criterion. In this case, the chosen criterion is a measure of similarity between the examples represented by the vertices.

The goal is to derive a hierarchical similarity-based network that attempts to (i) capture the community structure of the data; (ii) prioritize links among highly similar data instances; and (iii) search for a network with a desired average degree. The rationale is that instance data similarity structure will be expressed in the topology of networks constructed employing these criteria. The following procedure has been adopted to create a hierarchical similarity-based network.

An initial network includes all available examples as vertices, and has no edges, so that each vertex constitutes a single component. An iterative hierarchical agglomerative process starts that gradually connects vertex pairs, based on a given similarity threshold. Assuming the similarity measure takes values in the range $[0, 1]$, where 1 indicates highly similar examples, the similarity threshold is initialized with a high value, close to one.

An outer loop (the second While in Algorithm 1) inspects all vertex pairs whose similarity measure is above the current similarity threshold. It is responsible for identifying the potential edges to be added, i.e., those pairs whose similarity is above the threshold, and calls an inner loop, shown in Algorithm 2, to select the vertices or components to be actually joined. The similarity threshold is updated at each external iteration step in order to ensure that only the

5% most similar vertex pairs still unconnected are considered for potential connection. The whole process stops when all vertices have been connected into a single component forming a connected network.

Edge inclusion in a component stops when the component reaches a user defined average degree. In the inner loop depicted in Algorithm 2, an edge inclusion that joins components is performed if (i) the edge will link two highly similar vertices, and (ii) the two components share a high number of potential edges (i.e., highly similar vertices). A subset of the potential edges is actually added at each iteration, until the average degree of each component reaches a desired (user defined, for each component) value. The edges effectively added are those that, if included – thus causing their respective components C_i and C_j to be joined – will maximize the measure given by Equation 2.

$$interconnectivity(C_i, C_j) = \frac{1}{\#C_i + \#C_j} \sum_{\substack{x \in C_i, y \in C_j, \\ \exists edge(x,y)}} sim(x, y) \tag{2}$$

In the above equation, C_i and C_j denote components, $\#C$ represents the number of vertices in component C, $sim(x, y)$ denotes the similarity between vertices $x \in C_i$, $y \in C_j$. The equation is computed for all pairs of components defined in the current iteration, seeking the set of edges that maximizes its result. This approach ensures that components resulting from this iteration have maximum intra-component connectivity and minimum inter-component connectivity.

Algorithm 1. Construction of the Hierarchical Similarity Based Network

Input:
Set of vertices: $V = v_1,...,v_n$
Average Degree: $averageDegree$
Data similarity matrix: $similarity$
Output:
Network, given by a set of vertices and a set of edges: (V,E)

Components $C \leftarrow V$
Edges $E \leftarrow \emptyset$
$minSim \leftarrow$ similarity threshold to obtain the 5% most similar pairs of vertices
While ($\#C > 1$)
While ($\exists$ pair (x,y) | similarity(x,y)$\geq minSim$, $x \in C_i$, $C_i \in C$, $y \in C$-C_i)
Components Coalescing(C,E,$averageDegree$,$minSim$) % selection of components to be joined
$minSim \leftarrow$ similarity to add the 5% most similar pairs of vertices

Returns (V,E)

A characteristic of such a network is that similar examples – typically expected to be associated to the same class – are likely to define communities. In other words, they form groups of vertices that are densely connected among themselves, with few connections to external groups. Thus, one would expect

Algorithm 2. Components Coalition

Input:
Set of components: C
Set of Edges: E
Average Degree: $averageDegree$
Similarity Threshold: $minSim$
Output:
Set of components: C
Set of Edges: E

preSelectedEdges $\leftarrow \emptyset$
For each component C_i of C
$numberOfEdges \leftarrow$ ($averageDegree$ * $\#C_i$ / 2) - $\#E(C_i)$
If (numberOfEdges $\leq$ 0)
numberOfEdges $\leftarrow$ 1)
For all pair(i,j) | $i \in C_i$ and $j \in C$ - C_i % Edges (i,j) taken from a priority queue
If (similarity$(i,j) \geq minSim$)
preSelectedEdges(C_i,C_j)$\leftarrow$preSelectedEdges(C_i,C_j) $\cup$ (i,j)
$numberOfEdges--$
If ($numberOfEdges == 0$)
$break$
(C_a,C_b) $\leftarrow$ max(interconnectivity(C_i,C_j)) % selected components
$C_a \leftarrow C_a \cup C_b$ %join components
$E(C_a) \leftarrow E(C_a) \cup E(C_b) \cup$ preSelectedEdges(C_a,C_b) %add edges of the joined components
$C \leftarrow C$ - C_b %removes joined component
$A \leftarrow E$ - $E(C_b)$ %remove from E edges of the removed component

Returns (C,E)

the community structure of the network to reflect the underlying data similarity structure. The network would ideally have few connections between vertices in different communities, and the communities would be formed by groups of similar examples likely to belong to the same class. A probabilistic version of this algorithm has been fully described by Motta et al. [14].

The schema shown in Figure 1 illustrates our working hypothesis, that centrality measures bear a strong relation with the role of examples in a classification process. To illustrate this point, let us consider that two communities have been identified in a hypothetical network generated by the above process. Notice that:

1. *Boundary vertices*, located in the borders of the communities, labeled regions R1 in the figure, typically have low values for closeness, computed for the vertex relative to its component.
2. *Inner vertices*, located in the regions labeled R2 in the figure, are those well identified with a specific community. They typically have high values for closeness, again computed relative to the vertex component.

3. *Critical vertices*, located in the region labeled R3, are those placed across different communities. They typically have low values for closeness computed relative to their own community. Moreover, their betweenness values, computed in the context of the network as a whole, are high, as they are elements linking different communities.

We argue that these regions identified in the above network reflect, to some extent, the topology of the data examples relative to their possible classes. Thus, vertex centrality measures help identify the examples representative of the different regions. Hence, by analyzing such measures, or a combination of them, one may identify the interesting examples in an active learning process.

Boundary vertices correspond to examples that, although not prototypical of a class, have good chances of being properly classified. *Inner vertices* correspond to the prototypical examples that would be easily classified, whereas *Critical vertices* are likely to represent the problematic examples, those in the borders between classes. Thus, we suggest an active learning approach that adopts the following steps:

1. given the examples, generate the hierarchical similarity-based network;
2. partition this network into communities;
3. compute the closeness for all vertices in each community;
4. compute the betweenness for all vertices in the whole network;
5. rank vertices based on (a combination of) selected centrality measures to select examples

As we assume that the three types of region identifiable in the network communities define data set topology, a representative training sample should therefore include examples taken from the three regions from each major community. As measures for betweenness and closeness have different ranges, values are normalized in the interval [0,1]. To rank Critical vertices (in regions R3) one may consider the difference between the normalized measures of betweenness and closeness. A value closer to 1 is indicative of vertices in the critical region.

We shall first illustrate the approach on a data set of scientific papers. Figure 2(a) illustrates a similarity network obtained with the above algorithms from a corpus of nearly 600 papers in the subjects *Case-Based Reasoning*, *Inductive Logic Programming* e *Information Retrieval* (named corpus CBR-ILP-IR). Examples have been labeled by a human expert, based on their source vehicle, so that each subject is taken as a class.

To construct the network, the papers are described by their vector representation, according to the well known bag-of-words model [15]. A data similarity matrix has been computed with the cosine distance over the vector space model, for all document pairs. The average degree threshold has been set to five (see Section 4 for a discussion on this choice).

In the network graph shown in Figure 2(a) each vertex represents a paper, whose known class is indicated by the vertex shade. The same figure, in (b), shows the network after applying the community detection algorithm by Newman [13], mentioned in Section 2. One observes from the figure that most communities have a clear predominance of elements from a single class. This suggests

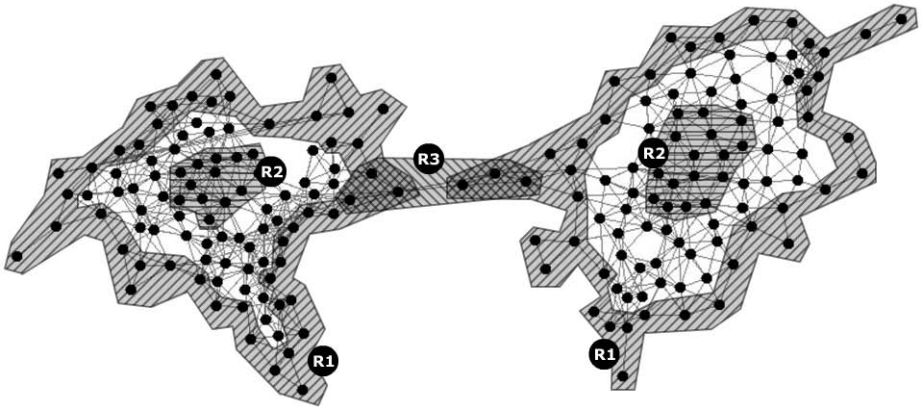


Fig. 1. Schema illustrating values of centrality measures and their relationship to the location of examples relative to their possible classes

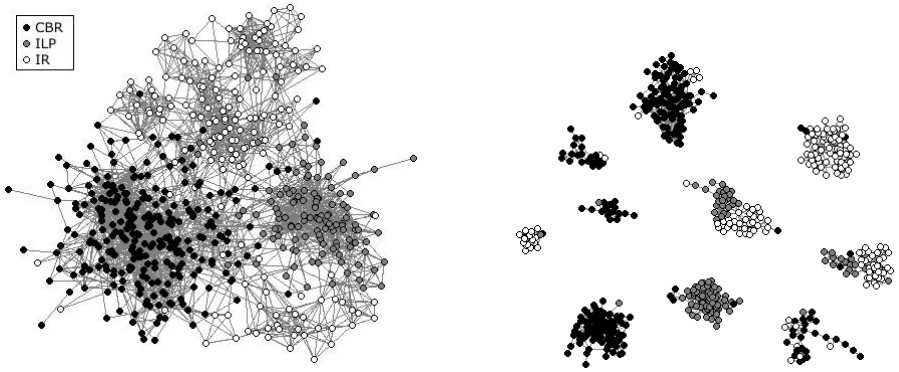
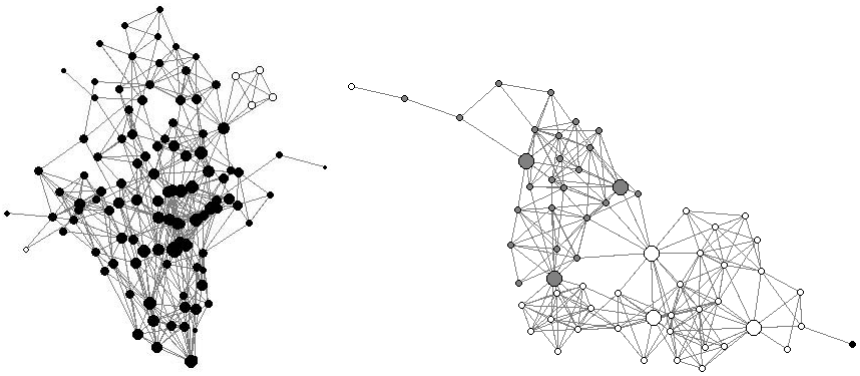


Fig. 2. (a) Hierarchical similarity network for the corpus of scientific papers (CBR, ILP e IR) and (b) communities identified in the network

that the community structure identified by the algorithm, without using any class information, reflects reasonably well the known class information.

Figure 3 (a) depict information about vertex closeness, computed for the larger community structure identified in the network. Vertex size maps measure values: greater values are shown as larger glyphs. As expected, one observes vertices with lower closeness in the boundary of the community.

Figure 3 (b) shows another community from the network, with vertex shade mapping the class of the corresponding example. Notice this community includes a single example from the (CBR) class, shown in black. In the same figure the 10% vertices of this community with higher values computed for betweenness are shown as larger glyphs. Inspection shows that these higher values are either in or close to the borders between classes.



(a) (b)

Fig. 3. (a): vertex size maps the value of vertex closeness, for a particular community. (b) shows class distribution in another community that has a predominance of two classes (shown in white and gray). Also in this figure, the larger vertices shown correspond to the 10% vertices from this community with higher values for betweenness.

4 Results

The evaluation process was carried out on 10 data sets from UCI repository and on the CBR-ILP-IR corpus already mentioned. We conducted two kinds of experiments on each data set: the first one to verify how many instances should be labeled in order to ensure the identification of all classes present in the training data set; and a second to measure the variation of the classifier error relative to the number of examples selected for labeling.

In both cases, results reported were averaged on 100 runs for data sets with cardinality lower than 500, and with 30 runs otherwise. Selection of training instances with the proposed approach was compared with a random selection. Similar results were obtained with Naïve Bayes and K-Nearest Neighbor (KNN) (with $K = 1$) classifiers. Figure 4 shows the classification errors for the KNN classifier on the CBR-ILP-IR data, considering different policies to select the training data, as described next.

First, the similarity based network is created for the data set. All networks were created by setting the desired average degree for the communities equal to 5. Various experiments conducted on these and other data sets have shown that when lower average degrees (up to 3) are set community structures are not well defined. By 'well defined' we mean communities densely connected internally and sparsely connect with other communities. On the other hand, when setting values above 5 the number of edges increased unnecessarily for the purpose of obtaining clearly defined community structures in the data.

The similarity matrices were computed with the cosine distance for the CBR-ILP-IR data and with the Euclidean distance for the numerical (normalized) data sets. The community structure was then extracted from the resulting network

using the aforementioned Newman´s algorithm. The following procedure has been adopted to select instances.

Communities are first sorted in decreasing order of their size (number of vertices). Then, the examples in each community are ranked by decreasing values of their closeness (in the community), and all the network examples are ranked in decreasing value of their betweenness.

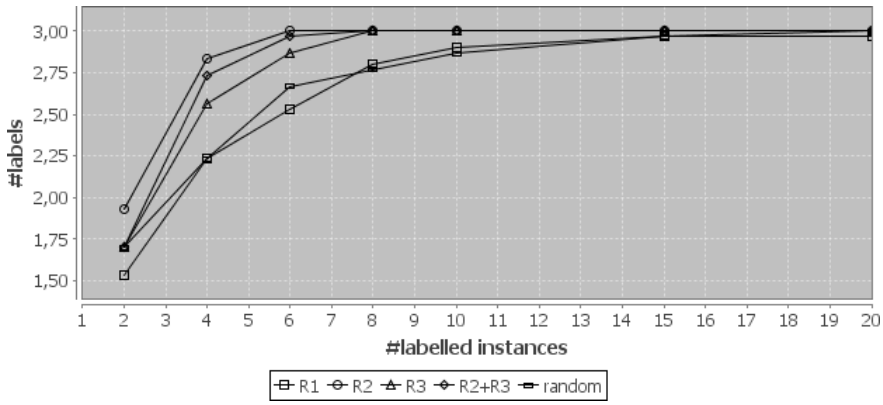
The overall process is conducted by iteratively scanning the ordered list of communities, selecting one sample from each community at each iteration. Several policies may be adopted to select instances, taking examples from the different regions (inner, boundary or critical). The following alternative policies have been considered to create test data sets in this study:

- Policy $R1$: in forming the test data set, priority is given to instances from the *boundary* region. At each iteration, the bottom ranked instance from each community is selected, i.e., the instance with the lowest closeness;
- Policy $R2$: priority is given to vertices from the *inner* region. The top ranked instance from each community is selected, i.e., the instance with the highest closeness;
- Policy $R3$: priority is given to vertices from the *critical* region. The instance with the highest critical value is selected from each community, computed from the difference $betweenness - closeness$;
- Policy $R2 + R3$: the test data set is formed selecting vertices based on a combination of *critical* and *inner* regions. At successive iterations, alternate between selecting from each community an instance from the Inner region (highest closeness) and an instance from the Critical region (highest critical value);

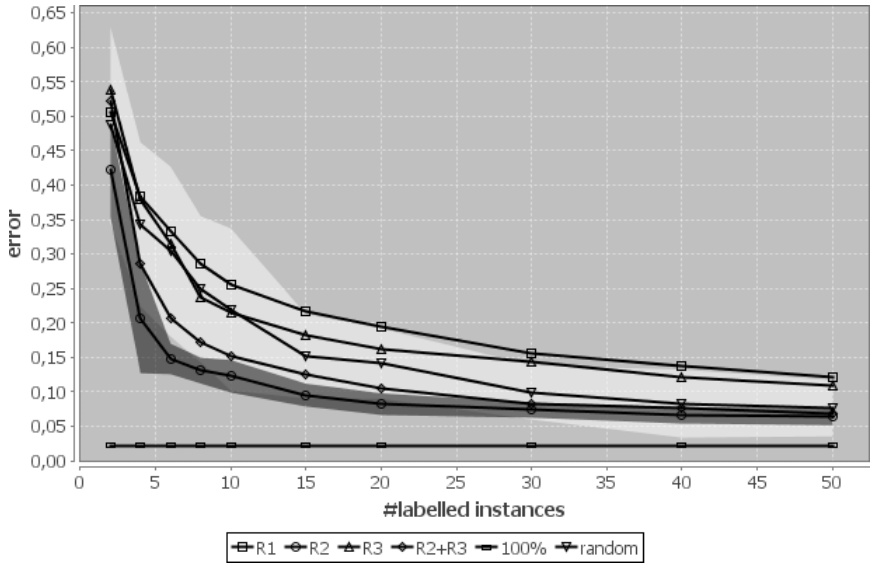
In all cases, once an instance is selected for the training data it is removed from further consideration. The iterative selection process stops when the desired number of training instances is reached. In these studies, this number has been set as equal to the number of communities identified in the network. We also generated training data sets of the corresponding size by selecting instances *randomly* and, for comparison purposes, employed a training data set including 100% of the labelled instances.

In all experiments we observed that selection of examples that correspond to vertices in Inner and Critical regions resulted in greater error reduction in the classifier output. Considered in isolation, Inner vertices provide the best examples for error reduction, followed by Critical vertices and then Boundary vertices.

Notice that policy $R3$ (selecting from critical region alone) does not produce good results, although this region is associated with higher classification uncertainty. In fact, the experiment indicates that the human expert must also classify some 'prototypical' examples (those in region $R2$). In these experiments an equal number of elements was selected from each region – we did not investigate whether assigning more weight to either regions $R2$ or $R3$ would improve the results.



(a)



(b)

Fig. 4. Comparing different policies to select training instances (average in 100 runs). (a) Number of examples required to identify all classes and (b) KNN classifier error for the CBR-ILP-IR corpus, considering examples selected with the proposed approach under different policies, and at random. For the sake of clarity, variance is represented in the figure only for the random selection policy (lighter region in the figure) and for the $R2$ policy (darker region).

In Figure 4 (a) one observes that the number of examples required to identify all classes is considerably reduced when examples are selected with either policy $R2$ or policy $R2 + R3$. Figure 4(b) shows the error of the KNN classifier as the number of labeled examples in the training set increases. The Naïve Bayes classifier presented a similar behavior. Errors have been computed for training

Table 1. Comparing the classification errors for different data sets. For each test data set we inform the number of known examples, the number of classes, the number of instances desired in the training set, equal to the average number of communities in each run, and the errors for the different instance selection policies. In the table, results from the policy that presented the lowest error (ignoring the error obtained with the training set that includes all examples) are highlighted in bold, whereas the second lowest error policy results are highlighted in italics.

dataset	#instances	#classes	#labelled instances	R1	R2	R3	R2+R3	random	100%
balance	625	3	14	0.312	**0.297**	0.311	*0.306*	0.316	0.223
ecoli	336	8	11	0.327	**0.263**	0.333	*0.29*	0.300	0.193
glass	214	7	10	*0.345*	**0.283**	0.451	0.361	0.352	0.090
ionosphere	351	2	12	0.276	*0.038*	0.082	**0.025**	0.172	0.011
iris	150	3	8	0.199	**0.057**	0.178	*0.111*	0.153	0.047
sonar	208	2	11	0.416	**0.331**	0.382	*0.342*	0.390	0.136
wdbc	569	2	14	0.155	**0.084**	0.108	*0.094*	0.095	0.049
wine	178	3	8	0.234	**0.087**	0.209	*0.091*	0.169	0.050
yeast	1484	10	20	0.654	**0.577**	0.627	0.594	*0.590*	0.473
zoo	101	7	7	0.288	**0.157**	0.352	*0.210*	0.299	0.040
CBR-ILP-IR	574	3	14	0.225	**0.103**	0.181	*0.130*	0.182	0.022

sets obtained with each of the above policies, and also a training set consisting of all known examples.

In Table 1 we synthesize the classification errors obtained for multiple data sets. Notice that all data sets, except the *glass* and the *yeast*, present similar behavior: the lowest error rates and lowest variances are obtained by selecting the training instances using policies $R2$ (inner vertices) and $R2 + R3$ (inner and critical vertices).

5 Conclusions and Further Work

We investigate a novel approach to support active learning in classification tasks, based on properties computed from hierarchical similarity networks derived from the known data instances. A discussion has been presented on how vertex centrality measures obtained from network communities enable identifying regions in the data space that characterize critical examples. Such measures may guide the selection of examples to be presented to a human expert in an active learning approach for classification tasks.

Results of applying the proposed approach on several data sets have been presented and discussed, including an illustrative example of its application on a corpus of scientific papers. These results indicate the potential of this approach for the problem. As further work, it would be desirable to investigate further the potential role of these and other network measures, under different classification algorithms (e.g., Support Vector Machines) and for distinct data set domains. How different network construction procedures may affect the instance selection process is also a topic that deserves further investigation.

Acknowledgments

The authors acknowledge the financial support of FAPESP (Grants 2008/04622-8 and 2009/03306-8) and CNPq (Grant 305861/2006-9).

References

1. Berry, M.: Survey of Text Mining: Clustering, Classification, and Retrieval. Springer, Heidelberg (2003)
2. Minghim, R., Levkowitz, H.: Visual mining of text collections, Tutorial Notes. In: Proc. of EUROGRAPHICS 2007, Computer Graphics Forum, pp. 929–1021 (2007)
3. Settles, B.: Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison (2009)
4. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: COLT 1998: Proceedings of the eleventh annual conference on Computational learning theory, pp. 92–100. ACM, New York (1998)
5. Cohn, D.A., Ghahramani, Z., Jordan, M.I.: Active learning with statistical models. Journal of Artificial Intelligence Research 4, 129–145 (1996)
6. Engelson, S., Dagan, I.: Minimizing manual annotation cost in supervised training from corpora. In: Proceedings of the 34th annual meeting on Association for Computational Linguistics, Morristown, NJ, USA, pp. 319–326. Association for Computational Linguistics (1996)
7. Hwa, R.: Sample selection for statistical grammar induction. In: Proceedings of EMNLP/VLC 2000, pp. 45–52 (2000)
8. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. Journal of Machine Learning Research 2, 45–66 (2002)
9. Hoi, S., Jin, R., Lyu, M.: Large-scale text categorization by batch mode active learning. In: WWW 2006: Proceedings of the 15th international conference on World Wide Web, pp. 633–642. ACM, New York (2006)
10. Newman, M.: The structure and function of complex networks. SIAM Review 45(2), 167–256 (2003)
11. Costa, L.F., Rodrigues, F.A., Travieso, G., Boas, P.V.: Characterization of complex networks: A survey of measurements. Advances In Physics 56, 167 (2007)
12. Wasserman, S., Faust, K.: Social Network Analysis: Methods and Applications. Cambridge University Press, Cambridge (1994)
13. Newman, M.: Fast algorithm for detecting community structure in networks. Physical Review E 69, 066133 (2004)
14. Motta, R., Almeida, L.J., Lopes, A.A.: Probabilistic similarity based networks for exploring communities. In: I Workshop on Web and Text Intelligence (SBIA-WTI 2008), Salvador, Brasil, pp. 1–8 (2008) (in Portuguese)
15. Baeza-Yates, R.A., Ribeiro-Neto, B.A.: Modern Information Retrieval. ACM Press/Addison-Wesley (1999)