# ERGAN: Generative Adversarial Networks for Entity Resolution

Jingyu Shao*, Qing Wang*, Asiri Wijesinghe*, Erhard Rahm†
*Research School of Computer Science, the Australian National University
{Jingyu.Shao, Qing.Wang and Asiri.Wijesinghe}@anu.edu.au
†Database Group, University of Leipzig
Rahm@informatik.uni-leipzig.de

***Abstract*—Entity resolution targets at identifying records that represent the same real-world entity from one or more datasets. A major challenge in learning-based entity resolution is how to reduce the label cost for training. Due to the quadratic nature of record pair comparison, labeling is a costly task that often requires a significant effort from human experts. Inspired by recent advances of generative adversarial network (GAN), we propose a novel deep learning method, called ERGAN, to address the challenge. ERGAN consists of two key components: a label generator and a discriminator which are optimized alternatively through adversarial learning. To alleviate the issues of overfitting and highly imbalanced distribution, we design two novel modules for diversity and propagation, which can greatly improve the model generalization power. We have conducted extensive experiments to empirically verify the labeling and learning efficiency of ERGAN. The experimental results show that ERGAN beats the state-of-the-art baselines, including unsupervised, semi-supervised, and unsupervised learning methods.**

***Index Terms*—Entity Resolution, Generative Adversarial Nets, Imbalanced Class Problem**

## I. Introduction

Entity Resolution (ER) is an important and ubiquitous component of real-world applications in various fields, such as national census, health sector, crime and fraud detection, bibliographic statistics, and online shopping [4]. Learning-based ER methods have been widely used in the past years. However, due to the quadratic nature of record pair comparison required by ER tasks [3], labeling is costly, time consuming, and highly imbalanced. This raises the difficulty of applying supervised learning methods for ER in many real-world applications.

To reduce the labeling effort, a number of semi-supervised learning methods have been proposed [11], [22], [19]. Some of them are proposed based on a low-density separation assumption, i.e. there exists a low-density "boundary" so that instances belonging to different classes can be distinguished [1], [13]. However, such a boundary may not always exist or can be clearly identified, especially when the number of labeled instances is small [8]. Some semi-supervised learning methods have ultilized the idea of self-learning, which firstly trains a classifier using labeled instances, and then selects unlabeled instances with predicted labels to train a classifier iteratively [11]. Although promising, these methods often lead to the issue of overfitting when labeled instances in training are limited [16].

In this paper, we focus on tackling the following two challenges, which cannot be handled by the existing ER methods: (1) **the overfitting problem**; (2) **the imbalanced class problem**. The overfitting problem happens when the number of labeled instances is limited and a learning model is powerful enough to remember all the features of training instances. In such cases, the learning model can correctly predict the classes of seen instances with high certainty, but fail to predict the classes of unseen instances, thus losing the generalization ability. For the imbalanced class problem, it is due to the fact that the number of matches (record pairs referring to the same entity) is far less than the number of non-matches in ER tasks. Traditionally, blocking techniques can help alleviate the imbalanced class problem by grouping potentially matched instances into the same cluster. However, selecting a blocking method also requires prior knowledge or sufficient training instances [21], [18], which is still hard to achieve under a very limited number of training instances.

Generative adversarial network (GAN) and its variants have recently emerged as a powerful deep learning technique for real-world applications across various domains such as image generation and natural language processing [7], [6]. Inspired by these advances, in this paper, we develop a novel generative adversarial network, called ERGAN, to solve the aforementioned challenges faced by ER applications. In ERGAN, there are two key components: (1) a *label generator* $G$ that aims to generate pseudo labels for unlabeled instances, and (2) a *discriminator* $D$ that aims to distinguish instances with pseudo labels from instances with real labels. The discriminator $D$ is trained using not only a small number of instances with real labels but also a large number of instances with high-quality pseudo labels. However, the question arises: how to ensure the high-quality of pseudo labels generated for unlabeled instances? Unfortunately, the existing GAN and its variants cannot guarantee this when the number of instances with real labels is limited. To address this question, our model ERGAN is designed to incorporate two modules: *diversity module* and *propagation module* into the label generator $G$ and the discriminator $D$, respectively. The diversity module enables the diversity of unlabeled instances during the sampling process, while the propagation module guarantees that only unlabeled instances with high-quality pseudo labels can

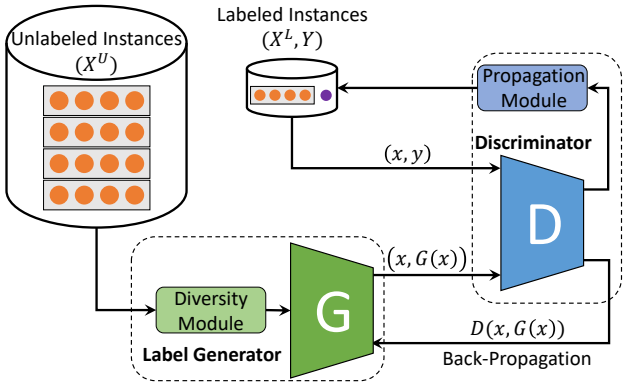


Fig. 1: **Overview of our framework.** Using only a limited number of labeled instances for training, ERGAN takes unlabeled instances as input and classifies them as being matches or non-matches.

be propagated into the training of the discriminator $D$. Then, $G$ and $D$ converge to the equilibrium point, achieving the global optimality.

Figure 1 shows an overview of ERGAN. It is worthy to note that, although we only consider ERGAN for entity resolution in this paper, the techniques of ERGAN for handing overfitting and imbalanced data can be much more widely applicable.

## II. Problem Formulation

Let $R$ be an ER dataset consisting of a set of records where each $r \in R$ is associated with a number of attributes $A$. Each record pair $(r_i, r_j)$ in $R$ corresponds to a feature vector $x$ where each element of $x$ indicates a feature value, e.g., the textual similarity of values in an attribute in $A$.

Let $X = \{x^{(ij)} | (r_i, r_j) \subseteq R \times R\}$ be the set of all feature vectors (i.e., instances) corresponding to record pairs in $R$ and $Y = \{M, N\}$ be a label space, where $M$ and $N$ refer to two labels *match* and *non-match*, respectively. There is a small subset $X^L \subseteq X$ of instances that are labeled, while the other instances in $X$ are unlabeled, i.e., $X^U = X - X^L$. We assume $|X^L| \ll |X^U|$, i.e., $X$ has a very limited number of labeled instances in $X^L$ but a large number of unlabeled instances in $X^U$. We denote $(X^L, Y)$ as a set of instances in $X^L$ and their labels in $Y$, and $(x^L, y) \sim (X^L, Y)$ as a pair of instance $x^L \in X^L$ and its label $y \in Y$. Our task is to tackle the ER classification problem as formulated below.

**Definition 1.** *Given a set $X$ of instances with $X = X^L \cup X^U$ and $|X^L| \ll |X^U|$, and a label space $Y = \{M, N\}$, the **ER classification problem** is to learn a model $\Lambda$ that can predict a label $\hat{y} \in Y$ for each unlabeled instance $x \in X^U$ w.r.t.*

$$max\ E(\Lambda)/|X^U| \tag{1}$$

*where* $E(\Lambda) = \sum_{x \in X^U \wedge \hat{y} = y} 1.$

Intuitively, $E(\Lambda)$ refers to the total number of unlabeled instances in $X^U$ whose labels are correctly classified by $\Lambda$.

## III. Proposed Method: ERGAN

Our proposed method ERGAN consists of two components: (1) a *label generator* $G$; and (2) a *discriminator* $D$. Both $G$ and $D$ are differentiable functions.

### A. Label Generator

In ERGAN, a label generator $G$ can obtain instances from $p(X^U)$, but does not know about $p(Y)$ nor $p(X, Y)$. Nevertheless, we know that $p(X^U) \approx p(X)$ because $X^U \subseteq X$ and $|X^U|/|X|$ is close to 1. The goal of $G$ is to learn a conditional distribution $p_g(Y|X^U) \approx p(Y|X^U)$, i.e., given an instance $x \sim p(X^U)$ as input, $G$ generates a pseudo label $\hat{y}$ for $x$. Ideally, the pseudo label $\hat{y}$ generated for an instance $x$ by $G$ should be the same as the real label of $x$. To simulate the conditional distribution $p(Y|X^U)$, the label generator $G$ receives feedback (i.e. gradients) from the discriminator $D$ and is trained iteratively through backpropagation.

**Diversity module.** One major difference of our ERGAN from the original GAN and its variants such as CatGAN [20] is that we consider the diversity of instances in the minibatch sampling process. More specifically, for all instances in $X$, we partition them into a number of non-overlapping subspaces alike in certain features $\{X_1, \ldots, X_b\}$ such that instances in the same subspace are more similar than those in different subspaces. Accordingly, labeled instances in $X^L$ and unlabeled instances in $X^U$ are partitioned into these $b$ subspaces, i.e., $X_i^L = X_i \cap X^L$ and $X_i^U = X_i \cap X^U$.

Let $\mathbf{v} = (\mathbf{v}_1, ..., \mathbf{v}_b)$ be a vector corresponding to $b$ subspaces, where each $\mathbf{v}_i = (v_i^1, \ldots, v_i^{n_i})^T \in [0, 1]^{n_i}$ and $n_i = |X_i^U|$. That is, each $v_i^j$ $(1 \le j \le n_i)$ is associated with an instance in $X_i^U$. Then, a minibatch of $m$ instances is selected from $X^U$ according to the following objective function:

$$\text{maximize} \quad ||\mathbf{v}||_{2,1} \quad \text{s.t.} \sum_{i,j} v_i^j = m \tag{2}$$

where $||\mathbf{v}||_{2,1}$ is a $l_{2,1}$-norm function defined as:

$$||\mathbf{v}||_{2,1} = \sum_{i=1}^{b} ||\mathbf{v}_i||_2 = \sum_{i=1}^{b} \sqrt{\sum_{j=1}^{n_i} {v_i^j}^2} \tag{3}$$

Here, $||\mathbf{v}_i||_2$ is the $l_2$-norm of $\mathbf{v}_i$. When $v_i^j = 1$, the instance in $X_i^U$ corresponding to $v_i^j$ is selected into the minibatch; otherwise, that instance is not selected. When the value of the $l_{2,1}$-norm is small, instances are selected from a small number of subspaces in $X^U$ and the diversity of instances is low. Conversely, when maximizing the $l_{2,1}$-norm in Eq. 2, instances are selected from as many subspaces in $X^U$ as possible and the diversity of instances is high.

**Objective function of $G$.** After a minibatch of unlabeled instances is selected from $X^U$ according to Eq. 2, the label generator $G$ generates a pseudo label $G(x_i)$ for each unlabeled instance $x_i$ in the minibatch. Then, $(x_i, G(x_i))$ is sent to the discriminator $D$. After receiving the gradient from $D$, $G$ updates its parameters according to the following objective:

$$\mathcal{L}_G = \min_G \quad \mathbb{E}_{x \sim p(X_i^U)}[\log(1 - D(x, G(x)))] \tag{4}$$

### B. Discriminator

Unlike GAN, a discriminator $D$ in our ERGAN does not know about the real distribution $p(X, Y)$. Instead, $D$ has access only to a limited number of instances with real labels, i.e. $(X^L, Y)$. The goal of $D$ is to distinguish whether a labeled instance $(x, G(x))$ is from the real distribution $p(X, Y)$, i.e., given a pair $(x, G(X))$ as input, $D$ generates a scalar value in $[0, 1]$ to indicate the probability that $G(x)$ is the same as the real label $y$ of $x$.

**Propagation module.** To achieve the above goal, as opposite to GAN and its variants in which the discriminator has the true distribution $p(X, Y)$, $D$ in ERGAN is designed to approximate the true joint distribution $p(X, Y)$ progressively through a propagation module. The general principle of propagation is that, the more confident the pseudo label $G(x)$ of an instance $x$ is the same as its real label $y$, the more likely such an instance is selected. Specifically, let $(X^t, G(X^t))$ denote all unlabeled instances with their pseudo labels at the $t$-th iteration of propagation. These instances are fed to $D$ to obtain their scores $D(X^t, G(X^t))$ that indicates the probabilities of their pseudo labels being the same as their real labels. Based on the scores, a subset $\Delta X^t \subseteq X^t$ of instances is selected according to the following objective function:

$$\begin{aligned} &\underset{\Delta X^t \subseteq X^t}{\operatorname{argmax}} \sum_{x \in \Delta X^t} D(x, G(x)) \\ &\text{subject to } |\Delta X^t| = \gamma \end{aligned} \tag{5}$$

where $\gamma$ is a hyper-parameter for the number of unlabeled instances being selected in the t-th iteration of propagation.

Then, this subset of instances with their high-quality pseudo labels $(\Delta X^t, \hat{Y})$ is propagated into the set of labeled instances $(X^*, Y)^t$ to train $D$, i.e.,

- $(X^*, Y)^0 = (X^L, Y)$
- $(X^*, Y)^t = (X^*, Y)^{t-1} \cup (\Delta X^t, \hat{Y})$

Hence, at the t-th iteration of propagation, $D$ has access to $(X^*, Y)^t$, which is a mixed set of labeled instances from $X^L$ (with real labels) and unlabeled instances from $X^U$ (with pseudo labels generated by $G$). The following holds:

$$(X^*, Y)^0 \subseteq (X^*, Y)^1 \subseteq \cdots \subseteq (X^*, Y)^t \tag{6}$$

Figure 2 shows an example of the propagation in two iterations, where the grey dash line indicates a boundary between two classes (red and blue) and is learned through propagation.

**Objective function of $D$.** The objective function of $D$ at the t-th iteration of propagation is defined as:

$$\begin{aligned} \mathcal{L}_D = \max_D \quad & \mathbb{E}_{x \sim p(X_i^U)} \log[(1 - D(x, G(x)))] \\ & + \lambda \mathbb{E}_{(x,y) \sim (X^*, Y)^t} \log[D(x, y)] \end{aligned} \tag{7}$$

where $\lambda \in [0, 1]$. In the following, we will explain how unlabeled instances with their pseudo labels, i.e., $(X^t, \hat{Y})$, is selected at the t-th iteration of propagation.

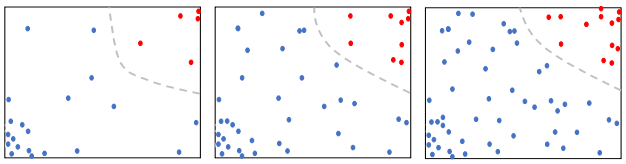


Fig. 2: **An illustration for propagation of ERGAN.** A boundary between two classes (red and blue) is learned through propagation.

### C. Choice of Hyper-parameters

Our algorithm is described in Algorithm 1. The number of subspaces $b$ is decided based on the attributes in each dataset. Suppose that a dataset has four attributes, we first obtain the median value for each attribute, and then partition instances into $4^2 = 16$ subspaces according to whether attribute values of each instance are above or below the median values of these four attributes [19]. $n$ is a hyper-parameter referring to the number of iterations for converging $G$ and $D$, and $t$ is decided by the total number $X^U$ of unlabeled instances and the number $\gamma$ of instances being propagated in each iteration, i.e. $t = \lceil \frac{|X^U|}{\gamma} \rceil$.

**Algorithm 1:** Minibatch stochastic gradient descent and label propagation of ERGAN

**Input:** $b$ subspaces in $X$; $X^U$; $(X^L, Y)$;
**Output:** $(X^*, Y)^t$ where $X^* = X$

Initialize $t = 0$; $(X^*, Y)^0 = (X^L, Y)$; $X^0 = X^1 = X^U$
**while** $X^t \neq \emptyset$ **do**
  **for** $n$ iterations **do** `// Batch training`
    Sample a minibatch $\{x_1, ..., x_m\}$ from $X^U$ w.r.t. Eq. 2
    Generate pseudo labels $\{(x_1, G(x_1)), ..., (x_m, G(x_m))\}$
    Sample a minibatch $\{(x_1^L, y_1), ..., (x_m^L, y_m)\}$ from $(X^*, Y)^t$
    Update the parameters of $D$ w.r.t. Eq. 7
    Update the parameters of $G$ w.r.t. Eq. 4
  t=t+1 `// Label propagation`
  Generate pseudo labels for $X^t$
  Select $\Delta X^t \subseteq X^t$ for propagation w.r.t. Eq. 5
  $(X^*, Y)^t = (X^*, Y)^{t-1} \cup (\Delta X^t, \hat{Y})$; $X^{t+1} = X^t - \Delta X^t$

## IV. Experimental Setup

We evaluate ERGAN to answer: 1) How does ERGAN perform in comparison with the state-of-the-art unsupervised, semi-supervised and fully supervised methods? 2) How do the design choices such as the diversity module, the propagation module, and GAN's architecture affect the performance of ERGAN?

**Datasets.** Four datasets are used in our experiments whose characteristics are summarized in Table I: the first three

TABLE I: **Characteristics of datasets.** The instances of these datasets are generated from their record pairs.

| Dataset | #Attributes ($\|A\|$) | #Instances ($\|X\|$) | Imbalance Rate | #Subspaces ($b$) |
|---|---|---|---|---|
| Cora | 4 | 837,865 | 1:49 | 16 |
| DBLP- ACM | 4/4 | 6,001,104 | 1:2,698 | 16 |
| DBLP- Scholar | 4/4 | 168,112,008 | 1:71,233 | 16 |
| NCVoter | 18/18 | 1,000,000 | 1:4,202 | 64 |

datasets contain the bibliographic records[1] and the last one contains real-world voter registration information of people from North Carolina in the USA[2].

**Baselines.** We compare ERGAN with the following baselines, whose details are described in Section VI: (1) *Unsupervised methods:* **Two-Steps** (**2S**) [3] and **Iterative Term-Entity Ranking and CliqueRank** (**ITER-CR**) [23]. (2) *Semi-supervised methods:* **Semi-supervised Boosted Classifier** (**SBC**) is the state-of-the-art semi-supervised learning method [11]. (3) Several state-of-the-art *fully supervised methods:* **Logistic Regression (LR)** and **Support Vector Machine (SVM)** are two supervised classifiers provided in Magellan that is an open-source ER solution [12]. **eXtreme Gradient boosting** (**XGboost**) is an ensemble learning based method [2]. **DeepMatcher** (**DM**) is a deep learning based approach specified for ER [14]. **Deep Transfer active learning** (**DTAL**) combines both transfer learning and active learning for handling ER tasks [10].

To compare with the baselines that use word embeddings, we use **ERGAN+WE** to refer to the model of ERGAN augmented with word embeddings for attribute values. In our ablation study, we use **ERGAN-D** and **ERGAN-P** to refer to a model being obtained by removing the diversity and propagation modules from ERGAN, respectively, and **ERNN** a model in which the GAN architecture (i.e. $G$ and $D$ are trained alternatively) is replaced by a single multi-layer perceptron for semi-supervised learning with the diversity module. We set $\lambda = 1$, $m \leq 100$, and $\gamma = |X^*|$. Our models use the same word embedding and similarity comparison techniques as the baselines.

**Measures.** We use the widely used F-Measure(FM) in ER tasks for performance evaluation [4].

## V. Experimental Results

We discuss the results of our experiments in this section.

### A. Performance Comparison

**Task 1.** We conduct an experiment to evaluate how our methods perform against the baselines. Following the previous work for the supervised methods DM [14] and DTAL [10], we split the datasets with 60% for training and the rest for testing.

Table II shows the results of the experiment, where the last three methods DM, DTAL and ERGAN+WE are deep-learning methods which use word embeddings for attribute values and the other methods use Jaccard similarity for comparing attribute values (without using word embeddings). We can see that, the unsupervised method 2S performs the worst among all the methods. However, the other unsupervised method ITER-CR performs better than SBC, SVM and LR due to its ability to leverage graph based structure. Compared with the fully supervised methods, the semi-supervised method SBC performs better than LR, comparably with SVM, but worse than XGBoost. Our method ERGAN performs better than any non-deep-learning method, but worse than the deep-learning methods with word embedding, i.e., DM, DTAL and ERGAN+WE. Nonetheless, our method ERGAN+WE outperforms all the baseline, including two deep-learning methods DM and DTAL, over all databases they have the results.

TABLE II: **Experimental results of f-measure with 60% training.** The results marked by $*$ are taken from the original papers and the others are obtained by running the code provided by the authors.

| Method | Datasets | | | |
|---|---|---|---|---|
| | Cora | DBLP-ACM | DBLP-Scholar | NCVoter |
| 2S [3] | 62.69 | 91.43 | 68.78 | 98.96 |
| ITER-CR* [23] | 89.00 | – | – | – |
| SBC [11] | 85.71 | 97.09 | 85.47 | 99.78 |
| SVM [12] | 88.95 | 97.19 | 85.71 | 98.48 |
| LR [12] | 80.25 | 95.56 | 83.84 | 99.37 |
| XGBoost [2] | 91.34 | 97.20 | 86.63 | **100** |
| ERGAN | **93.03** | **98.23** | **88.32** | **100** |
| DM [14] | 98.58 | 98.29 | 94.68 | **100** |
| DTAL* [10] | $98.68_{\pm 0.26}$ | $98.45_{\pm 0.22}$ | $92.94_{\pm 0.47}$ | – |
| ERGAN+WE | $\mathbf{98.72}_{\pm 0.15}$ | $\mathbf{98.51}_{\pm 0.23}$ | $\mathbf{94.73}_{\pm 0.35}$ | **100** |

**Observation 1.** *With sufficient training data,* ERGAN+WE *performs better than* ERGAN *due to the power of word embedding for records.* ERGAN+WE *has superior performance against all the baselines consistently.*

**Task 2.** To study performance under a limited number of instances with real labels, we further conduct an experiment using only a small percentage for training, ranging from 0.1% to 10% of the datasets, and the rest for testing.

Figure 3 shows the experimental results. ERGAN performs best among all the methods over all the datasets when training data is below 1%. ERGAN+WE performs poorly in this range. However, the performance of ERGAN+WE increases rapidly with increasing training data and exceeds all the other methods on all the datasets when training data reaches 10%. LR and DM have a similar trend as ERGAN+WE, but perform significantly worse. The semi-supervised method SBC performs better than ERGAN+WE only when training data is small, i.e. below 0.2% for Cora and below 0.9% for DBLP-ACM, DBLP-Scholar and NCVoter. The performance of SVM and XGBoost varies in datasets, i.e., perform well on Cora and DBLP-ACM, but badly on DBLP-Schloar and NCVoter. This demonstrates that the performance of SVM and XGBoost is sensitive to the imbalance rate of a dataset, and they fail to handle imbalanced data when no sufficient training data is available. For NCVoter, due to a clear boundary existing between matches and non-matches in the underlying distribution, the performance of all the methods that perform poorly for small training data can be dramatically improved

[1] Available from: *http://secondstring.sourceforge.net*

[2] Available from: *http://alt.ncsbe.gov/data/*

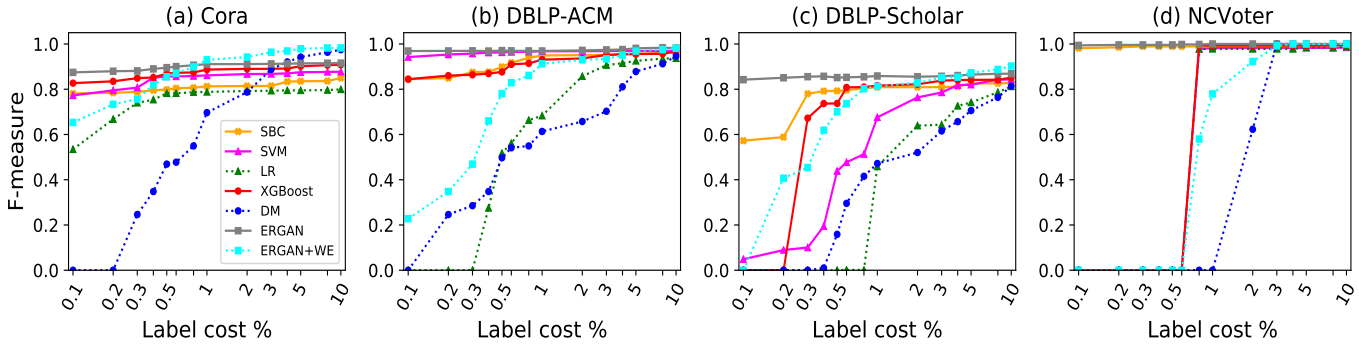


Fig. 3: **Experimental results of f-measure with 0.1% – 10% training.**

after using 0.6% or more training data. In general, we may conclude that, compared with the case of 60% training in Table II, the performance gain of the methods with word embedding against the methods without word embedding does not exist anymore. Instead, the methods with word embedding performs worse than most of the methods without word embedding when training data is small, i.e., below 1%.

**Observation 2.** *When decreasing training data,* ERGAN+WE *gradually performs worse than* ERGAN. *This is because,* ERGAN+WE *transforms instances into a high dimensional space through word embedding and thus requires much more labels in training than* ERGAN.

### B. Ablation Analysis

We conduct an ablation study to evaluate the effects of the key components of ERGAN, including the adversarial learning architecture, the diversity module and the propagation module, under different label costs, ranging from 0.1% to 60% for training. The results are presented in Table III. We observe that the performance of all the methods ERNN, ERGAN-D, ERGAN-P and ERGAN become stable and gradually converge when the label cost increases, e.g. in the case of 60% training. Nonetheless, ERGAN performs the best among all the methods, and the performance of the other methods varies in different datasets. In the following, we will discuss how each key component of ERGAN may affect the performance.

*Adversarial learning architecture.* The performance of ERNN generally lies in between ERGAN-D and ERGAN-P, and significantly worse than ERGAN. This indicates that the use of adversarial learning architecture by ERGAN helps improve the performance. Particularly when training data is limited, e.g., for 0.1% training, ERGAN improves around 3% on Cora and more than 8% on DBLP-ACM upon ERNN.

*Diversity module.* In Table III, the results of ERGAN-D are the worst among all the methods over all the datasets. This indicates that diverse instances are more informative for model training, which can improve the label efficiency. Specifically, with 0.1% training, ERGAN-D fails to work (i.e., f-measure value is 0) on three datasets except for Cora. This is because ERGAN-D lacks the diversity module and can only randomly select instances for training. As a result, all training instances are selected from the majority class (non-matches), and accordingly no matched instance can be classified correctly by ERGAN-D, i.e. all the instances are classified as non-matches. Since datasets in ER applications are usually highly imbalanced, training data without diversity may hardly contain instances from the minority class (matches) when labels are limited, thus leading to poor performance.

*Propagation module.* Table III shows that ERGAN-P generally has better performance than ERNN and ERGAN-D, and thus it may affect the performance of ERGAN least compared with the other two key components: the adversarial learning architecture and the diversity module, especially when the label cost is small, e.g. 0.1% and 1% training. Additionally, when the label cost is 60%, the performance of ERGAN-P and ERGAN is the same. This is because instances with real labels in 60% training data can provide sufficient information for learning, and the propagation of instances with pseudo labels becomes unnecessary.

**Observation 3.** *In* ERGAN*, all the three key components, i.e., the adversarial learning architecture, the diversity module and the propagation module, are necessary, each serving as an integral part of the entire framework.*

## VI. Related Work

Learning-based Entity Resolution (ER) approaches usually adopt a learning model to classify whether two records refer to the same entity. A recent supervised learning based approach is *Magellan* [12], which considered learning models including *Decision Tree*, *Random Forest* and *Support Vector Machine* (SVM). A widely used ensemble classifier is *extreme gradient boosting* (XGBoost) [2], which used the sparsity-aware algorithm and the weighted quantile sketch for approximate learning. One approach under unsupervised learning for ER is called *two-steps* (2S) [3], which first labeled a number (e.g. 10 percents of a dataset) of samples based on the similarity of record pairs, i.e. most similar and dissimilar ones, and then trained an SVM. A recent work is proposed by Jurek et. al. [9], which considered both ensemble learning and automatic self-learning for classification based on training labels which are automatically generated from different similarity measure schemes. The state-of-the-art semi-supervised learning approach is an ensemble learning based approach using ensemble

TABLE III: **Experimental results of f-measure with 0.1%, 1%, 20% and 60% training for ablation analysis.**

| Datasets | Cora | | | | DBLP-ACM | | | | DBLP-Scholar | | | | NCVoter | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0.1% | 1% | 20% | 60% | 0.1% | 1% | 20% | 60% | 0.1% | 1% | 20% | 60% | 0.1% | 1% | 20% | 60% |
| ERNN | 84.46 | 90.67 | 91.43 | 92.78 | 88.05 | 95.68 | 98.20 | 98.22 | 82.76 | 83.17 | 86.71 | 87.73 | 99.39 | 100 | 100 | 100 |
| ERGAN-D | 79.87 | 85.14 | 91.27 | 92.97 | 0 | 93.30 | 97.16 | 98.21 | 0 | 78.85 | 83.43 | 88.29 | 0 | 99.58 | 100 | 100 |
| ERGAN-P | 85.18 | 90.76 | 91.42 | 93.03 | 92.67 | 95.96 | 98.21 | 98.23 | 83.43 | 85.34 | 86.55 | 88.32 | 99.39 | 99.79 | 100 | 100 |
| ERGAN | 87.45 | 91.07 | 91.54 | 93.03 | 96.89 | 96.93 | 98.22 | 98.23 | 84.23 | 85.85 | 86.86 | 88.32 | 99.45 | 100 | 100 | 100 |

classifier Adaboost [17] for label prediction based on seed samples that have real labels.

Generative adversarial network (GAN) was proposed by Goodfellow et. al. [7]. The key idea of GAN is that two networks, a *generator* and a *discriminator*, play a minimax game so that they converge gradually to an optimal solution. The generator aims to generate fake instances to "fool" the discriminator by simulating the distribution of real instances, while the discriminator targets to distinguish fake instances (generated by the generator) from real instances.

In recent years, several attempts have been made to design deep learning solutions for ER tasks [14], [5]. Ebraheem et al. proposed DeepER, which uses bi-directional Recurrent Neural Networks (RNNs) with Long Short Term Memory (LSTM) units to learn a distributed representation for each record [5]. Mudgal et al. studied how to use deep learning techniques developed in natural language processing to handle the problems of attribute embedding, attribute summarization and attribute comparison [14]. A recent work proposed by Nie et al. [15] uses an align-compare-aggregate framework for a token level sequence-to-sequence ER which aims to solve the heterogeneous and dirty data problems.

## VII. Conclusion

In this paper, we have proposed a novel method, called ERGAN, to solve the ER classification problem with limited labeled instances. ERGAN incorporates the diversity of instances into sampling, prior to training the models. ERGAN consists of a label generator $G$ to generate pseudo labels for unlabeled instances, and a discriminator $D$ to distinguish instances with pseudo labels from instances with real labels. This method can be extended with word embedding for handling attribute values, leading to an enhanced method, called ERGAN+WE. Our experimental results show that the performance of our methods beats all the baselines.

## Acknowledgment

This work was partially funded by the Australian Research Council (ARC) under Discovery Project DP160101934.

## References

[1] Sugato Basu, Arindam Banerjee, and Raymond Mooney. Semi-supervised clustering by seeding. In *International Conference on Machine Learning (ICML)*, 2002.

[2] Tianqi Chen and Carlos Guestrin. Xgboost: a scalable tree boosting system. In *international conference on Knowledge Discovery and Data mining (SIGKDD)*, 2016.

[3] Peter Christen. Automatic record linkage using seeded nearest neighbour and support vector machine classification. In *international conference on Knowledge Discovery and Data mining (SIGKDD)*, 2008.

[4] Peter Christen. *Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection*. Springer Science & Business Media, 2012.

[5] Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. Distributed representations of tuples for entity resolution. *Proceedings of the VLDB Endowment*, 11(11):1454–1467, 2018.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems (NeurIPS)*, pages 2672–2680, 2014.

[8] Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

[9] Anna Jurek, Jun Hong, Yuan Chi, and Weiru Liu. A novel ensemble learning approach to unsupervised record linkage. *Information Systems*, 71:40–54, 2017.

[10] Jungo Kasai, Kun Qian, Sairam Gurajada, Yunyao Li, and Lucian Popa. Low-resource deep entity resolution with transfer and active learning. *arXiv preprint arXiv:1906.08042*, 2019.

[11] Mayank Kejriwal and Daniel P Miranker. Semi-supervised instance matching using boosted classifiers. In *European Semantic Web Conference*, pages 388–402. Springer, 2015.

[12] Pradap Konda, Sanjib Das, AnHai Doan, Adel Ardalan, Jeffrey R Ballard, Han Li, Fatemah Panahi, Haojun Zhang, Jeff Naughton, and Shishir Prasad. Magellan: toward building entity matching management systems over data science stacks. *Proceedings of the VLDB Endowment*, 9(13):1581–1584, 2016.

[13] Xiang Li, Yao Wu, Martin Ester, Ben Kao, Xin Wang, and Yudian Zheng. Semi-supervised clustering in attributed heterogeneous information networks. In *International Conference on World Wide Web (WWW)*, pages 1621–1629, 2017.

[14] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. Deep learning for entity matching: A design space exploration. In *Proceedings of the 2018 International Conference on Management of Data*, pages 19–34. ACM, 2018.

[15] Hao Nie, Xianpei Han, Ben He, Le Sun, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong. Deep sequence-to-sequence entity matching for heterogeneous entity resolution. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 629–638, 2019.

[16] Dorian Pyle. *Data preparation for data mining*. morgan kaufmann, 1999.

[17] Gunnar Rätsch, Takashi Onoda, and K-R Müller. Soft margins for adaboost. *Machine learning*, 42(3):287–320, 2001.

[18] Jingyu Shao and Wang Qing. Active blocking scheme learning for entity resolution. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, 2018.

[19] Jingyu Shao, Qing Wang, and Fangbing Liu. Learning to sample: an active learning framework. In *International Conference on Data Mining (ICDM)*, 2019.

[20] Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint:1511.06390*, 2015.

[21] Qing Wang, Mingyuan Cui, and Huizhi Liang. Semantic-aware blocking for entity resolution. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 28(1):166–180, 2016.

[22] Qing Wang, Dinusha Vatsalan, and Peter Christen. Efficient interactive training selection for large-scale entity resolution. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 562–573. Springer, 2015.

[23] Dongxiang Zhang, Long Guo, Xiangnan He, Jie Shao, Sai Wu, and Heng Tao Shen. A graph-theoretic fusion framework for unsupervised entity resolution. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 713–724. IEEE, 2018.