CODE: Contrastive Pre-training with Adversarial Fine-tuning for Zero-shot Expert Linking

Bo Chen[1], Jing Zhang[2*], Xiaokang Zhang[2], Xiaobin Tang[2], Lingfan Cai[2], Cuiping Li[2], Hong Chen[2], Peng Zhang[3], Jie Tang[1]

[1]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[2]Information School, Renmin University of China, Beijing, China
[3]Zhipu.AI, Beijing, China

Abstract

Expert finding, a popular service provided by many online websites such as Expertise Finder, LinkedIn, and AMiner, is beneficial to seeking candidate qualifications, consultants, and collaborators. However, its quality is suffered from lack of ample sources of expert information. This paper employs AMiner as the basis with an aim at linking any external experts to the counterparts on AMiner. As it is infeasible to acquire sufficient linkages from arbitrary external sources, we explore the problem of zero-shot expert linking. In this paper, we propose CODE, which first pre-trains an expert linking model by contrastive learning on AMiner such that it can capture the representation and matching patterns of experts without supervised signals, then it is fine-tuned between AMiner and external sources to enhance the model's transferability in an adversarial manner. For evaluation, we first design two intrinsic tasks, author identification and paper clustering, to validate the representation and matching capability endowed by contrastive learning. Then the final external expert linking performance on two genres of external sources also implies the superiority of the adversarial fine-tuning method. Additionally, we show the online deployment of CODE, and continuously improve its online performance via active learning.

Introduction

Online websites such as Expertise Finder[1], LinkedIn[2], and AMiner[3] provide valuable services of expert finding for governments or research groups to find consultants, collaborators, candidate qualifications, etc. However, expert information is dispersed across heterogeneous sources. For example, Google Scholar and AMiner maintain academic information, LinkedIn keeps skills and background, and news articles report the real-time activities of experts. A single source of information is far from comprehensive and convincing to support the high-quality expert finding, which demands for integrating heterogeneous expert information together. This paper employs AMiner, a free online academic search and mining system collecting over 100 million researcher profiles with 200 million papers from multiple databases (Tang et al. 2008), as the implementation basis. We target at linking heterogeneous expert information from external sources to AMiner. Figure 1 illustrates the online system[4] deployed with the proposed CODE to link an expert from a news article to the right candidate on AMiner.

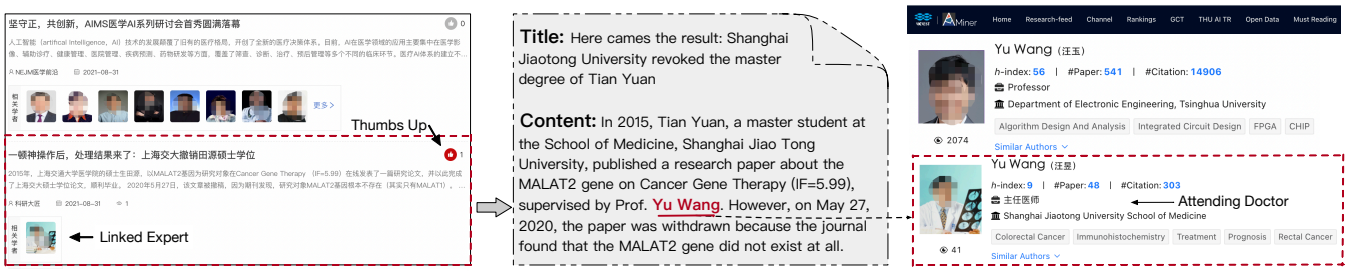
Entity linking is a related research field that links entities extracted from unstructured texts to those in a knowledge graph (Klie, de Castilho, and Gurevych 2020; Hou et al. 2020; Angell et al. 2021). However, prevailing methods often resort to the huge amount of labeled data to encode entities from heterogeneous sources into a unified space. Unfortunately, the linkages between external information and the AMiner experts are often arduous to obtain. For example, in AMiner, it usually spends up to several hours to correct the collected papers for a top expert by a skilled annotator. Moreover, the external information about experts come from arbitrary sources, making it unforeseeable for us to annotate the corresponding labels beforehand. In view of this, we pay attention to the problem of zero-shot expert linking. A natural question arises: *can we learn a universal expert linking model from abundant AMiner experts such that it can be transferable to unseen external expert linking?*

Besides the label scarcity issue, we also face challenges about how to represent an expert and match two experts, because: (1) an expert, consisting of different types of information such as demographic attributes, papers, or news, is neither a continuous signal as an image nor a discrete signal as a word, which demands a non-trivial method for expert representation beyond the standard image or word representation methods; (2) the gap of morphology, syntax, topics between AMiner and external sources is obvious, which motivates us to fine-tune the basic expert representation model on external sources to improve its transferability.

Present Work. We propose CODE, a COntrastive Pre-training with ADversarial Fine-tuning for ZEro-shot expert Linking model, to link experts from external sources to AMiner in the zero-shot setting. To address the label scarcity issue, CODE is first pre-trained on AMiner via contrastive learning (Wu et al. 2018). To enable this, we define the pre-training task as expert discrimination which samples instances from each AMiner expert, and pulls the instances sampled from the same expert close together but pushes

*Jing Zhang is the Corresponding Author.

[1]https://expertisefinder.com/
[2]http://www.linkedin.com
[3]https://www.aminer.cn/
[4]https://aminer.top3-talent.com/homepage



A screenshot of the deployed system. Linked experts are listed at the bottom of each news article.

The translated texts of second news article with the highlighted expert to be linked.

The expert candidates from AMiner. The highlighted one is the right expert.

Figure 1: The deployed online expert linking system (Left), which aims at linking expert from news articles to AMiner. The case of linking expert "Yu Wang" from the highlighted article is presented (Right).

those sampled from different experts far away from each other. An expert instance is defined as a set of papers owned by the expert, which is then encoded by BERT (Devlin et al. 2019) and matched with each other by an interaction-based fine-grained metric. After pre-training CODE, we apply the adversarial method to adapt it to unseen external sources when linking experts from them to the AMiner experts.

We demonstrate the validity of our model by a series of experiments, demonstrating that: (1) CODE is able to represent an expert well, based on which the authors of a paper can be correctly identified (the evaluation task of author identification) and the papers belonging to the same author can be correctly clustered (the evaluation task of paper clustering); (2) CODE is able to be transferred to unseen external sources such as news articles or LinkedIn pages to link the mentioned experts to the AMiner experts (the evaluation task of external expert linking).

We summarize our contributions as follows:

- We propose CODE consisting of a contrastive pre-training module and an adversarial fine-tuning module, to address zero-shot expert linking from external sources to AMiner.
- We define expert discrimination as the pre-training task along with an interaction-based metric function to characterize both the universal representation and fine-grained matching patterns of experts. Adversarial learning is used to improve the transferability of the pre-trained model.
- In addition to the extensive experiments which demonstrate the superiority of CODE, we also deploy CODE online, and further involve human in the loop to improve online performance via active learning. All codes and data are available at https://github.com/allanchen95/CODE.

Related Work

Entity Linking. Most endeavors of entity linking focus on linking entities from unstructured texts to those in a knowledge graph (KG) (Clark and Manning 2016; Kolitsas and Ganea 2018; Yamada et al. 2020; Klie, de Castilho, and Gurevych 2020; Hou et al. 2020; Angell et al. 2021). Recent work has investigated a cross-domain setting which links entities from heterogeneous texts like blogposts or news to a KG. They only train a model on the labeled source domain and directly apply it to different texts (Gupta, Singh, and Roth 2017; Le and Titov 2018; Logeswaran et al. 2019; Zemlyanskiy et al. 2021). CODE differs them in three aspects: 1) they have explicit labels, 2) an AMiner expert, consists of multiple papers, is more complex than an entity from the unstructured text, and 3) the information gap between AMiner and external sources prevents us from directly using the pre-trained model to external sources.

Contrastive Learning. Contrastive learning, which learns the data co-occurrence relationships via instance discrimination task (Wu et al. 2018), is a label-efficient representation learning regime. It has shown strength in various domains such as natural language process (Devlin et al. 2019; Yang et al. 2019; Brown et al. 2020), computer vision (Chen et al. 2020b; He et al. 2020), and graph (Qiu et al. 2020; Hu et al. 2020). Beyond learning representation, we need a fine-grained metric for matching experts.

The expert discrimination task is related to paper clustering, aiming to partition papers into a set of disjoint clusters corresponding to real experts (Zhang et al. 2018b; Qiao et al. 2019), and author identification, assigning the right authors to the anonymous papers (Chen et al. 2020a; Wang et al. 2020). Different from them, we contrast expert instances.

Adversarial Domain Adaptation. Adversarial learning has been extensively studied for the cross-domain transfer (Ganin et al. 2016; Zhai et al. 2020) such as word translation (Lample et al. 2018), text classification (Guo, Pasunuru, and Bansal 2020) and relation extraction (Shi et al. 2018) in different domains. Different from prior end-to-end adversarial learning, we adopt it for fine-tuning, which is an independent process following the pre-training stage.

Problem Formulation

This section defines an expert and formulates the problem of zero-shot expert linking in a contrastive way.

Definition 1 ***Expert.*** *An expert e is comprised by a set of support information $c_e = \{s_1, s_2, \cdots, s_{n_e}\}$, where s_i is a piece of support information. n_e denotes the size of c_e.*

The support information varies from different sources. If e is from a news article, c_e can be surrounding texts of

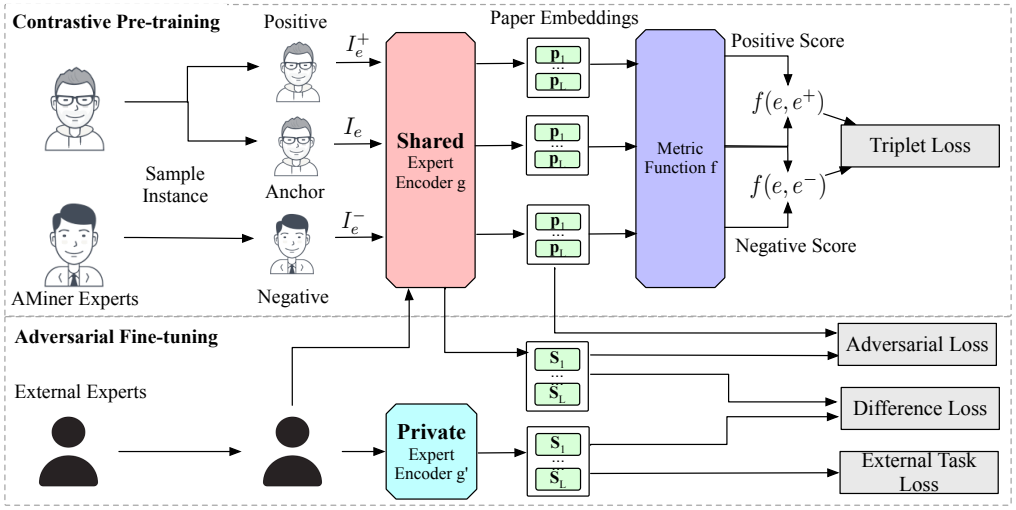


Figure 2: **Overview.** The pre-training module learns an expert encoder g and a metric f via discriminating the positive expert instance pair (I_e, I_e^+) from the negative one (I_e, I_e^-). Then it is fine-tuned on the external experts in an adversarial manner.

the expert name where s_i is one of the sentences. If e is from LinkedIn, c_e can be a homepage where s_i is one of the attributes such as summary, affiliations, etc. Particularly, we denote c_e of an AMiner expert as $c_e = \{p_1, \cdots, p_{n_e}\}$, where p_i is a paper containing title, keywords, venue, etc.

Problem 1 ***Zero-shot Expert Linking.*** *Given a set of experts $\{e\}$ on AMiner, we aim at pre-training an expert encoder g and a metric function $f : \{g(e), g(e')\} \to \{y\}$ to infer the alignment label between e and e', where $y = 1$ implies e and e' are equivalent and 0 otherwise. After pre-training, we fine-tune g and f on the external experts $\{\tilde{e}\}$ such that g and f are transferred to unseen external sources.*

Note that, we assume any experts from external sources can be certainly aligned to the counterparts in AMiner and leave the problem of non-existing alignment to the future.

The CODE Framework

This section introduces CODE (Figure 2), which consists of a contrastive pre-training module and an adversarial fine-tuning module. The former one pre-trains an expert encoder g and a metric function f purely on AMiner via contrastive learning, and the latter one fine-tunes g and f on both AMiner and the external data in an adversarial manner. For convenience, we hereinafter name the pre-trained model as CODE-pre, and also the final fine-tuned model as CODE.

Contrastive Pre-training Module

The pre-training module targets at learning an encoder g to capture the universal representation patterns of experts and a metric f to measure the fine-grained matches between experts. In light of this, we define expert discrimination as our pre-training task to increase the similarities of positive expert instances, while decreasing those of negatives ones. To achieve this, four questions should be answered carefully:

- **Q1:** What is an instance of an expert?
- **Q2:** How to encode an expert instance?
- **Q3:** How to measure the similarity of two instances?
- **Q4:** Which kind of loss function should be selected?

Q1: An Instance of An Expert. we define an instance as a set of randomly sampled papers of the expert c_e on AMiner, i.e. an instance I_e of e can be formulated as follows:

$$I_e = \{p_1, \cdots, p_L\}, \tag{1}$$

where $p_i \in c_e$ and L is the maximal number of sampled papers for each instance. Two instances sampled from the same expert is viewed as a positive pair, while the two instances from different experts are viewed as a negative pair.

Q2: BERT-based Expert Encoder. Since the support information of experts on multi-sources may be in different languages, we adopt the multi-lingual BERT (Wolf et al. 2020) to project the information into a unified semantic space.

In practice, we encode each paper as a basic representation unit of an expert instead of directly encoding the expert as a whole. We concatenate the paper attributes, including title, keywords, venues, etc, as the input p of BERT and apply a Multi-layer Perceptron (MLP) to get the paper embedding:

$$\mathbf{p} = g(p) = \text{MLP}(\text{CLS}(p)), \tag{2}$$

where $\text{CLS}(p)$ indicates the CLS token embedding of BERT.

Q3: Interaction-based Metric Function. Standard representation-based metric function usually aggregates all the paper embeddings of an instance as the expert embedding, based on which it estimates the similarity of two instances. However, the mixed expert embedding suffers from semantic drift. Thus, inspired by the similar ideas of information retrieval (Xiong et al. 2017; Dai et al. 2018), we propose an interaction-based metric to measure the fine-grained similarity between each paper pair of two instances.

Formally, we use Eq.(2) to obtain a set of paper embeddings $\{\mathbf{p}_m\}_{m=1}^{L}$ and $\{\mathbf{p}'_n\}_{n=1}^{L}$ for I_e and $I_{e'}$, respectively. Then we compute a similarity matrix $\mathbf{A}$ between I_e and $I_{e'}$. Each element α_{mn} is calculated by the normalized euclidean distance between the m-th paper in I_e and n-th paper in $I_{e'}$, i.e. $\alpha_{mn} = \|\mathbf{p}_m - \mathbf{p}'_n\|_2^2$. Then we adopt an RBF kernel aggregation function (Dai et al. 2018), which pays attention to how many similar paper pairs within two instances, to extract the similarity patterns. Specifically, we transform α_{mn} into a K-dimensional distribution (Eq.(4)), the k-th element of which is converted by the k-th RBF kernel with mean μ_k and variance σ_k (Eq.(3)) implying how likely α_{mn} corresponds to the k-th similarity pattern. Then we sum up over rows to represent the similarities between m-th paper in l_e and all the papers in $I_{e'}$, and sum up over columns to represent the similarity between I_e and $I_{e'}$ (Eq.(5)). Finally, we apply a MLP layer to obtain the similarity score (Eq.(6)).

$$K_k(\alpha_{mn}) = \exp\left[-\frac{(\alpha_{mn}-\mu_k)^2}{2\sigma_k^2}\right], \quad (3)$$

$$\mathbf{K}(\alpha_{mn}) = [K_1(\alpha_{mn}), \cdots, K_K(\alpha_{mn})], \quad (4)$$

$$\phi(e,e') = \sum_{m=1}^{L} \log \sum_{n=1}^{L} \mathbf{K}(\alpha_{mn}). \quad (5)$$

$$f(e,e') = \text{MLP}(\phi(e,e')) \quad (6)$$

Q4: Triplet Loss Function. We prefer the triplet loss instead of the widely-used contrastive loss (Chen et al. 2020b) in our problem. Contrastive loss encourages similar instances into a single point in the embedding space (Zhang et al. 2018b). However, experts may publish papers on different topics, making it weird to force all the papers into a single point. Thus the triplet loss, maintaining a relative distance between positive and negative pairs, is a better choice.

Specifically, for the anchor instance I_e, the positive counterpart I_e^+ is sampled from the same expert, while a negative one I_e^- is from a different expert. Given a set of triplets $\{(I_e, I_e^+, I_e^-)\}$, the triplet loss function is defined as:

$$\mathcal{L}^{\text{pre}}(\theta_g, \theta_f) = \sum_{(I_e, I_e^+, I_e^-)} \max\{0, m + f(e,e^-) - f(e,e^+)\}, \quad (7)$$

where m is a margin, $\theta_{g/f}$ are the parameters of g or f. To avoid trivial results, we omit overlaps within instance pairs.

Adversarial Fine-tuning Module

Intuitively, CODE-pre can be directly applied on external sources for zero-shot expert linking. However, the morphology, syntax, topics of the external information may be significantly different from that on AMiner, which encourages us to fine-tune CODE-pre to improve its transferability.

Most of the domain adaptation methods assume each domain is comprised of domain-agnostic and domain-private features, thus they learn a shared generator and a private generator for each domain (Liu, Qiu, and Huang 2017; Shi et al. 2018). However, as our goal is to link external experts to AMiner, we need to extract the features similar to AMiner from external sources as much as possible, such that the pre-trained metric f can better capture the similarity patterns between external and AMiner experts. Inspire by this, besides the shared generators in both domains, we only create a private generator for external experts to get rid of the dissimilar features compared with AMiner, shown in Figure 2.

Generator. Besides the shared generator g^{shared} pre-trained by the pre-training module, we create the same private generator g^{private} to extract the domain-private features from external experts. To enforce the shared and private generator to encode different aspects of features, we adopt orthogonality constraints as a difference loss (Liu, Qiu, and Huang 2017):

$$\mathcal{L}^{\text{diff}}(\theta_g^{\text{shared}}, \theta_g^{\text{private}}) = \sum_{i=1}^{N_{\text{ext}}} ||g^{\text{shared}}(s_i)^{\text{T}} g^{\text{private}}(s_i)||_F^2, \quad (8)$$

where $||.||_F^2$ is the squared Frobenius norm, N_{ext} is the number of pieces of the support information s_i.

Domain Discriminator. To cripple the external private features from the shared feature space, we design a domain discriminator for enforcing g^{shared} to abandon the private features from external sources. Given s_i from either AMiner or the external source, we use the shared generator g^{shared} to extract its features, and apply a classifier h to predict whether it is from the external source or AMiner. We adopt Gradient Reversed Layer (Ganin et al. 2016) to confuse h such that it cannot distinguish the source of support information:

$$\mathcal{L}^{\text{adv}}(\theta_g^{\text{shared}}, \theta_h) = \sum_{i=0}^{N_{\text{AMiner}}} \log(\hat{p}_i) + \sum_{i=0}^{N_{\text{ext}}} \log(1-\hat{p}_i), \quad (9)$$

$$\hat{p}_i = h\left(g^{\text{shared}}(s_i)\right) = \text{MLP}\left(g^{\text{shared}}(s_i)\right),$$

where $\hat{p}_i$ denotes the likelihood of s_i from AMiner, N_{AMiner} is the number of papers on AMiner, h is an MLP layer.

Task Predictor. Likewise, we design an external task predictor $\tilde{h}$, which predicts the source of private features, to further prevent the private features into the share feature space.

$$\mathcal{L}^{\text{ext}}(\theta_g^{\text{private}}, \theta_{\tilde{h}}) = -\sum_{i=0}^{N_{\text{ext}}} \log(1-\hat{p}_i), \quad \hat{p}_i = \tilde{h}(g^{\text{private}}(s_i)) \quad (10)$$

where $\hat{p}_i$ denotes the probability of the support information s_i is from AMiner. The classifier $\tilde{h}$ is the same as Eq.(9).

Training and Inference

The final loss function is defined as follows:

$$\mathcal{L}(\theta_g^{\text{shared}}, \theta_g^{\text{private}}, \theta_f, \theta_h) = \mathcal{L}^{\text{pre-train}} + \alpha\mathcal{L}^{\text{adv}} + \beta\mathcal{L}^{\text{diff}} + \gamma\mathcal{L}^{\text{ext}}, \quad (11)$$

where α, β, γ are trade-off hyper-parameters. During training, we first pre-train an expert encoder g^{shared} and a metric f on AMiner via Eq.(7). Then we fine-tune g^{shared} and f by adversarial learning via Eq.(11). During inference, we use the fined-tuned model to perform zero-shot expert-linking between AMiner and external sources.

Table 1: **Data statistics.** The support information of AMiner, News, and Linkedin are papers, news articles, and homepages, respectively. #Avg. candidates are the average number of candidate experts on AMiner for an author in the paper, a name in the news article, or a LinkedIn user.

	AMiner	News	LinkedIn
#Experts	45,187	1,824	1,665
#Support Information	399,255	20,658	50,000
#Avg. candidates	18	8.79	4.85

Experiments

In this section, we first evaluate the representation and matching capacity of the pre-trained model CODE-pre by two intrinsic tasks, author identification and paper clustering, on AMiner. Then we validate the transferability of fine-tuned model CODE by external expert linking, i.e., linking experts from the news article or LinkedIn to AMiner experts. For each experiment, we run 5 trials and report the mean results. Experimental details please refer to Appendix.

Datasets.

Table 1 summarizes statistics of three datasets.

- **AMiner.** We employ WhoIsWho[5], the largest manually-labeled name disambiguation dataset collected from AMiner, as the basis to be aligned.
- **News.** We collect news articles from several Chinese technique platforms such as sciencenet, jiqizhixin, etc, and extract names from these news articles by NER tools[6], then we link them to AMiner experts by a majority voting of three professional annotators' results.
- **LinkedIn.** We adopt the dataset from (Zhang et al. 2015).

Candidates. Given an external expert to be linked, we choose the AMiner experts with similar names as candidates. The similar names are obtained by moving the last name to the first or keeping the name initials except for the last name. For example, the similar names of "Bo Li" include "Li Bo", "B Li" and "L Bo".

Evaluation of the Pre-training Module

We adopt two intrinsic tasks, author identification (Chen et al. 2020a; Wang et al. 2020) and paper clustering (Zhang et al. 2018b), on AMiner, to evaluate the representation and matching capacity of CODE-pre via contrastive pre-training.

Author Identification assigns a new paper to the right expect, following the second track of competition[7]. Thus we adopt the same test set. We estimate the similarity score between a new paper and each candidate via (Eq.(6)), and return the expert with the highest similarity as the right answer.

Evaluation Metrics. We use HitRatio@K (HR@K, K=1,3) to measure the proportion of correctly assigned experts ranked in top K, and use MRR to measure the average reciprocal ranks of correctly assigned experts.

Table 2: Performance of Author Identification on AMiner.

Model	HR@1	HR@3	MRR
GBDT	0.873	0.981	0.927
Camel	0.577	0.737	0.644
HetNetE	0.582	0.759	0.697
CONNA	**0.911**	**0.985**	**0.949**
Unsupervised	0.713	0.875	0.808
Paper-paper pseudo labels	0.864	0.960	0.915
Paper-expert pseudo lables	0.892	0.970	0.933
Representation metric function	0.870	0.956	0.918
CODE-pre	0.898	0.964	0.934

Baselines. We employ the following methods to solve the task of author identification: **GBDT** (Li et al. 2013) is a feature-engineering model in KDD Cup 2013 (Roy et al. 2013), **Camel** (Zhang et al. 2018a) represents a paper by GRU with its title and keywords, an expert by one-hot embedding. **HetNetE** (Chen and Sun 2017) is similar to Camel except that each paper is represented by the author names, affiliations, venues in addition to the title and keywords, and **CONNA** (Chen et al. 2020a) is an interaction-based model. The basic interactions are built between the token embeddings of two attributes, then different attributes matrices are aggregated as the paper-level interactions, finally the paper-level matrices are aggregated as expert-level interactions.

Results. Table 2 shows the performance of author identification. We can see CODE-pre outperforms Camel and HetNetE by 31.6$\sim$32.1% in HR@1. Camel and HetNetE adopt a representation-based metric function, which embed a paper and an expert independently, and computes the similarity score between them. Thus they fail to capture the fine-grained similarities between papers.

CODE-pre is comparable with GBDT and CONNA. GBDT, extracts hand-crafted features between a paper and an expert, and CONNA, computes the fine-grained interactions between them, outperform Camel and HetNetE by a large margin, which demonstrates the efficacy of interaction-based strategy. CONNA slightly outperforms CODE-pre by 1.3% in HR@1, because CONNA, with the dedicated architecture and training criteria, is tailed for the author identification task. However, GBDT and CONNA aren't capable of being transferable to other domains, while CODE-pre first applies the expert encoder g to encode each paper into a basic embedding unit, and uses a metric f to compute the similarities between papers and experts. The g and f can be easily generalized to other domains, such as news articles or LinkedIn, by first encoding support information with g and then estimate the interactions with f similarly.

Paper Clustering aims at clustering papers belonging to the same expert together, following the first track of the competition[8]. We adopt the same test set, and use the hierarchical

[5]https://www.aminer.cn/whoiswho

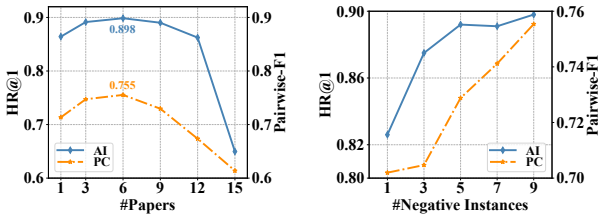
[6]http://thulac.thunlp.org/

[7]https://biendata.xyz/competition/aminer2019_2/

[8]https://biendata.xyz/competition/aminer2019/

Table 3: Performance of Paper Clustering on AMiner.

Model	P-Pre.	P-Rec.	P-F1
Louppe et al.	0.609	0.605	0.607
Zhang et al.	0.768	0.551	0.642
G/L-Emb	**0.835**	0.640	0.724
Unsupervised	0.332	0.591	0.425
Paper-paper pseudo labels	0.659	0.779	0.714
Paper-expert pseudo lables	0.715	0.786	0.749
Representation metric function	0.595	0.754	0.665
CODE-pre	0.724	**0.789**	**0.755**



(a) Paper Sampling. (b) Negative Sampling.

Figure 3: The Effect of (a) paper sampling and (b) negative sampling. AI and PC stand for author identification and paper clustering respectively.

agglomerative clustering algorithm (HAC) to cluster papers based on the paper embeddings output by g.

Evaluation Metrics. We use pairwise Precision, Recall, and F1-score (P-Pre., P-Rec, and P-F1) (Zhang et al. 2018b) to evaluate the clustering results of each name, and then calculate the macro metric by averaging metrics over all names.

Baselines. We compare with three state-of-the-art methods for paper clustering. **Louppe et al.** (Louppe et al. 2016) trains a similarity metric based on hand-crafted features to measure the similarities between papers, **Zhang et al.** (Zhang and Al Hasan 2017) constructs three graphs including the expert-expert graph, the expert-paper graph, and the paper-paper graph for each name. Then they learn graph embedding by preserving node connectivity on all the graphs, and **G/L-Emb** (Zhang et al. 2018b) learns paper embeddings on a global paper-paper network, and then fine-tunes the paper embeddings on a local paper-paper network built for each name by graph auto-encoding.

Results. Table 3 presents the performance of paper clustering. Overall, CODE-pre advances other baselines by 3.1∼14.8% in Pairwise-F1. Among them, Louppe et al. perform the worst, as it merely captures the pairwise similarities while ignoring the interplays with other papers. Zhang et al. and G/L-Emb, which build the local paper-paper graph in each name and leverage the graph structure to learn paper embeddings, outperform Louppe et al. by 3.5∼11.7% in Pairwise-F1. Besides local embeddings, G/L-Emb incorporates global information via preserving the connectivity in the global graph, making it outperform Zhang et al by +8.2% in Pairwise-F1. Although CODE-pre discards the graph information for its limitation in transferability, it still performs the best because of the striking representation capability endowed by the BERT encoder g and contrastive pre-training.

Ablation Study To verify the efficacy of different components in CODE-pre, we make four model variants: **Unsupervised** mimics the unsupervised industrial methods. We use BERT to obtain paper embeddings and average them of an expert as the expert embedding. Then we calculate the euclidean distance between the paper and expert, **Paper-paper pseudo labels** samples paper-paper pseudo labels instead of the pairs of instances, likewise two papers are viewed as a positive pair if they are sampled from the same expert and a negative one otherwise, **Paper-expert pseudo labels** samples paper-expert pseudo labels, and **Representation metric function** uses the representation-based metric function to replace Eq. (6) via averaging all the paper embeddings of an instance as the expert embedding.

The performance is shown in Table 2 and Table 3. The unsupervised model performs the worst, -18.5% in HR@1 and -33.0% in Pairwise-F1 compared with CODE-pre, which implies that the vanilla BERT fails to measure semantic correlations between papers or experts. Both the paper-paper and paper-expert pseudo labels underperform CODE-pre, -0.6∼3.4% in HR@1 and -0.6∼4.1% in Pairwise-F1, denotes contrasting two expert instances can result in better representations of papers and experts. The representation-based metric function also underperforms CODE-pre, -2.8% in HR@1 and -9.0% in Pairwise-F1, which emphasizes the superiority of interaction-based matching strategy.

Effect of Paper Sampling. We explore how the maximal number L (#Papers) of an instance affects performance. We vary L from 1 to 15 with interval 3 and present the performance of both tasks in Figure 3(a). We see that either too few or many papers will harm the performance. Few papers may result in dissimilar expert instances even if they are sampled from a same expert, while too many papers make positive expert instances easier to be distinguished, thus lead to a trivial solution and degrade the performance (You et al. 2020).

Effect of Negative Sampling. The number of negative instances also influences the performance. Many efforts (He et al. 2020; Chen et al. 2020b) have shown that more negative instances result in better performance. To verify this, we vary negative instances from 1 to 9 with interval 2 in both tasks. From Figure 3(b). We can see CODE-pre improves incessantly with the increment of negative instances. Due to the limitation of GPUs, the maximal number of negative instances is 9. But it is possible to improve the performance when GPUs with larger memory size are available.

Evaluation of the Adversarial Fine-tuning Module

We fine-tune CODE-pre between AMiner and external sources, and evaluate CODE on external expert linking task.

Baselines. We compared CODE with five classical domain adaptation baselines as: **Unsupervised** is the same as the unsupervised model variant for evaluating CODE-pre, **CODE-pre** directly encodes and links the external experts

Table 4: Performance of External Expert Linking.

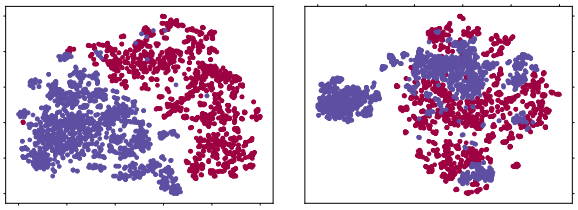
External Sources	News			LinkedIn		
	HR@1	HR@3	MRR	HR@1	HR@3	MRR
Unsupervised	0.329	0.731	0.559	0.805	0.963	0.886
CODE-pre	0.737	0.927	0.837	0.897	0.982	0.940
Chain Pre-training	0.739	0.927	0.839	0.895	0.978	0.939
DANN	0.743	0.928	0.842	0.901	0.983	0.943
ASP-MTL	0.746	0.930	0.843	0.903	0.983	0.944
CODE	**0.753**	**0.936**	**0.848**	**0.904**	**0.987**	**0.945**
CODE w/o $\mathcal{L}^{\text{adv}}$	0.721	0.923	0.825	0.873	0.981	0.927
CODE w/o $\mathcal{L}^{\text{diff}}$	0.745	0.927	0.841	0.901	0.985	0.942
CODE w/o $\mathcal{L}^{\text{ext}}$	0.743	0.925	0.84	0.898	0.983	0.941

to AMiner, **Chain Pre-training** (Logeswaran et al. 2019) chains a series of pre-training stages together. We first pre-train the vanilla BERT on both AMiner and the external data, then fine-tune it on the external data to obtain the fine-tuned BERT, base on which we train CODE-pre on AMiner via Eq. (7), **DANN** (Ganin et al. 2016) extracts domain-agnostic without domain-private features. We only use a shared generator to encode both AMiner and external expert, and **ASP-MTL** (Liu, Qiu, and Huang 2017) captures both the domain-shared and domain-private features. We create private generators for both AMiner and the external source.

Results. Table 4 shows the performance of linking external experts from news articles or LinkedIn users to AMiner experts. The unsupervised model performs the worst (-9.9∼42.4% in HR@1), as the vanilla BERT isn't fine-tuned at all. CODE-pre, pre-trained on AMiner without fine-tuning on the external data, underperforms CODE by 0.7∼1.6%. The Chain Pre-training only fine-tunes BERT encoder g but not metric f. Although DANN fine-tunes both g and f, it does not detach the private features from the shared space. ASP-MTL, builds both the private and shared encoders, outperforms the above two baselines. Compared with ASP-MTL, CODE performs better (+0.1∼0.7% in HR@1). As the goal of external expert linking is to extract the features similar to AMiner experts from external sources as much as possible, there is no need to extract the private features of AMiner. Moreover, we create several constraints to make the external private features special enough to be identified.

Ablation Study. To verify the efficacy of different components in CODE, we make three model variants: **CODE w/o $\mathcal{L}^{\text{diff}}$** removes the difference loss in Eq.(8), **CODE w/o $\mathcal{L}^{\text{adv}}$** removes the adversarial loss in Eq.(9), and **CODE w/o $\mathcal{L}^{\text{ext}}$:** removes the external task predictor loss in Eq.(10).

The performance is shown in Table 4. CODE w/o $\mathcal{L}^{\text{adv}}$ performs the worst (-3.1∼3.2% in HR@1 compared with CODE) denoting the domain discriminator takes the most important role in the fine-tuning process. Both CODE w/o $\mathcal{L}^{\text{diff}}$ and CODE w/o $\mathcal{L}^{\text{ext}}$ underperform CODE by 0.3∼1.0% in HR@1, which shows that both losses can enhance model performance. We also observe a desired phenomenon that CODE w/o $\mathcal{L}^{\text{diff}}$ performs slightly better than CODE w/o $\mathcal{L}^{\text{ext}}$, as we expect more about explicitly preventing external private features into the shared feature space.



(a) Before fine-tuning. (b) After fine-tuning.

Figure 4: T-SNE visualization before and after the adversarial fine-tuning for news articles (*Purple*) and AMiner (*Red*).

Visualization. We utilize t-SNE (Van der Maaten and Hinton 2008) to visualize papers on AMiner and sentences on news articles output by the encoder g before and after fine-tuning. Figure 4(a) implies when using CODE-pre, the embedding distributions between AMiner and news articles are not aligned, whille Figure 4(b) shows CODE, after fine-tuning, mitigates the domain shift.

Online Deployment

A screenshot of deployed system with CODE is shown in Figure 1. Practically, we first extract names by NER tools from a news article. Then for each name, instead of the experimental candidate selection strategy, we adopt ElasticSearch[9] to perform online fuzzy search. Finally, we apply CODE to estimate the similarity between each candidate and expert from a news article. To solve the case that an external expert may not be linked to any AMiner experts, we predefined a threshold and return the candidate with the highest score exceeding the threshold as right expert on AMiner.

Active Learning. Admittedly, assignment errors are inevitable, thus we allow users to provide feedback to our assignment results. Specifically, if users agree with the results, they can click "Thumbs Up" on the top left of each article shown on Figure 1, otherwise, they can fill in a feedback form to submit the right experts they think on AMiner. The feedback will be regarded as new training instances to further improve the online performance of CODE.

Conclusion and Discussion

Ethical Statement. CODE may violate personal privacy via automatically linking the news from experts' personal life to their professional information on AMiner. To avoid this, we have adopted the following two strategies: (1) we only integrate the external public information instead of the private information from the scientific media such as news.sciencenet.cn and jiqizhixin.com, and from professional social platforms such as LinkedIn; (2) before linking the information online, we send the email to the corresponding experts in AMiner to seek their authorization.

We propose CODE, consisting of a contrastive pre-training module and an adversarial fine-tuning module, to

[9]https://www.elastic.co

link experts from external sources to AMiner in the zero-shot setting. The former one performs the pre-training task of expert discrimination to learn an expert encoder for capturing the universal representation patterns of experts, and an interaction-based metric to characterize fine-grained matches between experts on AMiner. The later one adapts the pre-trained model to the unseen external sources when linking experts from them to AMiner in an adversarial manner. Experimental results connote the superiority of CODE. In the future, we plan to generalize CODE into more external sources and deploy it online.

Acknowledgments

We would like to thank Lingxi Zhang and Haipeng Ding for their help in data collection. We also thank the involved staffs in Zhipu.AI team for their help in online deployment of CODE. This work is supported by National Natural Science Foundation of China 62076245 and 61825602.

References

Angell, R.; Monath, N.; Mohan, S.; Yadav, N.; and McCallum, A. 2021. Clustering-based Inference for Biomedical Entity Linking. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2598–2608.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Chen, B.; Zhang, J.; Tang, J.; Cai, L.; Wang, Z.; Zhao, S.; Chen, H.; and Li, C. 2020a. CONNA: Addressing Name Disambiguation on The Fly. *IEEE Transactions on Knowledge and Data Engineering*.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020b. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*.

Chen, T.; and Sun, Y. 2017. Task-guided and path-augmented heterogeneous network embedding for author identification. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 295–304.

Clark, K.; and Manning, C. D. 2016. Deep reinforcement learning for mention-ranking coreference models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Dai, Z.; Xiong, C.; Callan, J.; and Liu, Z. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *Proceedings of the eleventh ACM international conference on web search and data mining*, 126–134.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186.

Efimov, D.; Silva, L.; and Solecki, B. 2013. KDD Cup 2013-Author-paper identification challenge: Second place team. In *KDD Cup 2013 Workshop*, 3.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 2096–2030.

Guo, H.; Pasunuru, R.; and Bansal, M. 2020. Multi-source domain adaptation for text classification via distancenet-bandits. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 7830–7838.

Gupta, N.; Singh, S.; and Roth, D. 2017. Entity linking via joint encoding of types, descriptions, and context. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2681–2690.

He, K.; Fan, H.; Wu, Y.; Xie, S.; and Girshick, R. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9729–9738.

Hou, F.; Wang, R.; He, J.; and Zhou, Y. 2020. Improving Entity Linking through Semantic Reinforced Entity Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 6843–6848.

Hu, Z.; Dong, Y.; Wang, K.; Chang, K.-W.; and Sun, Y. 2020. Gpt-gnn: Generative pre-training of graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1857–1867.

Klie, J.-C.; de Castilho, R. E.; and Gurevych, I. 2020. From zero to hero: Human-in-the-loop entity linking in low resource domains. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 6982–6993.

Kolitsas, N.; and Ganea, O.-E. 2018. End-to-End Neural Entity Linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Lample, G.; Conneau, A.; Ranzato, M.; Denoyer, L.; and Jégou, H. 2018. Word translation without parallel data. In *International Conference on Learning Representations*.

Le, P.; and Titov, I. 2018. Improving entity linking by modeling latent relations between mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Li, J.; Liang, X.; Ding, W.; Yang, W.; and Pan, R. 2013. Feature engineering and tree modeling for author-paper identification challenge. In *KDD Cup 2013 Workshop*, 5.

Liu, P.; Qiu, X.; and Huang, X. 2017. Adversarial multi-task learning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 1–10.

Logeswaran, L.; Chang, M.-W.; Lee, K.; Toutanova, K.; Devlin, J.; and Lee, H. 2019. Zero-Shot Entity Linking by Reading Entity Descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Louppe, G.; Al-Natsheh, H. T.; Susik, M.; and Maguire, E. J. 2016. Ethnicity sensitive author disambiguation using semi-supervised learning. In *international conference on knowledge engineering and the semantic web*, 272–287. Springer.

Qiao, Z.; Du, Y.; Fu, Y.; Wang, P.; and Zhou, Y. 2019. Unsupervised author disambiguation using heterogeneous graph convolutional network embedding. In *2019 IEEE international conference on big data (Big Data)*, 910–919. IEEE.

Qiu, J.; Chen, Q.; Dong, Y.; Zhang, J.; Yang, H.; Ding, M.; Wang, K.; and Tang, J. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1150–1160.

Roy, S. B.; De Cock, M.; Mandava, V.; Savanna, S.; Dalessandro, B.; Perlich, C.; Cukierski, W.; and Hamner, B. 2013. The microsoft academic search dataset and kdd cup 2013. In *KDD cup 2013 workshop*, 1–6.

Shi, G.; Feng, C.; Huang, L.; Zhang, B.; Ji, H.; Liao, L.; and Huang, H.-Y. 2018. Genre separation network with adversarial training for cross-genre relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 1018–1023.

Tang, J.; Zhang, J.; Yao, L.; Li, J.; Zhang, L.; and Su, Z. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 990–998.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Wang, H.; Wan, R.; Wen, C.; Li, S.; Jia, Y.; Zhang, W.; and Wang, X. 2020. Author name disambiguation on heterogeneous information network with adversarial representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 238–245.

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; Davison, J.; Shleifer, S.; von Platen, P.; Ma, C.; Jernite, Y.; Plu, J.; Xu, C.; Scao, T. L.; Gugger, S.; Drame, M.; Lhoest, Q.; and Rush, A. M. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 38–45.

Wu, Z.; Xiong, Y.; Yu, S. X.; and Lin, D. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3733–3742.

Xiong, C.; Dai, Z.; Callan, J.; Liu, Z.; and Power, R. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 55–64.

Yamada, I.; Asai, A.; Shindo, H.; Takeda, H.; and Matsumoto, Y. 2020. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6442–6454.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R. R.; and Le, Q. V. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; and Shen, Y. 2020. Graph Contrastive Learning with Augmentations. *Advances in Neural Information Processing Systems*, 33.

Zemlyanskiy, Y.; Gandhe, S.; He, R.; Kanagal, B.; Ravula, A.; Gottweis, J.; Sha, F.; and Eckstein, I. 2021. DOCENT: Learning Self-Supervised Entity Representations from Large Document Collections. *arXiv preprint arXiv:2102.13247*.

Zhai, Y.; Lu, S.; Ye, Q.; Shan, X.; Chen, J.; Ji, R.; and Tian, Y. 2020. Ad-cluster: Augmented discriminative clustering for domain adaptive person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9021–9030.

Zhang, B.; and Al Hasan, M. 2017. Name disambiguation in anonymized graphs using network embedding. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 1239–1248.

Zhang, C.; Huang, C.; Yu, L.; Zhang, X.; and Chawla, N. V. 2018a. Camel: Content-Aware and Meta-path Augmented Metric Learning for Author Identification. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, 709–718.

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*, 1485–1494.

Zhang, Y.; Zhang, F.; Yao, P.; and Tang, J. 2018b. Name Disambiguation in AMiner: Clustering, Maintenance, and Human in the Loop. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1002–1011.

Appendix

Table 5: Hyper-parameters of CODE.

Model architecture	
BERT MLP	$\mathbb{R}^{768\times512}$
#Kernels K	21
Metric MLP	$\mathbb{R}^{21\times21}$, $\mathbb{R}^{21\times1}$
Margin m	1.0
Adversarial MLP	$\mathbb{R}^{512\times100}$, $\mathbb{R}^{100\times2}$
LeakyReLU	Negative slope is 0.2
Learning rate	
μ_g for encoder	2e-5
μ_f for metric function	2e-3
μ_h for discriminator	1e-3
Learning rate decay	Exponential decay = 0.96
Loss function weight	
α for adversarial loss	0.1
β for difference loss	0.1
γ for external task loss	0.1
Batch size	
For pre-training on AMiner data	32
For fine-tuning on External sources	256
Others	
#Negative Instances	9
#Papers	6
Optimization algorithm	Adam
Pre-training epochs	20
Fine-tuning epochs	1

Implementation Details

The detailed hyper-parameters are listed in Table 5.

Running Environment We implement CODE by Python 3.6.8, PyTorch 1.1.0, and conduct the experiments on an Enterprise Linux Server with 56 Intel(R) Xeon(R) Gold 5120 CPU @ 2.20GHz, and a single NVIDIA Tesla V100 SXM2 with 32GB memory size.

Pre-training Module The dimension of the BERT output embedding is 768, which is then fed into a one-layer MLP in Eq.(2) to get the corresponding outputs:

$$\text{MLP}(\mathbf{X}) = \text{Tanh}(\mathbf{W}^T\mathbf{X}), \tag{12}$$

where $\mathbf{W} \in \mathbb{R}^{768\times512}$.

The two-layers MLP on top of the metric function in Eq.(6) is defined as follows:

$$\text{MLP}(\mathbf{X}) = \tanh(\mathbf{W}_2^T\text{LeakyReLU}(\mathbf{W}_1^T\mathbf{X})), \tag{13}$$

where $\mathbf{W}_1 \in \mathbb{R}^{21\times21}$ and $\mathbf{W}_2 \in \mathbb{R}^{21\times1}$.

In Eq.(3), we use 21 kernels, where μ is from 0 to 1 with interval 0.05. The kernel with $\mu = 0.0$ and $\sigma = 10^{-3}$ corresponds to the exact matching kernel, while σ is set as 0.1 for other kernels capture the semantic matches at different scales. The margin m of the triplet loss in Eq.(7) is set to 1.0.

Table 6: **Features extracted for GBDT model**. p: paper, a: an author in p, c: a candidate expert whose name is similar as a.

No.	Feature description
1	The number of the papers of c
2	The number of the coauthors of a in p
3	The number of the coauthors of c
4	The number of the same coauthors between a and c
5	Ratio of the same coauthors between a and c in p's coauthor names
6	Ratio of the same coauthors between a and c in c's coauthor names
7	Frequency of a's affiliation in c's affiliations
8	Ratio of a's affiliation in c's affiliations
9	Cosine similarity between a's affiliation and c's affiliations
10	Jaccards similarity between a's affiliation and c's affiliations
11	Distinct number of venues of c
12	Frequency of p's venue in c
13	Ratio of p's venue in c
14	Cosine similarity between p's venue and c's venues
15	Jaccards similarity between p's venue and c's venues
16	Cosine similarity between p's title and c's titles
17	Jaccards similarity between p's title and c's titles
18	Distinct number of keywords in c
19	Frequency of p's keywords of c
20	Ratio of p's keywords in c
21	Cosine similarity between p's keywords and c's keywords
22	Jaccards similarity between p's keywords and c's keywords

Adversarial Fine-tuning Module The two-layers MLP for the domain discriminator in Eq.(9) is defined as:

$$\text{MLP}(\mathbf{X}) = \mathbf{W}_2^T\text{LeakyReLU}(\mathbf{W}_1^T\mathbf{X}), \tag{14}$$

where $\mathbf{W}_1^T \in \mathbb{R}^{512\times100}$ and $\mathbf{W}_2^T \in \mathbb{R}^{100\times2}$. Likewise, the classifier $\tilde{h}$ of the Task Predictor is defined the same as above.

Training and Test Settings

Evaluation of the Pre-training Module. Pre-training is performed only on the AMiner dataset.

Training Settings. For the AMiner dataset shown in Table 1, we first filter the experts with less than 6 papers to satisfy the best number of the sampled papers for each expert. For each expert in the remaining AMiner dataset, we randomly sample $L = 6$ papers to comprise an expert instance. For each anchor expert instance, we sample 1 positive counterpart together with 9 negative counterparts, which results in 4,800 positive instance pairs and 43,200 negative instance pairs in total. For embedding a paper, we use [CLS] + title + keywords + name + organization + venue + [SEP] as the BERT input. The maximal length of the input tokens is set as 208.

Test Settings. Author Identification and Paper Clustering are two intrinsic tasks for testing the pre-training module.

- **Author Identification.** follows the second task of the name disambiguation competition[10]. Thus, we adopt the same test set as the competition. We randomly sample 17 negative experts together the ground truth expert to comprise the candidate list for a queried paper. The maximal number of papers sampled for each candidate expert is set as 100. For testing, we first leverage the pre-trained expert encoder g to generate the paper embedding for each candidate expert, then use the pre-trained metric function f to measure the similarity between a paper and each candidate expert. Finally, we rank all the candidate experts for a paper according to their similarity scores and return the top expert as the expert to be linked.
- **Paper Clustering.** follows the first task of the name disambiguation competition[11]. We directly use its test set. For testing, we apply the pre-trained expert encoder g to get the paper embeddings and then use the HAC algorithm to cluster the papers belonging to a same expert together. For a fair comparison, the true number of clusters in each name are provided to the HAC algorithm.

Implementation of baselines. The heuristic features of GBDT (Efimov, Silva, and Solecki 2013; Li et al. 2013) are specified in Table 6. Apart from GBDT, We implement other baselines following their released code and settings. The training and test settings of all the baselines are the same as CODE-pre.

- **Camel** (Zhang et al. 2018a): https://github.com/chuxuzhang/code_Camel_WWW2018
- **HetNetE** (Chen and Sun 2017): https://github.com/chentingpc/GuidedHeteEmbedding
- **CONNA** (Chen et al. 2020a): https://github.com/BoChen-Daniel/TKDE-2019-CONNA
- **louppe at el** (Louppe et al. 2016): https://github.com/glouppe/paper-author-disambiguation
- **Zhang et al** (Zhang and Al Hasan 2017): https://github.com/baichuan/disambiguation_embedding
- **G/L-Emb** (Zhang et al. 2018b): https://github.com/neozhangthe1/disambiguation/

Both Camel and HetNetE define an additional loss function on the indirect relationships between a paper and an expert generated by the pre-defined meta-paths. We ignore this part for the following reasons. Since author identification in their work aim to predict the authors of an anonymous paper, name ambiguity is not the key problem to be tackled. Thus they can collect all the papers published in some related venues as the training data. The connectivity of the resultant heterogeneous graph is good enough to find the indirect relationships between any two nodes. However, to disambiguate experts with same names, we only collect the papers with the same author names. The resultant graph is too sparse to find the indirect relationships.

Regarding the different baselines in the two tasks. The two evaluation tasks pay attention to various objectives, where Author Identification matches papers with authors, but Paper Clustering solely matches papers. The distinguished goals require different baselines. Actually, we have tried the Paper Clustering baselines for Author Identification but obtained poor performance. The paper-to-paper pseudo label that under-performs the paper-to-expert pseudo label on Author Identification also indicates the representation learning is better to be close with the goal of downstream tasks. The proposed expert-to-expert pseudo label can represent experts as well as papers when contrasting experts based on the interactions of their papers, resulting in a good performance on both tasks.

Evaluation of the Fine-tuning Module We evaluate of the fine-tuning module on two external datasets, News and LinkedIn, by the extrinsic task of external expert linking. We first fine-tune CODE-pre on both AMiner and the unlabeled external dataset, and evaluate it on the manually-labeled test set.

News. The fine-tuning settings for news are as follows:

- **Training.** We use 20,658 news articles for fine-tuning. We divide the contextual text of a name in a news article into sentences and treat each sentence as a piece of support information. We extract six sentences before and after a name in a news article as the support information. The maximal length of the input tokens for both the shared encoder and the private encoder is set as 64.
- **Test.** We annotate 1,622 names in news articles with linkages to AMiner experts for testing. We choose the candidates by name variants and sample up to 100 papers for each candidate.

LinkedIn. The settings for LinkedIn are as follows:

- **Training.** We use 50,000 LinkedIn homepages for fine-tuning. We select three common semi-structured attributes including affiliation, skills and summary in a homepage as the support information. The affiliation and the concatenated keywords in skills are separate pieces of support information. The long-text summary is divided into multiple pieces of support information. The maximal length of the input tokens for both the shared encoder and the private encoder is set as 64.
- **Test.** We annotate 1,329 linkages between LinkedIn users and AMiner experts for testing. Other settings are the same as News.