TrGraph: Cross-Network Transfer Learning via Common Signature Subgraphs

Meng Fang, Jie Yin, Xingquan Zhu, *Senior Member, IEEE*, and Chengqi Zhang, *Senior Member, IEEE*

Abstract—In this paper, we present a novel transfer learning framework for network node classification. Our objective is to accurately predict the labels of nodes in a target network by leveraging information from an auxiliary source network. Such a transfer learning framework is potentially useful for broader areas of network classification, where emerging new networks might not have sufficient labeled information because node labels are either costly to obtain or simply not available, whereas many established networks from related domains are available to benefit the learning. In reality, the source and the target networks may not share common nodes or connections, so the major challenge of cross-network transfer learning is to identify knowledge/patterns transferable between networks and potentially useful to support cross-network learning. In this work, we propose to learn common signature subgraphs between networks, and use them to construct new structure features for the target network. By combining the original node content features and the new structure features, we develop an iterative classification algorithm, TrGraph, that utilizes label dependency to jointly classify nodes in the target network. Experiments on real-world networks demonstrate that TrGraph achieves the superior performance compared to the state-of-the-art baseline methods, and transferring generalizable structure information can indeed improve the node classification accuracy.

Index Terms—Transfer learning, node classification, networked data

1 Introduction

Recent advances in communications and network technologies have witnessed the booming of social and networking oriented applications, such as social networks, human disease networks, and citation networks and so on. For these applications, a unique feature is that their data are represented by a network structure, in which nodes denote entities or instances (e.g., users in a social network or scientific publications in a citation network) and links denote relationships between nodes (e.g., friendship, co-authorship, or citation relationship). Such networks are very dynamic in nature and may be updated frequently over time. For example, new users introduced to a friendship network or new publications published in each year will result in new nodes or links being created in the network. To analyze these networks, node classification is one of the most important problems that often arise in many advanced applications, ranging from advertisement, question answering, to recommendation systems [1], [2], [3], [4], [5], [6], [7]. The success of these applications all requires nodes to be accurately classified in a network.

Given a small set of labeled nodes in a network, node classification aims to make use of labeled nodes to predict the labels of unlabeled nodes [1], [2]. This type of classification problems typically arise in the context of various social networking scenarios [3], [6], [7], where nodes are not only associated with content features, but also share dependency structure features with other nodes in the network. In the node classification problem setting, it is assumed that a small set of nodes are required to be labeled through a manual annotation process. For example, publications in citation networks can be categorized into specific topics, and users in friendship networks can also being labeled in regards to their interests, hobbies, or group characteristics (such as colleagues or relatives etc.). However, these labels may not be available to the majority of nodes in the network, either because there lacks available resources for a human-centered labeling process, or because new nodes with unknown labels are constantly be created over time. Therefore, the classification problem that we address in this work is to use labeled nodes in conjunction with their content and structure features to predict the labels of unlabeled nodes in an automated way.

The node classification problem is inherently more complicated than generic supervised learning tasks, where instances are considered to be independent of each other, so learning and classification are purely based on vector-based instance features. In a network context, nodes not only contain content features but also share dependency structures with other nodes in the neighborhood [8]. Both the content features of nodes and the structure of the network are useful for learning and classification. For example, authors who frequently co-author papers are often from

- *M. Fang is with the Centre for Quantum Computation & Intelligent Systems, Faculty of Engineering and Information Technology, University of Technology, Sydney, Australia. E-mail: Meng.Fang@student.uts.edu.au.*
- *J. Yin is with the Commonwealth Scientific and Industrial Research Organization, Australia. E-mail: Jie.Yin@csiro.au.*
- *X. Zhu is with the Department of Computer and Electrical Engineering & Computer Science, Florida Atlantic University. E-mail: xzhu3@fau.edu.*
- *C. Zhang is with the Centre for Quantum Computation & Intelligent Systems, Faculty of Engineering and Information Technology, University of Technology, Sydney, Australia. E-mail: chengqi.zhang@uts.edu.au.*

Manuscript received 15 Jan. 2014; revised 10 Aug. 2014; accepted 27 Feb. 2015. Date of publication 15 Mar. 2015; date of current version 3 Aug. 2015. Recommended for acceptance by J. Wang.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2015.2413789

similar research fields, and members of friendship tend to share similar interests or hobbies. To build accurate prediction models for node classification, a sufficient number of labeled nodes are essential to achieve the desired classification accuracy.

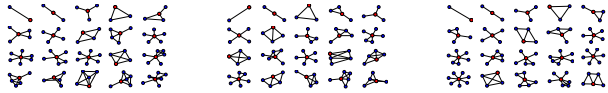
In reality, although networked data are easy to collect, the node labels are, however, either expensive to obtain or simply not available. This is particularly true for newly formed networks or for emerging entities of an existing network. The lack of sufficient labeled nodes makes it difficult to train a good classifier to classify unlabeled nodes. Fortunately, it is often the case that abundant labeled data may exist in many established networks from different yet relevant domains. To address this issue, transfer learning has emerged as a new machine learning framework, which aims at exploring external knowledge from auxiliary source domains to facilitate a new learning task in a target domain [9]. The basic idea is to uncover common factors shared between involved domains and use them as the bridge for knowledge transfer. To date, most existing studies on transfer learning have been mainly focused on traditional vector-based data, such as transactional data [10], image [11] and text [12], in which instances are assumed to be independent and identically distributed (i.i.d.). Little research work has been done to address effective and reliable cross-network transfer learning for node classification.

1.1 Key Challenges and Motivations

Cross-network transfer learning for node classification is a very challenging task, because there is no obvious indicator to identify the correspondence or relatedness between the source and target networks, and determine knowledge/patterns transferable across networks. In the following, we examine key challenges in cross-network transfer learning and seek opportunities to address them.

Network correspondence. Because the source and target networks may be formed for different purposes, the two networks can be largely distinct in that their nodes represent totally different types of entities, and the associated links indicate different relationships between nodes. Even in the case that two networks share similar relationships (such as citation relationship), each network may reveal different content features for its own nodes, making the feature spaces of the nodes from two networks have very little overlap, or no overlap at all. Therefore, performing transfer learning using the overlap of node content features would fail to achieve the desirable classification accuracy.

Transferable knowledge. Although the knowledge on node features is not necessarily transferable, two networks often share some common structure features. To illustrate this, let us examine two citation networks as one case study example; one is the CiteSeer network containing 3,327 nodes, and the other is the Cora network containing 2,708 nodes. The two networks are from relevant domains but have no overlapped nodes or edges. For each network, we extract its top frequent subgraphs using the method in [13], [14], as shown in Figs. 1a and 1b. It is clear that, the two networks share some frequent subgraphs, indicating that the networks have some common structure patterns with striking similarity although they may not have any nodes in common. Previous research [2], [4] has shown that, the structure information is a powerful source for the purpose of node classification, so that the nodes sharing the same local structures are likely to have the same label. Thus, discovering common structure patterns are the key to enable cross-network transfer learning.



(a) *Top frequent subgraphs of CiteSeer* (b) *Top frequent subgraphs of Cora* (c) *Common signature subgraphs*

Fig. 1. A case study example: the target network is Cora and the source network is CiteSeer. The two networks are from relevant domains, but have no overlapped nodes or edges. (a) and (b) show top frequent subgraphs extracted from CiteSeer and Cora, respectively. (c) shows common signature subgraphs discovered using our TrGraph algorithm.

Learning framework. After useful structure patterns are identified, a transfer learning framework needs to properly incorporate such knowledge to fulfill the underlying learning objective for node classification. In the network settings, the labels of connected nodes are correlated in a local neighborhood. This indicates that, closely connected nodes are likely to share the same label, and nodes on the same substructure tend to share the same label. Such label correlations should be considered together with transferred knowledge in the learning framework to achieve optimal performance for node classification.

Taking into account the three challenges in cross-network transfer learning, we aim to identify common structure patterns that are transferable from the source network and useful for the node classification in the target network.

Intuitively, although the source and target networks may not share common nodes or edges, as long as they are from relevant domains and built for relevant tasks, there potentially exist some interesting patterns with striking similarity between the networks. For example, in a citation network where nodes represent publications and edges denote citation relationships, a paper which addresses "sequential pattern mining" often cites papers related to Apriori-based pattern mining, pattern-growth based methods (such as FP-tree), and then other existing sequential pattern mining methods. This citation relationship, in fact, reveals the evolution of a research field, which is very common among scientific publications. It is very likely that we can find similar patterns across different citation networks, say one network is CiteSeer and the other network is Cora. Therefore, a key question is, what common structure patterns can be transferred effectively for boosting the node classification task in the target network?

In order to explore patterns transferable across networks, we report our case study example from two citation networks in Fig. 1. The two networks do not have any common nodes/edges at the node/edge level, and our objective is to check patterns discovered from two networks and investigate whether the two networks have any correspondence at the pattern level. As we can see, top frequent subgraphs found in CiteSeer (Fig. 1a) are not identical to those in Cora (Fig. 1b). So if we simply transfer top frequent subgraphs from the source network (CiteSeer), they may not be significant enough for representing the underlying structure of the target network (Cora), resulting in

grossly incorrect results for node classification in the target network. Therefore, it is desirable to transfer common structure patterns that occur frequently but also capture structural similarity between the networks. This motivates us to discover *common signature subgraphs* as transferable knowledge for cross-network node classification. Common signature subgraphs, as shown in Fig. 1c, are generalized subgraph representatives for transferring similar structure features across networks and thus have the potential to enhance node classification in the target network.

1.2 Our Contributions

We propose a novel approach to address cross-network transfer learning for node classification. Our key idea is to discover common signature subgraph patterns between the source and target networks and use them to boost the node classification accuracy in the target network.

For this purpose, we introduce a notation of *t-neighborhood structure*, which captures a node's local network structure within a maximum t-step distance centered at each single node. Based on this, we can extract a set of significant subgraphs respectively from the source and target networks, by aggregating t-neighborhood structures of all nodes in the network. After that, we devise an optimization problem to discover common signature subgraphs shared by the source and target networks. As domain-independent structure features, these common signature subgraphs are transferred from the source network so as to form new structure features for the nodes in the target network. Together with domain-dependent node features, we develop an iterative classification algorithm (ICA), TrGraph, that leverages label correlations to jointly classify the nodes in the target network. Experiments on real-world networks show that our proposed algorithm can successfully achieve knowledge transfer across networks and thus significantly improve the node classification accuracy in the target network.

2 Related Work

In this section, we review existing literature on node classification, state-of-the-art transfer learning, as well as frequent subgraph mining and influence pattern discovery.

2.1 Node Classification in Networks

The problem of node classification is an important learning task in social network analysis and data mining areas. Given a subset of labeled nodes in a network, node classification aims to use labeled nodes in conjunction with their content and structure features to classify the remaining unlabeled nodes [1], [2]. Previous research [4], [15], [16] has shown that, by jointly classifying the nodes, collective classification can achieve higher classification accuracies than traditional methods that classify each node separately. Eldardiry and Neville [17] proposed an ensemble method to reduce learning and inference variances in collective classification within domains where a same set of nodes are connected by multiple networks.

Collective classification approaches can be roughly categorized into global formulation-based methods and local classifier-based methods [16]. Global formulation-based methods aim to train a classifier that seeks to optimize a global objective function over the network, often based on a Markov random field (MRF) [18]. The extensions to MRF that take into account the observed attributes of data (i.e., nodes) are conditional random fields and relational Markov networks [19]. Global methods are usually computationally expensive in large-scale networks. On the other hand, local classifier-based methods have gained much attention recently for node classification for networked data [20], [21]. This group of methods employ an iterative process whereby a local classifier predicts the label of each node by using node content features and relational features derived from the current label predictions. After that, a collective inference algorithm recomputes the class labels, which will be used in the next iteration. This process continues until the predictions for all node labels are stabilized in the network.

Iterative classification algorithm is a local classifier-based method that is widely used in many studies [20], [21]. The basic assumption of ICA is that, for each node in the network, given the labels of its neighbors, the label of the node is independent of the features of its neighbors and non-neighbors, and the labels of all non-neighbors. In ICA, each node is represented by combining the node features and relational features constructed by using the labels of all the neighbors of the node. The relational features can be computed by using an aggregation function over the neighbors, such as `count`, `mode`, `proportion` and so on. Using both node content features and relational features, ICA trains a local classifier and recomputes the class labels of all nodes. This process continues until the algorithm converges. Local classifiers that have been used include naive Bayes [15], k-nearest neighbor [22], and logistic regression [23]. In this work, we adopt an ICA-like algorithm to jointly classify the nodes in the target network, which considers three types of features for each node, including node content features, structure features, and relational features.

2.2 Transfer Learning

Transfer learning is a machine learning paradigm which aims at borrowing knowledge from a related domain to help build an accurate classifier in the target domain, where the labeled data in the target domain is very limited [9].

2.2.1 Categories of Transfer Learning

According to the type of information that can be transferred, transfer learning approaches can be roughly grouped into three categories. The first category of transfer learning methods are based on instance transfer [24], [25], in which certain parts of the instances in the source domain are reused for learning in the target domain via instance weighting. TrAdaBoost [24] is one typical example of such methods, which assigns larger weights to the instances from the source domain that are more similar to the target instances. The second category is the parameter transfer approach [26], which assumes that the source and target learning tasks share similar parameters or prior distributions of the models, and thus transferring these parameters or priors can facilitate the target learning task. The third family of transfer learning aims to learn a good latent feature representation shared by two domains [27], [28], [29], [30]. In this case, the transferable knowledge is encoded in

the newly learned feature representation, which expects to improve the learning task in the target domain. In [28], Xue et al. proposed a cross-domain transfer learning algorithm that integrates labeled and unlabeled data from related domains for text classification. Wang et al. [30] presented a dual knowledge transfer approach for cross-language Web page classification. A commonality of most transfer learning methods is that they all assume the source and target data to be in the same feature space. However, in practice, it is common that the labeled data are scarce in their own feature space, whereas there may be a large amount of data in another feature space. Thus, our work relaxes this assumption and allows the nodes across networks to have different feature spaces.

There is also a body of research work on transfer learning from streaming data [31], [32], which aims to transfer knowledge from auxiliary source domains to aid an online target learning task. For example, Roy et al. [32] proposed a transfer learning algorithm that builds a transfer graph to capture the relation between the auxiliary social streams and the target video data. In this work, we also adopt the idea of graph transfer but focus on non-streaming scenarios for cross-network node classification.

2.2.2 Heterogeneous Transfer Learning

While most transfer learning methods assume that the data from different domains have the same feature space, heterogeneous transfer learning has been introduced as a new learning scenario, where the source and target data can be in different feature spaces and there may be no correspondence between data in these spaces.

Recently, researchers have attempted to solve heterogeneous transfer learning problems across image and text domains. Yang et al. [33] investigated exploring socially-annotated image data available on the Web to improve image clustering. They utilized the correlation between textual tags and image features in the annotated images to help estimate a good latent feature representation, through which better clustering results can be obtained. Hu et al. [34] leveraged auxiliary text documents to improve the tag recommendation performance for images. They proposed a factor alignment model which discovers the connection between image features and text features via the latent space of image tags. Zhu et al. [35] explored knowledge transfer from both unlabeled images and text data to enhance image classification performance. They attempted to enrich the representation of the target images with semantic concepts extracted from the auxiliary source data through a matrix factorization method. Similar to these methods, our work falls into the third category of transfer learning approaches, which focuses on learning a new and improved latent feature representation to improve the target learning task.

For existing heterogeneous transfer learning problem settings, they all require at least one primary key to link heterogeneous data sources, such as images sharing the same textual tag, or words occurring in both image tags and text documents. However, such an explicit correspondence does not exist in our problem. Instead, we resort to identifying common structure patterns transferable from source network and useful to generalize in the target network.

2.2.3 Transfer Learning on Relational Data and Networked Data

Although a significant amount of research has been done on transfer learning, most studies are limited to dealing with vector-based data, in which each instance is represented by a multi-dimensional feature vector, and all instances are assumed to be independent and identically distributed (i.i.d.). Several studies have been proposed to transfer knowledge on relational data (i.e., [36], [37]), where instances are non-i.i.d. and are represented by multiple relations. Mihalkova et al. [36] proposed a TAMAR algorithm to transfer relational knowledge with Markov logic networks (MLNs) across relational domains. Davis and Domingos [37] presented an approach to transferring relational knowledge based on a form of second-order logic. The basic idea of these algorithms is to discover structural regularities in the source domain in the form of Markov logic formulas and then instantiate these formulas with predicates from the target domain. These algorithms are based on the expressiveness of logics to establish semantic mappings to connect entities and their relationships from a source domain to a target domain. However, most real-world social networks have complex structures but lack explicit semantic mappings across two domains. In this context, how to perform effective transfer learning across general networks has remained a challenging task.

Only in recent years, attempts have been made to tackle transfer learning problems across networks. Ye et al. [38] presented a transfer learning algorithm to address the edge sign prediction problem in signed social networks. Because edge instances are not associated with a pre-defined feature vector, this work proposed to learn common latent topological features shared by the target and source networks, and then adopt an AdaBoost-like transfer learning algorithm with instance weighting to train a classifier. This differs from our work in that the edge prediction problems in two networks are assumed to share the same feature space and label space, so that the source edge instances can be reused via instance weighting. Fang et al. [39] proposed a transfer learning method that discovers common latent structure features as useful knowledge to facilitate collective classification in the target network. The proposed method discovers these latent features by constructing label propagation matrices in the source and target networks and mapping them into a shared latent feature space. Both of these methods have relied on the global graph similarities of both networks to construct implicit, latent structure features via nonnegative matrix factorization. The resulting latent features can be difficult to interpret. In contrast, our transfer learning framework focuses on finding explicit, local subgraph features that are more meaningful to capture localized structure information for node classification. Our solution also provides an explicit way to measure the relatedness of two networks by estimating the degree that they share on their subgraphs, which can be useful to avoid transferring from unrelated sources (i.e., negative transfer).

2.3 Frequent Subgraph Mining and Influence Pattern Discovery

To enable knowledge transfer across networks, we propose to find common signature subgraphs between the source

and target networks. This part of work is related to frequent subgraph mining [40], [41], [42] and influence pattern discovery in large networks [6], [8].

Frequent subgraph mining aims to discover subgraph structures whose frequency in a graph data set is above a minimum support threshold. Such high-frequency patterns are useful for many tasks, such as being used as features for graph classification [43] or for efficient graph database indexing [44], [45]. Many methods have been proposed for identifying frequent subgraphs within graph data sets, where the main challenge is to tackle the graph isomorphism. Early methods, such as AGM [40], used an adjacency matrix to represent graphs and employed a level-wise search, similar to the Apriori principle, for finding frequent subgraph patterns in an efficient way. Other methods made improvements by employing new coding mechanisms to avoid graph isomorphism. For example, gSpan [41] adopted a new lexicographic order to map each graph to a unique minimum depth-first search (DFS) code so as to mine frequently connected subgraphs efficiently. In summary, our proposed method is built upon existing research on frequent subgraph mining but further advances this field to identify common subgraphs shared by multiple networks as signature patterns to support transfer learning.

By using substructure patterns as tokens to study networks, existing research on influence pattern discovery [6], [8] has shown that typical patterns of influence exist in different networks, such as frequent cascade subgraphs. In other words, patterns that users pass their influence to friends or neighbors are largely similar. This is, in fact, the root of our research, because only if such patterns exist, we are able to use them to link different networks to achieve knowledge transfer for learning. In comparison, our work goes far beyond existing influence pattern mining and modeling (which normally focus on a single network) to link multiple networks for effective transfer learning.

3 Problem Formulation

We consider one source network G_s and one target network G_t for our node classification task. We focus on an inductive transfer learning setting, in which nodes in the source network G_s are fully labeled, while the target network G_t only has a small number of labeled nodes. The target network is represented as a graph $G_t = \{\mathcal{V}_t^u, \mathcal{V}_t^l, \mathcal{E}_t\}$, where $\mathcal{V}_t^l$ denotes a small set of labeled nodes in the network, and $\mathcal{V}_t^u$ denotes the set of nodes whose class labels are unknown and need to be predicted. $\mathcal{E}_t$ denotes the set of undirected edges connecting the nodes. Each node $v_t^i \in \mathcal{V}_t^u \cup \mathcal{V}_t^l$ is described by a feature vector $\mathbf{x}_t^i$. A mapping function $\mathcal{F}_t$ maps a node v_t^i to a class label $y_t^i \in \mathcal{Y}_t$, where $\mathcal{Y}_t$ denotes a set of class labels in the target domain. We treat the labeled nodes $\mathcal{V}_t^l$ as the training data and assume that $\mathcal{V}_t^l$ has the same distribution as the unlabeled node set $\mathcal{V}_t^u$. However, the quantity of the labeled nodes $\mathcal{V}_t^l$ is inadequate to train an accurate classifier for predicting the labels of unlabeled nodes $\mathcal{V}_t^u$ in G_t.

We also have a fully labeled source network $G_s = (\mathcal{V}_s^l, \mathcal{E}_s)$, where $\mathcal{V}_s^l$ denotes the set of labeled nodes and $\mathcal{E}_s$ denotes the set of undirected edges connecting the nodes. Each node $v_s^i \in \mathcal{V}_s^l$ is associated with a feature vector $\mathbf{x}_s^i$, and there exists a mapping function $\mathcal{F}_s$ that maps each node $v_s^i \in \mathcal{V}_s^l$ to its corresponding class label $y_s^i \in \mathcal{Y}_s$, where $\mathcal{Y}_s$ denotes a set of class labels in the source domain.

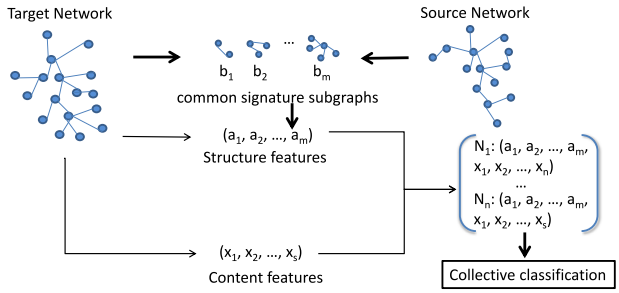


Fig. 2. Our proposed transfer learning framework for cross-network node classification.

We consider the most general cases of cross-network transfer learning, in which nodes in the source network G_s and the target network G_t can have different feature space and label space. In other words, nodes in source and target networks can denote different entities, or even if the same type of entities are represented, the nodes can have different content features. The classification problem of G_t can also differ from that of G_s, leading their respective label spaces $\mathcal{Y}_t$ and $\mathcal{Y}_s$ to be different. Thus, the mapping function $\mathcal{F}_s$ in G_s can be different from $\mathcal{F}_t$ in G_t. Although the knowledge on node features is not necessarily transferable, there may still exist some common properties between G_t and G_s. Given insufficient labeled nodes $\mathcal{V}_t^l$ from G_t, it is beneficial to leverage labeled nodes $\mathcal{V}_s^l$ from G_s to help classify unlabeled nodes $\mathcal{V}_t^u$ in G_t.

Formally, given the source network $G_s = (\mathcal{V}_s^l, \mathcal{E}_s)$ and the target network $G_t = \{\mathcal{V}_t^u, \mathcal{V}_t^l, \mathcal{E}_t\}$, the objective of our transfer learning task is to learn a classifier to predict the labels of unlabeled nodes $v_t^i \in \mathcal{V}_t^u$ in the target network G_t that maximizes the classification accuracy.

4 Methodology

The key issue of our transfer learning task is to identify knowledge/patterns which are transferable from the source network to the target network and then use them to facilitate the target learning task. Unlike traditional vector data that are independent and identically distributed, for networked data, the nodes are connected by links/edges to form a network. Closely connected nodes tend to have similar labels, and nodes sharing the same structure patterns are likely to have the same label [2], [4]. Therefore, we propose to discover common structure patterns shared by the source and target networks, and leverage these patterns to help the node classification task in the target network.

In Fig. 2, we outline our proposed transfer learning framework for cross-network node classification, which consists of three steps: (1) constructing structure features from the source and target networks; (2) discovering common signature subgraphs shared by the networks and using them to reconstruct structure features of the target network; and (3) combining reconstructed structure features and node content features to learn a classifier for jointly classifying nodes in the target network.

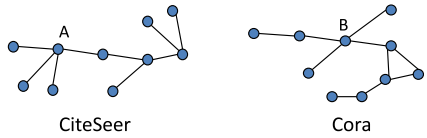


Fig. 3. Two examples of CiteSeer and Cora networks, where node "A" in CiteSeer and node "B" in Cora have similar neighborhood structures.

The technical details of the three steps are explained in the following sections.

4.1 Feature Construction

In this section, we discuss how to construct useful features for node classification. Because a network consists of a set of nodes and edges that connect the nodes, we would like to construct two types of features for each node: (1) content features, which represent the attributes that describe the characteristics of the node; and (2) structure features, which represent the structure information of the node with respect to its neighborhood in the network.

4.1.1 Content Features

For a given network, each node is usually associated with a set of pre-defined attributes that represent the node content. Therefore, we use the attribute values of each node to represent its content features as a feature vector. As noted before, because networks may be formed for different purposes, their node content features may not be generalizable from the source network to the target network.

4.1.2 Structure Features

Different from content features, a network has no value immediately available to represent its structure information, so there is no obvious vector form showing a node's structure features. In this work, we propose to use a subgraph-based approach to represent a node's structure information.

For this aim, we first explore the local neighborhood of a node to find its significant structure patterns. Assuming that the label of a node is only dependent on its local neighborhood structure within depth t, for any node v, we consider its t-neighborhood structure as its structure features. Formally, we define the t-neighborhood structure as follows.

Definition 1 (t-neighborhood structure). *Given a graph $G = (\mathcal{V}, \mathcal{E})$ and a node $v \in \mathcal{V}$, the t-neighborhood structure g_v of node v is a subtree, rooted from v and recursively traversing all of v's neighbors. For each neighbor, the traversal collects all the nodes until the shortest path from root v to the current node reaches a depth of t. The* **t***-neighborhood structure consists of all the visited nodes and the original links between them.*

In other words, for a node v in the network, considering v as the root node, we can crawl its t-neighborhood structure g_v using a breadth-first search (BFS) [46] originated from the node with depth t. The final results of t-step BFS rooted from node v are regarded as the structure features for node v. This procedure has the complexity of $O(|\mathcal{V}_v| + |\mathcal{E}_v|)$, where $|\mathcal{V}_v|$ indicates the number of nodes in g_v and $|\mathcal{E}_v|$ indicates the number of edges in g_v.

For the two examples of CiteSeer and Cora networks shown in Fig. 3, we illustrate t-neighborhood structures built for node "A" in the Citeseer network and node "B" in the Cora network, respectively, in Fig. 4. We can see that, node "A" and node "B" have the same one-neighborhood structures, but different t-neighborhood structures for $t = 2, 3,$ and 4.

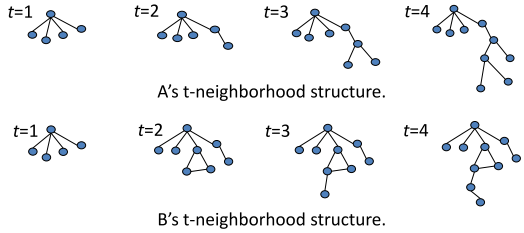


Fig. 4. The t-neighborhood structures for node "A" and node "B" in Fig. 3 when $t = 1, 2, 3, 4$.

In reality, nodes in the networks can have different neighborhood structures, while some of them might be sufficiently similar to each other. Thus, we propose a uniform way to represent a node's neighborhood structures by considering the similarity or distance between two subgraphs. Formally, given a node v, we define a mapping function $\mathcal{N} : (v, t) \rightarrow \mathbf{n}$, which generates a vector form $\mathbf{n}$ for describing the structure information of node v within depth t. As a result, we can map the node's t-neighborhood structures to the same feature space.

If we consider that the t-neighborhood structure is constituted by subgraphs, we can construct a set of subgraph bases to represent the neighborhood structure. Let $B = \{b_1, b_2, \ldots, b_k\}$ denote the entire space of subgraph bases. The mapping function $\mathcal{N}$ can be built based on these subgraph bases. Given a t-neighborhood structure of node v, the mapping function $\mathcal{N}$ uses these subgraph bases as input and outputs a feature value for each corresponding subgraph base. The generated values form a feature vector, named subgraph-based vector $A = \{a_1, a_2, \ldots, a_k\}$, where a_i is the corresponding feature value for subgraph base b_i. Based on the above idea, we formally define related notations as follows.

Definition 2 (Subgraph). *Given a graph $G = (\mathcal{V}, \mathcal{E})$ and a graph $g = (\mathcal{V}_g, \mathcal{E}_g)$, g is a subgraph of G if there exists an injective function $f : \mathcal{V}_g \rightarrow \mathcal{V}$ such that*

- $\forall v \in \mathcal{V}_g, f(v) \in G$;
- $\forall (u, v) \in \mathcal{E}_g, (f(u), f(v)) \in \mathcal{E}$.

Definition 3 (Subgraph bases). *Given a graph $G = (\mathcal{V}, \mathcal{E})$, the subgraph bases B is a collection of subgraphs $\{b_1, b_2, \ldots, b_k\}$, which is composed of subgraphs of G, and b_i indicates a subgraph base of B.*

Definition 4 (Subgraph activations). *Given a graph $G = (\mathcal{V}, \mathcal{E})$ and its subgraph bases $B = \{b_1, b_2, \ldots, b_k\}$, the subgraph activations A is a collection of values $\{a_1, \ldots, a_k\}$, where each activation value a_j corresponds to one base b_j of subgraph bases $B = \{b_1, b_2, \ldots, b_k\}$. A graph G and its subgraph bases B uniquely determine G's subgraph activations A.*

Definition 5 (Subgraph-based representation). *Given a graph $G = (\mathcal{V}, \mathcal{E})$, the subgraph bases B, and subgraph activations A, for any node $v \in \mathcal{V}$, the v's subgraph-based representation is a vector form, which is composed of the values of*

subgraph activations $A = \{a_1, a_2, \ldots, a_k\}$, *each* a_i *corresponding to a subgraph base* b_i *in* $B = \{b_1, b_2, \ldots, b_k\}$.

Now let us focus on how to find subgraph bases B from a graph G and calculate the corresponding values of subgraph activations A. One straightforward approach is to use t-neighborhood structures discovered from a graph as subgraph bases. This can be done by filtering the same t-neighborhood structures, and aggregating unique t-neighborhood structures as subgraph bases. That is,

$$B = \{b_1, b_2, \ldots, b_k\} = \text{unique}\{g_1, g_2, \ldots, g_n\}. \quad (1)$$

Given a t-neighborhood structure g, we can establish its relationship to subgraph bases $B = \{b_1, b_2, \ldots, b_k\}$. For each subgraph base b_i, we calculate the probability $\Pr(g|b_i)$ which indicates the causal probability of the ith cause b_i triggering the event g [47]. If the structure of g is likely to be the subgraph base b_i, the value of $\Pr(g|b_i)$ would be large. Otherwise, if no structure in g is likely to be b_i, the value of $\Pr(g|b_i)$ would be close to zero. The calculation of probability $\Pr(g|b_i)$ depends on the similarity between graph g and subgraph base b_i. The formal definition of $\Pr(g|b_i)$ is given later in this section.

Based on subgraph bases B, given any node and its t-neighborhood structure g, we obtain a new vector-based representation to capture its structure features, which is defined as follow:

$$\mathbf{n} = \{a_1, a_2, \ldots, a_k\} = \{\Pr(g|b_1), \Pr(g|b_2), \ldots, \Pr(g|b_k)\}, \quad (2)$$

where $\Pr(g|b_i)$ indicates the activation value a_i of g with respect to subgraph base $b_i \in B$.

We now discuss how to calculate the probability $\Pr(g|b_i)$. To this end, we first introduce several definitions as follows.

Definition 6 (Graph isomorphism). *Given two graphs* G *and* G', G *is a graph isomorphism of* G' *if there exists a bijective function* f *such that*

- $\forall (u, v) \in \mathcal{E}', (f(u), f(v)) \in \mathcal{E}$,
- $\forall (u, v) \in \mathcal{E}, (f(u), f(v)) \in \mathcal{E}'$.

Definition 7 (Subgraph isomorphism). *Given two graphs* G *and* G', G *is a graph isomorphism from* G *to* G' *if there exists a subgraph* $S \subseteq G$ *such that* f *is a graph isomorphism from* S *to* G.

Definition 8 (Common structure). *Given a graph* G_c *and two other graphs* G_1 *and* G_2, G_c *is a common subgraph of* G_1 *and* G_2 *if there exist subgraph isomorphisms both from* G_c *to* G_1 *and from* G_c *to* G_2.

Definition 9 (Maximal common subgraph). *Given a graph* G_m *and two other graphs* G_1 *and* G_2, G_m *is a maximal common subgraph of* G_1 *and* G_2 *if there do not exist other common subgraphs* G_c *of* G_1 *and* G_2 *that contain more nodes than* G_m.

It is worth noting that computing maximal common subgraph (MCS) between two graphs is known to be NP-complete. Optimal algorithms that follow different principles have been proposed to solve this problem; the first principle finds all common subgraphs of the two given graphs and chooses the largest, and the second builds the association graph between the two given graphs and then searches for the maximum clique of the latter graph. As they both involve a space state search process, the worst case time complexity is exponential with respect to the number of nodes in the graphs [48]. Alternatively, suboptimal algorithms have been developed to speed-up the efficiency, which typically adopt some heuristics to prune unfruitful search paths in the search space [49]. In our problem, since the sizes of two input graphs, t-neighborhood structure g and subgraph base $b_i \in B$, are very small, the time complexity of computing MCS between two small graphs is well acceptable.

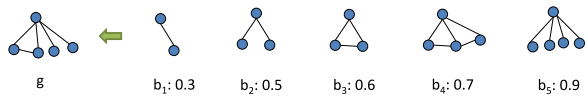


Fig. 5. Computing the values of subgraph activations given subgraph bases. $\{b_1, b_2, \ldots, b_5\}$ are subgraph bases and the decimals are the values of subgraph activations.

We define the maximal common subgraph between two graphs G_1 and G_2 as $\text{mcom}(G_1, G_2)$. For a graph $G = (\mathcal{V}, \mathcal{E})$, suppose that $|\mathcal{V}|$ and $|\mathcal{E}|$ denote the number of nodes and edges, respectively. For notational convenience, we let $|G|$ denote the number of nodes and edges of G, where $|G| = |\mathcal{V}| + |\mathcal{E}|$. The probability $\Pr(g|b_i)$ is calculated as follows:

$$\Pr(g|b_i) = \frac{|\text{mcom}(g, b_i)|}{\max(|g|, |b_i|)}. \quad (3)$$

If g has more common structures with b_i, $\Pr(g|b_i)$ would be larger, and if there is fewer common structures between g and b_i, $\Pr(g|b_i)$ would be closer to zero.

Formally, the probability $\Pr(g|b_i)$ for any given graph g and subgraph b_i has the following properties.

Property 1. *For any graph* g *and subgraph base* b_i, *the following conditions hold true:*

1) $0 < \Pr(g|b_i) \leq 1$;
2) $\Pr(g|b_i) = 1 \leftrightarrow g$ *and* b_i *are isomorphic to each other;*
3) *If* g_1 *and* g_2 *are isomorphic to each other, then* $\Pr(g_1|b_i) = \Pr(g_2|b_i)$.

Proof. According to the definition of Eq. (3), Properties (1)-(3) can be easily validated. □

Fig. 5 shows an example of computing the values of subgraph activations of graph g with respect to subgraph bases $\{b_1, b_2, \ldots, b_5\}$. The decimals indicate the value of subgraph activation a_i, computed as $\Pr(g|b_i)$, for the corresponding subgraph base b_i. Clearly, the larger the value of subgraph activation a_i, the more similar graph g and subgraph base b_i.

Algorithm 1 lists the procedure of constructing subgraph-based structure features for all nodes in network G.

4.2 Discovering Common Signature Subgraph Bases Between Networks

After useful features are extracted from the source network G_s and the target network G_t, our next step aims to discover patterns that are generalizable from G_s to G_t. As noted before, although the networks in relevant domains may have different content features for their nodes, they do share

similar structure patterns. Therefore, we propose to find an optimal set of common signature subgraph bases $B^*_{(k)}$ that capture common structure patterns between G_s and G_t.

Algorithm 1. Constructing Subgraph-based Structure Features

Input: A network $G = (\mathcal{V}, \mathcal{E})$,
Output: Subgraph bases B, Subgraph-based representation $\mathbf{n}_v$ of each node $v \in \mathcal{V}$.
1: Collect all t-neighborhood structures from network G;
2: Construct subgraph bases B by using unique t-neighborhood structures;
3: **for** each node $v \in \mathcal{V}$ **do**
4: Collect t-neighborhood structure g of node v;
5: Compute $\Pr(g|b_i)$, for each $b_i \in B$ using Eq. (3);
6: Obtain a subgraph-based representation $\mathbf{n}_v = \{\Pr(g|b_1), \ldots, \Pr(g|b_k)\}$.
7: **end for**

Let B_s and B_t denote the subgraph bases of the source network G_s and the target network G_t, respectively. The whole set of $B_s \cup B_t$ can be considered as candidates for finding an optimal set of common signature subgraph bases $B^*_{(k)}$. In other words, we want to find a "good" subset $B^*_{(k)}$ from all the subgraph base candidates $B_s \cup B_t$ to best represent both the target and source networks. Therefore, the common signature subgraph bases $B^*_{(k)}$ can be discovered by solving a likelihood-form optimization problem as:

$$B^*_{(k)} = \arg\max_{B_{(k)}} \prod_{v \in G_t} \Pr(g_v|B_{(k)}) \prod_{v \in G_s} \Pr(g_v|B_{(k)}). \quad (4)$$

Since we have $B_{(k)} = \{b_1, b_2, \ldots, b_k\}$, we then rewrite the above equation as follows:

$$B^*_{(k)} = \arg\max_{B_{(k)}} \prod_{v \in G_t} \prod_{b_i \in B_{(k)}} \Pr(g_v|b_i) \prod_{v \in G_s} \prod_{b_i \in B_{(k)}} \Pr(g_v|b_i). \quad (5)$$

Here, the source network G_s and the target network G_t are represented in the same space and the common signature subgraph bases can capture the common factors of the original neighborhood structures in both G_t and G_s.

To solve this optimization problem, we define

$$L = \prod_{v \in G_t} \prod_{b_i \in B_{(k)}} \Pr(g_v|b_i) \prod_{v \in G_s} \prod_{b_i \in B_{(k)}} \Pr(g_v|b_i). \quad (6)$$

Then we have

$$L = \prod_{b_i \in B_{(k)}} \left\{ \prod_{v \in G_t} \Pr(g_v|b_i) \prod_{v \in G_s} \Pr(g_v|b_i) \right\}. \quad (7)$$

We rewrite the above equation using a log form:

$$\log L = \sum_{b_i \in B_{(k)}} \log \left\{ \prod_{v \in G_t} \Pr(g_v|b_i) \prod_{v \in G_s} \Pr(g_v|b_i) \right\}, \quad (8)$$

and the optimal problem in Eq. (5) can be rewritten as:

$$B^*_{(k)} = \arg\max_{B_{(k)}} \log L. \quad (9)$$

Our optimization problem is then transferred to selecting an optimal set of b_i's that can maximize the sum of $\log\{\prod_{v \in G_t} \Pr(g_v|b_i) \prod_{v \in G_s} \Pr(g_v|b_i)\}$.

Considering $0 < \Pr(g_v|b_i) \leq 1$, we have

$$\log \left\{ \prod_{v \in G_t} \Pr(g_v|b_i) \prod_{v \in G_s} \Pr(g_v|b_i) \right\} < 0. \quad (10)$$

Thus, we can rewrite the optimal function as:

$$B^*_{(k)} = \arg\min_{B_{(k)}} \sum_{b_i \in B_{(k)}} \left\{ -\log \left\{ \prod_{v \in G_t} \Pr(g_v|b_i) \prod_{v \in G_s} \Pr(g_v|b_i) \right\} \right\}. \quad (11)$$

For each b_i from the candidate set $B_s \cup B_t$, we can compute $\Pr(g|b_i)$ for each node in both G_s and G_t. The time complexity is $O(n_c(|\mathcal{V}_s| + |\mathcal{V}_t|))$, where n_c is the number of subgraph base candidates. Using all the unique subgraph bases as candidates, we then select a set of $B^*_{(k)}$ with the lowest k values of $-\log\{\prod_{v \in G_t} \Pr(g_v|b_i) \prod_{v \in G_s} \Pr(g_v|b_i)\}$ using the quicksort method which has the time complexity of $O((|\mathcal{V}_s| + |\mathcal{V}_t|)\log(|\mathcal{V}_s| + |\mathcal{V}_t|))$.

4.3 Transferring Common Signature Subgraph Bases for Node Classification

After discovering common signature subgraph bases as generalizable features, our ultimate aim is to transfer these common structure features to help build an accurate classifier in the target network for node classification.

For our node classification task, because the nodes are not independent but are connected by links between each other as a network, the labels of connected nodes are dependent in a local neighborhood, and closely connected nodes are likely to share the same label. Previous research [4], [15] has demonstrated that collective classification can significantly improve the node classification accuracy by jointly classifying the nodes, as opposed to traditional classification methods that take vector inputs to classify the nodes separately. Therefore, we adopt an iterative classification algorithm as our underlying learning framework that makes use of label correlations to jointly classify the nodes in the target network.

To train an ICA classifier, for each node v in G_t, we take into consideration three types of features, including:

1) Content features: we use attribute values associated with node v as its content features $\mathbf{x}_v$.
2) Structure features: for a given node v, we first collect v's t-neighborhood structure g_v, and then compute v's new feature representation $\mathbf{n}_v$ based on common signature subgraph bases $B^*_{(k)} = \{b_1, b_2, \ldots, b_k\}$:

$$\begin{aligned} \mathbf{n}_v &= \{a_1, a_2, \ldots, a_k\}, \\ &= \{\Pr(g_v|b_1), \Pr(g_v|b_2), \ldots, \Pr(g_v|b_k)\}. \end{aligned}$$

The new feature representation $\mathbf{n}_v$ is used as structure features of node v. Together with node content

features, each node in the target network has a new feature representation $\mathbf{x}_{new} = (\mathbf{x}, \mathbf{n})$.

3) Relational features: for each node v, we also compute relational features by using an aggregation function, such as `count`, `mode`, and `proportion`, to collect the statistics information of the labels from the neighbors of node v. Note that, relational features may continuously change if any of node labels changes.

By combining node content features, structure features, and relational features derived from the current label predications, the ICA classifier iteratively trains a local classifier and updates the class labels of all nodes, which in turn generates new values of relational features, until the algorithm converges. The complexity of ICA is $O(\mathcal{B}T_f(|\mathcal{V}|))$, where $\mathcal{B}$ indicates the total number of iterations and $T_f(n)$ indicates the time required to train the local classifier, with $|\mathcal{V}|$ denoting the number of nodes in the network.

Algorithm 2 lists the detailed procedure of our transfer learning algorithm for cross-network node classification.

Algorithm 2. Transfer Learning for Cross-Network Node Classification

Input: The source network $G_s = (\mathcal{V}_s^l, \mathcal{E}_s)$, the target network $G_t = (\mathcal{V}_t^l, \mathcal{V}_t^u, \mathcal{E}_t)$, and a base classifier f,
Output: Labels of unlabeled nodes in $\mathcal{V}_t^u \in G_t$.

```
Collect t-neighborhood structures from G_s and G_t;
Construct the subgraph bases for G_s and G_t;
Learn common signature subgraph bases between G_s and G_t;
Reconstruct structure features of G_t using common signature subgraphs;
For each node v, the new features are x_new = (x, n);
for each node v_t^i in G_t do
    Compute relational features using only observed nodes in its neighborhood;
    Predict the label for an unlabeled node: y_t^i ← f(v_t^i);
end for
while all y_t^i's are not stabilized or number of iterations does not exceed a threshold do
    Generate an ordering O over nodes in G_t;
    for each node v_t^i ∈ O do
        Compute relational features using the current label predictions of its neighborhood;
        Predict the label for an unlabeled node: y_t^i ← f(v_t^i);
    end for
end while
Assign the predicted labels to V_t^u.
```

5 Experiments

To validate the performance of our proposed transfer learning algorithm, we carry out extensive experiments on four real-world networks.

5.1 Data Sets

We use four real-world networks in our experiments, including CiteSeer, Cora, WebKB and Terrorist Attacks.[1] For benchmark networks, their node features are different and node label spaces are also different indicating different classification problems. Table 1 summarizes the statistics of the four networks with detailed descriptions as follows.

TABLE 1
Statistics of the Four Benchmark Networks

Data Set	CiteSeer	Cora	WebKB	Attack
# of nodes	3,312	2,708	265	645
# of links	4,732	5,429	479	3,172
# of classes	6	7	5	6
# in largest Class	701	818	122	312
# in smallest Class	249	180	22	4
Avg. degree	0.00085	0.0014	0.013	0.015
Avg. closeness	0.0452	0.137	0.288	0.0155
Avg. clustering coeff.	0.14	0.24	0.21	0.76

CiteSeer is a citation network which consists of 3,312 publications and 4,732 citation links. Each node is represented by a 0/1-valued word vector indicating absence/presence of the corresponding word from a dictionary of 3,703 words, and is labeled as one of six classes: *Databases*, *Machine Learning*, *Information Retrieval*, *Artificial Intelligence*, *Human Computer Interaction*, and *Agents*.

Cora contains 2,708 publications labeled as one of seven classes: *Probabilistic Methods*, *Neural Networks*, *Case Based*, *Rule Learning*, *Reinforcement Learning*, *Genetic Algorithms* and *Theory*. It contains 5,429 citation links.

WebKB contains information about Web pages and their hyperlinks collected from computer science departments of various universities. We use the Wisconsin data which contains 265 Web pages and 479 hyperlink relationships. Each Web page is classified into one of five classes: *student*, *course*, *faculty*, *project* and *staff*.

Attack consists of 645 terrorist attacks each assigned one of six labels, indicating the type of the attack, including *Bombing*, *Weapon Attack*, *Kidnapping*, *Arson*, *NBCR Attach*, and *Other Attack*. Each node represents a terrorist attack and the link is created between the co-located attacks.

We can observe that CiteSeer and Cora have a larger number of nodes and links, while Attack has the largest average degree and average clustering coefficient and WebKB has the largest average closeness. The statistics show that there exists discrepancy in data distribution and network properties among the four networks. For each network, we consider a binary classification problem which takes the largest class as positive and the rest as negative.

5.2 Baselines

For evaluation, we compare our proposed *TrGraph* algorithm with the following four baseline methods.

ICA. This method simply uses the content features of the labeled nodes in the target network to train an ICA classifier for predicting unlabeled nodes [23].

Subgraph-based ICA (SICA). This method also relies only on the target network to perform collective classification. In addition to node content features, it also uses top-k frequent subgraphs in the target network as subgraph bases to construct structure features for training an ICA classifier.

Frequent subgraph-based transfer learning (FrGraph). This method is a variant of TrGraph, in which top-k frequent subgraphs, rather than common signature subgraphs,

1. http://www.cs.umd.edu/projects/linqs/projects/lbc/index.html

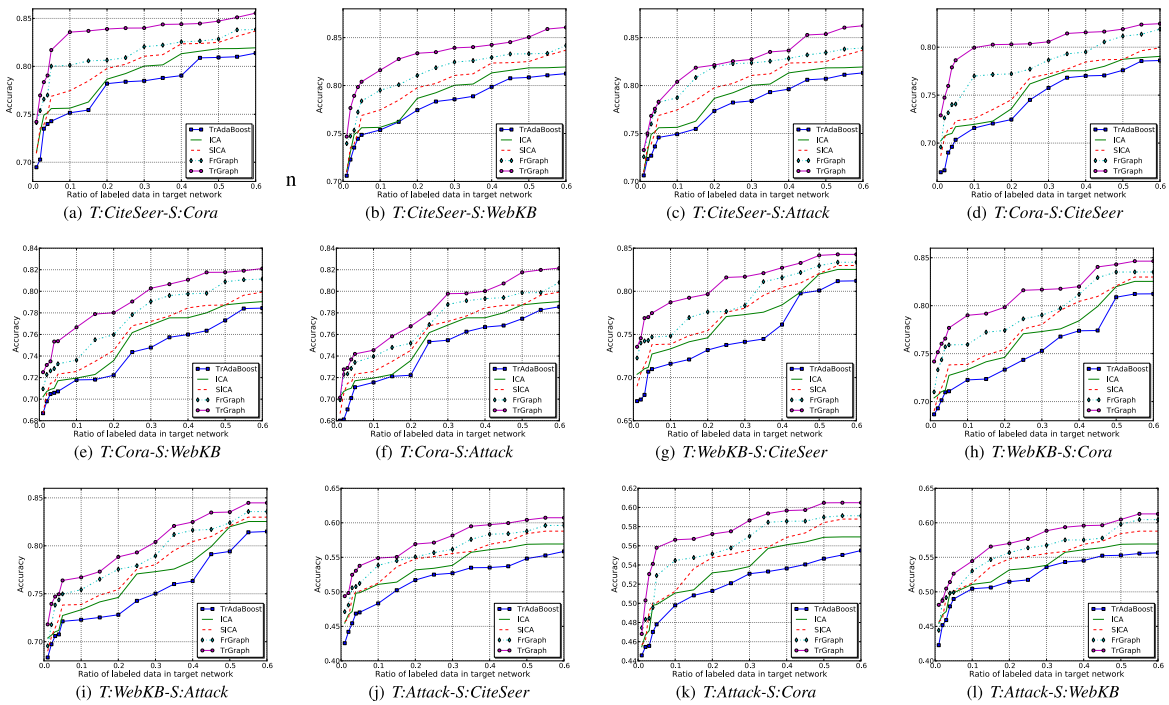


Fig. 6. Accuracy comparison of different algorithms on four data sets with respect to different percentages of the labeled nodes in target networks. "T" indicates target networks and "S" indicates source networks.

shared by the source and target networks are considered as subgraph bases to be transferred.

TrAdaBoost. This is a state-of-the-art transfer learning algorithm that reuses training data from the source domain via instance weighting [24]. To tailor it for networked data, we aggregate the feature spaces of the two networks and combine them with common signature subgraphs, and then build TrAdaBoost on the two networks for node classification. TrAdaBoost inherently ignores the linkages between nodes and directly uses labeled nodes as training instances for transfer learning.

For all the above methods, we use logistic regression as the base classifier and compute the node classification accuracy only on unlabeled target nodes. All ICA-style methods use `proportion` as the aggregation function to compute relational features in the target network.

To provide comprehensive validations for transfer learning, we take turns to consider each data set as the target network and the other three as the source network, respectively. In each setting, we randomly select a fixed percentage p of nodes from the target network as the labeled data according to their node indexes, and the rest of unlabeled nodes are used for testing. For evaluation, we repeat each algorithm three times and report the average classification results.

5.3 Classification Performance

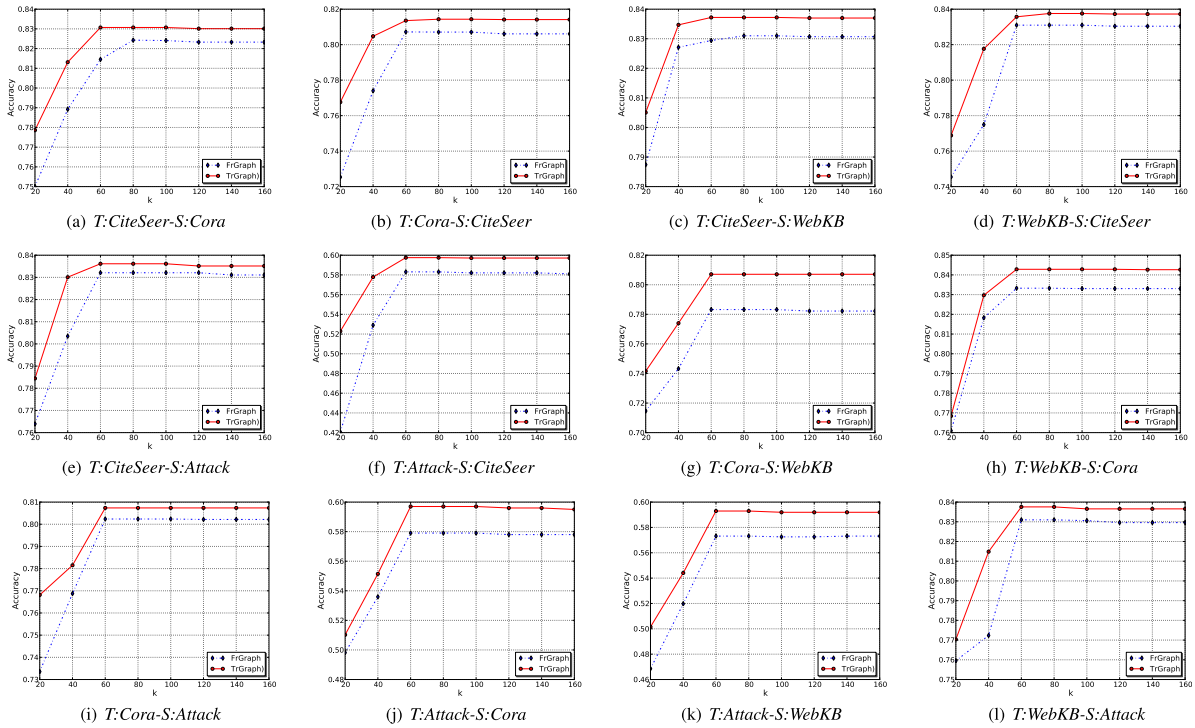
In the first set of experiments, we compare the classification accuracy of different methods with respect to different numbers of the labeled nodes in the target network. We set $k = 60$ indicating the number of the common signature subgraph bases that are transferred between the source and target networks. We vary the percentage p of the labeled nodes in the target network from 2 to 60 percent, and report the classification accuracy in Fig. 6.

We can observe that the proposed algorithm, TrGraph, consistently outperforms other baselines over all transfer learning settings. This confirms that, transferring common signature subgraphs across networks can significantly improve the node classification accuracy in the target network. Meanwhile, FrGraph is inferior to TrGraph, indicating that shared frequent subgraphs are not necessarily good candidates for effective knowledge transfer across networks. For the other two non-transfer-learning methods, SICA yields better performance than ICA. This indicates that, structure features, constructed via finding frequent subgraphs, can indeed help improve the collective classification accuracy. Expectedly, TrAdaBoost does not perform well on the cross-network node classification task. Because the feature spaces of the source and target networks are disparate and they only share common structure features, transfer learning via instance transfer becomes ineffective. Moreover, as TrAdaBoost does not consider label dependency between the nodes, its classification performance further degrades in the network settings.

An important observation is that, the classification accuracy of our TrGraph algorithm is positively correlated to the similarity between the source network and target network. For example, when CiteSeer is used as the target network, using Cora as the source networks yields a higher accuracy than using Attack. This is because both CiteSeer and Cora represent citation relationships between scientific publications. Thus, the two networks share more significant subgraphs in their network structures, which enables transfer learning to be more effective.

5.4 Impact of the Signature Subgraph Number k

In this experiment, we investigate the impact of the number of common signature subgraphs, k, of TrGraph on the node



(a) *T:CiteSeer-S:Cora* (b) *T:Cora-S:CiteSeer* (c) *T:CiteSeer-S:WebKB* (d) *T:WebKB-S:CiteSeer*

(e) *T:CiteSeer-S:Attack* (f) *T:Attack-S:CiteSeer* (g) *T:Cora-S:WebKB* (h) *T:WebKB-S:Cora*

(i) *T:Cora-S:Attack* (j) *T:Attack-S:Cora* (k) *T:Attack-S:WebKB* (l) *T:WebKB-S:Attack*

Fig. 7. Classification accuracy with respect to different numbers of common signature subgraphs (graph bases) k. "T" indicates target networks and "S" indicates source networks.

classification accuracy. The size of subgraph base candidates depends on the inherent characteristics of the network, such as the size of the network, and the value of t. For example, when $t = 1$, only the immediate neighbors of each node are considered to build subgraph bases. The sizes of candidates are 322 for CiteSeer, 267 for Cora, 124 for WebKB and 165 for Attack. For all transfer learning settings, we use 30 percent of nodes in the target network for training, and the rest of unlabeled nodes for testing.

Fig. 7 reports the classification accuracy of TrGraph and FrGraph with respect to different numbers of common signature subgraphs and frequent subgraphs, respectively. We can observe that, for both algorithms, their classification accuracies increase when more subgraphs are selected to be transferred across the networks, and TrGraph consistently outperforms FrGraph for any given number of subgraphs. For TrGraph, we can see that the classification accuracy becomes saturated when the value of k reaches around 60. When the value of k continues to increase, that is, $k > 80$, some discovered subgraph bases are found to be very complex and only appear once in the source network. Consequently, such subgraph bases cannot generalize well to classify unlabeled nodes in the target network.

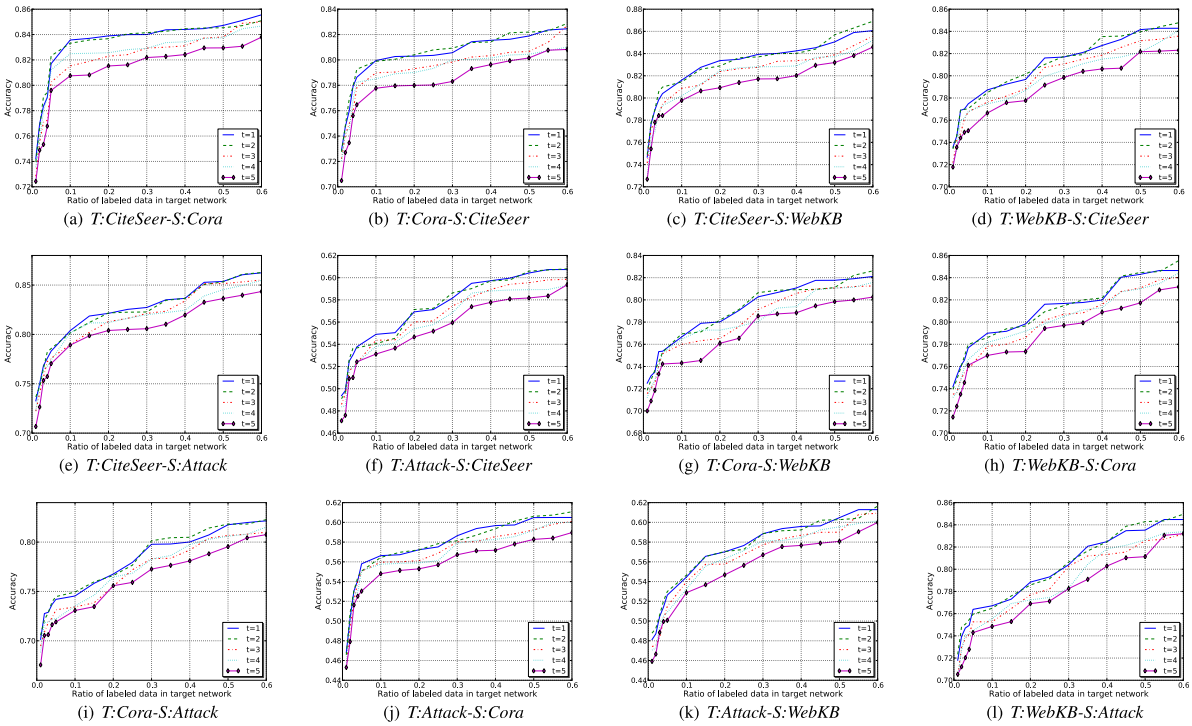
5.5 Impact of the Neighborhood Structure Depth t

We also carry out experiments to study the impact of t, the depth of the t-neighborhood structure for a node, on the node classification accuracy. When constructing t-neighborhood structures for a given node, if $t = 1$, we only consider a node's immediate neighbors. If $t > 1$, we need to recursively crawl the neighbors of a node's neighbors, so the number of nodes may quickly increases and the generated t-neighborhood structures may become more complex. In this set of experiments, we vary the values of depth t from 1 to 5 and generate t-neighborhood structures for different networks. This leads to different numbers of distinct subgraph bases (usually more than 100), so we set $k = 60$ to find common signature subgraph bases between the source and target networks.

Fig. 8 shows the classification accuracy of the proposed TrGraph algorithm with respect to different t values. We can clearly see that, for all the settings, TrGraph achieves the best classification accuracies when $t = 1$ or $t = 2$, although the accuracies for $t = 1$ are slightly better than setting $t = 2$. Also, the accuracies for $t = 3$ and $t = 4$ are better than that of $t = 5$. This shows that, as the value of t becomes larger, the generated subgraph structures would become more specific to the structure of individual networks. Consequently, the subgraph structures that are transferable from the source network are less potentially useful to be generalized in the target network, which limits the contribution of transfer learning to improving the node classification accuracy. From this set of experiments, we conclude that our proposed TrGraph algorithm achieves the highest node classification accuracy when we only consider one or two hop neighbors of the nodes for constructing and transferring structure features. This further confirms that local structure information is not only powerful for jointly classifying the nodes within a network, but also useful for performing cross-network node classification.

5.6 Efficiency Analysis

Lastly, we perform a theoretic and empirical analysis of the time efficiency of the proposed TrGraph algorithm. The most computationally expensive component of TrGraph is the feature transfer process, which consists of two parts:



(a) *T:CiteSeer-S:Cora* (b) *T:Cora-S:CiteSeer* (c) *T:CiteSeer-S:WebKB* (d) *T:WebKB-S:CiteSeer*

(e) *T:CiteSeer-S:Attack* (f) *T:Attack-S:CiteSeer* (g) *T:Cora-S:WebKB* (h) *T:WebKB-S:Cora*

(i) *T:Cora-S:Attack* (j) *T:Attack-S:Cora* (k) *T:Attack-S:WebKB* (l) *T:WebKB-S:Attack*

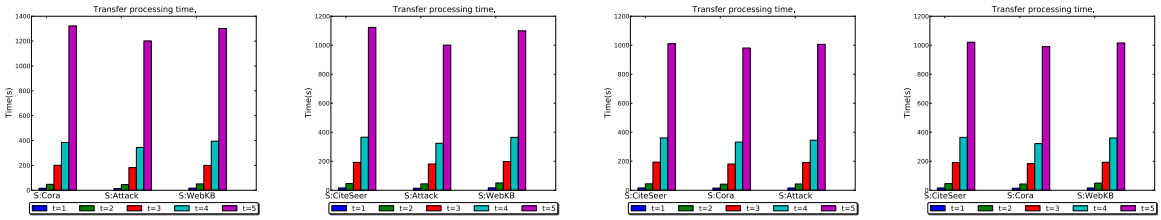
Fig. 8. Classification accuracy with respect to different values of t, which indicates the depth of t-neighborhood structure. "T" indicates target networks and "S" indicates source networks.

constructing the t-neighborhood structures for the nodes in the source and target networks and computing common signature subgraphs between the two networks. For a given node v, the complexity of finding its t-neighborhood structure g is $O(|\mathcal{V}_g| + |\mathcal{E}_g|)$, where $|\mathcal{V}_g|$ and $|\mathcal{E}_g|$ indicate the number of nodes and edges in subgraph g. Considering the total number of nodes in a network is $|\mathcal{V}|$, where $|\mathcal{V}_g| + |\mathcal{E}_g| \ll |\mathcal{V}|$, the time complexity will be far less than $O(|\mathcal{V}|^2)$. On the other hand, when discovering common signature subgraphs, for each subgraph base b_i, computing $\Pr(g|b_i)$ for all nodes in both G_s and G_t has the complexity of $O(n_c(|\mathcal{V}_s| + |\mathcal{V}_t|))$, where n_c is the number of subgraph base candidates. Using all unique subgraph bases from G_s and G_t as candidates, selecting an optimal set of common signature subgraph bases has the time complexity of $O((|\mathcal{V}_s| + |\mathcal{V}_t|)\log(|\mathcal{V}_s| + |\mathcal{V}_t|))$.

Because the value of t for constructing t-neighborhood structures largely affects the time efficiency of our TrGraph algorithm, we empirically study the processing time of the feature transfer component of TrGraph with respect to different t values, and report the results in Fig. 9. It is evident that the running time increases accordingly as the value of t increases. However, because TrGraph achieves the best classification accuracies when only considering local structures in the neighborhood, that is, $t = 1$ or $t = 2$, the feature transfer component is still computationally efficient.

6 Conclusion and Future Work

We presented a novel algorithm to address cross-network transfer learning for node classification. We argued that in network settings, a node classification task in a target network can be significantly improved if the knowledge from other auxiliary networks can be properly leveraged and transferred for learning. In reality, the node feature space and the label space of the networks can be largely (or even completely) different, the generalizable knowledge that can be transferred across networks is common structure patterns between the networks. Therefore, we proposed to construct t-neighborhood structures to represent each node's local structure information. We then solved an optimization problem to discover common signature subgraphs between the networks and used them to reconstruct new structure features of the target network. At last, we proposed an iterative classification algorithm to jointly classify the nodes in



(a) *T:CiteSeer* (b) *T:Cora* (c) *T:WebKB* (d) *T:Attack*

Fig. 9. The processing time of the feature transfer component of our proposed TrGraph algorithm with respect to different values of t, the depth of t-neighborhood structure. "T" indicates target networks and "S" indicates source networks.

the target network. Experiments showed that our proposed algorithm outperforms other baselines and the identified common signature subgraphs can indeed help improve the node classification accuracy.

This work can be extended from several directions. First, we will consider extending our proposed algorithm to intelligently select the "right" source network. Our proposed algorithm attempts to find a set of common significant subgraphs between the source and target networks and use them as features for transfer learning. This provides a potential way to measure the relatedness between two networks by estimating the degree that they share on their significant subgraphs. If two networks are related or similar, they would share most of their significant subgraphs. Otherwise, they would share few significant subgraphs. In this way, negative transfer can be effectively avoided. Second, our current transfer learning framework only considers one source network. We wish to extend it to utilize information from multiple source networks. One potential solution is to assign different weights to the source networks according to their relatedness to the target network. By taking weighted votes from multi-source networks, significant subgraphs are selectively chosen to be transferred to the target network. Finally, because networks can be represented as directed and/or weighted graphs in some applications, we would like to investigate the possibility of identifying directed subgraph patterns [50] for cross-network node classification.

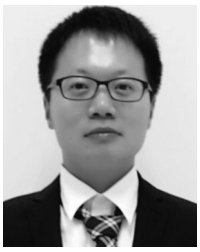
Acknowledgments

The authors thank the editor and the anonymous reviewers for their comments.

References

[1] S. Bhagat, G. Cormode, and S. Muthukrishnan, "Node classification in social networks," *Soc. Netw. Data Anal.*, pp. 115–148, 2011.
[2] C. C. Aggarwal and N. Li, "On node classification in dynamic content-based networks," in *Proc. 11th SIAM Int. Conf. Data Mining*, 2011, pp. 355–366.
[3] M. Pennacchiotti and A.-M. Popescu, "A machine learning approach to Twitter user classification," in *Proc. 5th Int. Conf. Weblogs Social Media*, 2011, pp. 281–288.
[4] P. Kazienko and T. Kajdanowicz, "Label-dependent node classification in the network," *Neurocomputing*, vol. 75, pp. 199–209, 2012.
[5] M. Kim and J. Leskovec, "The network completion problem: Inferring missing nodes and edges in networks," in *Proc. SIAM Int. Conf. Data Mining*, 2011, pp. 47–58.
[6] J. Leskovec, J. Singh, and J. Kleinberg, "Patterns of influence in a recommendation network," in *Proc. 10th Pacific-Asia Conf. Adv. Knowl. Discovery Data Mining*, 2006, pp. 380–389.
[7] H. Xu, Y. Yang, L. Wang, and W. Liu, "Node classification in social network via a factor graph model," in *Proc. Pacific-Asia Conf. Adv. Knowl. Discovery Data Mining*, 2013, pp. 213–224.
[8] J. Leskovec, M. McGlohon, C. Faloutsos, N. S. Glance, and M. Hurst, "Patterns of cascading behavior in large blog graphs," in *Proc. SIAM Int. Conf. Data Mining*, 2007, vol. 7, pp. 551–556.
[9] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[10] A. Evgeniou and M. Pontil, "Multi-task feature learning," in *Proc. Adv. Neural Inform. Process. Syst. 19*, 2007 vol. 19, pp. 41–48.
[11] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 759–766.
[12] T. Yang, R. Jin, A. K. Jain, Y. Zhou, and W. Tong, "Unsupervised transfer classification: application to text categorization," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2010, pp. 1159–1168.
[13] L. P. Cordella, P. Foggia, C. Sansone, and M. Vento, "A (sub) graph isomorphism algorithm for matching large graphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 10, pp. 1367–1372, Oct. 2004.
[14] L. Cordella, P. Foggia, C. Sansone, and M. Vento, "An improved algorithm for matching large graphs," in *Proc. IAPR-TC15 Workshop Graph-Based Representations Pattern Recognit.*, 2001, pp. 149–159.
[15] D. Jensen, J. Neville, and B. Gallagher, "Why collective classification improves relational classification," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery data Mining*, 2004, pp. 593–598.
[16] L. K. McDowell, K. Gupta, and D. Aha, "Cautious collective classification," *J. Mach. Learn. Res.*, vol. 10, pp. 2777–2836, 2009.
[17] H. Eldardiry and J. Neville, "Across-model collective ensemble classification," in *Proc. 25th AAAI Conf. Artif. Intell.*, 2011, pp. 343–349.
[18] R. Kindermann and J. L. Snell, "Markov random fields and their applications," *Am. Math. Soc.*, vol. 1, pp. 1–142, 1980.
[19] B. Taskar, P. Abbeel, and D. Koller, "Discriminative probabilistic models for relational data," in *Proc. 18th Conf. Uncertainty Artif. Intell.*, 2002, pp. 485–492.
[20] J. Neville and D. Jensen, "Iterative classification in relational data," in *Proc. AAAI Workshop Learn. Statist. Models Relational Data*, 2000, pp. 13–20.
[21] M. Bilgic, G. M. Namata, and L. Getoor, "Combining collective classification and link prediction," in *Proc. 7th IEEE Int. Conf. Data Mining Workshops*, 2007, pp. 381–386.
[22] L. K. McDowell, K. Gupta, and D. Aha, "Case-based collective classification," in *Proc. Florida Artif. Intell. Res. Soc. Conf.*, 2007, pp. 399–404.
[23] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, p. 93, 2008.
[24] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 193–200.
[25] W. Dai, G. Xue, Q. Yang, and Y. Yu, "Transferring naive Bayes classifiers for text classification," in *Proc. 22nd Nat. Conf. Artif. Intell.*, 2007, pp. 540–545.
[26] J. Gao, W. Fan, J. Jiang, and J. Han, "Knowledge transfer via multiple model local structure mapping," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 283–291.
[27] J. Blitzer, R. McDonald, and F. Perira, "Domain adaptation with structural correspondence learning," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2006, pp. 120–128.
[28] G. Xue, W. Dai, Q. Yang, and Y. Yu, "Topic-bridged PLSA for cross-domain text classification," in *Proc. 31st Annu. Int. ACM SIGIR Conf. Res. Develop. Inform. Retrieval*, 2008, pp. 627–634.
[29] S. Pan, X. Ni, J. Sun, Q. Yang, and Z. Chen, "Cross-domain sentiment classification via spectral feature alignment," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 751–760.
[30] H. Wang, H. Huang, F. Nie, and C. Ding, "Cross-language web page classification via dual knowledge transfer using nonnegative matrix tri-factorization," in *Proc. 34th Int. ACM SIGIR Conf. Res. Develop. Inform. Retrieval*, 2011, pp. 933–942.
[31] P. Zhao and S. Hoi, "Otl: A framework of online transfer learning," in *Proc. 28th Int. Conf. Mach. Learn.*, 2010, pp. 1231–1238.
[32] S. D. Roy, T. Mei, W. Zeng, and S. Li, "Socialtransfer: Cross-domain transfer learning from social streams for media application," in *Proc. 20th ACM Int. Conf. Multimedia*, 2012, pp. 649–658.
[33] Q. Yang, Y. Chen, G.-R. Xue, W. Dai, and Y. Yu, "Heterogeneous transfer learning for image clustering via the social web," in *Proc. Joint Conf. 47th Annu. Meeting ACL 4th Int. Joint Conf. Natural Lang. Process. AFNLP*, 2009, pp. 1–9.
[34] F. Hu, T. Chen, N. N. Liu, Q. Yang, and Y. Yu, "Discriminative factor alignment across heterogeneous feature space," in *Proc. Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2012, pp. 757–772.
[35] Y. Zhu, Y. Chen, Z. Lu, S. J. Pan, G.-R. Xue, Y. Yu, and Q. Yang, "Heterogeneous transfer learning for image classification," in *Proc. Nat. Conf. Artif. Intell.*, 2011, pp. 1304–1309.

[36] L. Mihalkova, T. Huynh, and R. Mooney, "Mapping and revising Markov logic netowrks for transfer learning," in *Proc. 22nd Nat. Conf. Artif. Intell.*, 2007, pp. 608–614.
[37] J. Davis and P. Domingos, "Deep transfer via second-order Markov logic," in *Proc. AAAI Workshop Transfer Lern. Complex Tasks*, 2008, pp. 217–224.
[38] J. Ye, H. Cheng, Z. Zhu, and M. Chen, "Predicting positive and negative links in signed social networks by transfer learning," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 1477–1488.
[39] M. Fang, J. Yin, and X. Zhu, "Transfer learning across networks for collective classification," in *Proc. IEEE 13th Int. Conf. Data Mining*, 2013, pp. 161–170.
[40] A. Inokuchi, T. Washio, and H. Motoda, "An Apriori-based algorithm for mining frequent substructures from graph data," in *Proc. 4th Eur. Conf. Principles Data Mining Knowl. Discovery*, 2000, pp. 13–23.
[41] X. Yan and J. Han, "gSpan: Graph-based substructure pattern mining," in *Proc. IEEE Int. Conf. Data Mining*, 2002, pp. 721–724.
[42] S. Pan and X. Zhu, "Graph classification with imbalanced class distributions and noise," in *Proc. 23rd Int. Joint Conf. Artif. Intell.*, 2013, pp. 1586–1592.
[43] H. Cheng, X. Yan, J. Han, and C. Hsu, "Discriminative frequent pattern analysis for effective classification," in *Proc. IEEE 23rd Int. Conf. Data Eng.*, 2007, pp. 716–725.
[44] X. Yan, P. S. Yu, and J. Han, "Graph indexing: A frequent structure-based approach," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2004, pp. 335–346.
[45] K. Shearer, H. Bunke, S. Venkatesh, and D. Kieronska, "Efficient graph matching for video indexing," in *Graph Based Representations in Pattern Recognition*. New York, NY, USA: Springer, 1998, vol. 12, pp. 53–62.
[46] E. K. Donald, "The art of computer programming," *Sorting Searching*, vol. 3, pp. 426–458, 1999.
[47] O. Maron, "Learning from ambiguity," Ph.D. dissertation, Massachusetts Inst. Technol., Cambridge, MA, 1998.
[48] H. Bunke, P. Foggia, C. Guidobaldi, C. Sansone, and M. Vento, "A comparison of algorithms for maximum common subgraph on randomly connected graphs," *Structural, Syntactic, Statist. Pattern Recognit.*, vol. 2396, pp. 123–132, 2002.
[49] I. Bomze, M. Budinich, P. Pardalos, and M. Pelillo, "The maximum clique problem," *Handbook Combinatorial Optimization*, vol. 4, pp. 1–74, 1999.
[50] S. Günemann and T. Seidl, "Subgraph mining on directed and weighted graphs," in *Proc. 14th Pacific-Asia Conf. Adv. Knowl. Discovery Data Mining*, 2010, pp. 133–146.



Meng Fang received the BEng and MEng degrees from Wuhan University, China. He is working towards the PhD degree at the Faculty of Engineering and Information Technology, University of Technology, Sydney. His research focuses on data mining and machine learning. He is currently part of the Centre for Quantum Computation & Intelligent Systems. He also works for CSIRO as a research student. Before he was an intern with Mcom Group at Microsoft Research Asia.

Jie Yin received the PhD degree in computer science from the Hong Kong University of Science and Technology. She is now a senior research scientist at Commonwealth Scientific and Industrial Research Organization, Australia. Her research interests include data mining and machine learning, and their applications to text mining, social media mining, and pervasive computing.

Xingquan Zhu received the PhD degree in computer science from Fudan University, Shanghai, China. He is an associate professor in the Department of Computer & Electrical Engineering and Computer Science, Florida Atlantic University. His research interests mainly include data mining, machine learning, and multimedia systems. He is an associate editor of the *IEEE Transactions on Knowledge and Data Engineering* (2014-date). He is a senior member of the IEEE.

Chengqi Zhang received the PhD degree from the University of Queensland, Brisbane, Australia, in 1991 and the DSc degree from Deakin University, Geelong, Australia, in 2002. Since December 2001, he has been a professor of information technology with the University of Technology, Sydney, Australia, where he has been the director of the UTS Priority Investment Research Centre for Quantum Computation & Intelligent Systems since April 2008. He is a senior member of the IEEE.