2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology

Clustering with Extended Constraints by Co-Training

Masayuki Okabe
Information and Media Center
Toyohashi University of Technology
Email: okabe@imc.tut.ac.jp

Seiji Yamada
National Institute of Informatics
Email: seiji@nii.ac.jp

***Abstract*—Constrained Clustering is a datamining technique that produces clusters of similar data by using pre-given constraints about data pairs. If we consider using constrained clustering for some practical interactive systems such as information retrieval or recommendation systems, the cost of constraint preparation will be the problem as well as other machine learning techniques. In this paper, we propose a method to complement the lack of constraints by using co-training framework, which extends training examples by leveraging two kinds of feature sets. Our method is based on a constrained clustering ensemble algorithm that integrates a set of clusters produced by a constrained k-means with random ordered data assignment, and runs the same algorithm on two different feature sets to extend constraints. We evaluate our method on a Web page dataset that provides two different feature sets. The results show that our method achieves the performance improvement by using co-training approach.**

I. Introduction

Constrained clustering [1] is a kind of semi-supervised learning technique [2] that utilizes labeled and unlabeled data to enhance learning performance. Difference from normal clustering is the use of background knowledge, which is given in the form of constraints about data pairs. Such constraints have two kinds, usually called *must-link* and *cannot-link* constraints. The former is constraints about data pairs that must be in a same cluster, while the latter is ones about data pairs that must be in different clusters. The challenge of constrained clustering is to develop utilization methods of such constraints. Some researches proposed to use them in the k-means algorithm as knowledge for assigning data to cluster centers, and some others proposed to use them as constraints for an optimization problem of distance metric learning or learning kernel matrix. [3], [4], [5].

Although the use of constraints is an effective approach, we have a problem in preparing constraints. Since constraints must be labeled as "must-link" or "cannot-link" manually by human, his/her cognitive cost seems very high. We need support to help users cut down such an operation cost. This problem is particularly serious when we consider to use constrained clustering in some practical interactive systems, such as information retrieval or recommendation systems.

Thus we propose a method to complement the lack of constraints by using co-training framework [6], which extends training example by leveraging two kinds of feature sets. In co-training process, two learners are repeatedly trained and are used to predict and select some unlabeled data to be added as pseudo training examples. In our case, we use co-training to increase the amount of constraints by repeating constrained clustering on two different feature sets.

Our method is based on a constrained clustering ensemble algorithm that integrates a set of clusters produced by a constrained k-means with random ordered data assignment. This is a realization of cluster ensemble framework that exploits partially coherent data group from clustering iteration and integrates them into a set of final clusters. Cluster variation can be created by changing data assignment order in a constrained k-means algorithm that is a modified version of COP-Kmeans proposed by Wagstaff [7]. This ensemble technique produces a kernel matrix for the final clustering. Since each element of the kernel matrix represents the confidence for a data pair to be in a same cluster or not, we use it to select pseudo constraints.

While we need to prepare two different feature sets for a dataset when we apply co-training, we can automatically extend the amount of constraints without any manual effort that consists of human operation such as data pair selection or judgement for the constraint label assignment.

This is a big advantage if we consider reducing the cost for labeling constraints in some interactive systems. We test our proposed method on a task of Web page clustering, which is one of well known task to support information retrieval from Web search engines. Web page is one of suitable data types for applying co-training because it contains various media such as texts, images, videos, hyperlinks and so on. Those media can be feature sets for co-training. We can combine various media to create two different feature sets for co-training. In our experiments, we used the WebKB Co-Training dataset that provide a set of Web pages from two categories and two feature sets to represent such Web pages. The two feature sets are actually two bags of words extracted from texts on each Web pages and anchor-texts on the hyperlinks pointing to the page. Since bags of words used in the two feature sets do not share common words so much, they will be applicable to co-training.

The rest of this paper is organized as follows. We first introduce a constrained clustering method based on a bagging

978-0-7695-4880-7/12 $26.00 © 2012 IEEE
DOI 10.1109/WI-IAT.2012.113

Algorithm 1 Ensemble of Constrained K-means

INPUT: Dataset $X = \{x_1, ..., x_{|X|}\}$,
Constraints S= $\{(i_1, j_1, y_1), ..., (i_{|S|}, j_{|S|}, y_{|S|})\}$,
k: No. of clusters
OUTPUT: Clusters $C = \{C_1, C_2, ..., C_k\}$

for $t = 1$ to T **do**
  Run constrained k-means procedure(**Algorithm2**). Then, create kernel matrix K^t

$$K^t(i,j) = \begin{cases} 1 & (x_i, x_j) \text{ belongs to same cluster} \\ -1 & (x_i, x_j) \text{ belongs to different clusters} \end{cases}$$

end for

Calculate final kernel matrix K

$$K = \sum_{t=1}^{T} K^t \tag{1}$$

Run kernel k-means algorithm with K, and return final set of clusters C

framework in Section II. Then we apply co-training framework for our constrained clustering algorithm in Section III. Section IV presents the results of the experiments conducted using the WebKB dataset. We finally conclude our work in Section V.

II. Clustering Algorithm

We first propose a constrained clustering algorithm that is based on a bagging based cluster ensemble technique and a constrained k-means with random data assignment order.

Bagging [8] is one of the ensemble learning techniques used to produce a classifier by integrating weak hypotheses generated by a weak learner that has outperforms random classifiers to some extent. Algorithm 1 is our clustering algorithm based on bagging. According to the normal description of the bagging, we can correspond each element of our algorithm as a

- Weak Learner → Constrained K-means((Argorithm2))
- Training data → Constraints (must/cannot-link).

A constrained cluster produced by the constrained k-means in each bagging step is used as a kernel matrix. Each element of this kernel matrix indicates whether or not the corresponding data pair belongs to the same cluster. Thus, the kernel matrix represents the link connections of the clusters. From the point of a weak learner, the modified COP-Kmeans predicts the existence of the link between any data pair in the clusters using the constraints as a set of training data. The kernel matrix is an aggregation of these predictions. The kernel matrices are summed up as a kernel. The final clustering result is generated using this final kernel matrix.

Algorithm 2 Constrained K-means

INPUT: Dataset X, Constraint Set S, No. of clusters k,
  Weights of Constraints $w_n (n = 1 \sim |S|)$
OUTPUT: Clusters C
Select initial cluster centers
for $r = 1$ to r_{max} **do**
  Assign a random value to each w_n
  Sort constraints in descending order according to w_n^t
  Assign constrained data pairs (x_i, x_j) to cluster centers in sorted order following procedure below
  Let (x_i, x_j) be data pair to be assigned, then each data will be assigned to one cluster centers according to following cases.
  if Both of (x_i, x_j) have not been yet assigned **then**
    Let c_i, c_j be nearest cluster centers for x_i and x_j respectively, then let $d(x_i, c_i)$, $d(x_j, c_j)$ be distances between each data and its nearest cluster center.
    if (x_i, x_j) is constrained by must-link **then**
      if $d(x_i, c_i) < d(x_j, c_j)$ **then**
        Assign x_i and x_j to c_i
      else
        Assign them to c_j
      end if
    else if (x_i, x_j) is constrained by cannot-link **then**
      if $c_i \neq c_j$ **then**
        Assign x_i to c_i, x_j to c_j, respectively
      else
        if $d(x_i, c_i) < d(x_j, c_j)$ **then**
          Assign x_i to c_i, x_j to second nearest center
        else
          Assign x_j to c_j, x_i to second nearest center
        end if
      end if
    end if
  else if x_i has been already assigned and x_j has not been yet assigned **then**
    Let c_i be cluster center to which x_i is assigned
    if x_i and x_j are constrained by must-link **then**
      Assign x_j to c_i
    else if x_i and x_j are constrained by cannot-link **then**
      Assign x_j to cluster center that is nearest to data and is different from c_i
    end if
  else if x_i has not been yet assigned and x_j has already been assigned **then**
    Let c_j be cluster center to which x_j is assigned
    if x_i and x_j are constrained by must-link **then**
      Assign x_i to c_j
    else if x_i and x_j are constrained by cannot-link **then**
      Assign x_i to cluster center that is nearest to data and is different from c_j
    end if
  end if
  Assign rest of data that are not constrained to their nearest cluster centers
  if Clustering result does not change from previous one **then**
    Return result and exit
  else
    Update cluster centers and go to next step
  end if
end for

Algorithm 2 lays out the entire procedure of our constrained k-means. Although it follows the basic procedure of the standard k-means algorithm, which assigns data to its nearest cluster center, its assignment process is rather complicated since we must consider the weight of constraint. There are mainly two parts to the assignment processes, which consists of the procedure for the constrained and unconstrained data, respectively. The latter one (for the unconstrained data) remains the same as that in a normal k-means process. What we need to consider is the process for the previous one (for constrained data). We must take several cases like those listed below into consideration.

- Whether or not one of the data pairs has already been assigned?
  Our algorithm assigns a constrained data pair at the same time. Since some data contain several constraints, one (or both) of the data may have already been assigned in some cases. We must prepare procedures for such situations.
- Which constraint the data pair has? - must-link or cannot-link?
  Depending on the situation described above, we must prepare different procedures according to which constraint the data pair has.

We describe the concrete procedure considering the above cases in Algorithm 2 (l.9 $\sim$ 42).

III. Extending Constraints by Co-Training

Based on the constrained clustering algorithm introduced in previous section, we apply co-training framework to extend the amount of constraints. Co-training is first introduced to classify Web pages with a small set of training examples. The main characteristic of this method is to utilize two kinds of feature sets prepared for one dataset. Though it can be applied to any dataset, the Web page is one of suitable data for co-training since the Web page has various representation methods using different types of media such as not only texts, images, audio and videos.

A classifier is learned on each feature set respectively. Thus we have two different classifiers for the same dataset. They are used to give probability or confidence for each unlabeled data to be positive or negative example. Then some data having high value are added as new training data. In this way, co-training extends training examples by adding data with pseudo judgement by two different classifiers.

Co-training is normally used for the classification learning, not constrained clustering like ours. Thus we need some adjustments for the application as follows.

- Since our objective is to extend must/cannot-link constraints for the clustering, unconstrained data pairs will be judged as must-link or cannot-link according to the result of constrained clustering.
- Since clustering result does not give any probability or confidence for a data pair to be must/cannot-link, just tells the pair belongs to a same cluster or different ones, we use the final kernel matrix K in Algorithm 1 to give such confidence values.

Let F_1 and F_2 be two different feature sets, K_1 and K_2 be kernel matrices created through the constrained clustering on F_1 and F_2, respectively. Co-training process repeats the following steps.

1) Run Algorithm 1 on F_1 and F_2, respectively, and get two kernel matrices K_1 and K_2.
2) Sort the values of K_1 in descending order. Then select top p and bottom n correspondent data pairs in the sorted list. Top p pairs are added as must-link constraints, and bottom n pairs are added as cannot-link constraints. Applying the same procedure to K_2, we can append total $2(p+n)$ pseudo constraints in a step of co-training.

Since we use two feature sets, we get two clustering results for a dataset. We select the better one depending on the practical usage. Parameters p, n and the number of repeats of the above cycle are set depending on the dataset.

IV. Experiments

We evaluated our proposed method on the WebKB Co-training dataset [1]. This dataset consists of 1051 web pages with two categories and has two different feature sets - Fulltext and Inlinks. Fulltext is a set of "texts" on the web pages. Inlinks is a set of "anchor texts" on the hyperlinks pointing to the page. Since both sets consists of texts, each data is represented by a bag of words. However words in a bag are clearly different from each other. We preprocessed Web pages by removing tags, some symbols and stopwords. Then we made feature vectors using the *tfidf* weighting method.

The objective of the experiments is to confirm whether co-training is useful for constrained clustering or not. Thus we simply investigate the clustering accuracy when co-training step goes on. The basic clustering algorithm is described in Section II. The bagging step T was set to 50. Distance metric used in the clustering was the Euclid distance.

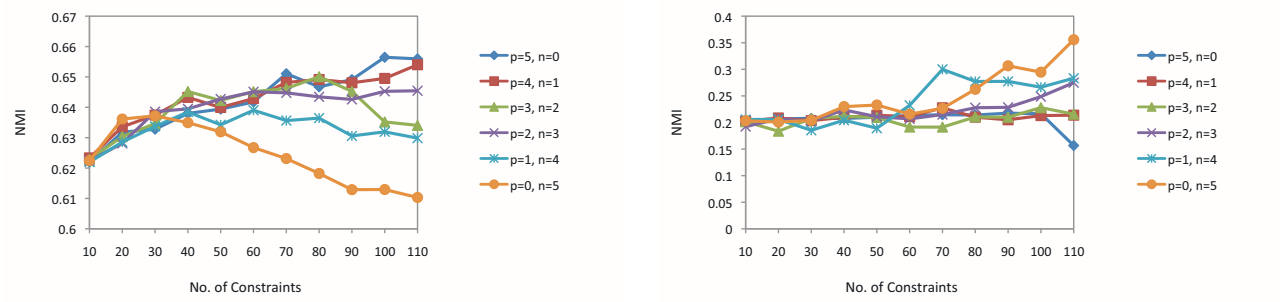
We used normalized mutual information (NMI) to measure the clustering accuracy. The NMI was calculated by using the following formula.

$$\mathrm{NMI}(C,T) = \frac{I(C,T)}{\sqrt{H(C)H(T)}}$$

where C is the set of cluster labels returned by algorithms and T is the set of true cluster labels. $I(C,T)$ is the mutual information between C and T, and $H(C)$ and $H(T)$ are the entropies.

Figure.1 shows the results. Figure.1(a) and Figure.1(b) show the results of the Fulltex and Inlinks feature sets

[1] http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-51/www/co-training/data/



(a) Fulltext (b) Inlinks

Figure 1. Results

respectively. In both graphs, horizontal axis indicates the number of extended constraints used for clustering, and vertical axis indicates the NMI value. We start co-training from situation that 10 true constraints are given, then repeats co-training cycle 10 times until total number of constraints reaches 110. We tested various parameter combinations about p and n, which are the numbers of pseudo constraints described in Section III. However we set 5 for the number of constraints to be added a feature set at a step of co-training.

As for the Fulltext feature set, pseudo must-link worked well, especially parameter combination $p = 5, n = 0$ showed the best performance. Pseudo cannot-link constraints were not effective in this feature set. On the other hand, pseudo cannot-link worked well in the Inlinks feature set. As seen the graph, parameter combination $p = 0, n = 5$ showed the best performance.

Comprared with the results of both feature sets, Fulltext outperformed much more than Inlinks. However the improvement of Inlinks is larger than Fulltext.

V. Conclusion

In this paper, we proposed a method to complement the lack of training examples for constrained clustering by using co-training framework, which extends pseudo training examples by leveraging two kinds of feature sets. We first introduced a constrained clustering ensemble algorithm that integrates a set of clusters produced by a constrained k-means with random ordered data assignment. Then we proposed to use co-training to extend constraints for the above clustering algorithm. In the co-training process, some data pairs are iteratively selected and added as pseudo constraints according to the value of kernel matrices that our proposed constrained clustering algorithm produces on two feature sets.

We evaluated our method on a task of Web page clustering using the WebKB Co-training dataset that provides two different feature sets. The results showed our method achieved the performance improvement by using co-training approach even if we provided only a small set of true constraints. We also found that the effectiveness of pseudo constraints differed on feature sets used in the clustering. Co-training can be an option to support semi-supervised constrained clustering while we need more reliable method to set parameters p and n depending on the feature set combination.

We will investigate more detailed behavior of our method and test it on many other datasets. For example, images on the Web can be a good candidate for the dataset because they are often annotated by text explanations. Such different kinds of features will help co-training in constrained clustering.

References

[1] S. Basu, I. Davidson, and K. L.Wagstaff, Eds., *Constrained Clustering*. CRC Press, 2008.

[2] O. Chapelle, B. Scholkopf, and A. Zien, *Semi-Supervised Learning*. The MIT Press, 2006.

[3] W. Tang, H. Xiong, S. Zhong, and J. Wu, "Enhancing semi-supervised clustering: A feature projection perspective," in *Proceedings of the 13th International Conference on Knowledge Discovery and Data Mining*, 2007, pp. 707–716.

[4] J. V. Davis, B. Julis, P. Jain, A. Sra, and I. S. Dhillon, "Information-theoretic metric learning," in *Proceedings of the 24th Int. Conf. on Machine Learning*, 2007, pp. 209–216.

[5] S. C. H. Hoi, R. Jin, and M. R. Lyu, "Learning nonparametric kernel matrices from pairwise constraints," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 361–368.

[6] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the eleventh annual conference on Computational learning theory*, ser. COLT' 98. New York, NY, USA: ACM, 1998, pp. 92–100. [Online]. Available: http://doi.acm.org/10.1145/279943.279962

[7] K. Wagstaff and S. Roger, "Constrained k-means clustering with background knowledge," in *Proceedings of the 18th International Conference on Machine Learning*, 2001, pp. 577–584.

[8] L. Breiman, "Bagging predictors," in *Machine Learning*, 1996, pp. 123–140.