Greedy Multi-Class Label Propagation

Hubert Cecotti
School of Computing and Intelligent Systems
Ulster University
Londonderry, Northern Ireland, UK
email: h.cecotti@ulster.ac.uk

***Abstract*—In many real-world applications such as image classification, labeled training examples are difficult to obtain while unlabeled examples are readily available. In this context, semi-supervised learning methods take advantage of both labeled and unlabeled examples. In this paper, a greedy graph-based semi-supervised learning (GGSL) approach is proposed for multi-class classification problems. The labels are propagated through different graphs, which are obtained with neighborhoods of different sizes. The method assumes that nearby points share the same label, by starting with a small neighborhood where a reliable decision can be obtained, and iterates with larger neighborhoods where more examples are needed to determine the label of an example. The experimental results on toy data-sets and real data-sets, such as handwritten digit recognition, demonstrate the effectiveness of the proposed approach if a well chosen distance is used. Finally, the method does not require the tuning of hyper-parameters. We show that it is possible to achieve a recognition rate of 97.16% on handwritten digits (MNIST) while considering only one labeled example per class in the training data-set.**

I. Introduction

In many real-world applications that rely on the classification of objects, the creation of those classifiers can suffer from a lack of labeled data because labeling a large amount of data requires a lots of time [1]. Thanks to internet and the emergence of the Big Data paradigm, it is now possible to easily access large databases of documents (texts and images). However, only a small part of those databases are labeled, and could be useful for classifiers based on supervised learning. In supervised learning methods such as Support Vector Machine (SVM), Multi-Layer Perceptron, that predict labels, only the labeled examples are used. In unsupervised learning problems such as clustering, only unlabeled data are used. To take advantage of the current situation where a large amount of unlabeled data and a minority of data is labeled, semi-supervised learning (SSL) methods have been proposed to take advantage of both labeled and unlabeled data for the creation of efficient classifiers.

Those approaches are typically used when only few labeled examples are available, while there exists a large amount of unlabeled examples. Due the emergence of large databases that must be classified, it is unrealistic to label a large database manually, and classical supervised method may not grasp the complexity of the data with only a limited number of labeled examples. In addition, it is relatively inexpensive to acquire new data that may improve the performance of a classifier.

In semi-supervised learning, several approaches have been proposed [2]. Transductive SVMs optimize margins of both labeled and unlabeled examples [3], [4]. Some other approaches use the cluster assumption. In this case, the classifier takes into account decision boundaries through low-density regions in the input feature space. Most of the techniques in semi-supervised learning are graph-based techniques [5], [6], [7], [8]. Most of the semi-supervised learning approaches rely on two assumptions. The first one is the cluster assumption. It assumes that examples associated to the same cluster, or the same group of clusters, will share the same label. The second hypothesis is the manifold assumption, which considers that examples that are close to each other will have the same label. The label prediction of an example x will depend on both the labeled and unlabeled examples that are very close to x. Depending on the data, those assumptions can be difficult to obtain. It is particularly the case for problems where a linear classifier can be used (e.g., two Gaussian distributions) where a large overlap between classes can occur, i.e., the data density remains high close to the decision surface.

The remainder of the paper is organized as follows: First, we define the graph based semi-supervised learning approach. Then, we propose a new greedy semi-supervised learning method in Section III. The presentation of the performance on toy data-sets and real data-sets are given in Section IV. Finally, the performance of the proposed technique is discussed in Section V.

II. Graph based Semi-Supervised Learning

In semi-supervised learning, many methods rely on the geometry of the data (e.g., distance between data points) by using both labeled and unlabeled examples to improve classification accuracy on regular supervised techniques [9]. Among the techniques that have been proposed in the literature, a key family of the semi-supervised learning methods are based on graphs. They offer flexibility and are relatively easy to implement.

The same way that in supervised learning, a classifier will benefit from discriminant inputs that can allow a linear separation between classes, graph based semi-supervised technique will benefit from a graph that respects the manifold and cluster assumption. For instance, the k-nearest neighbor (k-nn) classifier has some drawbacks in term of complexity, and for the choice of the parameter. However, a bad performance with k-nn may result from a poorly chosen distance. If the Euclidean distance is applied in a problem where this

distance does not respect the manifold assumption, then it may be difficult to correct the errors that are due to the distance.

In those methods, a graph $g = (V, E)$ is defined by the nodes $V = \{1, \ldots, n\}$, which represent all the n examples of a training database $X = \{x_1, \ldots, x_n\}$, and edges E, which represent the similarities between examples. The similarities are typically represented by a weight matrix $W \in \mathbb{R}^{n \times n}$. A cell $W(i, j)$ corresponds to the similarity between the example x_i and x_j, i.e., the edge (i, j) in E. If x_i and x_j are close to each other (they belong to the same neighborhood), then $W(i, j)$ has a non-zero value.

A key research problem is the definition of W, which can be non-trivial for real-world problems with large multivariate data that include non-trivial underlying structures. W can be instantiated in different ways. W can correspond to the direct k-nearest neighbor matrix.

$$W(i,j) \quad = \quad \begin{cases} 1 & \text{iff } x_i \in N(x_j) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $N(x_j)$ represents the set of examples in the neighborhood of x_j. In this case, W is sensible to the the parameter k, and the weights that are used for each neighbor in $N(x_j)$.

Another way to initialize W is to consider a Gaussian kernel:

$$W(i,j) \quad = \quad e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}} \tag{2}$$

where σ is the width of the kernel. In this case, W is defined as a symmetric positive function ($\forall (i, j) \in \{1, \ldots, n\}^2$, $W(i, j) = W(j, i)$ and $W(i, j) > 0$). The Gaussian kernel is a fine distance in pairwise graph-based methods. However, there is no completely reliable method for determining the optimal value of σ. It has been shown that a small change of σ can significantly change the classification performance [10]. In addition, it may be better to set different σ for different dimensions and classes.

We define now $W_s \in \mathbb{R}^{n \times k}$ and $E_s \in \mathbb{N}^{n \times k}$. W_s contains for each example x_i, the sorted weights of the k neighbor examples of x_i. E_s contains for each example x_i, the associated list of the edges corresponding to the closest neighbors. We have $W_s(i, 1) = 0$ and $E_s(i, 1) = i$, because each point is its closest neighbor in the graph. This representation allows to store the data in $O(kn)$, which is significantly smaller than a storage in $O(n^2)$.

A type of technique for semi-supervised learning is to propagate labels on the graph g. For the binary classification, there are $M = 2$ classes, we consider the set of l labeled examples X_l, with their respective label Y_l. Each labeled example is defined by (x_i, y_i), with $1 \leq i \leq l$. If x_i belongs to the first class, $y_i = -1$; if x_i belongs to the second class, $y_i = 1$. We also consider the set of u unlabeled examples X_u, with predefined labels Y_u set to 0. In the graph, X_l and X_u correspond to the nodes 1 to l, and $l+1$ to n, respectively. Label propagation techniques are iterative algorithms.

III. Greedy Semi-Supervised Learning

In this section, we present a greedy semi-supervised learning method. The main advantages of this method come from its ability to process several classes simultaneously, and more importantly, from its storage requirement as it does not need to use a large matrix W, which can be a significant issue for large databases. Moreover, typical graph-based semi supervised learning methods deal with label propagation only in binary classification. The multi-class variation of those algorithms uses a one-versus-rest strategy. For instance, the labeled examples of one class are set to $y_i = 1$ if x_i belongs to class C_1, $y_i = 0$ otherwise, with $1 \leq i \leq l$. Contrary to this common approach, the proposed method allows to process several classes directly. In other graph-based methods, parameters such as the size of the neighborhood (k), and the σ, which is often set empirically, are pre-defined. Instead of having a single iteration step, the greedy approach has two iterative stages.

The method takes as inputs W_s, E_s, and the associated ground truth Y, where $y_i = m$ if x_i belongs to the class $m \in \{1..M\}$, and $y_i = -1$ if x_i is unlabeled (it is a default value). The initial value of k, k_{min}, corresponds to the initial size of the neighborhood that is considered, and this value is set to the minimum number of labeled examples that can be found in one of the M classes. At the beginning of the procedure, an unlabeled example can have at least k examples of its own class in his neighborhood. k_{max} is the maximum number of neighbors that may be considered to establish a decision. First, we iterate with a neighborhood of size k until convergence, i.e., until it is not possible to label any new unlabeled examples. If all the initially unlabeled examples are not labeled, then we increment the value of k, and we repeat the method until all the examples are labeled, or $k = k_{max}$.

At each iteration, for each unlabeled examples x_i, we apply the following procedure: we define a vector $\mathbf{v}$, such that $\mathbf{v}(m) = 0$, $\forall m \in \{1..M\}$. The values of $\mathbf{v}$ are updated for each neighbor j, $1 \leq j \leq k_{max}$, of x_i that follows these conditions: $j > 1$ (it is not x_i), and $j \leq 1 + k$ or $|W_s(i, j) - W_s(i, 1 + k)| \leq \epsilon$, and the neighbor $Es(i, j)$ is currently labeled $Y(Es(i, j)) > 0$. The second condition assures that the neighborhood has a size k, while allowing extra neighbors if they are as close as the neighbor at the position $1 + k$. If the conditions are fulfilled, then $\mathbf{v}(Y(Es(i, j)))$ is incremented. Finally, we determine the probability for x_i as:

$$\mathbf{p}_i \quad = \quad \frac{\mathbf{v}(i)}{\sum_{m=1}^{M} \mathbf{v}(m)} \tag{3}$$

$$(p_{max}, p_{arg}) \quad = \quad \max(\mathbf{p}) \tag{4}$$

where p_{max} represents the maximum value in $\mathbf{p}$, and p_{arg} represents the argmax value. If $p_{max} \geq \alpha$ then y_i takes the value of p_{arg}, the code corresponding the class with the maximum score. If all the elements of $\mathbf{v}$ are null, then $\mathbf{p}_i$ keeps the prior probabilities to belong to one of the M

classes. In the next sections, we set $\alpha = 1$. When $\alpha = 1$, all the neighbors of the test example share the same label, it corresponds to a consensus decision: all the examples in the neighborhood of the unlabeled example share the same label. If the method stops when $k = k_{max}$, i.e., it was not possible to assign a label to each example, then a k-nn classifier is used to determine the remaining missing labels. Because it is a greedy method, the order of the examples can have an impact on the results. Several runs can be performed in parallel, and then the final result will be obtained by combining the results from those different runs, e.g., by taking the average of the scores from the different runs.

An example is depicted in Figure 1. It presents the different steps of the method with different values of k. At $k = 1$ and $k = 2$, there is no propagation because there exist no example that has as a closest neighbor one of the two examples that is already labeled. Thus, the size of neighborhood has to be increased until it finds a labeled example. On toy data-sets, the proposed method achieves the same type of performance as other SSL techniques such as label propagation [10] while avoiding setting values to different parameters.

IV. Results

A. Toy datasets

The classification results on standard state-of-the-art toy examples (double half-moon, double circles, normal distributions) are depicted in Figures 2, 3, 4 and 5, which shows the ground truth, the results with a k-nn classifier, and the proposed method Greedy-SSL. g50c is an artificial dataset of 550 examples (50 labeled examples) that was generated from two unit-covariance normal distributions (50 features) with equal probabilities ($M = 2$). The class means were adjusted so that the true Bayes error is 5%. Coil20 is a dataset of 32×32 gray scale images of twenty objects ($M = 20$) viewed from varying angles. This data-set contains 1440 examples (40 labeled). Due to their contents, g50c is less likely to satisfy the condition to apply the proposed method, while coil20 is more likely to possess a manifold structure as it contains images. Performance is evaluated by accuracies averaged over ten random choices of the labeled set. Each random set contains one example per class for g50c, and two examples per class for coil20. The first evaluation deals with the validity of the assumptions given in the introduction. We perform k-nn (k from 1 to 5) for all the examples in a data-set to determine to what extent the closest neighbor x_j of an example x_i share the same label, i.e., $y_i = y_j$. Tables I and II present the accuracy for coil20 and g50c when the decision from k-nn is obtained through a majority decision (Table I), and a consensus decision (Table II). In Table II, each cell of the table represents a couple recognition rate; rejection rate. The recognition rate is calculated with the examples where a consensus decision could be made. For $k = 1$, the accuracy of 100% on coil20 suggests that propagating the labels is less likely to involve errors compared to 80.18% of g50c.

TABLE I
Recognition rate (in %) for g50c and coil20 with k-nn (majority decision across the neighbors).

data-set	1	2	3	4	5
coil20c	100.0	100.0	99.93	100	89.93
g50c	80.18	81.27	88.00	88.91	90.73

TABLE II
Recognition and rejection rate (in %) for g50c and coil20 with k-nn (consensus decision across the neighbors).

data-set	2	3	4	5
coil20c	100.0/0.49	100.0/2.99	100.0/7.15	100.0/12.15
g50c	93.33/29.09	95.83/43.27	96.64/51.27	96.54/58.00

The accuracy with the proposed method is $97.54 \pm 1.31\%$ for Coil20, and $63.52 \pm 7.38\%$ for g50c. The performance for g50c is bad, and significantly under other results presented in the literature. This low performance can be explained by the greedy approach that was used. In addition, all the assumptions were wrong as indicated by the low accuracy in Table I.

The comparison with other state-of-the-art methods are presented in Table III. Comparisons with other methods on the same databases can be obtained in [4]. The best results on coil20c are obtained with the proposed method GSSL. However, GSSL offers the worst results on g50c, as it was expected due to the non respect of the assumptions.

B. Handwritten recognition

In this section, we apply the proposed method on a database of handwritten digit characters (MNIST). This database is used for benchmarks in supervised classifiers [11], [12]. Typical state-of-the-art methods include deep learning architecture (e.g., convolutional neural networks [13], [12]), SVM [14], or their combinations [15], [16]. It includes a pre-defined training and test database of 60000 and 10000 images, respectively. In order to optimize the respect of the manifold assumption, we use an appropriate distance based on local deformations [17], with images filtered after horizontal and vertical edge detection with the Sobel operator. This distance provides good performance for handwritten digit recognition, and it can take advantage of GPU and High Performance Clusters. The algorithm of the Image Deformation Model Distance is given in Algo. 1. In the evaluation, we set $w_0 = 2$, $w_1 = 1$, $w_2 = 2$, and $p = 2$. Those parameters correspond to the size of the displacement field (w_0), the size of the neighborhood around a pixel (w_1), the number of filtered images (w_2), and the order of the Minkowski distance (p).

For the evaluation of the method, we create the graph with

TABLE III
Recognition rate (in %) for g50c and coil20 with state-of-the-art methods.

database	SVM	RLS	LapSVM	LapRLS	GSSL
coil20c	75.40	74.00	96.00	95.70	97.54
g50c	90.30	91.50	94.60	94.80	63.52

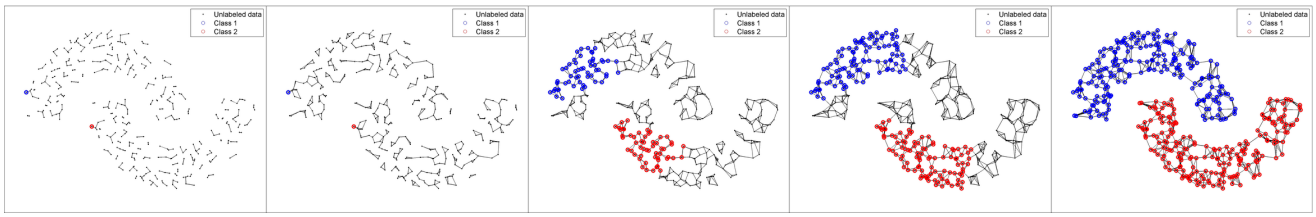


Fig. 1. Evolution of the labeling process in relation to different values of k, with the corresponding graph of the neighborhoods.

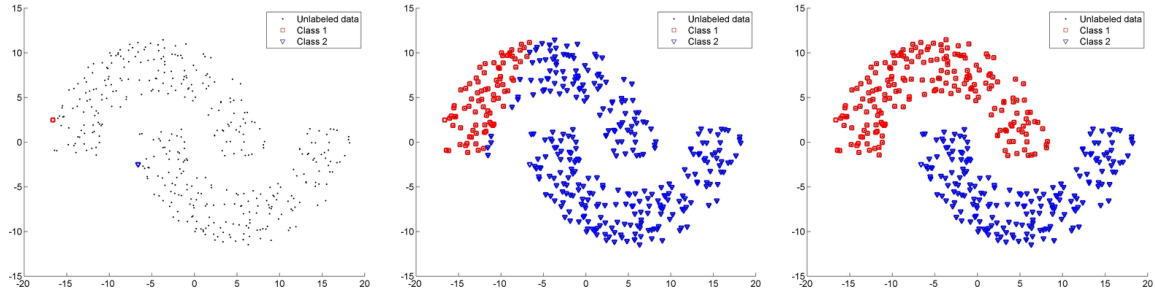


Fig. 2. Double moon.

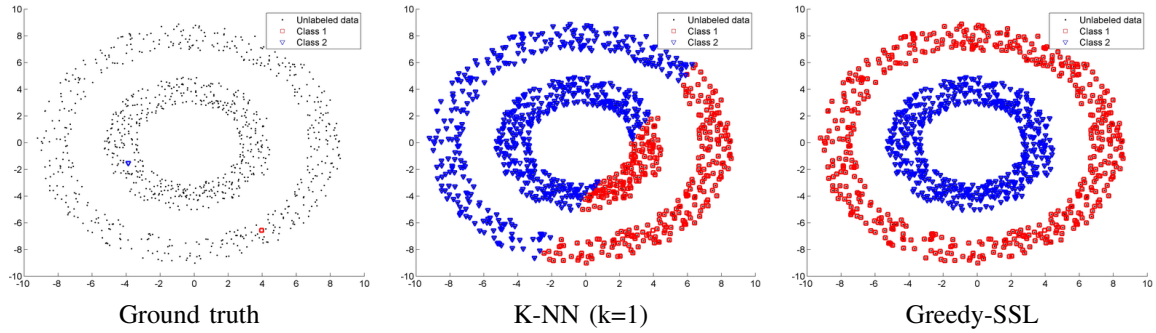
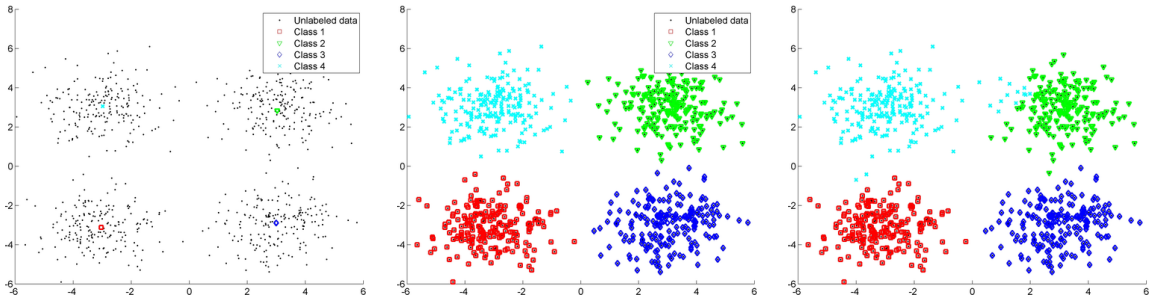


Fig. 3. Two circles.

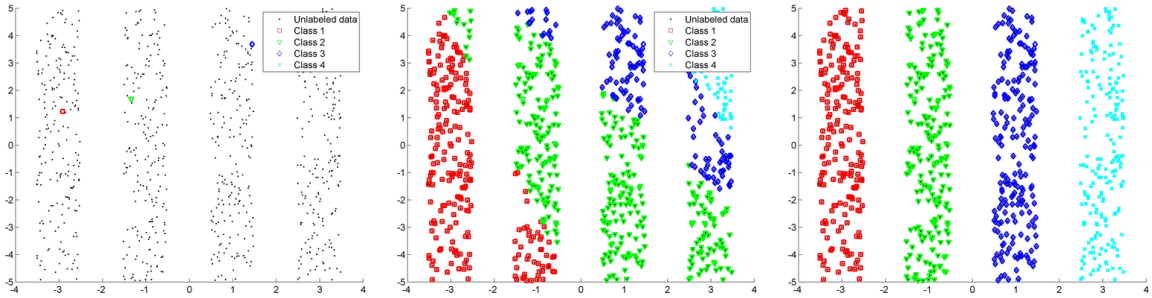
the whole training database. While the test could be included for transductive SSL, we only consider the training data-set in order to better compare the results with other results from the literature [18], [17], [11]. The original ground truth is chosen based on the nodes that correspond to the closest neighbor of the maximum number of nodes. If a node is often the closest neighbor of other nodes, then if it is labeled, it will propagate its label to a large number of nodes. By using this simple strategy, we sort the nodes by the descending number of times they appear as a first neighbor in a different node. We select the nodes that must be labeled in this list until the labeled examples contain a least one example of each class for the purpose of the experiment.

We have then two conditions. In the first one, we consider all the examples until we get at least one example per class. In the second condition, we only use the best examples from each class, i.e., the best seed for each class. In MNIST, we have 35 examples in the first condition, and 10 examples in the second, as there are 10 classes. The first condition corresponds to an active learning condition. While regular SSL deals with a pre-defined set of labeled examples, active learning is able to interactively query one or several oracles to obtain the label of some new examples. It can be a human expert of the data to be classified, or it can be simply the user. For the second condition, it is required to know the labels of the examples until it is possible to reach a number of 10 samples per class. With all the 60000 correctly labeled images from the training database, the accuracy on the test database is 99.36%. With only 10 labeled images in the training database, and after performing GSSL on only the training database, the accuracy on the test database is 97.16%. With the active learning condition, the accuracy reaches 97.35%. The evolution of the accuracy in the training database in relation to the number of labeled examples with GSSL, including the initial labeled examples, is depicted in Fig 6. With 10 labeled images, the accuracy on the training data-set is $96.81 \pm 0.06\%$, it is $97.07 \pm 0.07\%$ with 35 examples. The accuracy for the test, and for each class is presented in Table IV. It is worth noting the particular



Ground truth K-NN (k=1) Greedy-SSL

Fig. 4. Normal distributions.



Ground truth K-NN (k=1) Greedy-SSL

Fig. 5. Toy example.

low accuracy of the digit '7', with only about 86%, which suggests that there exists a large number of images that were used as bridges to mislabel the '7' images.

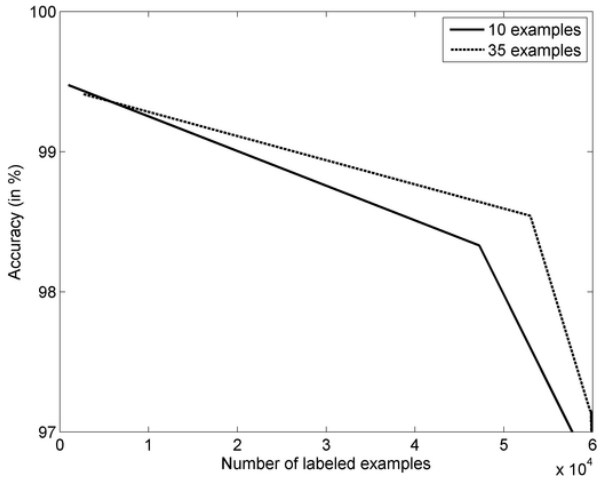


Fig. 6. Accuracy in relation of the number of examples in the training database with 10 and 35 seeds for GSSL.

V. Discussion and Conclusion

The graph is the heart of semi-supervised learning based on a graph. This graph is built in relation to a distance between examples in the data-set. In addition, other parameters, such as the neighborhood and its size, can be used to simplify the graph, and approximate the manifolds where the data lie, i.e., where the data density is supposed to concentrate.

Input: images I_1, I_2
Parameters: w_0, w_1, w_2, p
$d \leftarrow 0$
for $i_1 \leftarrow N_{min}, N_{max}$ **do**
  for $j_1 \leftarrow N_{min}, N_{max}$ **do**
    $i_5 \leftarrow 1$
    for $i_2 \leftarrow -w_0, w_0$ **do**
      for $j_2 \leftarrow -w_0, w_0$ **do**
        $s_2 \leftarrow 0$
        for $i_3 \leftarrow -w_1, w_1$ **do**
          for $j_3 \leftarrow -w_1, w_1$ **do**
            for $i_4 \leftarrow 1, w_2$ **do**
              $v_1 \leftarrow I_1(i_1 + i_3, j_1 + j_3, i_4)$
              $v_2 \leftarrow I_2(i_1 + i_3 + i_2, j_1 + j_3 + j_2, i_4)$
              $s_2 \leftarrow s_2 + |v_1 - v_2|^p$
            end for
          end for
        end for
        $s_1(i5) \leftarrow s_2$
        $i_5 \leftarrow i_5 + 1$
      end for
    end for
    $d \leftarrow d + min(s_1)$
  end for
end for
return d

Algorithm 1: Image Deformation Model Distance (IDMD) between I_1 and I_2.

TABLE IV
RECOGNITION RATE (IN %) ON THE TEST DATA-SET OF MNIST.

	# labeled examples		
class	10	35	60000
0	99.59	99.59	99.59
1	98.59	98.68	99.74
2	99.32	99.32	99.42
3	99.80	99.70	99.90
4	95.72	96.44	99.19
5	97.31	97.76	99.33
6	99.16	99.27	99.37
7	86.19	86.28	98.83
8	99.08	99.18	99.49
9	97.03	97.52	98.71
all	97.16	97.35	99.36

Those parameters, with the degree of the graph Laplacian, and the parameters of the distance, can be chosen with a cross-validation procedure on a grid of parameter values [19]. With a low number of labeled examples, it may be difficult to correctly estimate the right parameters.

For problems such as image classification, more adaptive distances shall be used such. For instance, other efficient distances that are mainly used for image retrieval could be used (e.g. structural similarity measures). We have shown that it is possible to use a greedy approach to perform semi-supervised learning when nearby points share the same label. Thanks to an approach that assigns step by step labels to the examples, this method does not propagate the labels in a single step, but the propagation is achieved through multiple iterations that increase the size of the graph. The method takes advantage of the manifold assumption that is critical because the cost of this assumption has a high impact on the performance of the classifier. It is worth mentioning that this approach shifts the machine learning problem from the SSL methods to the quality of the distances, and therefore the quality of the graphs. If the graph is well built, i.e., the assumptions are respected, then it is possible to use a greedy approach that does not require specific parameters.

With new technologies and with the Big data paradigm, it is possible to acquire a large number of examples. However, this large number of examples does not facilitate the efficiency of current SSL techniques because the large amount of data can correspond, or not, to the classification problem. It may be difficult to exploit the low density that can exist between classes, because the data space is very noisy due to the presence of examples that can be completely irrelevant to the problem. While we have shown that it is possible to achieve a good recognition rate with only few images on MNIST and few parameters, fast methods are required to both evaluate the graph, and to test the distances to new examples. Approximate k-nn strategies are therefore required to build large graphs [20]. In current databases, most of the examples are circumscribed to a single problem. For this reason, a challenge is to incorporate hierarchical views of the examples through adaptive distances or adaptive graphs. In addition, databases that are typically used such as MNIST do not correspond to what can be called Big-data. Despite 70000 images, MNIST lacks the number of examples, and the complexity of multimedia databases (e.g., the 80 Million Tiny Images [21]), which include different types of images. Further works will include clustering techniques to better select the labeled examples in the training data-set in an active learning setting.

REFERENCES

[1] W. Liu, J. Wang, and S.-F. Chang, "Robust and scalable graph-based semisupervised learning," in *Proc. of the IEEE*, vol. 100, no. 9, Sept. 2012.

[2] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," Carnegie Mellon University, Tech. Rep. Technical Report CMU-CALD-02-107, 2002.

[3] T. Joachims, "Transductive inference for text classification using support vector machines," in *Proc. the Int. Conf. on Machine Learning*, 1999, pp. 200–209.

[4] V. Sindhwani, P. Niyogi, and M. Belkin, "Beyond the point cloud: from transductive to semi-supervised learning," in *Proc. of the 22nd Int. Conf. on Machine Learning*, 2005, pp. 824–831.

[5] A. Blum and S. Chawla, "Learning from labeled and unlabeled data using graph mincuts," in *Proc. of the Int. Conf. on Machine Learning*, 2001, pp. 19–26.

[6] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: a geometric framework for learning from examples," *JMLR*, vol. 7, pp. 2399–2434, Dec. 2006.

[7] Y. Bengio, O. Dellalleau, and N. L. Roux, "Label propagation and quadratic criterion," in *Semi-supervised learning*, B. S. O. Chapelle and A. Zien, Eds. MIT Press, 2006, pp. 35–58.

[8] J. Wang, S.-F. Chang, X. Zhou, and S. T. C. Wong, "Active microscopic cellular image annotation by superposable graph transduction with imbalanced labels," in *Proc. of the Conf. Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.

[9] X. Zhu, "Semi-supervied learning literature survey," University of Wisconsin-Madison, Tech. Rep. Computer Sciences Technical Report 1530, 2005.

[10] F. Wang and C. Zhang, "Label propagation through linear neighborhoods," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 1, pp. 55–67, Jan. 2008.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[12] P. Simard, D. Steinkraus, and J. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *Proc. of the 7th Int. Conf. Document Analysis and Recognition (ICDAR)*, Aug. 2003, pp. 958–962.

[13] D. Cireşan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 3642–3649.

[14] D. DeCoste and B. Schölkopf, "Training invariant support vector machines," *Machine Learning*, vol. 46, no. 1-3, pp. 161–190, 2002.

[15] F. Lauer, C. Y. Suen, and G. Bloch, "A trainable feature extractor for handwritten digit recognition," *Pattern Recognition*, vol. 40, no. 6, pp. 1816–1824, 2007.

[16] X.-X. Niu and C. Y. Suen, "A novel hybrid cnn-svm classifier for recognizing handwritten digits," *Pattern Recognition*, vol. 45, no. 6, pp. 1318–1325, 2012.

[17] D. Keysers, T. Deselaers, C. Gollan, and H. Ney, "Deformation models for image recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 8, pp. 1422–1435, Aug. 2007.

[18] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, Apr. 2002.

[19] X. Zhang and W. S. Lee, "Hyperparameter learning for graph based semi-supervised learning algorithms," in *Advances in Neural Information Processing Systems (NIPS) 19*, 2006, pp. 1585–1592.

[20] M. Muja and D. G. Lowe, "Scalable nearest neighbor algorithms for high dimensional data," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2227–2240, Nov. 2014.

[21] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large dataset for non-parametric object and scene recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 1958–1970, Nov. 2008.