JMLR: Workshop and Conference Proceedings 16 (2011) 127–139 Workshop on Active Learning and Experimental Design

Active Learning with Clustering

Zalán Bodó ZBODO@CS.UBBCLUJ.RO
Zsolt Minier MINIER@CS.UBBCLUJ.RO
Lehel Csató LEHEL.CSATO@CS.UBBCLUJ.RO
Faculty of Mathematics and Computer Science
Babeş–Bolyai University, Cluj-Napoca, Romania

Editor: I. Guyon, G. Cawley, G. Dror, V. Lemaire, and A. Statnikov

Abstract

Active learning is an important field of machine learning and it is becoming more widely used in case of problems where labeling the examples in the training data set is expensive. In this paper we present a clustering-based algorithm used in the Active Learning Challenge[1]. The algorithm is based on graph clustering with normalized cuts, and uses k-means to extract representative points from the data and approximate spectral clustering for efficiently performing the computations.

Keywords: active learning, large scale spectral clustering, normalized cuts, support vector machines

1. Introduction

In active learning the learner *queries* data points from a large *data pool* that are thought to be the most informative (Settles, 2009). Active learners are useful when obtaining the label of a point is expensive. For example we can consider text categorization problems with a large number of categories – order of thousands or so – where data is easily collected but the assignment of documents to categories requires background knowledge and careful examination, being very time consuming when performed manually.

To find the labels of unlabeled examples, *oracles* are queried in different ways. A popular scenario is pool-based active learning (Lewis and Gale, 1994), where we assume a large data set with only a few labeled and the majority unlabeled examples. An item is chosen by inspection from the unlabeled pool. Other scenarios include query synthesis (Angluin, 1988), where queries are synthesized and novel examples can be generated, or stream-based selective sampling (Atlas et al., 1990), where the examples are coming successively and for each example one has to decide independently whether it is informative or not.

The central problem in active learning is the selection procedure, which can be reduced to *measure* the information content of the unlabeled points. This problem is called the *query strategy*. These can be based on the probabilistic output of a classifier, on the agreement between the members of a committee, based on the estimated reduction of error, to name a few (see eg. Lewis and Gale, 1994; Seung et al., 1992; Roy and McCallum, 2001).

In this paper we propose an active learning method based on spectral clustering (Shi and Malik, 2000) and large-scale approximate spectral clustering (Yan et al., 2009). Our

[1] http://www.causality.inf.ethz.ch/activelearning.php

algorithm is based on graph clustering with normalized cuts and uses the property that normalized cuts partition the data using a hyperplane (Rahimi and Recht, 2004). Therefore informativeness can be measured with the unthresholded cluster indicator values as produced by the clustering algorithm; this output can be interpreted as the output of a maximum margin-based classifier.

Since the simple heuristics of using the distance of a point from the separating hyperplane as a measure of informativeness – the smaller the better – was efficient (Tong and Koller, 2001), we apply this strategy in our algorithm. We mention that other semi-supervised or constrained clustering methods could be used, our choice of constrained spectral clustering leads to the query strategy as above whose application is straightforward, and additionally the spectral graph transducer proved effective on various data sets (Joachims, 2003).

The paper is structured as follows. Section 2 presents the components of our algorithm: spectral clustering and large-scale approximate spectral clustering (Section 2.1), spectral graph transducer (Section 2.2) and support vector machines (SVMs) for classification and active learning (Section 2.3). In Section 3 the proposed algorithm is presented in details and Section 4 describes the experiments and discusses the results.

1.1. Problem setting and notation

Let the training data be $X = X_L \cup X_U = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\} \cup \{\mathbf{x}_{\ell+1}, \ldots, \mathbf{x}_{N:=\ell+u}\}$, where X_L is the labeled and X_U is the unlabeled part. We assume that data is sampled i.i.d. from an unknown distribution. The goal in the challenge is to query $s \leq u$ labels of yet unlabeled data X_U from an *oracle* that are the most informative for the learning algorithm.

In the Active Learning Challenge the algorithm is evaluated on a separate *test* data set X_T, $|X_T| = t$. The performance is measured based on the number of queried labels, by iteratively increasing the number of known labels, ℓ, from 1 to N. That is, after querying s labels Y_s of some points X_s the labeled and unlabeled data sets change: $X_L = X_L \cup (X_s, Y_s)$, $X_U = X_U \setminus X_s$. We denote vectors by small boldfaces $\mathbf{a}, \mathbf{b}$; matrices by capital boldfaces $\mathbf{A}, \mathbf{B}$; while scalars and sets are denoted by normal letters $a, b, \ldots, A, B$. Furthermore $\mathbf{A}_{i\cdot}$ and $\mathbf{A}_{\cdot j}$ denotes the i-th row and j-th column of $\mathbf{A}$ respectively. We use $\mathbf{A}'$ to for the transpose of $\mathbf{A}$, and $\|\cdot\|$ for the Euclidean norm.

2. Active learning with spectral clustering

2.1. Large-scale spectral clustering

Spectral graph clustering techniques (von Luxburg, 2006) became popular in the last decade owing to their simplicity and efficiency. They minimize an objective function involving graph cuts. The two most popular cut objectives are the *ratio cut* and *normalized cut* (von Luxburg, 2006):

$$\mathrm{rcut}(A_1, A_2) = \sum_{i=1}^{2} \frac{\mathrm{cut}(A_i, \overline{A_i})}{|A_i|}, \quad \mathrm{ncut}(A_1, A_2) = \sum_{i=1}^{2} \frac{\mathrm{cut}(A_i, \overline{A_i})}{\mathrm{vol}(A_i)}, \tag{1}$$

where a cut (A_1, A_2) is defined as sum of edge weights between the two sets of graph vertices A_1 and A_2, and the volume of a partition is the sum of edge weights within the partition and all the vertices of the graph. Since exactly solving the above problems is NP hard, usually the relaxed versions are solved (Shi and Malik, 2000).

For the relaxation we introduce the similarity matrix $\mathbf{W}$ and the diagonal degree matrix $\mathbf{D}$ with $D_{ii} = \sum_j W_{ij}$; the unnormalized graph Laplacian (Chung, 1997) which is defined as $\mathbf{L} = \mathbf{D} - \mathbf{W}$. With these notations, the discrete normalized cut problem can be relaxed to solving the following optimization problem:

$$\begin{aligned} \mathbf{y}^* = \arg\min_{\mathbf{y}} \quad & \mathbf{y}'\mathbf{L}\mathbf{y} \\ \text{s.t.} \quad & \mathbf{y}'\mathbf{D}\mathbf{y} = 1, \quad \mathbf{y}'\mathbf{D}\mathbf{1} = 0 \end{aligned} \tag{2}$$

where $\mathbf{y}$ is the cluster indicator vector. The solution is $\mathbf{y}^* = \mathbf{D}^{-1/2}\mathbf{v}_2$, where $\mathbf{v}_2$ is the eigenvector corresponding to the second smallest eigenvalue of the symmetrically normalized graph Laplacian $\mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$. For crisp clusters the values in $\mathbf{y}^*$ are thresholded and treated as cluster indicators.

Having a time complexity of $O(N^3)$ and space complexity of $O(N^2)$, the application of (2) for large data sets is difficult, therefore efficient approximations are needed. A simple strategy is to reduce the number of points considered for clustering without losing too many of the characteristic features of the original data set.

Yan et al. (2009) proposed a fast approximate spectral clustering – k-means-based approximate spectral clustering or KASP – where the mis-clustering rate converges to zero as the number of extracted representative points grows. The representative points are obtained by using k-means clustering and the algorithm is as follows:

1. Perform k-means clustering on the whole data set.
2. Consider the output of k centers as the representative points.
3. Run a spectral clustering algorithm on the representative points.
4. Based on the clustering of the centers assign the initial points to the clusters determined by the spectral method.

For details of the algorithm see Yan et al. (2009). It was tested on some data sets and led to significant speedups and negligible degradation in clustering accuracy.

Normalized spectral clustering is a kernel method that shares similarities with the SVM (see Section 2.3). In (Rahimi and Recht, 2004) the authors showed that normalized spectral clustering can be expressed in terms of a hyperplane separating the unlabeled points maximizing the *gap* as given below:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \|\mathbf{w}'\mathbf{\Phi}\mathbf{D}^{-1/2}\|^2, \tag{3}$$

where $\mathbf{w}$ is the normal vector of the hyperplane and $\mathbf{\Phi}$ is the matrix of the transformed points, $\mathbf{\Phi}_{\cdot i} = \phi(\mathbf{x}_i)$, and ϕ is the feature mapping. The cluster indicator value for a point $\mathbf{x}_i$ equals $\mathbf{w}^{*\prime}\phi(\mathbf{x}_i)$, and similarly to the case of SVMs (using the *representer theorem* eg from Schölkopf and Smola, 2002) the *cluster indicator function* is written using kernels as:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i k(\mathbf{x}_i, \mathbf{x}), \tag{4}$$

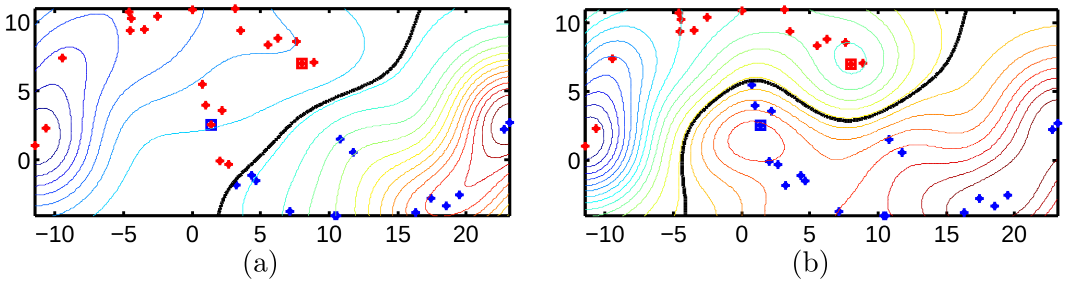

Figure 1: Separating hyperplanes – thick lines – for the *two moons* data set containing two labeled examples: (a) Normalized spectral clustering; (b) Normalized spectral graph transducer.

where $\boldsymbol{\alpha} = \mathbf{D}^{-1/2}\mathbf{v}_2$ from Eq. (2), and $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})'\phi(\mathbf{y})$ is the kernel function. Since the decision function parameters and the cluster indicators are equal ($\boldsymbol{\alpha} = \mathbf{y}^*$), the active learning heuristic of choosing the closest points to the decision surface can be applied (Settles, 2009). The result is that the cluster indicators can be used to predict the importance of a point: as the cluster indicator gets closer to zero, the point becomes increasingly important.

2.2. Constrained spectral clustering

The spectral graph transducer (SGT) method (Joachims, 2003) can be viewed as a constrained spectral clustering algorithm with explicit label constraints. The algorithm uses the ratio cut, but one can also define it using the normalized cut by simply changing the graph Laplacian to the symmetric normalized Laplacian. We obtain therefore a problem similar to the one presented in (Joachims, 2003):

$$\begin{aligned} \min_{\mathbf{z}} \quad & \mathbf{z}'\left(\mathbf{L}_{\text{sym}} + c\,\mathbf{D}^{-1/2}\mathbf{C}\mathbf{D}^{-1/2}\right)\mathbf{z} - 2c\,\mathbf{z}'\mathbf{D}^{-1/2}\mathbf{C}\boldsymbol{\gamma} \\ \text{s.t.} \quad & \|\mathbf{z}\| = 1, \quad \mathbf{z}'\mathbf{D}^{1/2}\mathbf{1} = 0 \end{aligned} \tag{5}$$

where $\mathbf{z} = \mathbf{D}^{1/2}\mathbf{y}$, $\mathbf{y}$ is the resulting cluster indicator, $\mathbf{L}_{\text{sym}} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ is the symmetric normalized graph Laplacian; $\boldsymbol{\gamma}$ contains the labels: $\gamma_i = \pm 1$ for labeled and 0 for unlabeled points, and $\mathbf{C}$ is a diagonal matrix with positive values only at the indexes of the labeled points.

The analysis from (Rahimi and Recht, 2004) can be applied in this case also, since the SGT narrows spectral clustering only by a quadratic constraint and therefore we can say that it also finds a separating hyperplane. Accordingly, the cluster indicator values returned by SGT can be viewed as decision function values $f(\mathbf{x}) = \mathbf{w}'\phi(\mathbf{x})$, where $\mathbf{w}$ is the normal of the separating hyperplane and ϕ is the feature map.

Consider the problem (5). Due to the representer theorem (Schölkopf and Smola, 2002) we again have the decision function as a linear combination of kernels as in Eq. (4). Moreover, we know that $\mathbf{y} = \mathbf{K}\boldsymbol{\alpha}$, and the resulting decision function (Belkin et al., 2006):

$$f(\mathbf{x}) = \sum_{i=1}^{N} (\mathbf{K}^{-1})_{i\cdot}\mathbf{y}\; k(\mathbf{x}_i, \mathbf{x}). \tag{6}$$

Figure 1 shows the separating hyperplanes obtained for the *two moons* data set using spectral clustering and spectral graph transducer.

Similarly to the case of spectral clustering, the absolute value of the cluster indicators returned by the SGT – the distance to the separating hyperplane – can be used to predict the *importance* of a point – a fast and popular *uncertainty sampling* technique for active learning for separating hyperplane-based methods like SVMs (Tong and Koller, 2001).

2.3. Learning with SVMs

Support vector machines – in their original formulation as binary classifiers – find an optimal hyperplane with maximal margin separating the negative examples from the positive ones (Boser et al., 1992; Cortes and Vapnik, 1995). By maximizing the margin of the separating hyperplane, a bound on the actual risk is lowered. The optimization problem is as follows:

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{\ell}\xi_i \\ \text{s.t.} \quad & y_i\left(\mathbf{w}'\mathbf{x}+b\right) \geq 1-\xi_i, \quad \xi_i \geq 0, \quad i=1,\ldots,\ell. \end{aligned} \tag{7}$$

where $\mathbf{w}$ is the normal vector to the separating hyperplane and ξ_i are the misclassification thresholds – or slack variables (Boser et al., 1992; Vapnik, 1995). The Lagrange formulation of this problem lowers the number of constraints, thus simplifies the optimization task (Boyd and Vandenberghe, 2004). The main advantage of the SVM formulation is its ability to deal with linearly non-separable data in a manner similar to the linear case. To handle linearly non-separable cases – instead of scalar products – we use kernel functions, two examples are the *polynomial* and *Gaussian (or RBF)* kernel functions (Schölkopf and Smola, 2002):

$$k_{\text{poly}}(\mathbf{x},\mathbf{z}) = (a\mathbf{x}'\mathbf{z}+b)^c, \qquad k_{\text{rbf}}(\mathbf{x},\mathbf{z}) = \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{z}\|^2\right). \tag{8}$$

Due to the representer theorem (Boser et al., 1992; Vapnik, 1995), the optimal weight vector $\mathbf{w}^*$ can be written as $\mathbf{w}^* = \sum_i \alpha_i^* \phi_i$ and consequently the resulting optimal classification function has the form

$$f^*(\mathbf{x}) = \sum_{i=1}^{\ell} \alpha_i^* y_i k(\mathbf{x}_i,\mathbf{x}) + b^*, \tag{9}$$

where $\boldsymbol{\alpha}^*$ and b^* denote the optimal weight parameters and the optimal bias respectively.

As mentioned in the previous section, we employ a similar method to the simple SVM-based active learning: we assume that a point with unknown label is the more *informative* the *closer* it is to the optimal separating hyperplane from Eq. (9).

We compute thus the distance of each unlabeled point $\mathbf{p}$ from the separating hyperplane $H^* = \{\mathbf{x} \mid f^*(\mathbf{x}) = 0\}$. This is $d(\mathbf{p},H^*) = |f^*(\mathbf{p})|/\|\mathbf{w}^*\|$, and since $\mathbf{w}^*$ is constant for a given hyperplane, it is sufficient to consider $d(\mathbf{p},H^*) \propto |f^*(p)|$ for comparison. We employ the fast method of querying the closest point(s) to the hyperplane, that is the points for which $|f(\mathbf{p})|$ is minimal, since the points near the separating hyperplane and near the margins tend to be more influential (Seung et al., 1992; Schohn and Cohn, 2000).

The choice of querying the points close to the separating hyperplane is once more motivated using the notion of *version space*, defined as the region in parameter space whose values classify *all* labeled data correctly. Tong and Koller (2001) implement an algorithm that selects points that reduce the version space as fast as possible by roughly halving it at each iteration. Since SVMs can be regarded as classifiers finding the center of the hypersphere with largest radius inside the version space – *version space duality* – choosing a point as close as possible to the center of the optimal hypersphere is often close to the center of the version space would practically halve the version space. Choosing the next query point as above leads thus to bisect and reduce the version space very fast.

3. The algorithm

Our proposed algorithm is a combination of constrained spectral clustering and k-means clustering. The algorithm based on a type of constrained spectral clustering, namely on the spectral graph transducer method since the amplitude of the decision function is a measure of informativeness. We implemented a *semi-supervised method* since we wanted to incorporate as much information as it was possible both from unlabeled and from test data sets. Therefore we use spectral graph transducer as long as there are unlabeled data. When all labels are known, support vector machines are applied.

We chose spectral clustering since it is a successful algorithm making no strong assumptions on the form of clusters, nor on the ratio of cluster sizes (Shi and Malik, 2000; von Luxburg, 2006). The spectral graph transducer – and thus our algorithm – is based on the semi-supervised *smoothness assumption*, which says that points in high-density regions should have similar labels (Chapelle et al., 2006).

Owing to the large number of training and test points, approximations are needed to speed up the computations. We decided to use a method that selects or generates *representative* points from the data set, and uses only the representative points instead of the entire data set. To this end we used k-means based approximate spectral clustering, which generates the representative points as the centers of the resulting clusters.

When the number of labeled examples is small, we extract the representative points from the unlabeled and test sets and we add them to the training data set *as unlabeled points*.

When the number of labeled examples becomes large, representative points are also built for the labeled examples – this is done for computational reasons. In this second case the labeled points and the ones generated from the unlabeled and test sets constitute the training data.

The proposed algorithm divides learning into four cases, depending on the number of labeled points: as the number of labeled points increases different methods are needed for efficiently performing the computations; θ is a threshold on the number of labeled points controlling the treatment of labeled data. The distinct cases of the algorithm are shown below.

1. If labeled points have homogeneous labels
    - Perform k-means-based approximate spectral clustering on the whole data set.

2. Else If $\ell < \theta$
   - Perform k-means on the unlabeled and test data.
   - Form the new data set from the labeled points and the centers of the clusters.
   - Perform SGT on the new data set.
3. Else If $\ell \geq \theta$
   - Perform k-means on the unlabeled and test data.
   - Perform k-means on the labeled data separately in each of the two classes.
   - Form the new data set from the centers of the obtained clusters.
   - Perform SGT on the new data set.
4. Else If $\ell = u$
   - Perform bagging with linear SVMs.

In the first case, when the label of only one point is known and while the labeled points belong to the same class, we perform approximate spectral clustering as described in Section 2.1. The labels of individual points are determined by the label of the cluster the point resides in. The informativeness of a point is determined by the closeness of the cluster indicator value to zero.

When more than one label assignments are known, we separate training into two cases depending on the number of data points. If the number of labeled points is less than a predetermined threshold θ, we first perform k-means clustering on the unlabeled and test data, and consider the resulting cluster centers as the new representative points. After forming the new data set from the centers and the labeled points we train a normalized SGT on it (Case 2).

If the number of labeled points is above the threshold we cannot deal anymore with these points separately because of their large number, therefore to reduce their number we cluster the labeled points in each of the two classes using k-means. Thus the new data set is formed by combining the cluster centers obtained from the unlabeled and test set with the cluster centers obtained from the labeled set. As in the previous case we train a normalized SGT using this data set (Case 3).

When all the data labels are known we use a bag of linear support vector machines for the binary classification task. Bagging is used to improve the learning algorithm, that is to reduce the average error of the model (Case 4).

4. Experiments and discussion of the results

The Active Learning Challenge was organized in the frame of the Pascal2 Challenge Program and is part of the AISTATS 2010 and WCCI 2010 conference competition programs. The goal of the challenge was to develop active learning methods for a pool-based learning scenario. The organizers provided $6 + 1$ development (6 development and 1 toy) and 6 final data sets.

The data sets are split in two, the first half contains the training and the second half contains the test data. The training, testing and querying steps proceed in an cycle: using the labeled and unlabeled data one trains the learning algorithm, and for all the examples provides prediction scores for the evaluation system. Based on some criterion a few examples are selected from the first half of the data set for querying its labels, and after obtaining them the process repeats until all the budget is spent – initially everybody is provided a sum of N ECU (Experimental Cash Units), where N is the total number of examples in the data set – or an AUC score of 1 is reached.

The algorithms used in our experiments were the following:

- ALG1 – the simple algorithm which initially uses normalized spectral clustering and then requests all the training labels and uses bagging with linear SVMs (no active learning).
- ALG2 – the algorithm described in Section 3, i.e. the method which starts with normalized spectral clustering when only the label of one point is known, then uses a normalized spectral graph transducer, and finally bagging with linear SVMs.
- SVM – the algorithm using linear SVMs as described in Section 2.3 or (Tong and Koller, 2001).

All the algorithms listed above are uncertainty sampling methods since they choose the most informative points based on how distant a point is from the separating hyperplane. We have already argued why and how normalized spectral clustering and spectral graph transducer can be used for this purpose in Sections 2.1 and 2.2.

In our experiments we used only the following data sets:

Data set	**Domain**	**Features**	**Size**
ALEX	Toy data set	11	5000
IBN_SINA	Handwriting recognition	92	10361
NOVA	Text categorization	16969	9733
A	Handwriting recognition	92	17535
D	Text categorization	12000	10000
F	Ecology	12	67628

because in the development phase we made experiments with data sets A, D and F, and when the final data sets appeared we chose the data sets most similar to these.

For our algorithm the threshold θ was set to 2^8, while at the final step we performed bagging with 20 linear support vector machines.

The k-means clustering has two parameters: the first k denotes the number of clusters formed from the unlabeled and test data, while k_{lab} is the number of clusters containing labeled points; k was set to $(|X_U| + |X_T|)/100$, while k_{lab} to 100.

Spectral clustering and SGT uses the affinity matrix $\mathbf{W}$ for calculating the graph Laplacian. Here we used the complete graph of the examples using the Gaussian similarity,

$$W_{ij} = \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right), \tag{10}$$

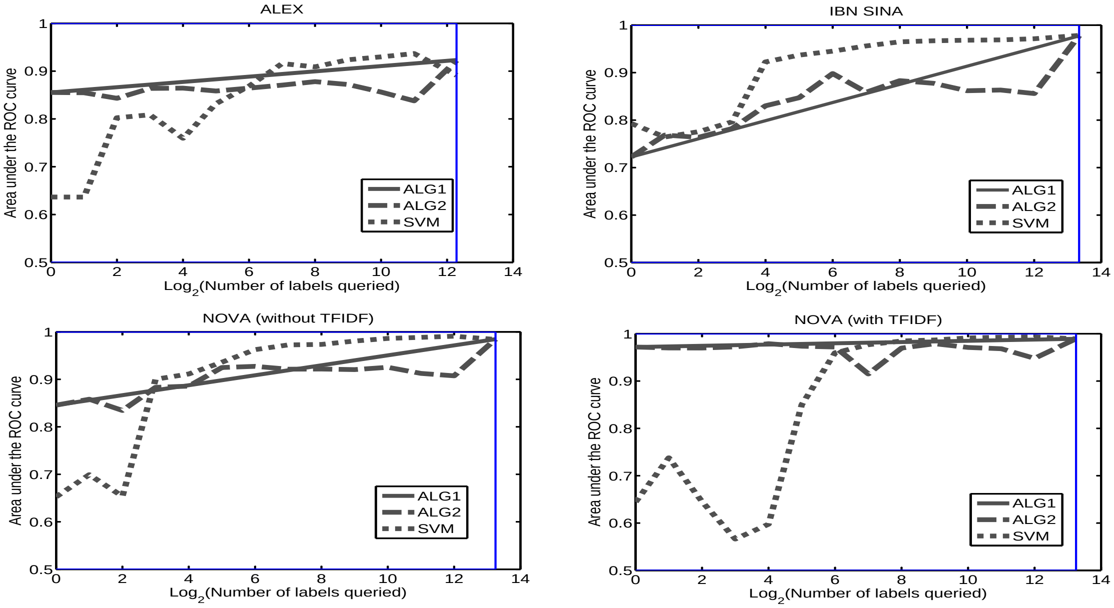

Figure 2: ROC curves for different development data sets: ALEX, IBN_SINA, NOVA, and NOVA using the tf×idf transformation.

	ALEX	IBN_SINA	NOVA	NOVA
				(with tf×idf)
ALG1	91.86/**77.40**	97.73/69.98	98.46/**83.03**	98.93/**96.12**
ALG2	92.23/72.63	97.79/68.70	98.46/80.75	98.93/93.35
SVM	89.12/68.46	97.83/**82.23**	98.41/81.44	98.92/71.28

Table 1: Table showing the exact results (AUC/ALC) obtained for the development data sets with algorithms ALG1 (the fast method without active learning), ALG2 (algorithm described in Section 3), and SVM (linear SVM using the distance of a point from the hyperplane as an informativeness measure). The best results are typeset in boldface.

where the width parameter σ specifies the distance below which the neighborhood relationship means *similarity*; it was set as the mean norm of the feature vectors in the data set (Chapelle et al., 2006). Furthermore, we set the following parameters of the algorithm: $c = 1000$, $d = 80$ (eigenvalues of the normalized Laplacian) and $\mathbf{C}$ was set to the identity matrix, that is no differentiation between the labeled points was made. These parameters were set based on the experimental results and conclusions from (Joachims, 2003).

For the NOVA and D data sets we performed principal component analysis (PCA) and used only the first $r = 50$ principal components; using this value we obtained the best results on NOVA. Since the training data is textual, before performing PCA we transformed

	A	**D**	**D**	**F**
			(with tf×idf)	
$\boxed{\textbf{ALG1}}$	84.96/4.19	*95.24/74.49*	96.41/86.10	96.27/38.07
	(ranked 22/22)		(ranked 1/22)	(ranked 15/16)
ALG2	84.52/-13.99	95.20/67.30	96.38/63.22	96.28/28.16
SVM	55.92/12.91	95.23/68.42	96.35/84.24	96.52/60.24

Table 2: Results obtained (AUC/ALC) for the challenge data sets using algorithms ALG1, ALG2 and SVM. We included an additional test using data set D without the tf×idf transformation (second column). The algorithm used in the challenge (ALG1) is indicated by the rectangular frame.

the feature vectors using the *tf×idf* transformation (term frequency × inverse document frequency) (Baeza-Yates and Ribeiro-Neto, 1999), but we also report results without applying this transformation. In these data sets we also normalized the vectors to unit length before the learning process.

When using the linear SVM for active learning and while the labeled set contains points from only one class – this includes the first step as well – we calculate the rank of the i-th data point as $1/(1 + \|\mathbf{z} - \mathbf{x}_i\|^2)$, where $\mathbf{z}$ is the mean of the labeled points. Other solutions for the one-class problem would be the application of the Gaussian similarity or one-class SVMs (Schölkopf et al., 2001). We used the above similarity measure since no additional parameter is involved in this way, and it provided good results on the development data.

The methods were implemented in MATLAB using the sample code provided by Isabelle Guyon for the challenge[2]. For SVMs we used LIBSVM (Chang and Lin, 2001) and performing fast k-means was accomplished using the package written by Charles Elkan (Elkan, 2003)[3].

The time complexity of the algorithm presented in Section 3 is $O(N \cdot i \cdot (k + k_{\mathrm{lab}} + n) + k^3 + p^3)$, assuming that $N \approx t$. In the formula k and k_{lab} denote the desired number of clusters as defined beforehand, n denotes the dimension of the data, i is the maximum iteration count in k-means and SVM (LIBSVM) instances, and $p = \max\{k_{\mathrm{lab}}, \theta\}$.

The methods were evaluated using two evaluation measures: a *local* and a *global* score. The local score shows the performance (Area Under the ROC Curve, AUC) of the method in the last step of the query process. The global score (Area under the Learning Curve, ALC) characterizes the method by integrating the local score over the queries. To obtain the final global score, ALC is normalized using the following formula:

$$\frac{\mathrm{ALC} - \mathrm{A}_{rand}}{\mathrm{A}_{max} - \mathrm{A}_{rand}}, \tag{11}$$

where A_{max} is the area under the ideal learning curve and A_{rand} is the area under the "lazy" learning curve, that is the learning curve obtained using random predictions. Figure

[2] http://www.causality.inf.ethz.ch/al_data/Sample_code.zip

[3] http://cseweb.ucsd.edu/∼elkan/fastkmeans.html

2 shows the results obtained for the development data sets. The graphs are obtained by plotting the AUC values in terms of the number of labeled points using a $\log_2$ scaling for the x-axis.

Tables 1 and 2 show the AUC/ALC results (in percentage) obtained for some of the development and final data sets, respectively. Based on the results on the development sets we have chosen to run ALG1 on the final data sets; although it is not an active learning algorithm, it is fast and performs sufficiently well on some of the data sets. However, since the algorithm is evaluated in two points only (i.e. with a single label and with all the labels), if the initial clustering does not fit well, a low global score is obtained. This happened in the case of data sets A and F: at the first step we obtained AUC/ALC scores of $19.22\%/-61.55\%$ and $41.80\%/-16.41\%$, respectively. The results obtained for the final data sets by ALG1 are evidently superior to the performances provided by ALG2; this can be explained by the fact that although the learning curves obtained for ALG2 have monotonically increasing tendency, at the beginning the increases are too small to beat ALG1. This can be caused by k-means, since for a smaller amount of data points the cluster centers are not sufficiently representative.

Other reasons of obtaining low scores can be the inadequate parameter settings. For example spectral clustering is very sensitive to the similarity graph: a suitable similarity function has to be chosen and its parameters have to be set carefully. Additionally, sparsification schemes can be considered for large data sets and better performance, which involves further important parameters.

Domain knowledge was also important in the challenge. For data set D – which shared similar characteristics with NOVA – we applied the tf×idf transformation for giving larger weights for some words based on their distribution in the corpus, used PCA to filter out noise and represent document vectors in a lower dimensional space, and finally normalized each vector to unit length. Applying these techniques used frequently in text categorization we achieved a performance improvement of almost 12% for ALG1. For ALG2 a lower global score is obtained since for 16 and 32 labeled points surprisingly low performances were recorded, in spite of the superior results in the remaining 12 evaluation points.

5. Future work

As a further research direction of this topic we plan to study other large-scale approaches for spectral clustering and SGT. We also plan to study how the application of the decision function in Eq. (4) influences the results of the KASP and RASP (Yan et al., 2009) algorithms. Another direction would be the application of kernel instead of "linear" k-means, however this introduces at least one new parameter. Finally, other graph construction methods are to be investigated, for example heuristics for computing the width parameter of the Gaussian similarity measure using the label information from the training data.

Acknowledgements

The authors acknowledge the partial support of the Romanian Ministry of Education and Research via grant PNII 11-039/2007.

We would also like to thank the reviewers for their work, for the comments, useful suggestions and supporting critiques.

References

Dana Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.

Les Atlas, David Cohn, Richard Ladner, M. A. El-Sharkawi, R. J. Marks, M. E. Aggoune, and D. C. Park. Training connectionist networks with queries and selective sampling. In *Advances in Neural Information Processing Systems (NIPS)*, pages 566–573, 1990.

Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley, 1999.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

Bernhard E. Boser, Isabelle Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. *Computational Learning Theory*, 5:144–152, 1992.

Steven P. Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, Cambridge, UK, 2004.

Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-Supervised Learning.* MIT Press, 2006.

Fan Chung. *Spectral Graph Theory.* American Mathematical Society, 1997.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273, 1995.

Charles Elkan. Using the triangle inequality to accelerate k-means. In Tom Fawcett and Nina Mishra, editors, *ICML*, pages 147–153. AAAI Press, 2003.

Thorsten Joachims. Transductive learning via spectral graph partitioning. In *ICML*, pages 290–297. AAAI Press, 2003.

David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International Conference on Research and Development in Information Retrieval*, pages 3–12, London, UK, July 1994. Springer Verlag.

Ali Rahimi and Ben Recht. Clustering with normalized cuts is clustering with a hyperplane. *Statistical Learning in Computer Vision*, 2004.

Nicholas Roy and Andrew McCallum. Toward optimal active learning through sampling estimation of error reduction. In *Proc. 18th International Conf. on Machine Learning*, pages 441–448. Morgan Kaufmann, San Francisco, CA, 2001.

Greg Schohn and David Cohn. Less is more: Active learning with support vector machines. In *Proc. 17th International Conf. on Machine Learning*, pages 839–846. Morgan Kaufmann, San Francisco, CA, 2000.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels.* The MIT Press, Cambridge, MA, 2002.

Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alexander J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

H. Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proc. 5th Annual ACM Workshop on Comput. Learning Theory*, pages 287–294, New York, NY, 1992. ACM Press.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, 1995.

Ulrike von Luxburg. A tutorial on spectral clustering. Technical Report 149, Max Planck Institute for Biological Cybernetics, August 2006.

Donghui Yan, Ling Huang, and Michael I. Jordan. Fast approximate spectral clustering. In *SIGKDD*, pages 907–916. ACM, 2009.