Multi-instance Graphical Transfer Clustering for Traffic Data Learning

Shan Xue, Jie Lu, Jia Wu, Guangquan Zhang, and Li Xiong

Abstract— **In order to better model complex real-world data and to develop robust features that capture relevant information, we usually employ unsupervised feature learning to learn a layer of features representations from unlabeled data. However, developing domain-specific features for each task is expensive, time-consuming and requires expertise of the data. In this paper, we introduce multi-instance clustering and graphical learning to unsupervised transfer learning. For a better clustering efficient, we proposed a set of algorithms on the application of traffic data learning, instance feature representation, distance calculation of multi-instance clustering, multi-instance graphical cluster initialisation, multi-instance multi-cluster update, and graphical multi-instance transfer clustering (GMITC). In the end of this paper, we examine the proposed algorithms on the Eastwest datasets by couples of baselines. The experiment results indicate that our proposed algorithms can get higher clustering accuracy and much higher programming speed.**

I. Introduction

TRANSFER learning is desirable for the cases when the data contains different features or distribution changes so that statistical models need not to be rebuilt from scratch using newly collected training data [1]. Unlike traditional machine learning, in transfer learning domain, the source task is used to restore useful knowledge for new tasks to extract from and the target tasks is the task to apply source task on the novel task [2]. It focuses on learning situations of cross feature spaces and distributions. The inner workings of an unsupervised transfer learning problem are complex with many interacting components.

Multi-instance learning [3] is a variation of problems that deal with the incomplete knowledge on examples, in which, patterns are given as bags and each bag consists of some instances. The labels are assigned to bags, rather than instances. In a binary multi-instance learning classification problem [4], a typical assumption is that a bag should be labeled as positive if at least one of its instances is positive; and negative if all of its instances are negative. The goal of the multi-instance learning classification problem is to learn a classifier that can predict labels of new bags or instances. However, almost all of the transfer learning methods are designed to solve traditional single instance learning problems [5] [6] [7], while in many cases transfer learning can be better formulated as multi-instance learning problems. Therefore, if we want to transfer knowledge from some domains to others, it requires the solution of multi-instance transfer learning [8].

Graphical models can be viewed as estimating a function $f(\cdot)$ on the graph [9]. $f(\cdot)$ should satisfy both be close to the given labels y_L on the labeled nodes and be smooth on the whole graph at the same time. This can be expressed in a regularisation framework where the first term is a loss function, and the second term is a regulariser. Moreover, it is more important to construct a good graph [10] from unsupervised transfer learning and then to choose among the methods.

Despite of the challenge of real-time analysis in traffic information services [11], another challenge is on how to develop a computing infrastructure by big data to deal with the large volumes and variety of data so that the multiple data sources can be involved.

The aims of this paper is to provide solutions for unsupervised transfer learning within the following situations, (1) the multi-instance domain where the feature spaces are heterogeneous; (2) the unsupervised clustering algorithm which will be trained by unlabeled datasets; (3) evolutionaries in proposed algorithm which can get optimal solutions faster; and (4) knowledges that can be gained from long-life learning which transfer learning mechanism contributes. Meanwhile, the proposed algorithm in this paper can fully support most application scenarios, such as large-scale traffic datasets learning.

II. Graphical Clustering in Multi-instance Transfer Learning

A. Multi-instance Learning

DEFINITION 1 **(Bag [12])**: A bag $B_i = \{B_i^{(I)}\}$ contains a number of objectives (instances), in which $B_i^{(I)}$ denotes sub-bag with instances in B_i. A bag B_i's label is denoted by $y_i \in \mathcal{Y}$, with $\mathcal{Y} = \{-1, +1\}$. A bag is labeled positive if one or multiple instances in the bag is positive, and negative otherwise. $\boldsymbol{\mathcal{B}} = \{B_1, \cdots, B_p\}$ denotes the set of p bags, and the aggregation of all objectives in $\boldsymbol{\mathcal{B}}$ is denoted by $\mathcal{I} = \{I_1, \cdots, I_q\}$, where q is the total number of instances.

Shan Xue is with the Lab of Decision Systems & e-Service Intelligence (DeSI), the Centre for Quantum Computation & Intelligent Systems (QCIS), Faculty of Engineering and Information Technology (FEIT), University of Technology Sydney, Australia, and is affiliated with the School of Management, Shanghai University, China (Shan.Xue@student.uts.edu.au).

Jie Lu and Guangquan Zhang are with the Lab of Decision Systems & e-Service Intelligence (DeSI), the Centre for Quantum Computation & Intelligent Systems (QCIS), Faculty of Engineering and Information Technology (FEIT), University of Technology Sydney, Australia ({jie.lu, guangquan.zhang}@uts.edu.au).

Jia Wu is with the Centre for Quantum Computation & Intelligent Systems (QCIS), Faculty of Engineering and Information Technology (FEIT), University of Technology Sydney, Australia (jia.wu@uts.edu.au).

Li Xiong is with the School of Management, Shanghai University, China (xiongli8@shu.edu.cn).

This work is supported by the Australian Research Council (ARC) (No. DP140101366), and the Shanghai Education Commission (No. 14ZS085).

Algorithm 1 Instance Feature Representation

Input:

$\mathcal{I}$: An instance dataset;

$\mathcal{G}$: A graph dataset;

n_f: The number of features to be selected.

Output:

$\mathbf{f} = \{f_1, \cdots, f_{n_f}\}$: A set of features.

$\mathbf{f} = \emptyset$, $\rho = 0$;
for each feature f_k in $\mathbf{f}$ **do**
  $L' \leftarrow$ Apply $\mathcal{I}$ and $\mathcal{G}$ to obtain the graphical matrix;
  $r(f_k) \leftarrow$ Apply L' to compute the score of feature f_k;
  if $|\mathbf{f}| < max$ or $r(f_k) > \rho$ **then**
    $\mathbf{f} \leftarrow \mathbf{f} \cup f_k$
  end if
  if $|\mathbf{f}| \geq max$ **then**
    $\mathbf{f} \leftarrow \mathbf{f} / \arg\min_{f_i \in \mathbf{f}} r(f_i)$
  end if
  $\rho = \min_{f_i \in \mathbf{f}} r(f_i)$.
end for
return $\mathbf{f}$.

Similarly, the set of all positive (or negative) bags can be denoted by $\boldsymbol{\mathcal{B}}^+$ (or $\boldsymbol{\mathcal{B}}^-$).

DEFINITION 2 (**Instance Feature Representation [12]**): According to multi-instance bag constraint, all instances in a negative bag are negative. So a negative bag $B_i \in \boldsymbol{\mathcal{B}}^-$ can be represented by an instance feature vector $\mathbf{x}_i^B = \sum_{j=1}^{m_i} \mathbf{x}_i^j / m_i$, where m_i denotes the number of instances in $B_i \in \boldsymbol{\mathcal{B}}^+$, the instance with the largest distance from negative bags is used to denote the positive bag. So the instance feature vector for a positive bag is $\mathbf{x}_i^B = arg\ min_{\mathbf{x}_i^j \in B_i} \sum_{\mathbf{x}_k^B \in \boldsymbol{\mathcal{B}}^-} exp(-\|\mathbf{x}_i^j - \mathbf{x}_k^B\|^2/t)$.

In this paper, we employ the instance feature representation and extend it to the Algorithm 1. In Algorithm 1, we extract a set of instances $\mathcal{I}$ from the bag set $\mathbb{B}$ and it corresponds to the graph set $\mathcal{G}$. ρ denotes the minimum value of the scores $r(f_k)$. The main idea of this algorithm is that, after score all the selected features, if the number of features $\mathbf{f}$ are smaller than max or the scores of the new feature f_k are larger than ρ, f_k will be included into $\mathbf{f}$; and if the number of features $\mathbf{f}$ are larger than max, $\mathbf{f}$ will be upgraded by new scores. For each feature f_k, the processes above will be applied and ρ is going to be updated.

B. Multi-instance Graphical Clustering

The algorithm BAg-level Multi-instance Clustering (BAMIC) [13] clusters all the training bags into k disjoint groups $G_i (1 \leq i \leq k)$ each containing a number of training bags. Given two bags of instances $A = \{a_1, \cdots, a_m\}$ and $B = \{b_1, \cdots, b_n\}$, the maximal and minimal Hausdorff distances are defined as Eq. (1) and Eq. (2) respectively, where $\|a - b\|$ measures the distances between instances a and b which takes the form of Euclidean distance.

Algorithm 2 Distances of Multi-instance Clustering

Input:

$\mathbb{B}$: A bag dataset;

$\mathcal{G}$: A graph dataset.

Output:

Bag_dist: Distance metric used to calculate distances between bags and graphs.

Bag_dist(B_i, B_j) $\leftarrow$ Distance between bags by applying Eq. (7);
Bag_dist(B_i, G_j) $\leftarrow$ Distance between bag and graph by applying Eqs. (3) and (7);
Bag_dist(G_i,G_j) $\leftarrow$ Distance between graphs by applying Eqs. (4)-(6);
return Bag_dist.

$$\max H(A, B) = max\{\max_{a \in A} \min_{b \in B} \|a - b\|, \max_{b \in B} \min_{a \in A} \|b - a\|\} \tag{1}$$

$$\min H(A, B) = \min_{a \in A, b \in B} \|a - b\| \tag{2}$$

DEFINITION 3 (**Average Hausdorff Distance [13]**): In Eq. (3), where $|\cdot|$ measures the cardinality of a set. In words, $aveH(\cdot, \cdot)$ averages the distances between each instance in one bag and its nearest instance in the other bag. Conceptually speaking, average Hausdorff distance takes more geometric relationships between two bags of instances into consideration than those of maximal and minimal Hausdorff distances.

$$aveH(A, B) = \frac{\sum_{a \in A} \min_{b \in B} \|a - b\| + \sum_{b \in B} \min_{a \in A} \|b - a\|}{|A| + |B|} \tag{3}$$

DEFINITION 4 (**Graph Kernel [14]**): Given two multi-instance bags B_i and B_j which are presented as graphs $G_{bag}(\{\mathbf{x}_{bagu}\}_{u=1}^{n_{bag}}, \{\mathbf{e}_{bagv}\}_{v=1}^{m_{bag}})$, $bag = i, j$, where n_{bag} and m_{bag} are the number of nodes and edges in G_{bag}, respectively.

$$k_G(B_i, B_j) = \sum_{a=1}^{n_i} \sum_{b=1}^{n_j} k_{node}(\mathbf{x}_{ia}, \mathbf{x}_{jb}) + \sum_{a=1}^{m_i} \sum_{b=1}^{m_j} k_{edge}(\mathbf{e}_{ia}, \mathbf{e}_{jb}), \tag{4}$$

where k_{node} and k_{edge} are positive semi-definite kernels. To avoid numerical problem, k_G is normalised to

$$k_G(B_i, B_j) = \frac{k_G(B_i, B_j)}{\sqrt{k_G(B_i, B_i)}\sqrt{k_G(B_j, B_j)}}. \tag{5}$$

The k_{node} and k_{edge} can be defined in many ways. Here we simply define k_{node} using Gaussian RBF kernel [15] as

$$k_{node}(\mathbf{x}_{ia}, \mathbf{x}_{jb}) = exp(-\gamma \|\mathbf{x}_{ia} - \mathbf{x}_{jb}\|^2). \tag{6}$$

Algorithm 3 Multi-instance Graphical Cluster Initialisation

Input:

$U = \{X_1, \cdots, X_N\}$: Unlabeled multi-instance training set;

k: Number of clusters.

Output:

$\boldsymbol{\omega}$: The cluster selection labels.

$C_j \leftarrow$ Randomly selected k training bags as the initial medoids;
repeat
  for $j \in \{1, 2, \cdots, k\}$ **do**
    $G_j = \{C_j\}$;
  end for
  for $i \in \{1, 2, \cdots, N\}$ **do**
    $index = \arg\min_{j \in \{1,2,\cdots,k\}} \text{Bag\_dist}(X_i, C_j) \leftarrow$ Bag_dist by applying Algorithm 2;
    $G_{index} = G_{index} \cup \{X_i\}$;
  end for
  for $j \in \{1, 2, \cdots, k\}$ **do**
    $C_j = \arg\min_{A \in G_j} (\sum_{B \in G_j} \text{Bag\_dist}(A, B)/|G_j|) \leftarrow$ Bag_dist by applying Algorithm 2;
  end for
until {the clustering results do not change};
$\boldsymbol{\omega} \leftarrow$ Apply $index$ and C_j;
return $\boldsymbol{\omega}$.

Algorithm 4 Multi-instance Multi-Cluster Update

Input:

$\mathbb{B}$: A bag dataset;

$\mathcal{G}$: A graph dataset.

Output:

$\mathbf{Y}$: A vector of bag labels for $\mathbb{B}$.

$\boldsymbol{\omega} \leftarrow$ The selection vector of bags by Algorithm 3;
for Clusters $d = 1 : k$ **do**
  $G_{C^d} \leftarrow$ Dynamic value of cluster d in graph G;
  $[v_{C^d}, G_{C^d}] = v(\boldsymbol{\omega}_{C^d}) \leftarrow$ Initial cluster-based dynamic values of cluster d;
  $v(\boldsymbol{\omega}_{C^d}) - M = 0 \leftarrow$ Initial generic retirement value;
end for
while $\exists\ d : v_{C^d} > 0$ **do**
  $C^* = \arg\max_d \{M\} \leftarrow$ Best cluster;
  $\mathbf{Y} = [\mathbf{Y}, G_{C^*}] \leftarrow$ Best node in cluster C^*;
  $\omega_{G_{C^*}} = \mathbf{B}_{G_{C^*}} \leftarrow$ Set sampled $\mathbf{B}_{G_{C^*}}$ at $\omega_{G_{C^*}}$;
end while
for Clusters $d = 1 : k$ **do**
  $[v_{C^d}, G_{C^d}] = v(\boldsymbol{\omega}_{C^d, S_{C^*}}^{t_{C^*}}) \leftarrow$ Update cluster-based dynamic values of cluster d;
  $M : v^d(\boldsymbol{\omega}_{C^d}, M) - M = 0 \leftarrow$ Update cluster index by applying Eq. (9);
end for
return $\mathbf{Y}$.

DEFINITION 5 (**Bag Kernel [14]**): Given two multi-instance bags B_i and B_j which contains n_i and n_j instances, respectively.

$$k_g(B_i, B_j) = \frac{\sum_{a=1}^{n_i} \sum_{b=1}^{n_j} W_{ia} W_{jb} k(\mathbf{x}_{ia}, \mathbf{x}_{jb})}{\sum_{a=1}^{n_i} W_{ia} \sum_{b=1}^{n_j} W_{jb}}, \quad (7)$$

where $W_{ia} = 1/\sum_{u=1}^{n_i} w_{au}^i$, $W_{jb} = 1/\sum_{v=1}^{n_j} w_{bv}^j$, and $k(\mathbf{x}_{ia}, \mathbf{x}_{jb})$ is defined as similar as Eq.(6).

For multi-instance graphical clustering, Algorithm 2 defines the calculation of all the distances shown in multi-instance clustering. For the three kinds of distance, i.e., distances between bags, distances between bag and graph, and distances between graphs, we denote a function Bag_dist output from Algorithm 2. In Algorithm 3, we initial the multi-instance graphical clustering when unlabeled multi-instance training set comes. The data structures and knowledge are saved in such clustered graph and can be sequentially upgraded. A set of cluster selection labels is output from Algorithm 3.

C. Clustering Strategies of Multiple Clusters Update

The idea is to partition a large graph in smaller subgraphs (clusters) based on Brown and Smith [16], which can be computed efficiently. Let C^d, $d = 1, \cdots, k$ be disjoint nodes of the entire node set $\mathcal{N}$, i.e., $C^d \cap C^e = \emptyset$, and $\cup_{d=1}^k C^d = \mathcal{N}$. Denote $\mathbf{x}_{C^d}$ the random variables in cluster C^d, and $\boldsymbol{\omega}_{C^d}$ the cluster specific evidence. The number of nodes in cluster C^d will be in the order of one to around ten. The ranking is based on dynamic programming within clusters, given the current information. Once we collect data in clusters, we update the probabilities, use dynamic programming again, and get a new ranking. This provides the basis for the selection.

$$v_i(\boldsymbol{\omega}) = \sum_{j=1}^{k} p(x_i = j|\boldsymbol{\omega}) \left[r_i^j + \delta \cdot v(\boldsymbol{\omega}_i^j) \right] \quad (8)$$

DEFINITION 6 (**Multi-instance Graphical Clustering [17]**): Assume that dynamic programming is set up for each cluster, given the current evidence. By considering a variation of (Eq. (8)), with a generic retirement value M instead of 0 in the decision rule, we have expected value for cluster d given by:

$$\begin{aligned} v^d(\boldsymbol{\omega}, M) = \max_{i \in C^d} \{ & \sum_{j=1}^{k} p(x_i = j) \big[r_i^j + \delta \max_{s \in C^d / \{i\}} \\ & \{ \sum_{l=1}^{k} p(x_s = l | x_i = j)(r_s^l + \cdots), M \} \big], M \}, \end{aligned} \quad (9)$$

where $v^d(\boldsymbol{\omega}, M) \to M$.

In the algorithm of multiple clusters update for multi-instance sequential clustering (Algorithm 4), we rank the cluster according to Definition 6. Dynamic programming in the best cluster gives the first node. We update the probability model for all the clusters, given the observation (sample outcome) in the selected node. All graphical clusterings are also modified based on the updated probabilities. Then, we choose

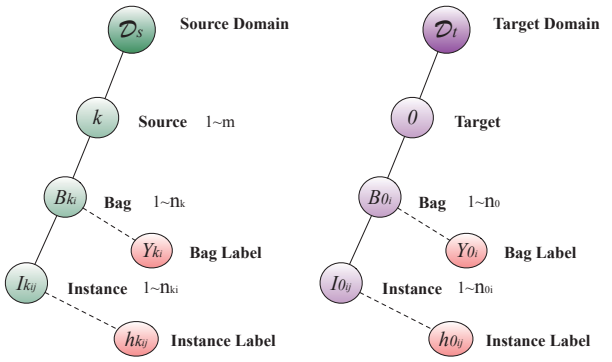

Fig. 1. Data structure of GC-MITL

Algorithm 5 GMITC: Graphical Multi-instance Transfer Clustering

Input:

$\mathbb{B}_s = \{\mathbf{B}_1, \cdots, \mathbf{B}_k, \cdots, \mathbf{B}_{n_k}\}$: The bags in $\mathcal{D}_s$;

$\mathbf{B}_0$: The bag in $\mathcal{D}_t$;

Output:

$G_{\mathbf{C}}$, $\mathbf{C} = \{C_1, \cdots, C_k\}$: A clustered graph.

$G_{\mathbf{C}} \leftarrow$ Initial clustered graph of source domain;

$\mathbf{Y}_{\mathbb{B}_s} \leftarrow$ Multiple cluster update for cluster labels of source bags by applying Algorithm 4;

for all $\mathbf{B}_0$ **do**

  $\mathbf{f} \leftarrow$ Features explored by Algorithm 1;

  Bag_dist$(B_{0i}, B_{ki}) \leftarrow$ Distances between target bags and source bags, calculated by Algorithm 2;

  $\boldsymbol{\omega}_{\mathbf{B}_0} \leftarrow$ Cluster selection labels of target bags by applying Algorithm 3;

  $\mathbf{Y}_{\mathbf{B}_0} \leftarrow$ Multiple cluster update for cluster labels of incoming target bags by applying Algorithm 4;

end for

if $\mathbf{Y}_{\mathbf{B}_0} \in \mathbf{Y}_{\mathbb{B}_s}$ **then**

  $G_{\mathbf{C}} \leftarrow G_{\mathbf{C}} \cup \mathbf{B}_0$;

else

  $G_{\mathbf{C}} \leftarrow G_{\mathbf{C}} \cup \{G_{C^{\mathbf{B}_0}}, \mathbf{B}_0\}$

end if

return $G_{\mathbf{C}}$.

the best cluster at the second stage using these clusterings. We proceed until all the nodes have been observed or there are no more clusterings greater than 0.

D. Graphical Multi-instance Transfer Clustering

In processing high-dimensional data, unsupervised learning is commonly used for exploratory purposes. Before learning, we may only have limited knowledge on how data is distributed. It can be grouped into multiple but unknown numbers of clusters with arbitrary shapes, reside on multiple low-dimensional manifolds, encompass mixed data structures (e.g., clusters and manifolds), or may contain no structure at all.

DEFINITION 7 (**Unsupervised Transfer Learning [1]**): Given a source domain $\mathcal{D}_s$ with a learning task $\mathcal{T}_s$, a target domain $\mathcal{D}_t$ and a corresponding learning task $\mathcal{T}_t$, unsupervised transfer learning aims to help improve the learning of the target predictive function $f_t(\cdot)$ in $\mathcal{D}_t$ using the knowledge in $\mathcal{D}_s$ and $\mathcal{D}_t$, where $\mathcal{D}_t$ is different from $\mathcal{D}_s$ and labels $\mathcal{Y}_s$ and $\mathcal{Y}_t$ are not observable.

III. Experiments

A. Experimental Conditions

We implement the proposed method using MATLAB tool and validate its performance on East-West dataset. We first test the proposed algorithms on 12 kinds of data standardisation methods. Then, we apply the optimised standardisation methods on instance clustering and bag clustering, respectively. In our experiments, the algorithms are evaluated in terms of cluster accuracy and program speed. Besides, the clustering results are compared with the real cluster label which we can find in the original dataset.

To make the dataset into 3 sources and make them different but relevant, we used the well-known East-West dataset and randomly selected equal amount of features from each bag. The randomness is also a good way to help us test the clustering effects. The structure of the dataset are shown in Table I.

B. Instance Clustering v.s. Bag Clustering

This experiment compares cluster effects on instances and bags. In instance clustering, we only cluster the instances ($n_i = 213$) of East-West dataset where each instance has 24 features.

Step 1, Standard values of 24 features. Different results of standardisation are shown in follows, A. No standardisation; B. Delete none 0-1 features; C. Standard none 0-1 features by average method; D. Standard all features by average method; E. Standard none 0-1 features by zscore function from MATLAB; F. Standard all features by zscore function

TABLE I
PROCESSED EAST-WEST DATASET FOR TRANSFER CLUSTERING

Bags No.	Instances	Source1 Attributes	Source2 Attributes	Source3 Attributes	Clusters
14	4	Ran #8	Ran #8	Ran #8	-
15	16	Ran #8	Ran #8	Ran #8	+
18	16	Ran #8	Ran #8	Ran #8	+
26	4	Ran #8	Ran #8	Ran #8	-
33	4	Ran #8	Ran #8	Ran #8	-
45	16	Ran #8	Ran #8	Ran #8	-
50	16	Ran #8	Ran #8	Ran #8	-
65	9	Ran #8	Ran #8	Ran #8	+
70	4	Ran #8	Ran #8	Ran #8	-
84	16	Ran #8	Ran #8	Ran #8	-
88	4	Ran #8	Ran #8	Ran #8	-
90	16	Ran #8	Ran #8	Ran #8	+
92	9	Ran #8	Ran #8	Ran #8	-
100	16	Ran #8	Ran #8	Ran #8	+
101	9	Ran #8	Ran #8	Ran #8	+
104	16	Ran #8	Ran #8	Ran #8	+
120	9	Ran #8	Ran #8	Ran #8	+
125	9	Ran #8	Ran #8	Ran #8	+
128	4	Ran #8	Ran #8	Ran #8	-
129	16	Ran #8	Ran #8	Ran #8	+

TABLE II
INSTANCE CLUSTERING ON 12 METHODS

No.	A	B	C	D	**E**	F	**G**	H	I	J	K	L
1	45.54%	61.03%	61.03%	60.09%	**67.61%**	60.09%	**60.09%**	67.61%	66.2%	60.09%	60.09%	61.03%
2	45.54%	61.03%	61.03%	60.09%	**67.61%**	60.09%	**60.09%**	67.61%	66.2%	60.09%	60.09%	61.03%
3	45.54%	61.03%	61.03%	60.09%	**67.61%**	60.09%	**60.09%**	67.61%	66.2%	60.09%	60.09%	61.03%
4	46.48%	61.03%	61.03%	60.09%	**67.61%**	60.09%	**60.09%**	67.61%	66.2%	60.09%	60.09%	61.03%
5	45.54%	61.03%	61.03%	60.09%	**67.61%**	60.09%	**60.09%**	67.61%	66.2%	60.09%	60.09%	61.03%
6	45.54%	61.03%	61.03%	60.09%	**67.61%**	60.09%	**60.09%**	67.61%	66.2%	60.09%	60.09%	61.03%
7	45.54%	61.03%	61.03%	60.09%	**67.61%**	60.09%	**60.09%**	67.61%	66.2%	60.09%	60.09%	61.03%
8	45.54%	61.03%	61.03%	60.09%	**67.61%**	60.09%	**60.09%**	67.61%	66.2%	60.09%	60.09%	61.03%
9	45.54%	61.03%	61.03%	60.09%	**67.61%**	60.09%	**60.09%**	67.61%	66.2%	60.09%	60.09%	61.03%
10	45.54%	61.03%	61.03%	60.09%	**67.61%**	60.09%	**60.09%**	67.61%	66.2%	60.09%	60.09%	61.03%

from MATLAB column by column; G. Standard all features by zscore function from MATLAB as matrix; H. Standard none 0-1 features by self-defined zscore (without abs); I. Standard none 0-1 features by self-defined zscore (with abs); J. Standard all features by self-defined zscore (without abs); K. Standard all features by self-defined zscore (with abs); and L. Standard none 0-1 features to 0-1 values by rules. **Step 2**, We treat the East-West dataset as the source domain and randomly separate 24 features into 3 sources, i.e., source 1, source 2 and source 3. Each source has 8 features, respectively. **Step 3**, Cluster source1, source2 and source3 separately. **Step 4**, Decide the final clusters across domain, i.e., cluster 1 and cluster 2. **Step 5**, Label cluster 1 and cluster 2 by negative (0) and positive (1) which without any supervise. **Step 6**, Calculate accuracy of instance clustering.

According to the instance clustering results shown in Table II, we find that only A has different results after source randomly selection; some methods can get 3 totally different clustering results on 3 source domains ($y = [1, 0, 0, 0]$), i.e., B (61.03%), C (61.03%), E (67.61%), H (67.61%), I (66.2%); and some methods can get robust clustering results (no matter whether clustering results on each source are the same, 2 in same or all not same, the clustering result of cross domain are the same) to the randomly source selections, i.e., D (60.09%), F (60.09%), G (60.09%), J (60.09%), K (60.09%).

While, the experiment of bag clustering clusters bags ($n_b = 20$) of the East-West dataset. The overall bags consist of instances ($n_i = 213$) and they have unbalanced instances amounts. To make cross domain transfer clustering, we randomly select 3 sources from 24 features of all instances. Meanwhile we examine the 8 different bag distance calculation methods on the 2 optimised data standardisations. The details of 16 methods and their results are explained as follows.

The 8 different bag distance calculation methods on bag clustering for source domain are, A1. Average Hausdorff Distance, Eq. (3); A2. Maximal Hausdorff Distance, Eq. (1); A3. Minimal Hausdorff Distance, Eq. (2); B. Graph Kernel without edges, Eqs. (4)-(6); C1. Bag Kernel, Eq. (7), where delta set by Maximal Hausdorff Distance, Eq. (1); C2. Bag Kernel, Eq. (7), where delta set by Minimal Hausdorff Distance, Eq. (2); C3. Bag Kernel, Eq. (7), where delta set by Average Hausdorff Distance, Eq. (3); and C4. Bag Kernel, Eq. (7), where delta set by general distance calculation.

We examine them on 2 optimised data standardisation methods E and G, respectively. The results are shown in the following tables (Table III and IV).

TABLE III
BAG CLUSTERING OF 12 METHODS ON STANDARDISATION E

No.	**A1**	A2	A3	B	C1	C2	C3	C4
1	**75.00%**	55.00%	50.00%	55.00%	60.00%	50.00%	50.00%	50.00%
2	**75.00%**	75.00%	50.00%	55.00%	50.00%	50.00%	50.00%	50.00%
3	**75.00%**	75.00%	65.00%	55.00%	50.00%	50.00%	50.00%	55.00%
4	**75.00%**	75.00%	65.00%	55.00%	50.00%	55.00%	50.00%	50.00%
5	**75.00%**	75.00%	55.00%	55.00%	50.00%	55.00%	50.00%	50.00%
6	**75.00%**	75.00%	55.00%	55.00%	50.00%	55.00%	50.00%	50.00%
7	**75.00%**	75.00%	50.00%	55.00%	50.00%	50.00%	50.00%	55.00%
8	**75.00%**	75.00%	50.00%	55.00%	50.00%	50.00%	50.00%	50.00%
9	**75.00%**	75.00%	50.00%	55.00%	50.00%	55.00%	50.00%	50.00%
10	**75.00%**	75.00%	50.00%	55.00%	50.00%	55.00%	50.00%	50.00%

TABLE IV
BAG CLUSTERING OF 12 METHODS ON STANDARDISATION G

No.	A1	A2	A3	B	C1	C2	C3	C4
1	25.00%	45.00%	35.00%	60.00%	55.00%	55.00%	55.00%	55.00%
2	25.00%	45.00%	35.00%	60.00%	55.00%	55.00%	55.00%	55.00%
3	45.00%	45.00%	35.00%	60.00%	55.00%	55.00%	55.00%	55.00%
4	25.00%	45.00%	35.00%	60.00%	55.00%	55.00%	55.00%	60.00%
5	25.00%	45.00%	35.00%	60.00%	55.00%	55.00%	55.00%	55.00%
6	25.00%	45.00%	35.00%	60.00%	55.00%	55.00%	55.00%	55.00%
7	25.00%	45.00%	35.00%	60.00%	55.00%	55.00%	55.00%	55.00%
8	25.00%	45.00%	35.00%	60.00%	55.00%	55.00%	55.00%	55.00%
9	25.00%	45.00%	35.00%	60.00%	55.00%	55.00%	55.00%	55.00%
10	25.00%	45.00%	35.00%	60.00%	55.00%	55.00%	55.00%	55.00%

The average accuracy, maximum accuracy and minimum accuracy of these 16 methods are shown in Fig. 2. The results indicate that the method A1 that applied on the first standardisation method E gets the highest results and robustness. Meanwhile, both the standardisation methods

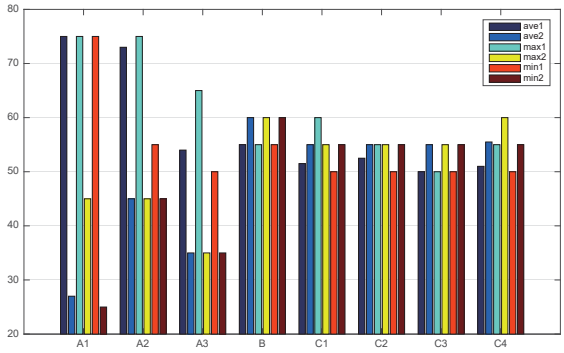

Fig. 2. Accuracies of average, maximum and minimum on 8 bag distance methods with 2 standardisation methods

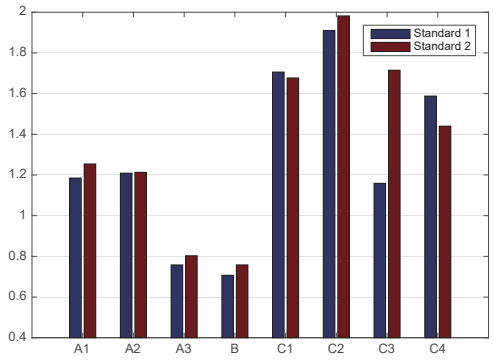

Fig. 3. Programming speed of 8 bag distance methods with 2 standardisation methods

and bag distance calculation methods strongly affect multi-instance clustering accuracy and graphical method shows the best results.

C. Programming Speed Evaluations

The instance clustering speeds on the E and G methods are 12.322s and 25.205s. However, the 2 methods combines 8 bag distance calculation methods show a much faster programming speed. The average speed of bag clustering is 1.316s (1.278s on E and 1.355s on G), the highest speed is 1.982s (C2 on G), and the lowest speed is 1.707s (B on E). That exactly proofs the contributions of multi-instance graphical transfer clustering are not only on accuracy, but also on the programming speed. In further study into the speed differences, we find insource clustering consumes a lot of time on instance clustering; but in bag clustering, the bag distance calculation is the most time consuming process. That is to say, multi-instance algorithm with transfer learning algorithm dramatically help save programming time on one same clustering task. Meanwhile, bag clustering also increases cluster accuracy.

IV. Conclusion and Future Work

In this paper, we proposed a graphical multi-instance transfer clustering algorithm, namely GMITC, supporting by a set of algorithms of instance feature representation, distances of multi-instance clustering, multi-instance graphical cluster initialisation and multi-instance multi-cluster update. We apply them to a real-world traffic problem, i.e., a well-known East-West Challenge. Since the original East-West dataset is for machine learning and multi-instance classification, we pre-process it for unsupervised transfer learning use. The randomness of source selection also help test the experiment and indicate the robustness. The experiment results show confirmations that multi-instance clustering jointly with graphical transfer learning (unsupervised) do good in both cluster accuracy and programming speed. The further study of this research will be on multi-instance graphical transfer clustering on online large datasets.

References

[1] S. J. Pan and Q. Yang, "A survey on transfer learning," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 22, no. 10, pp. 1345–1359, 2010.

[2] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, "Transfer learning using computational intelligence: A survey," *Knowledge-Based Systems*, vol. 80, pp. 14–23, 2015.

[3] Z. Fu, A. Robles-Kelly, and J. Zhou, "Milis: Multiple instance learning with instance selection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 5, pp. 958–977, 2011.

[4] B. Babenko, N. Verma, P. Dollár, and S. J. Belongie, "Multiple instance learning with manifold bags," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 81–88.

[5] X. Cao, Z. Wang, P. Yan, and X. Li, "Transfer learning for pedestrian detection," *Neurocomputing*, vol. 100, pp. 51–57, 2013.

[6] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 4, pp. 594–611, 2006.

[7] P. Huang, G. Wang, and S. Qin, "Boosting for transfer learning from multiple data sources," *Pattern Recognition Letters*, vol. 33, no. 5, pp. 568–579, 2012.

[8] D. Zhang and L. Si, "Multiple instance transfer learning," in *Data Mining Workshops, 2009. ICDMW'09. IEEE International Conference on*. IEEE, 2009, pp. 406–411.

[9] J. Wu, X. Zhu, C. Zhang, and P. S. Yu, "Bag constrained structure pattern mining for multi-graph classification," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 26, no. 10, pp. 2382–2396, 2014.

[10] J. Wu, S. Pan, X. Zhu, and Z. Cai, "Boosting for multi-graph classification," *Cybernetics, IEEE Transactions on*, vol. 45, no. 3, pp. 416–429, 2015.

[11] S. Xue, L. Xiong, S. Yang, and L. Zhao, "A self-adaptive multi-view framework for multi-source information service in cloud its," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–16, 2015.

[12] J. Wu, X. Zhu, C. Zhang, and Z. Cai, "Multi-instance multi-graph dual embedding learning," in *Data Mining (ICDM), 2013 IEEE 13th International Conference on*. IEEE, 2013, pp. 827–836.

[13] M.-L. Zhang and Z.-H. Zhou, "Multi-instance clustering with applications to multi-instance prediction," *Applied Intelligence*, vol. 31, no. 1, pp. 47–68, 2009.

[14] Z.-H. Zhou, Y.-Y. Sun, and Y.-F. Li, "Multi-instance learning by treating instances as non-iid samples," in *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009, pp. 1249–1256.

[15] T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola, "Multi-instance kernels." in *ICML*, vol. 2, 2002, pp. 179–186.

[16] D. B. Brown and J. E. Smith, "Optimal sequential exploration: Bandits, clairvoyants, and wildcats," *Operations research*, vol. 61, no. 3, pp. 644–665, 2013.

[17] G. Martinelli and J. Eidsvik, "Dynamic exploration designs for graphical models using clustering with applications to petroleum exploration," *Knowledge-Based Systems*, vol. 58, pp. 113–126, 2014.