CoLink: An Unsupervised Framework for User Identity Linkage

Zexuan Zhong[†] [*] , Yong Cao[‡], Mu Guo[‡] and Zaiqing Nie[§]
[‡]Microsoft Research, China
[†]University of Illinois at Urbana-Champaign, USA
[§]Alibaba AI Labs, China
zexuan2@illinois.edu, {yongc, muguo}@microsoft.com, zaiqing.nzq@alibaba-inc.com

Abstract

Nowadays, it is very common for one person to be in different social networks. Linking identical users across different social networks, also known as the User Identity Linkage (UIL) problem, is fundamental for many applications. There are two major challenges in the UIL problem. First, it's extremely expensive to collect manually linked user pairs as training data. Second, the user attributes in different networks are usually defined and formatted very differently which makes attribute alignment very hard. In this paper we propose CoLink, a general unsupervised framework for the UIL problem. CoLink employs a co-training algorithm, which manipulates two independent models, the attribute-based model and the relationship-based model, and makes them reinforce each other iteratively in an unsupervised way. We also propose the sequence-to-sequence learning as a very effective implementation of the attribute-based model, which can well handle the challenge of the attribute alignment by treating it as a machine translation problem. We apply CoLink to a UIL task of mapping the employees in an enterprise network to their LinkedIn profiles. The experiment results show that CoLink generally outperforms the state-of-the-art unsupervised approaches by an F1 increase over 20%.

Introduction

Social network services (SNS) play a very important role in people's everyday life. It's very common that one real-world person appears in different SNS, including public social networks on the web, such as LinkedIn and Facebook, and private social networks such as employee networks inside an enterprise. Linking the user identities among different social networks, a.k.a the UIL problem, is generally required to get better and deeper understanding of individual users which usually results in better business intelligence. For example, linking different social network users that refer to the same person can help create an integrated person profile (Zhang et al. 2015); understand user migration patterns in social media (Kumar, Zafarani, and Liu 2011); and help social network websites recommend potential friends more precisely.

Specifically, linking the enterprise users to their public social network profiles brings huge opportunities for improving enterprise effectiveness, as it connects the public knowledge to the enterprise knowledge, which are usually complementary. For example, the relationships and life-related profiles in one's public social networks are usually missing from his/her enterprise profile. Such information may greatly help different enterprise scenarios including internal collaboration, talent hiring, and cross-enterprise businesses.

Although machine learning algorithms have been widely used in the UIL problem, the effort of training data annotation is not trivial. Firstly, finding linked user pairs is extremely time-consuming since it requires one to search over the entire networks and carefully evaluate a large amount of candidate pairs. It also requires the human labellers to have a wide range of domain knowledge, for example, one has to know that "SDE" is the acronym for "Software Development Engineer" before he/she starts to work on the users who have software development backgrounds. Secondly, not all the social network profile data can be exposed to human labellers for privacy protection reasons, especially when the profile comes from internal enterprise networks.

Researchers have proposed unsupervised approaches for the UIL problem which requires no labelled data. Such solutions are either heuristic approaches or machine learning based approaches with auto-generated training data. Lacoste-Julien et al. (2013) propose a greedy approach which updates the relationship similarity iteratively. The attributes are aligned with heuristic string similarity functions. However, the selected string similarity functions are very much sensitive to the profile data, which makes them hard to generalize. Liu et al. (2013) try to automatically collect a training data set based on the rareness of the user names from the forum data, and then employ Supported Vector Machine (SVM) to classify whether two users are linked or not. The performance very much depends on the quantity and quality of the auto-generated training data which is sensitive to the targeted UIL tasks. There is lack of general unsupervised solutions in the literature.

Linking users between two social networks requires to carefully compare the user attributes in both networks, such as name, job title, location etc. As a result, aligning the attribute values is essential to the UIL problem. The attribute alignment is usually handled with string similarity comparison. For example, Kong, Zhang, and Yu (2013) convert the attribute value text into bag-of-words vectors with

[*]This work was done when the first author was on an internship with Microsoft Research.

TF-IDF weights, and then compute the similarity with the inner product and the cosine similarity. Lacoste-Julien et al. (2013) use a smoothed weighted Jaccard similarity for the property scores. Besides, attribute comparison are also widely studied in the Entity Resolution literature, which aims to link entries in a database. For example, Soft TF-IDF similarity (Cohen, Ravikumar, and Fienberg 2003) has been shown to perform well when comparing name-based entities. Bilenko and Mooney (2003) proposed a SVM based similarity which can be learned from bag-of-words vectors of matched string pairs. However, the traditional string similarity functions shares the following two weaknesses.

- No general approach for attribute variations. For the same attribute field, its values could be very different in different social networks. Taking the job title as an example, "SR CONSULTANT PROD" has a corresponding variation "Senior Consultant"; "ESCAL ENG" equals to "Escalation Engineer". The variations follow different patterns, such as acronyms, abbreviations, synonyms and translations, in different social networks. Traditional string similarity functions can only cover some patterns, but never all. There is no general way to handle alignment among different variations.
- Unable to find implicit connections. Sometimes catching the implicit connections between different attributes is required. For example, the office location "SUNNYVALE-1020/6221" in one's enterprise profile has strong correlation to the location attribute "San Francisco Bay Area" in his/her LinkedIn profile, although there is almost nothing in common in string-level comparison. Traditional string similarity functions won't apply in such scenarios.

In this paper, we propose CoLink, a general unsupervised framework for the UIL problem. The social network profile data can be naturally split into two independent views: attributes and relationships, which satisfy the requirement of the co-training algorithm (Blum and Mitchell 1998) perfectly. CoLink employs two independent models, an attribute-based model and a relationship-based model, and a co-training algorithm to reinforce them in an iterative way. Both the attribute-based model and the relationship-based model are binary classifiers that decide whether two users are linked or not. They can be grounded with any machine learning or heuristic algorithms. Therefore, CoLink as a framework can be applied to any UIL problem as long as the user profiles contain attributes and relationships.

We further employ the sequence-to-sequence learning algorithm (Sutskever, Vinyals, and Le 2014) ("*sequence-to-sequence*" in short) in the implementation of the attribute-based model of CoLink, which provides a general solution for aligning attributes among different social networks. Instead of treating the attribute alignment as string similarity comparison, we actually try to "translate" the attribute value from one "language", the specific style of one network, to the other. Acronyms, abbreviations, synonyms and even implicit connections are all regarded as special translations. We choose *sequence-to-sequence* as it has demonstrated to be effective for machine translation (Wu et al. 2016). Specifically, *sequence-to-sequence* has two advantages for CoLink. First, it automatically captures the word-level mapping and sequence-level mapping with almost no feature engineering; Second, it only requires positive examples (aligned attribute pairs) as training data which relaxes the effort of sampling negative examples.

We apply CoLink to an UIL application where we try to link the employees in an enterprise network to their LinkedIn profiles. We enumerate different settings and different model implementations to show the robustness of CoLink. We further compare CoLink with the state-of-the-art unsupervised approaches. The experiment results show that CoLink generally outperforms the state-of-the-art unsupervised approaches by an F1 increase of 20%. We summarize our contributions as follows.

- We are the first to employ the co-training algorithm for the UIL problem. User attributes and relationships in the social network profile are naturally separated views, which makes co-training a perfect and cost-free solution.
- We are the first to model the attribute alignment problem as machine translation. We ground the attribute-based model with *sequence-to-sequence* which can be easily generalized with almost no feature engineering.
- We conduct extensive experiments to show the effectiveness of the proposed solution by comparing to the state-of-the-art unsupervised approaches, enumerating different settings and models.

CoLink

We present the details of the CoLink in this section, including the co-training algorithm, the seed rules for starting the co-training algorithm and the implementations of the attribute-based model and the relationship-based model.

Problem Definition

Let $G = \{V, E, A\}$ denote a social network, where $V = \{v_1, v_2, \dots, v_N\}$ is a set of vertices, representing all users in the network; $E \subseteq V \times V$ is a set of edges among the vertices, representing the relationships among users; A is a $N \times d$ user attribute matrix, where d is the number of the attribute fields. The UIL problem is defined as follows.

Input: The input includes a source social network $G^s = \{V^s, E^s, A^s\}$ and a target social network $G^t = \{V^t, E^t, A^t\}$

Output: The output is a set of user pairs $S \subseteq V^s \times V^t$, representing the linked user pairs from the source network to the target network.

Here we assume that there is no user duplication in both the source network and the target network, i.e. there does not exist any two users inside one network which link to one real-world person. Therefore, the linked pair set S must obey the *one-to-one* linkage constraint, i.e. there are no two pairs in S share a same vertex.

CoLink Framework

The CoLink framework is based on the co-training algorithm as shown in Algorithm 1. We define two separate models in the framework: an attribute-based model f_{att} and a relationship-base model f_{rel}. Both models make the binary classification predictions, classifying a given user pair to be positive (linked) or negative (unlinked). The co-training algorithm reinforce the two models in an iterative way. At each co-training iteration, both models are retrained with the linked pair set S. The high-quality linked pairs generated with both models are then merged into S for the next iteration until S converges. At the very beginning, an initial linked pair set, seed set in short, is required to kick off the co-training process, which can be generated with a set of seed rules. The training of the attribute-based model and the relationship-based model may require negative examples depending on what algorithms they employ. The process of sampling negative examples is not included in Algorithm 1.

```
   Input: a source social network $G^s$, a target social
          network $G^t$
   Output: a set of user pairs $S$
 1 $S \leftarrow$ the set of seed pairs generated with seed rules;
 2 repeat
 3     /* generate pairs from attribute-based model */
 4     $D_{att} \leftarrow f_{att}(S, G^s, G^t)$ ;
 5     /* generate pairs from relationship-based model */
 6     $D_{rel} \leftarrow f_{rel}(S, G^s, G^t)$ ;
 7     /* join two sets and remove conflicting pairs */
 8     $D \leftarrow merge(D_{att}, D_{rel})$ ;
 9     $S \leftarrow S \cup D$ ;
10 until $D = \emptyset$;
11 return $S$ ;
```

Algorithm 1: The co-training algorithm in CoLink.

The co-training algorithm won't revise the linked pairs generated from the previous iterations. Therefore the errors brought from early iterations won't be fixed later. An alternative to this algorithm is to do a final revision after the co-training converges. That is, reconstructing S with the final models generated from the co-training process. We will discuss the results with and without the final revision in the experiment section.

Seed Rules

In order to kick off the co-training algorithm, a small seed set of linked user pairs are required. A straight-forward way to get the seed set is to generate it with hand-crafted rules, which we call seed rules. These seed rules may consider the following facts from the targeted social networks.

- user name uniqueness; If a source network user and a target network user have identical user name which are unique in both networks, the two users are very much likely to represent the same person.
- attribute value mapping; Although it's not applicable to collect all possible attribute mappings across two networks, it's reasonable to find one or two value mappings.
- relationship propagation; If two users are linked, it is highly probable to find linked user pairs in their related users.

The choice of seed rules will directly impact CoLink performance. It's natural that fine-tuned high-quality seed rules have positive impact while coarse-tuned noisy seed rules have negative impact. The robustness of CoLink lies on how sensitive it is to the seed rules, which will be discussed later in the experiment section.

Attribute-based Model

The attribute-based model predicts the linked user pairs by only considering the user attributes. It can utilize any classification algorithm. In this paper, we try two different machine learning algorithms: *sequence-to-sequence* and SVM.

Sequence-to-sequence The traditional string similarity approaches can poorly handle attribute alignment due to their different variations. We use *sequence-to-sequence* because the attribute alignment is similar to a machine translation problem. Acronyms, abbreviations, synonyms and even implicit connections are all special cases of translation.

We employ the *sequence-to-sequence* network structure that was proposed by Sutskever, Vinyals, and Le (2014). The network consists of two parts: the sequence encoder and the sequence decoder. Both encoder and decoder use a deep Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) architecture. The encoder deep LSTM reads the input sequence X and generates representation vectors at each word position. The vectors are further fed into an attention layer (Bahdanau, Cho, and Bengio 2014) to output an overall representation of the input sequence considering the output word position. The hidden states h_k of decoder deep LSTM are further fed into a dense layer $z_k = W \cdot h_k$, where the output z_k holds the dimension of the vocabulary size, to predict the output word y_k.

We follow previous work to train the *sequence-to-sequence* network with the linked attribute value pairs. However, instead of predicting the output sequence, in CoLink we turn the learned *sequence-to-sequence* network into a binary classifier. Firstly, we use the network to generate a linkage likelihood of the user pair. Secondly, we select a linkage likelihood threshold above which the user pair is considered as linked.

To calculate the linkage likelihood of two attribute values, we fix the input and the output sequence of the *sequence-to-sequence* network. Then we apply softmax function to decoder output z_k to approximate the conditional probability of word y_k as follows,

$$P(y_k|y_0, \dots, y_{k-1}, X) = \frac{exp(z_k^j)}{\sum_{j=1}^{|z_k|} exp(z_k^j)}, \quad (1)$$

where j is the word index in vocabulary. As we are interested in making the linkage likelihood of different attribute pairs comparable, we want it to be independent to the sequence length. Therefore, we define the linkage likelihood of two attribute text strings to be the averaged conditional probability of each output word.

$$Likelihood(X, Y) = \{\prod_{k=1}^{K} P(y_k|y_0, \dots, y_{k-1}, X)\}^{\frac{1}{K}} \quad (2)$$

We train a separate *sequence-to-sequence* network for each attribute field. The overall linkage likelihood of a user pair is an average of the linkage likelihood from all attribute fields. The final linkage decision is made by comparing the linkage likelihood with a selected threshold. The threshold is selected empirically by considering the linkage likelihood of the training pairs. We study the selection of the threshold in the experiment section.

Support Vector Machine Traditional classification algorithms like SVM can also be employed in the attribute-based model. Unlike *sequence-to-sequence*, which only requires positive training examples (linked pairs), SVM requires negative examples. As the user pair space $V^s \times V^t$ is extremely large, the positive examples are actually very sparse in the whole space. Given the linked pairs at each co-training iteration, we pick the same quantity of random user pairs as negative examples. We follow previous work (Kong, Zhang, and Yu 2013; Lacoste-Julien et al. 2013; Cohen, Ravikumar, and Fienberg 2003) to generate SVM features including bag-of-words similarity, Jaccard similarity and Jaro string distance. We use the Radial Basis Function (RBF) as the SVM kernel.

Relationship-based Model

The relationship-based model uses only user relationships to collect linked pairs. Finding identical vertices across two networks based on only relationships is frequently studied as the Network Alignment problem (Singh, Xu, and Berger 2008; Bayati et al. 2009; Korula and Lattanzi 2014).

The relationship-based model can use any algorithm for aligning networks based on relationships. As we focus more on the co-training algorithm and the *sequence-to-sequence* attribute-based model, in this paper, we use a heuristic model which is based on an assumption that if two users, from different networks, share a lot of mutually related users, which have been already linked, they are very likely to be linked too. We define a function $RelSim(v^s, v^t)$ to measure the relationship similarity of two users v^s, v^t from the source and the target networks respectively. v^s, v^t must meet certain constraints, such as having similar user name or identical user account id etc., otherwise the similarity is set to zero. The $RelSim(v^s, v^t)$ equals to the common related user count.

$$RelSim(v^s, v^t) = |\{\langle \hat{v}^s, \hat{v}^t \rangle\}| \tag{3}$$

where $\{\langle \hat{v}^s, \hat{v}^t \rangle\} \subseteq S$, $\langle \hat{v}^s, v^s \rangle \in E^s$, $\langle \hat{v}^t, v^t \rangle \in E^t$. Given social networks G^s and G^t and the linked pair set S, the relationship-based model finds additional linked pairs where the similarity function $RelSim$ is larger than a pre-set threshold of 2.

Experiments

We evaluate the effectiveness of CoLink by comparing it to the state-of-the-art unsupervised approaches. We also study the seed rules and the selection of the linkage likelihood threshold to better understand how they may impact on the linking results.

Data Set

We choose a real-world data set to evaluate CoLink, in which one social network is LinkedIn, while the other network is an internal enterprise user network. We crawl over 2.4 million public LinkedIn profiles from the web. The LinkedIn profile webpages are parsed to get attributes like name, organization, job title, location etc. Since LinkedIn connections are not public, we parse the “People Also Viewed” list (maximum 10 profile outlinks) in the profile page to connect profiles. The enterprise user network contains over 220K users. The relationships between enterprise users are obtained from the enterprise's Active Directory. For example, if two users attend the same meeting, there will be a relationship between them. In enterprise networks, the user attributes are name, job title and office location. The data set summary is listed in table 1.

Table 1: Data set summary.

		Enterprise	**LinkedIn**
Network	# vertices	221,869	2,480,410
	# relationships	4,411,721	16,069,575
Attribute	# names	221,869	2,468,072
	# job titles	132,430	2,207,871
	# locations/offices	148,471	2,466,961

We randomly select 658 users from the enterprise network and try to link them to their LinkedIn profiles manually. We finally collect 594 linked pairs, while 64 users have no linked LinkedIn profile found. We use precision, recall and F1 measure of the linked pairs as evaluation metrics.

Candidate Filter

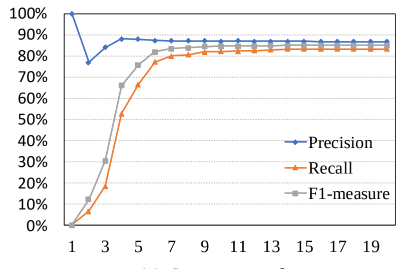
Given the vertex numbers in both networks, enumerating every possible user pairs in the space of $V^s \times V^t$ is unrealistic. Instead, we construct a candidate filter which removes a huge number of user pairs that are very unlikely to be linked. The candidate filter considers the following attributes.

- user name; The first names and last names in a user pair must meet certain similarity conditions, such as exact match, name initial, nickname etc.
- organization; The LinkedIn user's organization attribute text must contain the enterprise name.

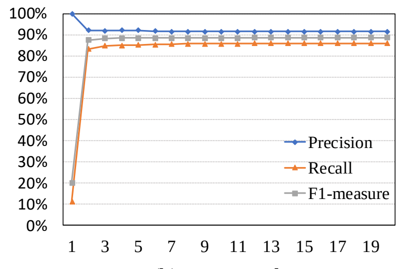
After filtering, we get 758,046 candidate user pairs which cover all the linked pairs in the ground-truth set. The candidate pairs are grouped by the source network (the enterprise network in this paper) vertices. Therefore, in each candidate pair group $g(v^s) = \langle v^s, \{v^t\}_{candidates} \rangle$ there could be only one correct linked pair according to the *one-to-one* linkage constraint.

Sequence-to-sequence

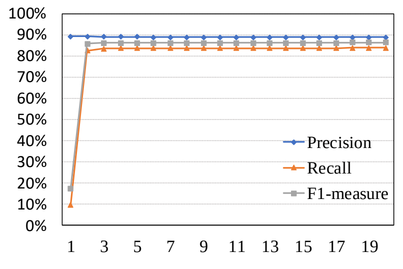
The *sequence-to-sequence* network in our experiments consists of a deep LSTM encoder with attention network and a deep LSTM decoder. The encoder deep LSTM and decoder deep LSTM both have 2 LSTM stacked, because we find that a encoder or decoder with more than 2 layers won't



(a) Coarse-tuned



(b) Fine-tuned



(c) Noisy

Figure 1: Seed rule sets comparison. P/R/F1 trend with co-training iterations startin.

bring performance increase any more for the UIL task. In each LSTM, the recurrent unit has a size of 512. Every word is first turned into a 512 embedding vector before being fed into the encoder and decoder. The training time of the *sequence-to-sequence* model depends on the training data size. On average, it takes about 30 minutes to get the model trained on 100K attribute pairs with a Tesla K40 GPU.

Seed Rules

In order to test the robustness of CoLink, we try the following 3 seed rule sets.

- Coarse-tuned set; There are only two rules in this set. First, it is the only pair in the candidate pair group $g(v^s)$; second, the title attribute must contain the string "account manager" in both networks. We collect 81 initial linked pairs using this rule set.
- Fine-tuned set; There are also two rules in this set. First, it is the only pair in the candidate pair group $g(v^s)$; second, there are at least 2 users in v^s's related people that share the same names with 2 users in v^t's related people. We collect 19,241 initial linked pairs using this rule set.
- Noisy set; Based on the result of the fine-tuned seed rule set, we replace 20% pairs with randomly selected user pairs. The result initial linked pair set remains the same size.

We employ *sequence-to-sequence* as the attribute-based model. The likelihood threshold remains the same for all three seed rule sets. Figure 1 shows the CoLink performance with different seed rule sets. The coarse-turned seed rule set generates very few linked pairs with strong sampling bias which cause the precision drop at the beginning of the co-training. After more linked pairs been added, the precision increases as the sampling bias decreases. The fine-tuned seed rule set gets a initial recall of 10% which brings a large quantity of training data for the model at the beginning. As a result, the very first trained *sequence-to-sequence* model already gets a pretty good recall. In the rest co-training iterations, only the long-tail linked pairs are added. The noisy seed rule set is used to test the noise tolerance capability of CoLink. As shown in Figure 1-c, the co-training process does not diverge because of the noisy initial linked pair set.

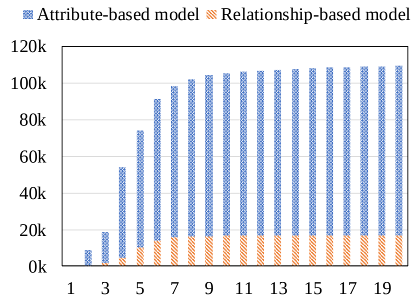


Figure 2: Linked pairs growth with co-training iterations from coarse-tuned seed pairs.

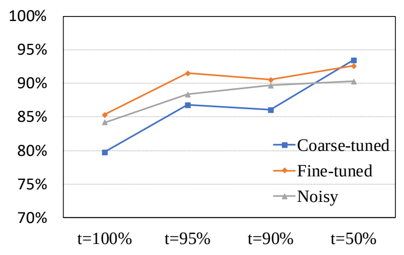
Co-training

We employ co-training by separating the relationship features from the attribute features. The attribute-based model and the relationship-based model both find new pairs at each iteration, and then reinforce each other. We keep the statistics of the result linked pairs from each model in Figure 2. In this task, the attribute-based model generates more pairs than the relationship model because we don't have full LinkedIn relationship data. We crawl the "People Also Viewed" list items from the public LinkedIn profiles which only provide no more than 10 relationships for each user.

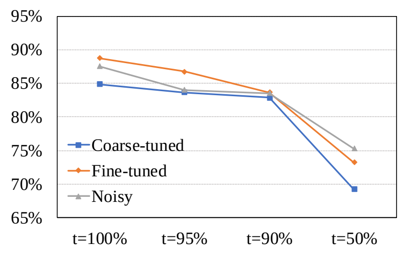
Likelihood Threshold

We set a likelihood threshold in the *sequence-to-sequence* attribute-based model. The threshold is a percentage which indicates how many training pairs are above the threshold. For example, a threshold of 100% means that every training pair gets a linkage likelihood over the threshold, while a threshold of 50% means that only 50% of the training pairs are above the threshold. The absolute threshold value is then calculated by finding a maximum value that meets the required percentage.

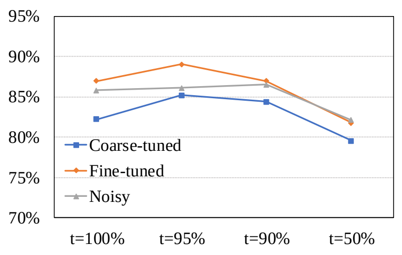
Figure 3 shows the comparison among different threshold values. Using a stricter threshold (a smaller percentage) results in a higher precision and a relatively lower recall. The threshold value can be used to adjust the trade-off between precision and recall depending on the requirements of the specific UIL tasks. In our task, we choose a threshold of 95%.



(a) Precision



(b) Recall



(c) F1-measure

Figure 3: Sequence-to-sequence linkage likelihood threshold comparison.

Table 2: Performance comparison of different approaches.

Method	P	R	F1
Random-select	49.31	54.21	51.64
SiGMa	91.00	44.28	59.57
Alias-disamb	82.35	58.92	68.69
CoLink (S2S+Coarse-tuned)	86.74	83.67	85.18
CoLink (S2S+Coarse-tuned+Rev)	89.51	86.20	87.82
CoLink (S2S+Fine-tuned)	**91.47**	**86.70**	**89.02**
CoLink (S2S+Fine-tuned+Rev)	89.22	86.36	87.77
CoLink (SVM+Fine-tuned)	84.16	62.63	71.81

Comparison Results

We demonstrate the effectiveness of CoLink by comparing it to the state-of-the-art unsupervised approaches including SiGMa and Alias-disamb. We also compare the CoLink results with different seed rule sets and attribute-based models to better understand the impact of the different settings. Table 2 shows the comparison results of different approaches. We apply the same candidate filter for all approaches. The detailed discussion are as follows.

- **Random-select:** We randomly pick one target network user v^t from the candidate group $g(v^s)$ and add $\langle v^s, v^t \rangle$ to the linked pair set S. This is the baseline of our task. Since we filter the candidate pairs with name and organization constrains, the baseline gets an F1 of 51.6%.
- **SiGMa:** SiGMa (Lacoste-Julien et al. 2013) is a heuristic approach that starts with an initial linked pair set, and iteratively expands the set with newly found linked pairs by propagating the linkage similarity in the networks. The similarity functions in SiGMa are based on Jaccard coefficient. We directly use the initial linked pair set generated from the fine-turned seed rule set. SiGMa achieves good precision but low recall because the Jaccard similarity function can only handle a fixed number of attribute variations.
- **Alias-disamb:** In Alias-disamb (Liu et al. 2013), the training data is automatically generated by evaluating the rareness of user names in both networks. We label the candidate pairs with top 1% rarest names as positive examples, and candidate pairs with top 1% most common name as negative examples, which result in 15,160 training pairs. We then train a binary SVM classifier using the generated data. We port most of the features that proposed in their paper except the ones that rely on the data not available in our task, such as post content, image etc.
- **CoLink (S2S+Coarse-tuned):** The coarse-turned seed rule set generates only 81 linked pairs with strong bias (all titles must contain "account manager"). Although the precision drops at the initial iterations, the final F1 only drops less than 4% compared to the best result. This shows the robustness of CoLink which can start with a small biased seed set.
- **CoLink (S2S+Coarse-tuned+Rev):** As mentioned in the earlier sections, a revision after convergence could help fix errors from early iterations. *CoLink (S2S+Coarse-tuned+Rev)* adds a revision process after the convergence of *CoLink (S2S+Coarse-tuned)*, which reconstructs S with the final models of *CoLink (S2S+Coarse-tuned)*. In *CoLink (S2S+Coarse-tuned)*, the biased initial training set hurts the precision of the result model very much at the beginning. The revision process may fix the errors generated by the early stage models. There is an F1 increase of over 2% compared to *CoLink (S2S+Coarse-tuned)*.
- **CoLink (S2S+Fine-tuned):** The fine-tuned seed rule set generates a high quality seed set, and thus results in a better overall performance. *CoLink (S2S+Fine-tuned)* achieves the best result, which outperforms Alias-disamb by a F1 increase of over 20%. This shows that the selection of the seed set does impact the performance of CoLink, although CoLink is less sensitive to the seed set selection.
- **CoLink (S2S+Fine-tuned+Rev):** As shown in Figure 1-b, the precision is decreasing along the co-training iterations when using the fine-tuned seed rules. More and more incorrectly linked user pairs are generated in the later models. These errors are going to hurt the final model trained in the revision process. It is observed that there is 1.25% F1 drops compared to *CoLink (S2S+Fine-tuned)*.
- **CoLink (SVM+Fine-tuned):** We also employ SVM in the attribute-based model of CoLink. We directly utilize the SVM features of the Alias-disamb experiment. Comparing to the Alias-disamb, *CoLink (SVM+Fine-tuned)* achieves over 3% F1 increase as the co-training algorithm constructs the training data set iteratively. How-

Table 3: Selected attribute examples and their similarity scores. Cosine Similarity (Kong, Zhang, and Yu 2013), Jaccard Similarity (Lacoste-Julien et al. 2013), Jaro Distance (Cohen, Ravikumar, and Fienberg 2003) and LCS-based Fuzzy-match (Vosecky, Hong, and Shen 2009) are listed for comparison.

Enterprise	**LinkedIn**	**Seq-to-seq likelihood**	**Cosine Similarity**	**Jaccard Similarity**	**Jaro Distance**	**LCS-based Fuzzy-match**
SUNNYVALE-1020/6221	San Francisco Bay Area	0.931	0	0	0.181	0.028
PARIS-ISSY/O3E17F	Parijs en omgeving, Frankrijk	0.703	0	0	0.575	0.250
PFE	Premier Field Engineer	0.817	0	0	0.529	0.067
SR SDE	Senior Software Development Engineer	0.733	0	0	0.272	0.087
ESCAL ENG	Escalation Engineer	0.843	0	0	0.650	0.500
SR CONSULTANT PROD	Senior Consultant	0.722	0.387	0.323	0.618	0.444

ever, the F1 drops over 17% compared with the *CoLink (S2S+Fine-tuned)* line. This demonstrates the effectiveness of *sequence-to-sequence* in the attribute alignment. We will further discuss the attribute alignment in the following section.

Attribute Alignment

By using *sequence-to-sequence*, CoLink can handle a large variety of attribute alignment problems that can hardly be solved with traditional string similarity functions. Table 3 shows several selected attribute examples, which are supposed to be aligned, and their similarity scores from different approaches, which all generate scores in $[0, 1]$. We also list the similarity measures from four traditional string similarity methods for comparison. Compared to traditional string similarity functions, *sequence-to-sequence* linkage likelihood is much more amenable to different attribute alignment scenarios including acronyms, abbreviations, semantic connections like geography correlation and language translation, etc. It can be easily applied to other UIL tasks with almost no feature engineering thanks to the representational capability of *sequence-to-sequence*.

Related Work

The User Identity Linkage problem was first formalized by Zafarani and Liu (2009) and became a heated research topic in recent years. Their solution is a set of specific rules based on several hypothesis created from empirical observations on user name patterns in the targeted SNS. These rules do not generalize to other UIL applications where the user name patterns do not apply. Liu et al. (2013) automatically generate training data by evaluating the rareness of the user names in both networks, and then trained a binary SVM classifier considering the whole user profiles to find identical users. Their solution is very sensitive to the quality and the quantity of the generated training data. Lacoste-Julien et al. (2013) propose a greedy approach called SiGMa which updates relationship similarity iteratively. However, the selected string similarity functions are sensitive to the profile data, which makes them hard to generalize. The existing unsupervised approaches are all sensitive to the similarity measures for user profiles, which hurts the generality.

There are many supervised approaches which extract features from user profiles and employ a supervised classifier models to predict linked user pairs. Goga et al. (2013) create a logistic regression classifier based on distance-based similarity features of attributes to do binary classification on candidate pairs. Liu et al. (2014) use bag-of-words based and distance-based features and address the UIL problem as a multi-objective optimization. Zhang et al. employ bag-of-words based features and basic relationship features, together with an energy-based model with local and global consistency (Zhang et al. 2015). Many embedding based approaches were also proposed recently (Mu et al. 2016; Man et al. 2016; Tan et al. 2014). However, the supervised approaches all require expensive data annotation effort.

Entity resolution approaches (Cohen, Ravikumar, and Fienberg 2003; Bhattacharya and Getoor 2007), which link entries in a database to real-world entities, may also be adapted to help in the UIL problem. Specifically, Yu uses Jaccard and Cosine similarity as textual similarity (Yu 2014); Bilenko and Mooney (2003) propose a learn-able SVM-based similarity which may be used to align user profiles. The above approaches share the weakness of the existing UIL approaches, which either require expensive annotated training data, or can hardly handle attribute variations well.

Similarly, network alignment approaches, which aim to find identical vertices across networks based on relationships, may also help in the UIL problem. Singh, Xu, and Berger propose IsoRank (Singh, Xu, and Berger 2008) which propagates pair-wise vertex similarity. More network propagation based approaches that extended IsoRank were also proposed in (Liao et al. 2009; Kollias, Mohammadi, and Grama 2012). Such propagation based approaches may serve in the CoLink framework as relationship-based models.

Conclusion

We propose CoLink, a general unsupervised framework for the User Identity Linkage problem. CoLink separates the user profile into two independent views: user attributes and relationships, and creates independent attribute-based and relationship-based model from them. By using co-training algorithm, the two models are reinforced in an iterative way, and therefore requires no data annotation effort. We address attribute alignment as a translation problem instead of typical string similarity measurement, and employ *sequence-to-sequence* in the attribute-based model. By employing *sequence-to-sequence*, CoLink can easily generalize as *sequence-to-sequence* requires almost no feature engineering and positive data samples only. We apply CoLink to the task of linking enterprise users to their LinkedIn profiles. CoLink generally outperforms the state-of-the-art unsupervised approaches by an F1 increase of 20%.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bayati, M.; Gerritsen, M.; Gleich, D. F.; Saberi, A.; and Wang, Y. 2009. Algorithms for large, sparse network alignment problems. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*.

Bhattacharya, I., and Getoor, L. 2007. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1(1):5.

Bilenko, M., and Mooney, R. J. 2003. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 39–48. ACM.

Blum, A., and Mitchell, T. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*.

Cohen, W.; Ravikumar, P.; and Fienberg, S. 2003. A comparison of string metrics for matching names and records. In *Kdd workshop on data cleaning and object consolidation*, volume 3, 73–78.

Goga, O.; Perito, D.; Lei, H.; Teixeira, R.; and Sommer, R. 2013. Large-scale correlation of accounts across social networks. *University of California at Berkeley, Berkeley, California, Tech. Rep. TR-13-002*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*.

Kollias, G.; Mohammadi, S.; and Grama, A. 2012. Network similarity decomposition (nsd): A fast and scalable approach to network alignment. *IEEE Trans. Knowl. Data Eng.*

Kong, X.; Zhang, J.; and Yu, P. S. 2013. Inferring anchor links across multiple heterogeneous social networks. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*.

Korula, N., and Lattanzi, S. 2014. An efficient reconciliation algorithm for social networks. *Proceedings of the VLDB Endowment*.

Kumar, S.; Zafarani, R.; and Liu, H. 2011. Understanding user migration patterns in social media. In *AAAI*.

Lacoste-Julien, S.; Palla, K.; Davies, A.; Kasneci, G.; Graepel, T.; and Ghahramani, Z. 2013. Sigma: Simple greedy matching for aligning large knowledge bases. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*.

Liao, C.-S.; Lu, K.; Baym, M.; Singh, R.; and Berger, B. 2009. Isorankn: spectral methods for global alignment of multiple protein networks. *Bioinformatics*.

Liu, J.; Zhang, F.; Song, X.; Song, Y.-I.; Lin, C.-Y.; and Hon, H.-W. 2013. What's in a name?: an unsupervised approach to link users across communities. In *Proceedings of the sixth ACM international conference on Web search and data mining*.

Liu, S.; Wang, S.; Zhu, F.; Zhang, J.; and Krishnan, R. 2014. Hydra: Large-scale social identity linkage via heterogeneous behavior modeling. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*.

Man, T.; Shen, H.; Liu, S.; Jin, X.; and Cheng, X. 2016. Predict anchor links across social networks via an embedding approach. In *IJCAI*.

Mu, X.; Zhu, F.; Lim, E.-P.; Xiao, J.; Wang, J.; and Zhou, Z.-H. 2016. User identity linkage by latent user space modelling. In *KDD*.

Singh, R.; Xu, J.; and Berger, B. 2008. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences of the United States of America*.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*.

Tan, S.; Guan, Z.; Cai, D.; Qin, X.; Bu, J.; and Chen, C. 2014. Mapping users across networks by manifold alignment on hypergraph. In *AAAI*.

Vosecky, J.; Hong, D.; and Shen, V. Y. 2009. User identification across multiple social networks. In *2009 First International Conference on Networked Digital Technologies*.

Wu, Y.; Schuster, M.; Chen, Z.; Le, Q. V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Yu, M. 2014. Entity linking on graph data. In *Proceedings of the 23rd International Conference on World Wide Web*, 21–26. ACM.

Zafarani, R., and Liu, H. 2009. Connecting corresponding identities across communities. *ICWSM*.

Zhang, Y.; Tang, J.; Yang, Z.; Pei, J.; and Yu, P. S. 2015. Cosnet: connecting heterogeneous social networks with local and global consistency. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.