A supervisory approach to semi-supervised clustering

A SUPERVISORY APPROACH TO SEMI-SUPERVISED CLUSTERING

Bryan Conroy, Yongxin Taylor Xi, Peter Ramadge

Dept. of Electrical Engineering, Princeton University, Princeton, N.J.

ABSTRACT

We propose a new approach to semi-supervised clustering that utilizes boosting to simultaneously learn both a similarity measure and a clustering of the data from given instance-level must-link and cannot-link constraints. The approach is distinctive in that it uses a supervising feedback loop to gradually update the similarity while at the same time guiding an underlying unsupervised clustering algorithm. Our approach is grounded in the theory of boosting. We provide three examples of the clustering algorithm on real datasets.

Index Terms— Clustering methods, Algorithms, Pattern classification, Learning systems

1. INTRODUCTION

In many fields, the increasingly massive size of experimental datasets has required sophisticated machine learning algorithms to better organize and interpret the data. One important example is clustering [1], where data is partitioned into groups based on a selected pairwise similarity measure.

The importance of clustering is reflected in the variety of existing clustering algorithms. One of the most popular is K-means [2] and its exemplar-based variant K-medoids, and more recent algorithms include spectral clustering [3], and affinity propagation (AP) [4]. AP operates in a sequence of message passing rounds, each propagating soft information about cluster exemplars and eventually discovering and converging to a number of clusters. Spectral clustering represents the data as a graph with weights on the edges indicating the similarity of the nodes. It then uses a kernel derived from this graph to project the data to a lower-dimensional space in which similar points are aggregated. A final postprocessing step may be required to group the projected data into clusters.

Although clustering has customarily been completely unsupervised, recent studies [5] have explored semi-supervised clustering, where side information about cluster membership guides cluster formation. Typically, the side information is specified as a set of partial labels, or instance-level constraints in the form of must-link pairs and cannot-link pairs. Since partial labels can be converted to instance-level constraints, but not vice versa, we only consider the latter.

There are two main approaches in semi-supervised clustering [5]. In similarity-propagation methods, a preprocessing step adapts the similarity measure between data points to facilitate compliance with pairwise constraints. Then an existing unsupervised algorithm is used to cluster under the new similarity measure. In search-based methods the clustering algorithm is modified using the label information to bias the search for an appropriate clustering. A cost function can penalize constraint violations, as in constrained K-means [6], or require hard constraints to be satisfied during the clustering process. For example, [7] describes a search-based method that augments the data nodes in AP with 'meta-points' to enforce instance-level constraints.

The main contribution of this paper is Boost Base Clustering (BBC), a new similarity-propagation approach for semi-supervised clustering. Our approach designs a "supervisor" for any unsupervised clustering algorithm that takes a pairwise data similarity measure as input and returns a clustering. The supervisor uses the instance-level constraints to modify a global similarity measure such that must-link datapoints are made more similar and cannot-link datapoints more dissimilar. The method is distinctive in that it does not modify the similarity measure once at the beginning; instead it uses feedback to continually supervise similarity learning throughout the iterations of the underlying unsupervised clustering algorithm. This is done via the paradigm of boosting [8].

The paper is organized as follows. We introduce and discuss the supervised clustering algorithm in §2. The performance of the algorithm is examined experimentally in §3 and conclusions drawn in §4.

2. APPROACH AND ALGORITHM

We have N data points $\{x_i \in R^p\}_{i=1}^N$, and a set of instance-level constraints: $y_{ij} = 1$ for $(i,j) \in M$ (must-link pairs) and $y_{ij} = -1$ for $(i,j) \in C$ (cannot-link pairs). We assume $M \cap C = \emptyset$ and $(i,j) \in M$ (resp. C) implies $(j,i) \in M$ (resp. C). The goal is to learn a clustering of the data satisfying the given constraints that also effectively propagates the label information to unlabeled examples. Essentially, we have a classification problem on data pairs with the instance-level constraints playing the role of training examples. Let $m = |M| + |C|$ denote the number of training examples. A pairwise classifier is a matrix $U \in \mathbb{R}^{N \times N}$ with entries in the set $\{\pm 1\}$. If $U_{ij} = 1$, then x_i, x_j are in the same cluster; otherwise they are not. We are told the values of U_{ij} on the training set

$(i,j) \in M \cup C$ and we must learn the remaining $(n(n-1)-m)/2$ entries. However, to be a clustering we require U to be transitive: $U_{ij}=1$ and $U_{jk}=1$ implies then $U_{ik}=1$.

Our approach to the problem uses "supervising" feedback to direct an unsupervised clustering algorithm. We call the unsupervised clustering algorithm the base clusterer . The base clusterer takes as input a pairwise data similarity matrix S and produces a pairwise classification matrix Y. Thus, one can think of the base clusterer as a family of base classifiers, parameterized by the input similarity matrix S. Let $w = \{w_{ij} \geq 0, (i,j) \in M \cup C, \sum_{(i,j)\in M\cup C} w_{ij} = 1\}$ be a distribution on $M \cup C$; w_{ij} reflects the relative importance of correctly classifying training pair (i,j). Our intent is to call the base clusterer in a sequence of rounds. At $t=1$ we start with similarity matrix $S(1)$ and weights $w(1)$; the base clusterer yields the pairwise classifier $Y(1)$. Based on the classification errors of the training data, we then update the weights to $w(2)$ and the similarity matrix to $S(2)$, and so on. Note that we use $w(t+1)$ to guide the selection of the next similarity measure $S(t+1)$, thus closing the loop on a feedback mechanism that supervises the clustering.

We now fill in the details of the approach. The merit of any pairwise classification Y w.r.t. weights w is quantified by the weighted error:

$$\varepsilon = \textstyle\sum_{(i,j)\in M\cup C} w_{ij} \mathbb{1}[y_{ij} \neq Y_{ij}] \tag{1}$$

The weights can then be updated, as in AdaBoost [8], with constraints given more or less weight according to how (incorrectly or correctly) they are classified:

$$w_{ij}(t+1) = w_{ij}(t) e^{-\alpha(t) y_{ij} Y_{ij}(t)} / Z(t) \tag{2}$$

where $\alpha(t) = (1/2)\ln[(1-\varepsilon(t))/\varepsilon(t)]$ reflects the confidence of the clustering $Y(t)$ under weights $w(t)$ and the normalization factor $Z(t)$ ensures $\sum w_{ij}(t+1) = 1$.

The similarity of x_i, x_j is given by the negative Mahalonobis distance:

$$S_{ij}(A) = -(x_i - x_j)^T A (x_i - x_j) \tag{3}$$

where $A \in \mathcal{S}^p_+$, the cone of real $p \times p$ PSD matrices. This measure has been used extensively in the clustering literature for identifying the linear transformation $x \rightarrow A^{(1/2)}x$ that best separates the clusters (see, e.g., [9]). The similarity is flexible and easy to optimize, but assumes that the clusters are linearly separable in the ambient space. We require A to be selected based on the weights w. To this end, consider the objective function:

$$\begin{aligned} F(A) &= \textstyle\sum_{(i,j)\in M\cup C} w_{ij} y_{ij} S_{ij}(A) & (4)\\ &= \textstyle\sum_{(i,j)\in M} w_{ij} S_{ij}(A) - \sum_{(i,j)\in C} w_{ij} S_{ij}(A) \end{aligned}$$

By (2), violated constraints have larger weights and by (4), $F(A)$ increases if the similarity of misclassified pairs in M increases or the similarity of misclassified pairs in C decreases.

Algorithm 1 Semi-supervised clustering algorithm

```
Given data $X = [x_1, \ldots, x_N]$ and constraints $y_{ij} = 1, (i,j) \in M$, $y_{ij} = -1, (i,j) \in C$
$A(1) = I_{p\times p}$
$w_{ij}(1) = \frac{1}{|M|+|C|}, \forall (i,j) \in M \cup C$
$Y = \mathbf{0}_{N\times N}$
for $t = 1$ to $T$ do
    $\varepsilon(t) = 0.5$
    while $\varepsilon(t) \geq 0.5$ do
        $A(t) = P(A(t) + \gamma G)$
        $S_{ij}(t) = -(x_i - x_j)^T A(t)(x_i - x_j)$
        Run base clusterer with $S(t)$ to obtain $Y(t)$
        $\varepsilon(t) = \sum_{(i,j)\in M\cup C} w_{ij}(t) \mathbb{1}[y_{ij} \neq Y_{ij}(t)]$
    end while
    $\alpha(t) = \frac{1}{2}\ln[(1-\varepsilon(t))/\varepsilon(t)]$
    $Z(t) = \sum_{(i,j)\in M\cup C} w_{ij}(t) e^{-\alpha(t) y_{ij} Y_{ij}(t)}$
    $w_{ij}(t+1) = w_{ij}(t) e^{-\alpha(t) y_{ij} Y_{ij}(t)} / Z(t)$
    $Y = Y + \alpha(t) Y(t)$
end for
```

Fig. 1*:* The BBC Algorithm

Hence, we update A using projected gradient ascent of $F(A)$. This is similar to, but distinct from, the distance metric learning in [9]. Let $G(A) \in R^{p\times p}$ be the gradient of F with respect to A and set

$$A(t+1) = P\left(A(t) + \gamma G(A(t))\right)$$

where $\gamma > 0$ and $P(A)$ projects A onto $\mathcal{S}^p_+$. The projection is found via an eigen-decomposition:

Lemma 1. *Let $B \in \mathcal{S}^p$ have eigen-decomposition $B = \sum_{i=1}^p \lambda_i v_i v_i^T$. Then the orthogonal projection of B onto $\mathcal{S}^p_+$ is $P(B) = \sum_{i=1}^p \max(\lambda_i, 0) v_i v_i^T$.*

Proof. Write $B = P + N$ with $P = \sum_{i=1}^p \max(\lambda_i, 0) v_i v_i^T$ PSD and $N = \sum_{i=1}^p \min(\lambda_i, 0) v_i v_i^T$ NSD. Note $\langle N, P\rangle = 0$. P is the unique minimum distance projection of B onto $\mathcal{S}^p_+$ if and only if $\forall K \in \mathcal{S}^p_+$, $\langle B-P, K-P\rangle \leq 0$ [10]. We have $\langle B-P, K-P\rangle = \langle N, K\rangle - \langle N, P\rangle = \langle N, K\rangle \leq 0$, since K is PSD and N is NSD. □

The parameter γ controls the rate of increase of $F(A)$. We want to increase $F(A)$ just enough to ensure that A better matches the constraints but we don't want to be too greedy. As in Adaboost, the base clusterer only needs to be better than chance on the training data, i.e., $\varepsilon(t) < 0.5$. Thus, if $\varepsilon(t) \geq 0.5$, we further adjust the similarity by another update $P(A(t) + \gamma G(A(t)))$, and re-do the previous step. This is repeated as necessary until $\varepsilon(t) < 0.5$.

The final pairwise classification is given by the composite classifier $H = \text{sgn}(Y)$, where $Y = \sum_{t=1}^T \alpha(t) Y(t)$ aggregates the pairwise classifiers over the T rounds. The algorithm is summarized in Fig. 1. After T rounds, the composite classifier H produces a classification for each pair of datapoints. However, for H to represent a clustering, it must additionally satisfy transitivity. Thus, additional care must be taken to derive a hard clustering from H or Y.

By the results of boosting theory [11], classification error on the instance-level constraints decays exponentially with t.

Lemma 2. *If $\varepsilon(t) < 0.5$ for $t = 1, \ldots, T$, then the composite classifier H produces at most $\exp(-2\sum_{t=1}^{T}(e(t))^2)$ percentage of errors on the instance-level constraints, where $e(t) = \frac{1}{2} - \varepsilon(t)$.*

Proof. For a binary classification problem, the training error rate is upper-bounded by $\prod_t 2\sqrt{\varepsilon(t)(1-\varepsilon(t))}$, which can be further bounded by $\exp(-2\sum_{t=1}^{T}(e(t))^2)$ [11]. □

The generalization error, i.e., the error rate on $(i,j) \notin M \cup C$, is more complex to quantify. We note that Adaboost is known (empirically) to increase the generalization accuracy even after the training error reaches 0. This has been explained in terms of margin theory [12]. The convergence of the proposed algorithm is closely related with the convergence of Adaboost. In general, Adaboost need not converge, although in practice it almost always does [13].

We examine two schemes for ensuring transitivity. The first computes the connected components of the graph with adjacency matrix $H^+ = [\max(H_{ij}, 0)]$, which we refer to as BBC (CC). This is simple and efficient, involving a depth-first search of the graph but it does not allow a choice of K.

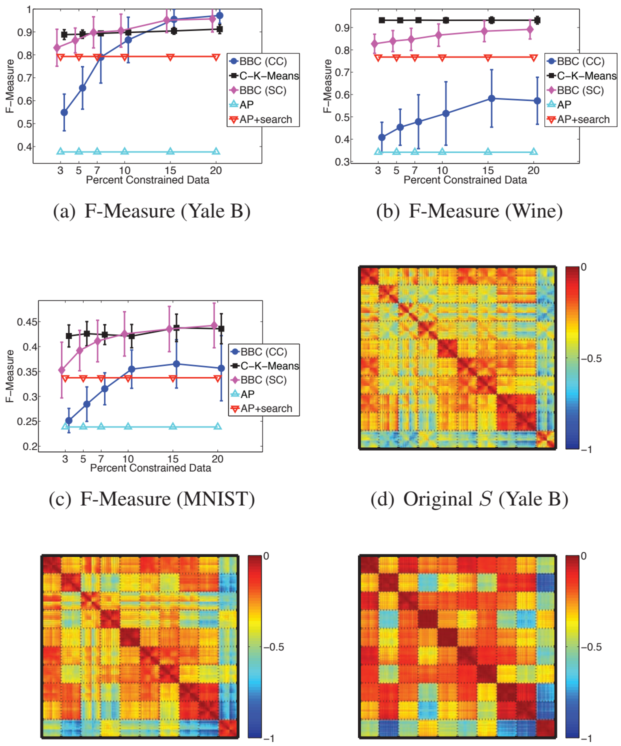
Alternatively, in the spirit of spectral methods, e.g. [3], a clustering can be computed by analyzing the spectrum of the Laplacian of a weighted undirected graph derived from Y. This offers greater flexibility in specifying K. Let $W = Y + |\min(Y)|\mathbf{1}\mathbf{1}^\mathbf{T}$ denote the weighted adjacency matrix of an undirected graph defined on the data. The edge between x_i and x_j is weighted by a translated version of Y_{ij}. This is a confidence measure of classification: a larger W_{ij} indicates higher confidence that x_i and x_j are in the same cluster. Let D be a diagonal matrix with entries given by row sums of W. The Laplacian of the graph is $L = D - W$. It is well known from spectral graph theory [14] that the number of connected graph components K is the multiplicity of the 0 eigenvalue of L. Moreover, the corresponding K-dimensional eigenspace is spanned by the set of vectors $v_1, \ldots, v_K$, where v_i denotes the cluster membership of the i^{th} cluster: $v_i(j) = 1$ if x_j belongs to cluster i, and 0 otherwise. Thus, given any basis $Q = [q_1, \ldots, q_K]$ for the 0-eigenspace, the cluster membership vectors can be determined from an appropriately chosen rotation matrix R, with $V = QR$. The approach of [3] can recover the v_i by iteratively rotating the given basis Q. The clustering is then trivially determined from the resultant v_i. This can be extended to a connected graph, as is sometimes the case with the graph defined by W. Here, the basis $q_1, \ldots, q_K$ is comprised of the K eigenvectors with smallest eigenvalues. The cluster label l_j for data j is then $l_j = \arg\max_i v_i(j)$. We refer to this scheme as BBC (SC).

3. EXPERIMENTAL RESULTS

For brevity we only present results using AP [4] as the base clusterer . This allows us to focus on the interesting case when the number of clusters is not specified a-priori. We first test our algorithm by clustering a subset of the Yale Face Database B [15] (10 subjects, 65 640×480 grayscale images/subject, frontal pose with varying illumination). This is an interesting test, since Euclidean distance is at best an approximate feature similarity measure under illumination and background variations. Each image was standardized by: (1) subtracting the mean of all images, (2) normalizing to unit norm, (3) projecting to the top 100 principal components. Step (3) reduces dimensionality by a factor of 3000, while preserving 97% of the total variance. Thus, $p = 100$, and A in (3) is 100×100.

For AP, the diagonal entries of A give the prior preferences for each datapoint being a cluster exemplar; these values directly influence the number of clusters identified. Search strategies exist to tune the preference for a prescribed number of clusters K. However, without prior knowledge of K, the preference is typically specified as the median value of all similarities [4]. We took the latter approach and reset the preferences to the median similarity after each similarity update. The supervision was provided by a set of partial labels (3% - 20%) from which all implied instance-level constraints were derived. For each label percentage, 500 random labeling instantiations were generated and results averaged over these runs. We evaluated each of the clusterings using the F-measure $F = 2PR/(P+R)$, where P and R are precision and recall, respectively. Given a binary matrix $U \in \mathbb{R}^{N \times N}$, a positive result $U_{ij} = 1$ corresponds to x_i and x_j being clustered. With this understanding, $P = tp/(tp+fp)$ and $R = tp/(tp+fn)$, where tp,fp, and fn denote true-positive, false-positive, and false-negative rates, respectively.

A plot of the F-measure is given in Figure 2(a). Here, BBC(CC) uses connected components postprocessing and AP is unsupervised clustering. BBC(SC) uses spectral postprocessing using knowledge of K. Also included are the results of the semi-supervised constrained K-means algorithm (C-K-Means) [6]. AP+search is AP using a search strategy for the preference parameter to obtain K clusters. Note that BBC(CC) should be compared with AP, and BBC(SC) with AP+search and C-K-Means. To remove bias due to the partial supervision, labeled datapoints were excluded from the analysis of all semi-supervised algorithms, while results for the unsupervised AP algorithms include all datapoints. Thus, the F-measures for BBC(CC) and BBC(SC) directly show the degree to which labeled information is effectively propagated to unlabeled examples. Additionally, as label percentage increases, BBC(CC) better estimates the true number ($K = 10$) of clusters ($\hat{K} = 31, 25, 20, 16, 12$, and 11, for 3%,5%,7%,10%,15%, and 20% partial labels, respectively). Note that without prior information, the base clustering algorithm AP returns 48 clusters on this dataset.



(a) F-Measure (Yale B)

(b) F-Measure (Wine)

(c) F-Measure (MNIST)

(d) Original S (Yale B)

(e) Boosted S, 3% labels (Yale B)

(f) Boosted S, 5% labels (Yale B)

Fig. 2*:* Results of algorithm on Yale B, MNIST, and UCI Wine.

Figure 2 contrasts the original pairwise similarity matrix (with $A = I$) on the Yale B face data (Figure 2(d)), with the final similarity measure from the proposed algorithm for 3% and 5%, label information (Figures 2(e) and 2(f), respectively). Here, the data are sorted by class label. The result for 3% is more block-diagonally dominant than the original similarity, and is progressively more-so with more label information. The ratio of average inter-class distance to average intra-class distance increases from 2.55 (original) to 4.4 for 3% and 13.2 for 5% label information.

Results of the BBC(CC) and BBC(SC) algorithms on the wine dataset ($K = 3$) and the MNIST handwritten digits dataset ($K = 10$) are shown in Figures 2(b) and 2(c), respectively. Results are averaged over 50 random labeling instantiations for each label percentage. On the wine dataset, BBC(CC) returns $\hat{K} = 12, 10, 9, 7, 5$, and 4 clusters over the label percentage range, while unsupervised AP returns 15 clusters. On the MNIST dataset, BBC(CC) returns $\hat{K} = 46, 40, 34, 27, 24$, and 21 clusters, while unsupervised AP returns 52 clusters,

4. CONCLUSION

The proposed instance constraint algorithm showed improved performance on three tested datasets. The face dataset demonstrated significant improvement in clustering performance from limited instance-level constraints. The algorithm also gave better estimates of the true number of classes as the percentage of partial labels increased. The wine dataset and MNIST datasets also showed improved performance results, commensurate with constrained k-means. One feature of the algorithm that remains to be explored is that it can be built around a variety of unsupervised clustering algorithms. Here we examined its use only with affinity propagation.

5. REFERENCES

[1] A.K. Jain, M.N. Murty, and P.J. Flynn, “Data clustering: a review,” *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[2] J. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Proc. Fifth Berkeley Symp. on Math. Statist. and Prob.*, 1967.

[3] L. Zelnik-Manor and P. Perona, “Self-tuning spectral clustering,” in *NIPS 17*. 2004, pp. 1601–1608, MIT Press.

[4] B.J. Frey and D. Dueck, “Clustering by passing messages between data points,” *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[5] N. Grira, M. Crucianu, and N. Boujemaa, “Unsupervised and semisupervised clustering: a brief survey,” in *A Review of Machine Learning Techniques for Processing Multimedia Content, MUSCLE European Network of Excellence (FP6)*, 2004.

[6] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl, ,” *International Conference on Machine Learning*, 2001.

[7] I. Givoni and B. Frey, “Semi-supervised affinity propagation with instance-level constraints,” in *12th Artifical Intelligence and Statistics*, 2009.

[8] Y. Freund and R.E. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting,” in *European Conf. on Comput. Learning Theory*, 1995, pp. 23–37.

[9] E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell, “Distance metric learning with application to clustering with side-information,” in *NIPS 15*. 2002, MIT Press.

[10] D.G. Luenberger, *Optimization by vector space methods*, Wiley, 1969.

[11] Y. Freund and R.E. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting,” *J. Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[12] R.E. Schapire, Y. Freund, P. Bartlett, and W.S. Lee, “Boosting the margin: A new explanation for the effectiveness of voting methods,” *The Annals of Statistics*, 1998.

[13] C. Rudin, I. Daubechies, and R.E. Schapire, “The dynamics of adaboost: Cyclic behavior and convergence of margins,” *J. Mach. Learn. Res.*, vol. 5, pp. 1557–1595, 2004.

[14] C. Godsil and G. Royle, *Algebraic Graph Theory*, Springer, 2001.

[15] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman, “From few to many: Illumination cone models for face recognition under variable lighting and pose,” *IEEE Trans. Pattern Anal. Mach. Intelligence*, vol. 23, no. 6, pp. 643–660, 2001.