Meta Diagram based Active Social Networks Alignment

Yuxiang Ren
IFM Lab
Florida State University
Tallahassee, USA
yuxiang@ifmlab.org

Charu C. Aggarwal
IBM Research AI
New York, USA
charu@us.ibm.com

Jiawei Zhang
IFM Lab
Florida State University
Tallahassee, USA
jiawei@ifmlab.org

***Abstract*—Network alignment aims at inferring a set of anchor links matching the shared entities between different information networks, which has become a prerequisite step for effective fusion of multiple information networks. In this paper, we will study the network alignment problem to fuse online social networks specifically. The challenges of the network alignment in social networks mainly come from three perspectives, e.g., *network heterogeneity*, *paucity of training data*, and *one-to-one constraint*. To resolve these challenges, a novel network alignment model, namely Active Iterative Alignment (*ActiveIter*), is introduced in this paper. *ActiveIter* defines a set of *inter-network meta diagrams* for anchor link feature extraction, adopts *active learning* for effective label query and uses *greedy link selection* for anchor link cardinality filtering. Extensive experiments were performed and have demonstrated the effectiveness of *ActiveIter* compared with state-of-the-art baseline methods. A full version of this paper can be accessed in [1].**

***Keywords*-Heterogeneous Network; Network Alignment; Active Learning; Data Mining;**

I. Introduction

The network alignment problem [2, 3] denotes the task of inferring the set of anchor links [4] between the shared information entities in different networks. Network alignment has concrete applications in the real world, which can be applied to discover the set of shared users between different online social networks [2, 4], identify the common protein molecules between different protein-protein-interaction (PPI) networks [3, 5], and find the mappings of POIs (points of interest) across different traffic networks [2]. In this paper, we will use online social networks to elucidate the proposed model.

Online social networks have very complex structures, involving different categories of nodes and links. Users' personal preference may steer their online social activities, and the network structure can provide insightful information for differentiating users between networks. Furthermore, the nodes in online social networks can be also attached with various types of attributes. Based on such an intuition, both the network structure and attribute information should be incorporated in the network alignment model building.

Most of the existing network alignment models are based on supervised learning [4]. Pre-labeled anchor links can provide necessary information for understanding the patterns of aligned user pairs in their information distribution, especially compared with the unsupervised alignment models [2, 3]. However, for the real-world social networks, cross-network anchor link labeling is not an easy task, since it requires tedious user-account pairing and manual user-background checking, which can be very time-consuming and expensive.

In this paper, we propose to study the heterogeneous network alignment problem based on the active learning setting, which is formally referred to the *Active heterogeNeous Network Alignment* (ANNA) problem. The ANNA problem is a novel yet difficult task, and the challenges mainly come from three perspectives, e.g., *network heterogeneity*, *paucity of training data*, and *one-to-one constraint* [6]. To address these challenges, we will introduce a novel network alignment model *Active Iterative Alignment* (*ActiveIter*). To model the diverse information available in social networks, *ActiveIter* adopts the *attributed heterogeneous social network* concept to represent the complex network structure, and a unified feature extraction method based on a novel concept namely *meta diagram* will be utilized in *ActiveIter*. Active learning will be adopted in *ActiveIter* to deal with the paucity of training data. An innovative query strategy is proposed to make sure that *ActiveIter* can select mis-classified false-negative anchor links more precisely. *ActiveIter* can outperform other non-active models with less than 10% of extra training instances which has the additional benefits of reducing the time complexity.

A full version of this paper is available in [1].

II. Concept and Problem Definition

Definition 1 (Attributed Heterogeneous Social Networks): The *attributed heterogeneous social network* can be represented as $G = (\mathcal{V}, \mathcal{E}, \mathcal{T})$, where $\mathcal{V} = \bigcup_i \mathcal{V}_i$ and $\mathcal{E} = \bigcup_i \mathcal{E}_i$ represent the sets of diverse nodes and complex links in the network. The set of attributes associated with nodes in $\mathcal{V}$ can be represented as set $\mathcal{T} = \bigcup_i \mathcal{T}_i$.

For the *attributed heterogeneous social networks* with shared users, they can be represented as the *aligned attributed heterogeneous social networks* (or *aligned social networks* for short).

Definition 2 (Aligned Social Networks): Given online social networks $G^{(1)}$, $G^{(2)}$ sharing common users, they can be represented as the *aligned social networks* $\mathcal{G} = \big((G^{(1)}, G^{(2)}), \mathcal{A}^{(1,2)}\big)$, where $\mathcal{A}^{(1,2)}$ represents the set of undirected anchor links connecting the common users.

Problem Definition: Given a *aligned social networks* $\mathcal{G} = ((G^{(1)}, G^{(2)}), \mathcal{A}^{(1,2)})$, we can represent all the potential anchor links between networks $G^{(1)}$ and $G^{(2)}$ as set $\mathcal{H} = \mathcal{U}^{(1)} \times \mathcal{U}^{(2)}$, where $\mathcal{U}^{(1)}$ and $\mathcal{U}^{(2)}$ denote the user sets in $G^{(1)}$ and $G^{(2)}$ respectively. For the known links, we can group them as a labeled set $\mathcal{L} = \mathcal{A}^{(1,2)}$. The remaining anchor links with unknown labels are those to be inferred, and they can be formally denoted as the unlabeled set $\mathcal{U} = \mathcal{H} \setminus \mathcal{L}$. In the ANNA problem, we aim at building a mapping function $f : \mathcal{H} \to \mathcal{Y}$ to infer anchor link labels in $\mathcal{Y} = \{0, +1\}$ subject to the *one-to-one* constraint, where class labels $+1$ and 0 denote the existing and non-existing anchor links respectively. In the ANNA problem, we are also allowed to query for the label of links in set $\mathcal{U}$ with a pre-specified budget b, i.e., the number of allowed queries. Besides learning the optimal variables in the mapping function $f(\cdot)$, we also aim at selecting an optimal query set $\mathcal{U}_q$ to improve the performance of the learned mapping function $f(\cdot)$ as much as possible.

III. PROPOSED METHOD

A. Meta Diagram based Proximity Features

To effectively categorize the diverse information in the *aligned social networks*, we introduce the *aligned network schema* concept as follows.

Definition 3 (Aligned Social Network Schema): The schema of the aligned social networks $\mathcal{G} = ((G^{(1)}, G^{(2)}), \mathcal{A}^{(1,2)})$ can be represented as $S_{\mathcal{G}} = ((S_{G^{(1)}}, S_{G^{(2)}}), \{\text{anchor}\})$. Here, $S_{G^{(1)}} = (\mathcal{N}_{\mathcal{V}}^{(1)} \cup \mathcal{N}_{\mathcal{T}}, \mathcal{R}_{\mathcal{E}} \cup \mathcal{R}_{\mathcal{A}})$, where $\mathcal{N}_{\mathcal{V}}^{(1)}$ and $\mathcal{N}_{\mathcal{T}}$ denote the set of node types and attribute types in the network, while $\mathcal{R}_{\mathcal{E}}$ represents the set of link types in the network, and $\mathcal{R}_{\mathcal{A}}$ denotes the set of association types between nodes and attributes. In a similar way, we can represent the schema of $G^{(2)}$ as $S_{G^{(2)}} = (\mathcal{N}_{\mathcal{V}}^{(2)} \cup \mathcal{N}_{\mathcal{T}}, \mathcal{R}_{\mathcal{E}} \cup \mathcal{R}_{\mathcal{A}})$.

In the above definition, to simplify the representations, (1) the attribute types have no superscript, since lots of attribute types can be shared across networks; and (2) the relation types also have no superscript, and the network they belong to can be easily differentiated according to the superscript of user/post node types connected to them. Based on the definition, we can represent the Twitter network schema as $S_{G^{(1)}} = (\mathcal{N}^{(1)}, \mathcal{R})$, $\mathcal{N}^{(1)} = \{\text{User}^{(1)}, \text{Post}^{(1)}, \text{Word}, \text{Location}, \text{Timestamp}\}$ (or $\mathcal{N}^{(1)} = \{\text{U}^{(1)}, \text{P}^{(1)}, \text{W}, \text{L}, \text{T}\}$ for short) and $\mathcal{R} = \{\text{follow, write, at, check-in}\}$. The Foursquare network schema has exactly the same representation.

Meta paths [7] may suffer from two major disadvantages. Firstly, meta path cannot characterize rich semantics. For instance, given two users $u_i^{(1)}$ and $u_j^{(2)}$ with check-in records "$u_i^{(1)}$: (Chicago, Aug. 2016), (New York, Jan. 2017), (Los Angeles, May 2017)", and "$u_j^{(2)}$: (Los Angeles, Aug. 2016), (Chicago, Jan. 2017), (New York, May 2017)" respectively, based on meta path P_5 and P_6, user pair $u_i^{(1)}$, $u_j^{(2)}$ have a lot in common and are highly likely to be the same user, since they have either checked-in the same locations (for 3 times) or at the same time (for 3 times). However, their activities are totally "dislocated" as they have never been at the same place for the same moments. Secondly, different meta paths denote different types of connections among users, and assembling them in an effective way is another problem. To solve these two challenges, we introduce *Inter-Network Meta Diagram*.

Definition 5 (Inter-Network Meta Diagram): Give a network schema as $S_{\mathcal{G}} = ((S_{G^{(1)}}, S_{G^{(2)}}), \{\text{anchor}\})$, an *inter-network meta diagram* can be formally represented as a directed acyclic subgraph $\Psi = (\mathcal{N}_\Psi, \mathcal{R}_\Psi, N_s, N_t)$, where $\mathcal{N}_\Psi \subset \mathcal{N}^{(1)} \cup \mathcal{N}^{(2)}$ and $\mathcal{R}_\Psi \subset \mathcal{R} \cup \{anchor\}$ represents the node, attribute and link types involved, while $N_s, N_t \in \{\text{U}^{(1)}, \text{U}^{(2)}\} \wedge N_s \neq N_t$ denote the source and sink user node types from network $G^{(1)}$ and $G^{(2)}$ respectively.

Several *meta diagram* examples have been shown in Table I. The operator $\text{P}_i \times \text{P}_j$ denotes the stacking of meta paths P_i and P_j via the common node types shared by them. *Meta path* [7] actually is a special type of *meta diagram* in the shape of path. More description of the *meta diagram* and the difference among several relating concepts can be seen from the full version of this paper in [1].

Definition 6 (Meta Diagram Proximity): Based on the meta diagram Φ_k, the meta diagram proximity between users $u_i^{(1)}$ and $u_j^{(2)}$ in $\mathcal{G}$ can be represented as

$$s_{\Phi_k}(u_i^{(1)}, u_j^{(2)}) = \frac{2|\mathcal{P}_{\Phi_k}(u_i^{(1)}, u_j^{(2)})|}{|\mathcal{P}_{\Phi_k}(u_i^{(1)}, \cdot)| + |\mathcal{P}_{\Phi_k}(\cdot, u_j^{(2)})|}.$$

where $\mathcal{P}_{\Phi_k}(u_i^{(1)}, u_j^{(2)})$ is the set of meta diagram Φ_k instances connecting $u_i^{(1)}$ and $u_j^{(2)}$, and $\mathcal{P}_{\Phi_k}(u_i^{(1)}, \cdot)$ is the set of Φ_k instances going out from user $u_i^{(1)}$.

B. Active Network Alignment Model

Formally, the feature vector extracted for anchor link $l \in \mathcal{H}$ can be represented as vector $\mathbf{x}_l \in \mathbb{R}^d$ (parameter d denotes the feature size). Meanwhile, we can denote the label of link $l \in \mathcal{L}$ as $y_l \in \mathcal{Y}$ ($\mathcal{Y} = \{0, +1\}$), which denotes the existence of anchor link l between the networks. For the existing anchor links in set $\mathcal{L}_+$, they will be assigned with $+1$ label; while the labels of anchor links in $\mathcal{U}$ are unknown.

The *discriminative* component can effectively differentiate the positive instances from the non-existing ones, which can be denoted as mapping $f(\cdot; \theta_f) : \mathbb{R}^d \to \{+1, 0\}$ parameterized by θ_f. A linear model is used to fit the link instances, and the *discriminative* model to be learned can be represented as $f(\mathbf{x}_l; \mathbf{w}) = \mathbf{w}^\top \mathbf{x}_l + b$, and $\theta_f = [\mathbf{w}, b]$. By incorporating the bias term b into the weight vector $\mathbf{w}$, the *discriminative* loss function on the labeled set $\mathcal{L}_+$ is:

$$L(f, \mathcal{L}_+; \mathbf{w}) = \sum_{l \in \mathcal{L}_+} \big(f(\mathbf{x}_l; \mathbf{w}) - y_l\big)^2 = \sum_{l \in \mathcal{L}_+} (\mathbf{w}^\top \mathbf{x}_l - y_l)^2.$$

For a unlabeled anchor link $l \in \mathcal{U}$, we can represent its inferred "label" as $y_l = f(\mathbf{x}_l; \mathbf{w})$. In the *generative* component, we can represent the generated anchor link label

Table I
SUMMARY OF INTER-NETWORK META DIAGRAMS.

ID	Notation	Meta Diagram	Semantics
P_1	$\mathrm{U} \rightarrow \mathrm{U} \leftrightarrow \mathrm{U} \leftarrow \mathrm{U}$	User $\xrightarrow{follow}$ User $\xleftrightarrow{anchor}$ User $\xleftarrow{follow}$ User	Common Anchored Followee
P_2	$\mathrm{U} \leftarrow \mathrm{U} \leftrightarrow \mathrm{U} \rightarrow \mathrm{U}$	User $\xleftarrow{follow}$ User $\xleftrightarrow{anchor}$ User $\xrightarrow{follow}$ User	Common Anchored Follower
P_3	$\mathrm{U} \rightarrow \mathrm{U} \leftrightarrow \mathrm{U} \rightarrow \mathrm{U}$	User $\xrightarrow{follow}$ User $\xleftrightarrow{anchor}$ User $\xrightarrow{follow}$ User	Common Anchored Followee-Follower
P_4	$\mathrm{U} \leftarrow \mathrm{U} \leftrightarrow \mathrm{U} \leftarrow \mathrm{U}$	User $\xleftarrow{follow}$ User $\xleftrightarrow{anchor}$ User $\xleftarrow{follow}$ User	Common Anchored Follower-Followee
P_5	$\mathrm{U} \rightarrow \mathrm{P} \rightarrow \mathrm{T} \leftarrow \mathrm{P} \leftarrow \mathrm{U}$	User $\xrightarrow{write}$ Post $\xrightarrow{at}$ Timestamp $\xleftarrow{at}$ Post $\xleftarrow{write}$ User	Common Timestamp
P_6	$\mathrm{U} \rightarrow \mathrm{P} \rightarrow \mathrm{L} \leftarrow \mathrm{P} \leftarrow \mathrm{U}$	User $\xrightarrow{write}$ Post $\xrightarrow{checkin}$ Location $\xleftarrow{checkin}$ Post $\xleftarrow{write}$ User	Common Checkin
$\Psi_1(\mathrm{P}_1 \times \mathrm{P}_2)$	$\mathrm{U} \leftrightarrow \mathrm{U} \xleftrightarrow{anchor} \mathrm{U} \leftrightarrow \mathrm{U}$	$\begin{tikzcd} \text{User} \arrow[r, bend left, "follow"] & \text{User} \arrow[l, bend left, "follow"] \arrow[r, leftrightarrow, "anchor"] & \text{User} \arrow[r, bend right, "follow"'] & \text{User} \arrow[l, bend right, "follow"'] \end{tikzcd}$	Common Aligned Neighbors
$\Psi_2(\mathrm{P}_5 \times \mathrm{P}_6)$	$\begin{tikzcd} & & \mathrm{L} & & \\ \mathrm{U} \arrow[r] & \mathrm{P} \arrow[ur] \arrow[dr] & & \mathrm{P} \arrow[ul] \arrow[dl] & \mathrm{U} \arrow[l] \\ & & \mathrm{T} & & \end{tikzcd}$	$\begin{tikzcd} & & \text{Location} & & \\ \text{User} \arrow[r, "write"] & \text{Post} \arrow[ur, "checkin"] \arrow[dr, "at"'] & & \text{Post} \arrow[ul, "checkin"'] \arrow[dl, "at"] & \text{User} \arrow[l, "write"'] \\ & & \text{Timestamp} & & \end{tikzcd}$	Common Attributes
$\Psi_3(\mathrm{P}_1 \times \mathrm{P}_5 \times \mathrm{P}_6)$	$\begin{tikzcd} \mathrm{U} \arrow[rrrr, leftrightarrow] & & & & \mathrm{U} \\ & & \mathrm{L} & & \\ \mathrm{U} \arrow[uu] \arrow[r] & \mathrm{P} \arrow[ur] \arrow[dr] & & \mathrm{P} \arrow[ul] \arrow[dl] & \mathrm{U} \arrow[uu] \arrow[l] \\ & & \mathrm{T} & & \end{tikzcd}$	$\begin{tikzcd} & \text{User} \arrow[rr, leftrightarrow, "anchor"] & & \text{User} & \\ & & \text{Location} & & \\ \text{User} \arrow[uur, "follow"] \arrow[r, "write"] & \text{Post} \arrow[ur, "checkin"] \arrow[dr, "at"'] & & \text{Post} \arrow[ul, "checkin"'] \arrow[dl, "at"] & \text{User} \arrow[uul, "follow"'] \arrow[l, "write"'] \\ & & \text{Timestamp} & & \end{tikzcd}$	Common Aligned Neighbor & Attributes

as $sign\big(f(\mathbf{x}_l;\mathbf{w})\big) \in \{+1, 0\}$. Furthermore, a subset of the anchor links in $\mathcal{U}$ will be selected to query for the labels, which can be denoted as set $\mathcal{U}_q$. The true label of anchor link $l \in \mathcal{U}_q$ after query can be represented as $\tilde{y}_l \in \{+1, 0\}$. Depending on whether the labels of links are queried or not, we can specify the loss function for set $\mathcal{U}$ as

$$\begin{aligned} & L(f, \mathcal{U}; \mathbf{w}) = L(f, \mathcal{U}_q; \mathbf{w}) + L(f, \mathcal{U} \setminus \mathcal{U}_q; \mathbf{w}) \\ & = \sum_{l \in \mathcal{U}_q} (\mathbf{w}^\top \mathbf{x}_l - \tilde{y}_l)^2 + \sum_{l \in \mathcal{U} \setminus \mathcal{U}_q} \Big(\mathbf{w}^\top \mathbf{x}_l - sign\big(f(\mathbf{x}_l;\mathbf{w})\big)\Big)^2. \end{aligned}$$

The anchor links to be inferred between networks are subject to the *one-to-one* cardinality constraint[6]. The cardinality constraint on anchor links should be effectively incorporated in model building, which will be modeled as the mathematical constraints on node degrees in this paper. To represent the user node-anchor link relationships in networks $G^{(1)}$ and $G^{(2)}$ respectively, we introduce the user node-anchor link incidence matrices $\mathbf{A}^{(1)} \in \{0,1\}^{|\mathcal{U}^{(1)}| \times |\mathcal{H}|}$, $\mathbf{A}(2) \in \{0,1\}^{|\mathcal{U}^{(2)}| \times |\mathcal{H}|}$. Entry $A^{(1)}(i,j) = 1$ iff anchor link $l_j \in \mathcal{H}$ is connected with user node $u_i^{(1)}$ in $G^{(1)}$, and it is similar for $A^{(2)}$. We can represent the labels of links in $\mathcal{H}$ as vector $\mathbf{y} \in \{+1, 0\}^{|\mathcal{H}|}$, where entry $y(i)$ represents the label of link $l_i \in \mathcal{L}$. Therefore, the *one-to-one* constraint on anchor links can be denoted as follows:

$$\mathbf{0} \leq \mathbf{A}^{(1)}\mathbf{y} \leq \mathbf{1}, \text{ and } \mathbf{0} \leq \mathbf{A}^{(2)}\mathbf{y} \leq \mathbf{1}.$$

By combining the loss terms introduced by the labeled, queried and remaining unlabeled anchor links together with the cardinality constraint, we can represent the joint optimization objective function as

$$\begin{aligned} & L(f, \mathcal{L}_+; \mathbf{w}) + \alpha \cdot L(f, \mathcal{U}_q; \mathbf{w}) + \beta \cdot L(f, \mathcal{U} \setminus \mathcal{U}_q; \mathbf{w}) \\ & = L(f, \mathcal{H}; \mathbf{w}) = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2, \end{aligned}$$

where $\mathbf{X} = [\mathbf{x}_{l_1}^\top, \mathbf{x}_{l_2}^\top, \cdots, \mathbf{x}_{l_{|\mathcal{H}|}}^\top]^T$ denotes the feature matrix of all the links in set $\mathcal{H}$. We set the weight scalar α and β as 1, because we assume that each link is equally important for training. In this paper, we design an hierarchical alternative variable updating process for solving the problem:

- **External Iteration Step (1)**: Fix $\mathcal{U}_q$, Update $\mathbf{y}$, $\mathbf{w}$.
  - ■ **Internal Iteration Step (1-1)**: Fix $\mathcal{U}_q$, $\mathbf{y}$, Update $\mathbf{w}$.

With $\mathbf{y}$, $\mathcal{U}_q$ fixed, the objective function is

$$\min_{\mathbf{w}} \frac{c}{2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 + \frac{1}{2} \|\mathbf{w}\|_2^2.$$

The optimal solution of the objective function is:

$$\mathbf{w} = \mathbf{H}\mathbf{y} = c(\mathbf{I} + c\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y},$$

where $\mathbf{H} = c(\mathbf{I} + c\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$ is a constant matrix.

■ **Internal Iteration Step (1-2)**: Fix $\mathcal{U}_q$, $\mathbf{w}$, Update $\mathbf{y}$.
With $\mathcal{U}_q$, $\mathbf{w}$ fixed, the objective function is

$$\begin{aligned} & \min_{\mathbf{y}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \\ & s.t.\ y_l \in \{+1, 0\}, \forall l \in \mathcal{U} \setminus \mathcal{U}_q, \\ & \quad y_l = \tilde{y}_l, \forall l \in \mathcal{U}_q \text{ and } y_l = +1, \forall l \in \mathcal{L}_+, \\ & \quad \mathbf{0} \leq \mathbf{A}^{(1)}\mathbf{y} \leq \mathbf{1}, \text{ and } \mathbf{0} \leq \mathbf{A}^{(2)}\mathbf{y} \leq \mathbf{1}. \end{aligned}$$

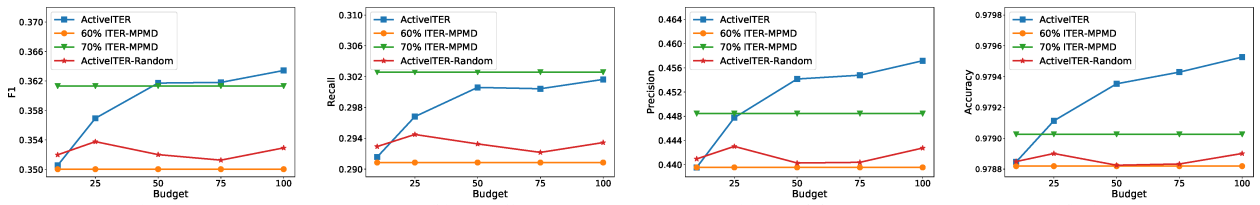
In this paper, we will use the greedy link selection algorithm proposed in [6] based on values $\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}$.

- **External Iteration Step (2)**: Fix $\mathbf{w}$, $\mathbf{y}$, Update $\mathcal{U}_q$.

Instead of selecting the optimal set $\mathcal{U}_q$ at one time, we propose to choose several link instances greedily in each iterations. Due to the one-to-one constraint, the unlabeled anchor links no longer bears equal information, and querying for labels of potential positive anchor links will be more "informative" compared with negative anchor links. Among the unlabeled links, *ActiveIter* selects a set of mis-classified false-negative anchor links (but with a large positive score) as the potential candidates, benefits introduced by whose label queries includes both their own label corrections and other extra label gains of their conflicting negative links at the same time. Formally, among all the unlabeled links in $\mathcal{U}$, we can represent the set of links classified to be positive/negative instances in the previous iteration step as $\mathcal{U}^+ = \{l | l \in \mathcal{U}, y_l = +1\}$ and $\mathcal{U}^- = \{l | l \in \mathcal{U}, y_l = 0\}$. Based on these two sets, the group of potentially mis-classified false-negative anchor link candidates as set

$$\begin{aligned} \mathcal{C} = \{l | l \in \mathcal{U}^-, \exists l', l'' \in \mathcal{U}^+ & \text{ that conflicts with } l, \\ & \hat{y}_{l'} \sim \hat{y}_l \gg \hat{y}_{l''} > 0\}, \end{aligned}$$

where statement "l'/l'' conflicts with l" denotes l'/l'' and l are incident to the same nodes respectively. Operator $\hat{y}_{l'} \sim \hat{y}_l$ represents $\hat{y}_{l'}$ is close to $\hat{y}_l$. All the links in set $\mathcal{C}$ will be



(a) F1 (b) Recall (c) Precision (d) Accuracy

Figure 1. Performance Analysis under different budgets when $\theta = 50$ and $\gamma = 60\%$ for *ActiveIter* and *ActiveIter-Rand*.

Table II
PERFORMANCE COMPARISON OF DIFFERENT METHODS FOR NETWORK ALIGNMENT UNDER DIFFERENT SAMPLE-RATIOS

		Sample Ratio γ				
metrics	methods	60%	70%	80%	90%	100%
F1	*ActiveIter-100*	**0.363**	**0.369**	**0.397**	**0.404**	**0.422**
	ActiveIter-50	0.361	0.362	0.396	0.399	0.410
	ActiveIter-Rand-50	0.352	0.360	0.383	0.391	0.402
	Iter-MPMD	0.350	0.361	0.385	0.387	0.400
	SVM-MPMD	0.049	0.082	0.090	0.092	0.131
	SVM-MP	0.000	0.000	0.000	0.000	0.000
Precision	*ActiveIter-100*	**0.457**	**0.460**	**0.491**	**0.499**	**0.518**
	ActiveIter-50	0.450	0.450	0.489	0.492	0.503
	ActiveIter-Rand-50	0.440	0.447	0.471	0.480	0.493
	Iter-MPMD	0.439	0.448	0.474	0.475	0.489
	SVM-MPMD	0.311	0.343	0.424	0.361	0.449
	SVM-MP	0.000	0.000	0.000	0.000	0.000
Recall	*ActiveIter-100*	**0.301**	**0.308**	**0.334**	**0.339**	**0.356**
	ActiveIter-50	0.300	0.303	0.333	0.336	0.347
	ActiveIter-Rand-50	0.293	0.302	0.322	0.330	0.340
	Iter-MPMD	0.290	0.302	0.322	0.327	0.338
	SVM-MPMD	0.027	0.047	0.056	0.053	0.077
	SVM-MP	0.000	0.000	0.000	0.000	0.000
Accuracy	*ActiveIter-100*	0.979	0.979	**0.980**	**0.980**	**0.981**
	ActiveIter-50	0.979	0.979	0.979	0.980	0.980
	ActiveIter-Rand-50	0.978	0.979	0.979	0.979	0.980
	Iter-MPMD	0.978	0.979	0.979	0.979	0.980
	SVM-MPMD	0.980	0.980	0.980	0.980	0.980
	SVM-MP	**0.980**	**0.980**	0.980	0.980	0.980

sorted according to value $\hat{y}_l - \hat{y}_{l''}$, and, instead of adding one by one, the top k candidates will be added to $\mathcal{U}_q$ in this iteration. Here, k denotes the query batch size.

IV. Experiments

The dataset used in experiments consists of two heterogeneous networks: Foursquare and Twitter. The methods in experiments include SVM-MP, SVM-MPMD, *Iter-MPMD*, *ActiveIter-Rand*, and *ActiveIter*, and we will use F1, Recall, Precision and Accuracy as evaluation metrics. 10-folds cross validation is utilized in experiments, where 1 fold is used as the training set. In order to simulate the setting without enough labeled data, we further sample a small proportion of instances from the 1-fold training set as the final training set. The sampling process is controlled by sample-ratio γ.

In Table II, *ActiveIter-50* denotes *ActiveIter* with 50 query budget, and *ActiveIter-100* has a query budget of value 100. The comparison between SVM-MP and SVM-MPMD shows the effectiveness of the inter-network meta diagram. Furthermore, we can make comparison between *ActiveIter-100* with certain γ and *Iter-MPMD* with $\gamma + 10\%$, where *Iter-MPMD* uses additional $1,670$ training instances, while *ActiveIter-100* merely queries for additional 100 instances. According to Table II, *ActiveIter-100* with far less training data can still outperform *Iter-MPMD* with great advantages in most of the cases.

The effects of the parameter budget b can be seen in Figure 1. It shows that *ActiveIter* can achieve better prediction results consistently along with querying critical labels continuously, but *ActiveIter-Rand* can not improve prediction output with random labels. Besides, with far less (less than 100 additional) training instances, method *ActiveIter* based on active learning can achieve comparable and even better results than the non-active method *Iter-MPMD* with 1,670 extra training instances. More experimental results are available in [1].

V. Conclusion

In this paper, we study the ANNA problem and propose an active learning model *ActiveIter* based on meta diagrams to solve this problem. Meta diagrams can be extracted from the network to constitute heterogeneous features. In the active learning model *ActiveIter*, we propose an innovative query strategy in the selection process to in order to query for the optimal unlabeled links. Extensive experiments conducted on two real-world networks demonstrate that *ActiveIter* has very outstanding performance compared with the state-of-the-art baseline methods.

Acknowledgment

This work is partially supported by FSU and by NSF through grant IIS-1763365.

References

[1] Y. Ren, C. Aggarwal, and J. Zhang, "Meta diagram based active social networks alignment," *arXiv:1902.04220*, 2019.

[2] J. Zhang and P. Yu, "Multiple anonymized social networks alignment," in *ICDM*, 2015.

[3] J. Flannick, A. Novak, B. Srinivasan, H. McAdams, and S. Batzoglou, "Graemlin: general and robust alignment of multiple large interaction networks." *Genome research*, 2006.

[4] X. Kong, J. Zhang, and P. Yu, "Inferring anchor links across multiple heterogeneous social networks," in *CIKM*, 2013.

[5] R. Singh, J. Xu, and B. Berger, "Pairwise global alignment of protein interaction networks by matching neighborhood topology," in *RECOMB*, 2007.

[6] J. Zhang, J. Chen, J. Zhu, Y. Chang, and P. Yu, "Link prediction with cardinality constraint," in *WSDM*, 2017.

[7] Y. Sun, J. Han, X. Yan, P. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *PVLDB*, 2011.