Interpretable and Low-Resource Entity Matching via Decoupling Feature Learning from Decision Making

Zijun Yao[1,2] **Chengjiang Li**[1,2] **Tiansi Dong**[3] **Xin Lv**[1,2] **Jifan Yu**[1,2]
Lei Hou[1,2*] **Juanzi Li**[1,2] **Yichi Zhang**[4] **Zelin Dai**[4]
[1]Department of Computer Science and Technology, BNRist;
[2]KIRC, Institute for Artificial Intelligence
Tsinghua University, Beijing 100084, China
[3]B-IT, University of Bonn, Germany
[4]Alibaba Group, Hangzhou, China
{yaozj20@mails, houlei@}.tsinghua.edu.cn
dongt@bit.uni-bonn.de

Abstract

Entity Matching (EM) aims at recognizing entity records that denote the same real-world object. Neural EM models learn vector representation of entity descriptions and match entities end-to-end. Though robust, these methods require many annotated resources for training, and lack of interpretability. In this paper, we propose a novel EM framework that consists of Heterogeneous Information Fusion (HIF) and Key Attribute Tree (KAT) Induction to decouple feature representation from matching decision. Using self-supervised learning and mask mechanism in pre-trained language modeling, HIF learns the embeddings of noisy attribute values by inter-attribute attention with unlabeled data. Using a set of comparison features and a limited amount of annotated data, KAT Induction learns an efficient decision tree that can be interpreted by generating entity matching rules whose structure is advocated by domain experts. Experiments on 6 public datasets and 3 industrial datasets show that our method is highly efficient and outperforms SOTA EM models in most cases. Our codes and datasets can be obtained from https://github.com/THU-KEG/HIF-KAT.

1 Introduction

Entity Matching (EM) aims at identifying whether two records from different sources refer to the same real-world entity. This is a fundamental research task in knowledge graph integration (Dong et al., 2014; Daniel et al., 2020; Christophides et al., 2015; Christen, 2012) and text mining (Zhao et al., 2014). In real applications, it is not easy to decide whether two records with ad hoc linguistic descriptions refer to the same entity. In Figure 1, e_2 and e_3 refer to the same publication, while e_1 refers to a different one. *Venue*s of e_2 and e_3 have different expressions; *Author*s of e_3 is misplaced in its *Title* field.

	Title	Author	Venue	Conference (redundant)
e_1	Data Mining Techniques	missing	SIGMOD Conference	International Conference on Management of Data
e_2	Data Mining: Concepts and Techniques	J. Han, J. Pei, M. Kamber	SIGMOD Record	missing
e_3	Data mining: Concepts & Techniques by Jiawei Han	misplaced	ACM SIGMOD Record	missing

Figure 1: Published papers as entity records.

Early works include feature engineering (Wang et al., 2011) and rule matching (Singh et al., 2017; Fan et al., 2009). Recently, the robustness of Entity Matching has been improved by deep learning models, such as distributed representation based models (Ebraheem et al., 2018), attention based models (Mudgal et al., 2018; Fu et al., 2019, 2020), and pre-trained language model based models (Li et al., 2020). Nevertheless, these modern neural EM models suffer from two limitations as follows.

Low-Resource Training. Supervised deep learning EM relies on large amounts of labeled training data, which is extremely costly in reality. Attempts have been made to leverage external data via transfer learning (Zhao and He, 2019; Thirumuruganathan et al., 2018; Kasai et al., 2019; Loster et al., 2021) and pre-trained language model based methods (Li et al., 2020). Other attempts have also been made to improve labeling efficiency via active learning (Nafa et al., 2020) and crowdsourcing techniques (Gokhale et al., 2014; Wang et al., 2012). However, external information may introduce noises, and active learning and crowdsourcing still require additional labeling work.

Lack of Interpretability. It is important to know why two entity records are equivalent (Chen et al., 2020), however, deep learning EM lacks inter-

* Corresponding to L.Hou (houlei@tsinghua.edu.cn)

pretability. Though some neural EM models analyze the model behavior from the perspective of attention (Nie et al., 2019), attention is not a safe indicator for interpretability (Serrano and Smith, 2019). Deep learning EM also fails to generate interpretable EM rules in the sense that they meet the criteria by domain experts (Fan et al., 2009).

To address the two limitations, we propose a novel EM framework to decouple feature representation from matching decision. Our framework consists of Heterogeneous Information Fusion (HIF) and Key Attribute Tree (KAT) Matching Decision for low-resource settings. HIF is robust for *feature representation* from noisy inputs, and KAT carries out interpretable decisions for entity matching.

In particular, HIF learns from unlabeled data a mapping function, which converts each noisy attribute value of entity into a vector representation. This is carried out by a novel self-supervised attention training schema to leverage the redundancy within attribute values and propagate information across attributes.

KAT Matching Decision learns KAT using decision tree classification. After training, KAT carries out entity matching as a task of the classification tree. For each entity pair, it first computes multiple similarity scores for each attribute using a family of metrics and concatenates them into a comparison feature vector. This classification tree can be directly interpreted as EM rules that share a similar structure with EM rules derived by domain experts.

Our EM method achieves at least SOTA performance on 9 datasets (3 structured datasets, 3 dirty datasets, and 3 industrial datasets) under various extremely low-resource settings. Moreover, when the number of labeled training data decreases from 60% to 10%, our method achieves almost the same performance. In contrast, other methods' performances decrease greatly.

The rest of the paper is structured as follows. Section 2 defines the EM task; Section 3 presents HIF and KAT-Induction in details; Section 4 reports a series of comparative experiments that show the robustness and the interpretability our methods in low-resource settings; Section 5 lists some related works; Section 6 concludes the paper.

2 Task Definitions

Entity Matching. Let T_1 and T_2 be two collections of entity records with m aligned attributes $\{\mathcal{A}_1, \cdots \mathcal{A}_m\}$. We denote the i^{th} attribute values of entity record e as $e[\mathcal{A}_i]$. Entity matching aims to determine whether e_1 and e_2 refer to the same real-world object or not. Formally, entity matching is viewed as a binary classification function $T_1 \times T_2 \rightarrow \{True, False\}$ that takes $(e_1, e_2) \in T_1 \times T_2$ as input, and outputs $True$ ($False$), if e_1 and e_2 are matched (not matched).

Current neural EM approaches simultaneously embed entities in low-dimensional vector spaces and obtain entity matching by computations on their vector representations. Supervised deep learning EM relies on large amounts of labeled training data, which is time-consuming and needs costly manual efforts. Large unlabelled data also contain entity feature information useful for EM, yet has not been fully exploited by the existing neural EM methods. In this paper, we aim at decoupling feature representation from matching decision. Our novel EM model consists of two sub-tasks: learning feature representation from unlabeled data and EM decision making.

Feature Representation from Noisy Inputs. Entity records are gathered from different sources with three typical noises in attribute values: *misplacing*, *missing*, or *synonym*. *Misplacing* means that attribute value of $\mathcal{A}_i$ drifts to $\mathcal{A}_j (i \neq j)$; *missing* means that attribute values are empty; *synonym* means that attribute values with the same meaning have different literal forms. Our first task is to fusion noisy heterogeneous information in a self-supervised manner with unlabelled data.

Interpretable EM. Domain experts have some valuable specifications on EM rules as follow: (1) an EM rule is an *if-then* rule of feature comparison; (2) it only selects a part of key attributes from all entity attributes for decision making; (3) feature comparison is limited to a number of similarity constraints, such as $=, \approx$ (Fan et al., 2009; Singh et al., 2017). Our second task is to realize an interpretable EM decision process by comparing feature representation per attribute by utilizing a fixed number of quantitative similarity metrics and then training a decision tree using a limited amount of labeled data. Our interpretable EM decision making will ease the collaboration with domain experts.

3 Methodology

In this section, we introduce (1) a neural model, Heterogeneous Information Fusion (HIF), for the task of feature representation, and (2) a decision

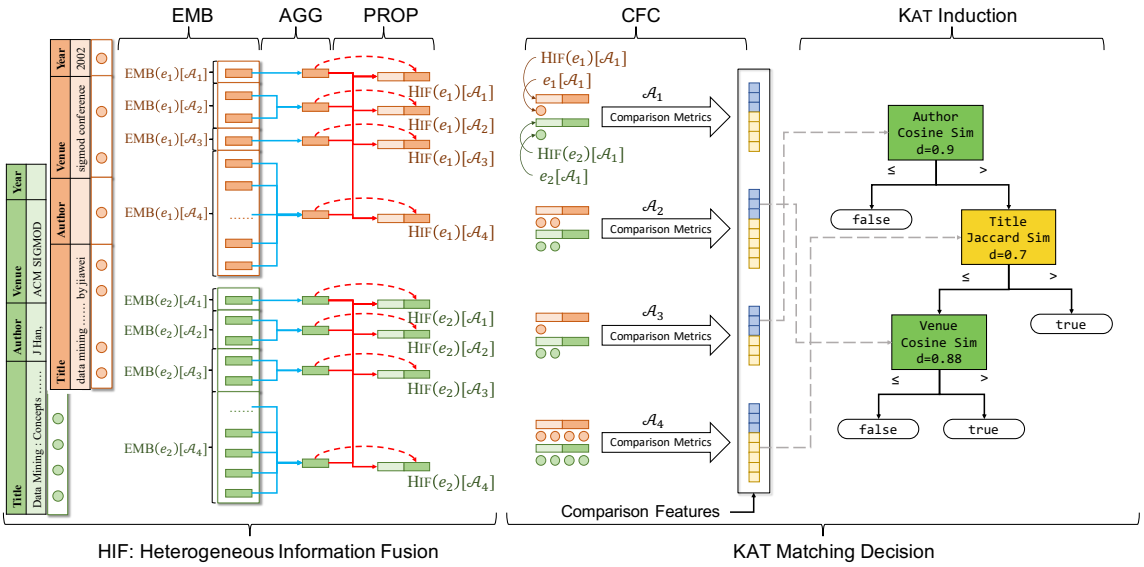

Figure 2: The decoupled EM model comprising the heterogeneous information fusion module and the matching decision making module. We use circles and rectangles to denote words and vectors, respectively. Cyan lines with arrow indicate word information aggregation via intra-attribute attention. Red lines with arrow show attribute information propagation. In the comparison features vector, blue squares are similarity scores by comparing on $\text{HIF}(e_1)[\mathcal{A}_i]$, $\text{HIF}(e_2)[\mathcal{A}_i]$ and yellow squares are similarity scores by comparing on $e_1[\mathcal{A}_i], e_2[\mathcal{A}_i]$ directly. EMB, AGG, PROP, CFC, and KAT-Induction are calculation components specified in Section 3.

tree, Key Attribute Tree (KAT), for the task of interpretable EM. Figure 2 illustrates the overall workflow of our method. The following subsections dive into details of the two tasks and propose a novel training scheme for low resource settings by exploiting unlabelled entity records.

3.1 HIF for Entity Attribute Embedding

$\text{HIF} : T \rightarrow \mathbb{R}^{m \times d}$ is a function that maps entity records into vector representations. An attribute value $e[\mathcal{A}_i]$ of a record e is mapped to a d dimensional vector, written as $\text{HIF}(e)[\mathcal{A}_i] \in \mathbb{R}^d$. HIF treats attribute values as strings of words and performs word embedding (EMB), word information aggregation (AGG), and attribute information propagation (PROP) successively.

Word Embedding (EMB). Word embedding is a pre-train language model that contains features learned from a large corpus. We convert numerical and encoded attribute values into strings of digits or alphabets. For Chinese attribute values, we do word-segmentation using *pkuseg* (Luo et al., 2019). Then, we mark the beginning and the end of an attribute value with two special tokens, namely $\langle\text{BEG}\rangle$ and $\langle\text{END}\rangle$. Finally, we pad each attribute value with $\langle\text{PAD}\rangle$ so that they are represented in the same length l. The representation after padding is illustrated as below:

$$\underbrace{(\langle\text{BEG}\rangle, w_1, w_2, \cdots \langle\text{END}\rangle, \langle\text{PAD}\rangle, \cdots, \langle\text{PAD}\rangle)}_{\text{length} = l}$$

Let W be the set of words, each word $w \in W$ is mapped into a vector, and each attribute value is mapped into a matrix. Formally, $\text{EMB} : W^N \rightarrow \mathbb{R}^{N \times d_e}$ maps N words into an $N \times d_e$ matrix by executing a look-up-table operation. N is the dictionary size. In particular, we have $\text{EMB}(e)[\mathcal{A}_i] \in \mathbb{R}^{l \times d_e}$, in which d_e is the dimension of word embedding vectors. It is worth noting that $\langle\text{PAD}\rangle$ is embedded to zero vector to ensure that it does not interfere with other non-padding words in the following step.

Word Information Aggregation (AGG). Summing up the l word embeddings as the embedding of an attribute value will neglect the importance weight among the l words. We leverage a more flexible framework, which aggregates word information by weighted pooling. The weighting coefficients α_i for different words are extracted by multiplying its embedding vector with a learnable, and attribute-specific vector $\mathbf{a}_i \in \mathbb{R}^{d_e \times 1}$. Subscript i implies that α_i and $\mathbf{a}_i$ are associated with the i^{th} attribute $\mathcal{A}_i$. The weighting coefficients are normalized by *Softmax* function among words. Finally, we enable a non-linear transformation (e.g.,

ReLU) during information aggregation with parameters $\mathbf{W}_{ai} \in \mathbb{R}^{d_e \times d_a}$. Formally, AGG maps each attribute value of entity record e into a d_a dimensional vector $\text{AGG}(\text{EMB}(e)[\mathcal{A}_i]) \in \mathbb{R}^{d_a}$ as below:

$$\text{AGG}(\text{EMB}(e)[\mathcal{A}_i]) = \text{ReLU}\left(\alpha_i \, \text{EMB}(e)[\mathcal{A}_i] \, \mathbf{W}_{ai}\right)$$

$$\alpha_i = \text{Softmax}(\text{EMB}(e)[\mathcal{A}_i] \, a_i)^\top \in \mathbb{R}^{1 \times l}$$

Attribute Information Propagation (PROP). The mechanism of attribute information propagation is the key component for noise reduction and representation unification. This mechanism is inspired by the observation that missing attribute values often appear in other attributes (e.g., *Venue* and *Conference* in Figure 1, Mudgal et al. (2018) also reported the misplacing issue).

We use "Scaled Dot-Product Attention" (Ashish et al., 2017) to propagate information among different attribute values. We use parameters $\mathbf{Q}, \mathbf{K}, \mathbf{V}_i$ to convert $\text{AGG}(\text{EMB}(e)[\mathcal{A}_i])$ into query, key, and value vectors, respectively (Notice that only $\mathbf{V}_i$ is attribute-specific). $\mathbf{A} \in \mathbb{R}^{m \times m}$ is the attention matrix. $\mathbf{A}_{ij}$ denotes the attention coefficients from the i^{th} attribute to the j^{th} attribute:

$$\mathbf{A}_{ij} = \text{Softmax}\left(\frac{\mathbf{q}_i \cdot \mathbf{k}_j}{\sqrt{m}}\right)$$
$$\mathbf{q}_i = \text{AGG}(\text{EMB}(e)[\mathcal{A}_i]) \, \mathbf{Q}$$
$$\mathbf{k}_j = \text{AGG}(\text{EMB}(e)[\mathcal{A}_i]) \, \mathbf{K}$$
$$\mathbf{v}_i = \text{AGG}(\text{EMB}(e)[\mathcal{A}_i]) \, \mathbf{V}_i$$

Record notation e is omitted in vectors $\mathbf{q}, \mathbf{k}, \mathbf{v}$ for brevity. To keep the identity information, each attribute value after attribute information propagation is represented by the concatenation of the context and the value vector:

$$\text{PROP}(\text{AGG}(e))[\mathcal{A}_i] = \text{ReLU}\left(\mathbf{v}_i \left\| \sum_{j \neq i} \mathbf{A}_{ij} \mathbf{v}_j \right.\right)$$

HIF outputs with Multiple Layer Perceptron (MLP). The whole process can be summarized as follows:

$$\text{HIF}(e) = \text{MLP} \circ \text{PROP} \circ \text{AGG} \circ \text{EMB}(e) \in \mathbb{R}^{m \times d}$$

After HIF, each attribute $\mathcal{A}_i$ of an entity record e has a feature embedding $\text{HIF}(e)[\mathcal{A}_i]$.

3.2 KAT for Matching Decision

KAT Matching Decision consists of two steps: comparison feature computation (CFC) and decision making with KAT. CFC computes similarity score for each paired attribute features by utilizing a family of well-selected metrics, and concatenate these similarity scores into a vector (comparison feature). KAT takes comparison feature as inputs, and perform entity matching with a decision tree.

Comparison Feature Computing (CFC). Given a record pair (e_1, e_2), CFC implements a function that maps (e_1, e_2) to a vector of similarity scores $\text{CFC}(e_1, e_2)$. The similarity score $\text{CFC}(e_1, e_2)$ is a concatenation of a similarity vector between paired attribute values (i.e., $e_1[\mathcal{A}_i], e_2[\mathcal{A}_i]$) and a similarity vector between their vector embeddings (i.e., $\text{HIF}(e_1)[\mathcal{A}_i], \text{HIF}(e_2)[\mathcal{A}_i]$).

To compare paired attribute values, we follow Konda et al. (2016) and classify attribute values into 6 categories, according to the type and the length, each with a set of comparison metrics for similarity measurement. For example, for Boolean attributes, we compare their attribute value by inspecting whether they are exactly match; for numerical attributes, we compute their absolute difference as well as their string distance. More details are presented in Appendix A.

For attribute value embeddings, we choose three metrics: the cosine similarity, which is the normalized projection distance; the L_2 distance, which measures the distance of two vectors in the finite dimensional Hilbert space; and the Pearson coefficient, which further normalized the cosine similarity by the mean and the variance. In this way, we convert entity record pair into similarity score vector of attributes. Each dimension indicates the similarity degree of one attribute from a certain perspective.

KAT Induction. In the matching decision, we take $\text{CFC}(e_1, e_2)$ as input, and output binary classification results. We propose Key Attribute Tree, a decision tree, to make the matching decision based on *key attribute* heuristic, in the sense that some attributes are more important than others for EM. For example, we can decide whether two records of research articles are the same by only checking their *Title* and *Venue* without examining their *Conference*. Focusing only on key attributes not only saves computations, but also introduces interpretability that has two-folded meanings: (1) each dimension of $\text{CFC}(e_1, e_2)$ is a candidate feature matching which can be interpreted as a component of an EM rule; (2) the decision tree learned by

KAT can be converted into EM rules that follow the same heuristics as the EM rules made by domain experts (Fan et al., 2009).

3.3 Model Training

HIF and KAT Induction are trained separately.

HIF Training. We design a self-supervised training method for HIF to learn from unlabeled data. Our strategy is to let the HIF model predict manually masked attribute values. We first represent attribute values, as strings of words, by Weighted Bag Of Words (WBOW) vectors, whose dimensions represent word frequencies. Then, we manually corrupt a small portion of entity records in $T_1 \cup T_2$ by randomly replacing (mask) their attribute values with an empty string, which forms a new table T'. HIF takes T' as input and uses another MLP to predict the WBOW of masked attribute values. HIF is trained by minimizing the Cross-Entropy between the prediction and the ground-truth WBOW:

$$\min_{\text{HIF}} \text{CrossEntropy}\left(\text{MLP}(\text{HIF}(T')), \text{WBOW}\right)$$

KAT Induction Training. KAT is trained with a normal decision tree algorithm. We constrain its depth, in part to maintain the interpretability of transformed EM rules. We use xgboost (Tianqi and Carlos, 2016) and ID3 algorithm (Quinlan, 1986) in the experiments. To preserve interpretability, the booster number of xgboost is set to 1, which means it only learns one decision tree. For $(e_1, e_2, True) \in D$, KAT takes CFC(e_1, e_2) as input, and $True$ as the target classification output.

4 Experiments

4.1 Experimental Setup

4.1.1 Datasets

In order to evaluate our model comprehensively, we collect multi-scaled datasets ranging from English corpus and Chinese corpus, including *Structured* datasets, *Dirty* datasets, and *Real* datasets. *Structured* and *Dirty* datasets are benchmark datasets[1] released in (Mudgal et al., 2018). The *Real* datasets are sampled from a real E-commerce platform, a portion of which are manually labeled to indicate whether they are the same entity or not. The *real* datasets have notably more attributes than the *structured* or *dirty* datasets.

[1] http://pages.cs.wisc.edu/~anhai/data1/deepmatcher_data/

Type	Dataset	#Attr.	#Rec.	#Pos.	#Neg.	Rate
Structured	I-A_1	8	2,908	132	407	10%
	D-A_1	4	4,739	2,220	10,143	1%
	D-S_1	4	13,270	5,347	23,360	1%
Dirty	I-A_2	8	2,908	132	407	10%
	D-A_2	4	4,739	2,220	10,143	1%
	D-S_2	4	13,270	5,347	23,360	1%
Real	Phone	36	940	1,099	2,241	10%
	Skirt	20	9,708	6,371	18,202	1%
	Toner	13	7,065	4,551	13,481	1%

Table 1: Statistics of the datasets. #Attr. is the number of attributes, #Rec. is the number of entity records, and #Pos. (#Neg.) is the number of labeled positive (negative) pairs. I-A indicates matching between iTunes-Amazon. D-A indicates matching between DBLP-ACM. D-S indicates matching between DBLP-Google Scholar. We use subscripts 1, 2 to distinguish between *Structured* and *Dirty* data.

Statistics of these datasets are listed in Table 1. We focus on setting of *low resource* EM and use Rate% of labelled data as training set. The validation set uses the last 20% labeled pairs, and the rest pairs in the middle are the test set. This splitting is different from the *sufficient resource* EM (Mudgal et al., 2018; Konda et al., 2016) where up to 60% pairs are used in the training set. For *I-A*$_1$, *I-A*$_2$, and *Phone*, we use 10% labeled pairs as training data, because some of the baselines will crash, if the training data is too small.

We remove trivial entity pairs from the *Real* datasets, as *Structured* and *Dirty* datasets have been released. For *Real* datasets, we remove matching pairs with large Jaccard similarity (0.32 for Phone, 0.36 for others) and non-matching pairs with small Jaccard similarity (0.3 for Phone, 0.332 for others).

4.1.2 Baselines

We implement 3 variants of our methods with different KAT Induction algorithms. **HIF+KAT$_{\text{ID3}}$** and **HIF+KAT$_{\text{XGB}}$** inducts KAT with ID3 algorithm and xgboost respectively constraining maximum depth to 3. **HIF+DT** inducts KAT with ID3 algorithm with no constraints on the tree depth. We include reproducibility details in Appendix B.

We compare our methods with three SOTA EM methods, among which two are publicly available end-to-end neural methods, and one is feature engineering based method.

1. **DeepMatcher** (Mudgal et al., 2018) (DM) is a general deep-learning based EM framework with multiple variants—*RNN* DM-RNN,

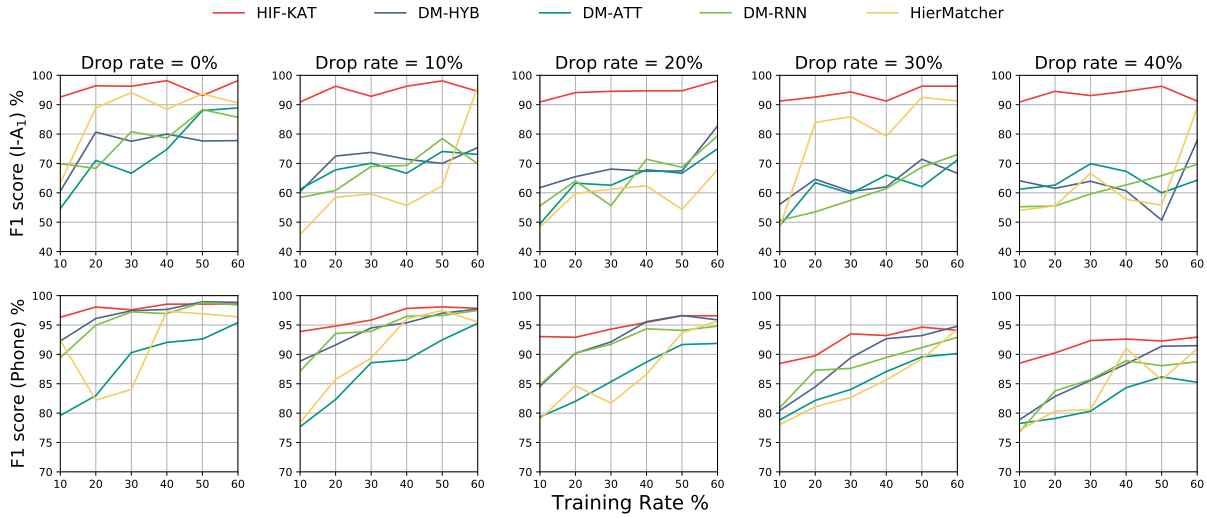

Figure 3: Results for robustness. HIF+KAT refers to HIF+KAT$_{XGB}$. Each two subgraphs in the same column correspond to the same drop rate (Drop rate is marked on the top of each column). Each five subgraphs in the same row correspond to the same dataset. x-axis is the rate of labelled data used in training. y-axis is the F_1 score.

Attention DM-ATT, and *Hybrid* DM-HYB—depending on what building block it chooses to construct[2].

2. **HierMatcher** (Fu et al., 2020) is also an end-to-end neural EM method that compare entity records at the word level[3].

3. **Magellan** (Konda et al., 2016) integrates both automatic feature engineering for EM and classifiers. Decision tree is used as the classifier of Magellan in our experiments.

For ablation analysis, we replace a single component of our model with a new model as follows: **HIF+LN** replaces KAT with a linear classifier; **HIF+LR** replaces KAT with a logistic regression classifier; **HIF-ALONE** removes comparison metrics of attribute values (yellow segment of comparison features in Figure 2). We also do ablation analysis for **HIF-ALONE** as follows: **HIF-WBOW** replaces outputs of HIF with d-dimensional WBOW vectors using PCA. **HIF-EMB** replaces the outputs of HIF with the mean pooling of word embeddings.

4.1.3 Evaluation Metrics

We use F_1 score as the evaluation metric. Experiment results are listed in Table 2 and Table 4. All the reported results are averaged over 10 runs with different random seeds.

[2] https://github.com/anhaidgroup/deepmatcher

[3] https://github.com/cipnlu/EntityMatcher

4.2 Experimental Results

General Results. We evaluate the performance of our model against 3 SOTA models under low resource settings, where only 1% or 10% of the total amount of labeled pairs are used for training (See Table 1). Comparative experiment results on the 9 datasets are listed in Table 2.

Our decoupled framework achieves SOTA EM results on all the nine datasets, and demonstrates significant performance on *Dirty* datasets, with a boosting of 4.3%, 14.7%, and 8.4% in terms of F_1 score on I-A_2, D-A_2, D-S_2, compared to the best performance of baselines on their corresponding datasets. Our methods also outperforms all baselines on *Structured* and two *Real* datasets (the same as Magellan on Toner). The out-performance on *Real* datasets is marginal because attribute values in *Real* datasets are quite standard, which means that our model does not have many chances to fix noisy attribute values. Still, our methods achieve a high F_1 score ($\geq$ 94.9%) in *Real* datasets. These results indicate out methods are both effective under low resource settings and robust to noisy data.

Effectiveness to Low Resource Settings We reduce the training rate from 60% to 10% to see whether our method is sensitive to the number of labeled record pairs as training resources. Experimental results are shown in Figure 3. HIF+KAT (red line) achieves a stable performance as the number of labeled record pairs decreases, while the F_1 score of DeepMatcher and HierMatcher decrease simultaneously. Besides, our methods continuously

Methods	I-A_1	D-A_1	D-S_1	I-A_2	D-A_2	D-S_2	Phone	Skirt	Toner
DM-RNN	63.6	85.4	74.8	42.3	45.7	39.0	90.0	67.6	68.6
DM-ATT	55.8	82.5	79.0	46.5	45.2	57.8	80.3	54.4	48.8
DM-HYB	60.9	86.6	78.0	49.5	46.2	60.4	91.9	64.2	67.4
HierMatcher	61.9	37.5	68.2	37.8	32.6	45.8	86.2	61.7	55.2
Magellan	92.3	93.7	85.1	50.6	65.6	71.1	93.6	96.6	**97.2**
HIF+DT	**96.0**	96.4	87.5	**54.9**	80.1	74.2	**94.9**	**96.7**	**97.2**
HIF+KAT_{ID3}	95.8	**96.6**	**88.2**	51.6	79.0	**79.5**	94.5	**96.7**	**97.2**
HIF+KAT_{XGB}	90.6	93.3	87.9	41.5	**80.3**	**79.5**	94.4	96.2	**97.2**
HIF+LN	77.9	21.0	54.7	41.6	-	78.5	72.2	62.8	86.0
HIF+LR	84.2	87.1	84.6	46.5	-	68.1	87.5	41.7	62.0
HIF-WBOW	93.0	92.7	75.4	43.2	47.9	43.7	91.6	66.3	74.0
HIF-EMB	91.1	90.9	76.6	30.8	53.9	46.8	89.9	65.7	79.8
HIF-ALONE	94.6	96.1	82.9	45.6	73.5	63.2	91.8	63.0	72.9

Table 2: F_1 score of all methods under low resource setting(%). Dash (-) indicates classifier fails to converge.

outperform DeepMatcher and HierMatcher, ranging from low resource setting to sufficient resource setting. These results indicate that by exploring unlabelled data, HIF alleviates the reliance on labeled record pairs.

Effectiveness to Noisy Heterogeneous Data. We manually aggravate the quality of datasets by randomly dropping $p\%$ of attribute values ($p\%$ ranges from 0% to 40%), and see to what degree the feature representations delivered by HIF will affect the EM decision matching. From left to right, columns of subgraphs in Figure 3 demonstrates results with increasing dropping rate. On the I-A_1 dataset, the influence of dropping rate is marginal to HIF+KAT , whose F_1 score fluctuates around 95%. In contrast, F_1 scores of both DeepMatcher and HierMatcher will decrease if more attribute values are dropped. On the Phone dataset, the dropping rate's influence is not severe to HIF+KAT, especially when the training rate is low. These results show that HIF is efficient in recovering noisy heterogeneous inputs.

4.3 Case Study for Interpretablity

The interpretability of our model means that the process of decision making of KAT can be easily transformed into EM rules whose structure is recommended by domain experts. Figure 4 illustrates a tree decision process of KAT that determines whether two records denote the same publication in the D-A_1 (DBLP and ACM) datasets. Each path from the root to a leaf node of the tree structure can

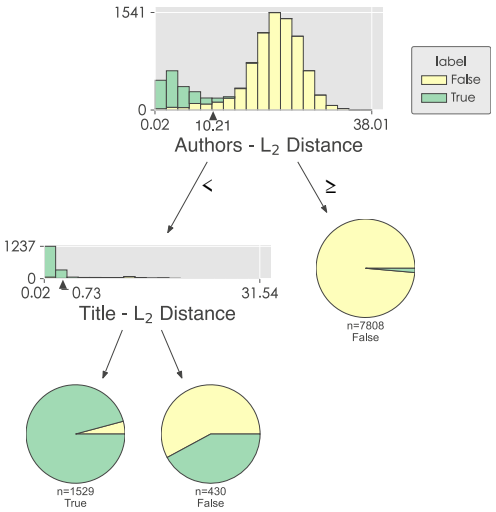

Figure 4: The Key Attribute Tree generated by HIF+KAT_{XGB} for D-A_1 dataset.

be converted into an EM rule as follows:

Rule 1: if $L_2\,(\text{HIF}(e_1), \text{HIF}(e_2))\,[\text{Authors}] \geq 10.21$
then e_1, e_2 are not a match;
Rule 2: if $L_2\,(\text{HIF}(e_1), \text{HIF}(e_2))\,[\text{Authors}] < 10.21$
$\wedge\, L_2\,(\text{HIF}(e_1), \text{HIF}(e_2))\,[\text{Title}] < 0.73$
then e_1, e_2 are a match;
Rule 3: if $L_2\,(\text{HIF}(e_1), \text{HIF}(e_2))\,[\text{Authors}] < 10.21$
$\wedge\, L_2\,(\text{HIF}(e_1), \text{HIF}(e_2))\,[\text{Title}] \geq 0.73$
then e_1, e_2 are not a match

They can be further read as descriptive rules:
Rule 1: if two records have different authors, they will be different publications.
Rule 2: if two records have similar authors and similar titles, they will be the same publication.
Rule 3: if two records have similar authors and dissimilar titles, they will not be the same publication.
The soundness of such rules can be examined by our experience.

Important features of KAT are as follows: (1) KAT is conditioned on attribute comparison; (2) KAT only selects a few key attributes to compare features. In our example, there are 4 attributes, *Author, Title, Venue* and *Conference* in D-A_1 dataset, KAT only selects *Title* and *Author* for EM decision making. The transformed rules meet the specifications of manually designed EM rules of domain experts (Fan et al., 2009; Singh et al., 2017). This kind of interpretability will ease the collaboration with domain experts, and increase the trustworthiness, compared with uninterpretable end-to-end Deep learning EM models.

4.4 Discussions

Ablation Analysis. Experiment results for ablation models are listed in Table 2. On the one hand, HIF+LN and HIF+LR generally outperforms DeepMatcher and HierMatcher on 7 datasets with on-par performance on 2 *Real* datasets. This indicates that HIF and CFC together extract better comparison features than end-to-end neural methods under low resource settings. On the other hand, HIF+LN and HIF+LR are weaker than the tree induction classifier, suggesting that KAT is more reliable.

Compared with HIF-KAT_{ID3}, Magellan, and HIF-ALONE, HIF-KAT_{ID3} achieves the highest performance, indicating that comparison on both attribute value embeddings and the original attribute values are important. Compared with HIF-ALONE, HIF-WBOW, and HIF-EMB, HIF-ALONE outperforms HIF-WBOW and HIF-EMB on the *Dirty* datasets, showing the positive effects of its information reconstruction.

Finally, comparing HIF+KAT with HIF+DT, we find that HIF+KAT has better performances than HIF+DT on most of the datasets, except for (I-A_2 and Phone). This shows that non-key attributes may disturb decision making.

Efficiency. Table 3 shows the running times of our methods and of the two neural baselines. Our methods are highly efficient for inference, because our methods are highly parallel and are memory-saving. For example, on Phone datasets our methods can inference in a single batch, while HierMatcher can only run in a batch size of 4 with 24GiB RAM. The training efficiency of our method is comparable with baselines, because when the training data is small enough, baseline models may finish one epoch training with only few batches.

Sufficient Resource EM. Table 4 shows the results with sufficient training data following the split method of Mudgal et al. (2018); Fu et al. (2020). Our method outperforms other methods on 4 datasets, and slightly fall behind on 5 datasets.

Epoch	I-A_1	D-A_1	D-S_1	Phone	Skirt	Toner
DM-HYB	0.98	1.0	2.3	12.7	5.1	2.5
HierMatcher	0.47	0.3	0.7	41.7	4.0	1.4
HIF+KAT_{ID3}	0.45	1.0	1.5	2.2	5.5	3.2
Train	I-A_1	D-A_1	D-S_1	Phone	Skirt	Toner
DM-HYB	86	434	958	1,418	2,984	1,473
HierMatcher	37	139	309	3,799	2,809	1,082
HIF+KAT_{ID3}	344	819	1,085	1,097	1,669	968
Test	I-A_1	D-A_1	D-S_1	Phone	Skirt	Toner
DM-HYB	2.4	31.7	67.1	56.9	229.6	113.9
HierMatcher	2.0	25.1	50.1	113.0	181.1	74.4
HIF+KAT_{ID3}	**0.4**	**1.0**	**1.4**	**2.2**	**5.4**	**3.1**

Table 3: (Epoch) Training time for one epoch & (Train) Training time until finish & (Test) Testing time. All the results are recorded in seconds.

Methods	I-A_1	D-A_1	D-S_1	I-A_2	D-A_2	D-S_2	Phone	Skirt	Toner
DM-RNN	83.1	98.8	93.5	67.1	94.8	89.6	98.2	91.6	90.9
DM-ATT	83.8	98.8	93.7	62.2	94.1	90.4	95.7	93.2	91.6
DM-HYB	83.5	98.8	95.0	64.0	95.9	92.6	98.7	94.2	92.0
HierMatcher	79.1	98.5	94.3	77.1	96.1	93.0	96.5	95.4	94.7
HIF+DT	95.5	97.6	91.7	60.0	87.8	77.1	97.5	99.7	99.8
HIF+KAT_{ID3}	95.9	98.1	90.2	59.3	89.7	80.5	94.9	99.3	99.6
HIF+KAT_{XGB}	95.5	98.1	90.1	63.3	89.3	80.4	96.5	99.7	99.9

Table 4: F_1 scores of all methods under sufficient resource setting(%).

5 Related Works

The way of extracting comparison features falls into two categories: monotonic and non-monotonic. Monotonic features are (negatively) proportional similarities between attribute values. They can be calculated by symbolic rules, such as Jaccard similarity, Levenshtein similarity (Fan et al., 2009; Wang et al., 2011; Konda et al., 2016; Singh et al., 2017), or learned from differentiable comparison operations, such as subtracting, point-wise multiplication (Fu et al., 2019; Ebraheem et al., 2018; Fu et al., 2019). Non-monotonic features are hidden representations of end-to-end neural networks, such as *Softmax* or *Sigmoid* based similarity scores (Fu et al., 2020), attention based scores (Nie et al., 2019), or simply embedding based features (Mudgal et al., 2018; Li et al., 2020).

EM with limited resources has recently intrigued research interest (Thirumuruganathan et al., 2018; Kasai et al., 2019). Existing explorations seek solution from leveraging external data to improving annotation efficiency. External data can be aggregated via transfer learning (Zhao and He, 2019; Thirumuruganathan et al., 2018; Kasai et al., 2019; Loster et al., 2021), or via pre-training language models (Li et al., 2020). For better annotations, researchers tried active learning (Kasai et al., 2019; Nafa et al., 2020; Sarawagi and Bhamidipaty, 2002; Arasu et al., 2010), or crowd sourcing techniques (Wang et al., 2012; Gokhale et al., 2014).

The interpretability of neural models will contribute to the trust and the safety. It has become

one of the central issues in machine learning. Chen et al. (2020) examines interpretability in EM risk analysis. There are also attempts to explain from the perspective of attention coefficients (Mudgal et al., 2018; Nie et al., 2019).

6 Conclusion

We present a decoupled framework for interpretable entity matching. It is robust to both noisy heterogeneous input and the scale of training resources. Experiments show that our method can be converted to interpretable rules, which can be inspect by domain experts and make EM process more reliable.

In the future, it is intriguing to explore more efficient ways to explore unlabeled data, such as levering connections among entities, or combine with pre-trained language models. It is also valuable to explore how to use our heterogeneous information fusion module to boost other EM methods, such as injecting HIF representation as supplementary information into end-to-end models.

Acknowledgments

This work is supported by Science and Technology Innovation 2030 - New Generation of Artificial Intelligence Project (2020AAA0106501), the NSFC Key Project (U1736204), the NSFC Youth Project (62006136), the Federal Ministry of Education and Research of Germany as part of the competence center for machine learning ML2R (01IS18038C), and the grant from Alibaba Inc.

Ethical Considerations

Intended Use. The reported technique is intended for reliable entity matching in large scale E-commercial products, where attribute values are mostly heterogeneous descriptive sentences. The 'low resource' feature is intended to avoid heavy labor force. The 'interpretability' is intended to risk control in entity matching.

Misuse Potential. As matching/alignment technique, our method may be misused in matching private information.

Failure Modes. Our method provides a promising way to have domain experts check the generated rules, thus reducing the failure risk.

Energy and Carbon Costs. The efficiency test in Section 4.4 shows that our method costs less computations and is more energy saving than existing methods.

References

Arvind Arasu, Michaela Götz, and Raghav Kaushik. 2010. On active learning of record matching packages. In *SIGMOD'10*.

Vaswani Ashish, Shazeer Noam, Parmar Niki, Uszkoreit Jakob, Jones lion, Aidan N. Gomez, Kaiser Lukasz, and Polosukhin Illia. 2017. Attention is all you need. In *NIPS'17*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *TACL*, 5.

Zhaoqiang Chen, Qun Chen, Boyi Hou, Zhanhuai Li, and Guoliang Li. 2020. Towards interpretable and learnable risk analysis for entity resolution. In *SIGMOD'20*.

P Christen. 2012. *Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection*. Springer: Data-centric systems and applications.

Vassilis Christophides, Vasilis Efthymiou, and Kostas Stefanidis. 2015. Entity resolution in the web of data. *Synthesis Lectures on the Semantic Web*.

Obraczka Daniel, Schuchart Jonathan, and Rahm Erhard. 2020. EAGER: embedding-assisted entity resolution for knowledge graphs. In *ICDM'20*.

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *SIGKDD'14*.

Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. 2018. Distributed representations of tuples for entity resolution. In *VLDB'18*.

Wenfei Fan, Xibei Jia, Jianzhong Li, and Shuai Ma. 2009. Reasoning about record matching rules. In *VLDB'09*.

Cheng Fu, Xianpei Han, Jiaming He, and Le Sun. 2020. Hierarchical matching network for heterogeneous entity resolution. In *IJCAI'20*.

Cheng Fu, Xianpei Han, Le Sun, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong. 2019. End-to-end multi-perspective matching for entity resolution. In *IJCAI'19*.

Chaitanya Gokhale, Sanjib Das, AnHai Doan, Jeffrey F. Naughton, Narasimhan Rampalli, Jude W. Shavlik, and Xiaojin Zhu. 2014. Corleone: hands-off crowdsourcing for entity matching. In *SIGMOD'14*.

Jungo Kasai, Kun Qian, Sairam Gurajada, Yunyao Li, and Lucian Popa. 2019. Low-resource deep entity resolution with transfer and active learning. In *ACL'19*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR'15*.

Pradap Konda, Sanjib Das, Suganthan G. C. Paul, AnHai Doan, Adel Ardalan, Jeffrey R. Ballard, Han Li, Fatemah Panahi, Haojun Zhang, Jeffrey F. Naughton, Shishir Prasad, Ganesh Krishnan, Rohit Deep, and Vijay Raghavendra. 2016. Magellan: Toward building entity matching management systems. In *VLDB'16*.

Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. 2020. Deep entity matching with pre-trained language models. In *VLDB'20*.

Michael Loster, Ioannis Koumarelas, and Felix Naumann. 2021. Knowledge transfer for entity resolution with siamese neural networks. *Journal of Data and Information Quality (JDIQ)*.

Ruixuan Luo, Jingjing Xu, Yi Zhang, Xuancheng Ren, and Xu Sun. 2019. Pkuseg: A toolkit for multi-domain chinese word segmentation. *CoRR*, abs/1906.11455.

Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. 2018. Deep learning for entity matching: A design space exploration. In *SIGMOD'18*.

Youcef Nafa, Qun Chen, Zhaoqiang Chen, Xingyu Lu, Haiyang He, Tianyi Duan, and Zhanhuai Li. 2020. Active deep learning on entity resolution by risk sampling. *CoRR*, abs/2012.12960.

Hao Nie, Xianpei Han, Ben He, Le Sun, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong. 2019. Deep sequence-to-sequence entity matching for heterogeneous entity resolution. In *CIKM'19*.

J. Ross Quinlan. 1986. Induction of decision trees. *Machine learning*, 1(1):81–106.

Sunita Sarawagi and Anuradha Bhamidipaty. 2002. Interactive deduplication using active learning. In *SIGKDD'02*.

Sofia Serrano and Noah A Smith. 2019. Is attention interpretable? In *ACL'19*.

Rohit Singh, Venkata Vamsikrishna Meduri, Ahmed K. Elmagarmid, Samuel Madden, Paolo Papotti, Jorge-Arnulfo Quiané-Ruiz, Armando Solar-Lezama, and Nan Tang. 2017. Generating concise entity matching rules. In *SIGMOD'17*.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional skip-gram: Explicitly distinguishing left and right context for word embeddings. In *NAACL'18*.

Saravanan Thirumuruganathan, Shameem A Puthiya Parambath, Mourad Ouzzani, Nan Tang, and Shafiq Joty. 2018. Reuse and adaptation for entity resolution through transfer learning. *CoRR*, abs/1809.11084.

Chen Tianqi and Guestrin Carlos. 2016. Xgboost: A scalable tree boosting system. In *SIGKDD'16*.

Jiannan Wang, Tim Kraska, Michael J. Franklin, and Jianhua Feng. 2012. Crowder: Crowdsourcing entity resolution. *VLDB'12*.

Jiannan Wang, Guoliang Li, Jeffrey Xu Yu, and Jianhua Feng. 2011. Entity matching: How similar is similar. In *VLDB*.

Chen Zhao and Yeye He. 2019. Auto-em: End-to-end fuzzy entity-matching using pre-trained deep models and transfer learning. In *WWW'19*.

Wayne Xin Zhao, Yuexin Wu, Hongfei Yan, and Xiaoming Li. 2014. Group based self training for e-commerce product record linkage. In *COLING'14*.

Attribute Type	Comparison Metrics
boolean	Exact matching distance
number	Exact matching distance, Absolute distance, Levenshtein distance, Levenshtein similarity
string of length $[1, 1]$	Levenshtein distance, Levenshtein similarity, Jaro similarity, Jaro Winkler similarity, Exact matching distance, Jaccard similarity with QGram tokenizer,
string of length $[2, 5]$	Jaccard similarity with QGram tokenizer, Jaccard similarity with delimiter tokenizer, Levenshtein distance, Levenshtein similarity Cosine similarity with delimiter tokenizer, Monge Elkan similarity, Smith Waterman similarity,
string of length $[6, 10]$	Jaccard similarity with QGram tokenizer, Cosine similarity with delimiter tokenizer, Levenshtein distance, Levenshtein similarity, Monge Elkan similarity
string of length $[10, \infty]$	Jaccard similarity with QGram tokenizer, Cosine similarity with delimiter tokenizer

Table 5: Comparison metrics for different types of attributes.

A Compairison Metrics

We classify attributes into 6 categories: boolean, number, natural language string of length $[1, 1]$, natural language string of length $[2, 5]$, natural language string of length $[6, 10]$, natural language string of length $[10, \infty]$. Each has a family of comparison metrics. These metrics are listed in Table 5.

We do not distinguish between 'distance' and 'similarity' as distance can be converted to similarity metric by taking its reciprocal.

B Reproducibility Details

Each epoch of HIF training is evenly divided into 3 batches. The *Title* attribute values were padded to $l = 64$, and the other attribute values are all padded to $l = 32$. We modify the padding size on large datasets, so that our the experiments can be conducted on a single GPU. Chinese datasets are embedded with Tencent Embedding (Song et al., 2018) and English datasets use fastText embeddings (Bojanowski et al., 2017). Multi-head mechanism is used in the attention module. The embedding size d_e for Chinese is 300, and for English is 200. AGG converts embedding into d_a dimensional vectors, where $d_a = 100$. PROP further outputs with a 2-layer MLP with dimension size $d = 64$. The query vector and the key vector in the attention layer of PROP are 16 dimensional vectors. During training, attribute values are masked at a probability $p = 0.4$. The Adam optimizer (Kingma and Ba, 2015) is used for HIF . Training rate and L_2 weight decay are 0.01 and 10^{-5}.

KAT_{XGB} is implemented using `xgboost 0.9` with objective function *binary: logistic*. KAT_{ID3} is implemented using `scikit-learn 0.24`. HIF is implemented with `PyTorch 1.4.0` in `Python 3.7.6`. The comparison feature metrics in Table 5 are implemented with `py-entitymatching 0.4.0`. We also use `Numpy 1.19.2` for matrix calculation. All the experiments are evaluated on a single NVIDIA 3090 GPU with 24GiB GRAM.

C More Experimental Results

Table 2 in the main text only shows the F_1 measure of the all the methods. Here, we supplement the experimental results with *precision* (P = $\frac{\text{TP}}{\text{TP+FP}}$), *recall* (R = $\frac{\text{TP}}{\text{TP+FN}}$) on the 9 datasets for more comprehensive analysis. Experimental results are listed in Table 6. Our methods achieve the highest precision and recall on most of the datasets.

Methods	I-A_1			D-A_1			D-S_1		
	P	R	F_1	P	R	F_1	P	R	F_1
DM-RNN	69.1	60.9	63.6	81.7	90.3	85.4	69.9	80.9	74.8
DM-ATT	54.2	58.4	55.8	75.3	91.2	82.5	75.0	83.5	79.0
DM-HYB	58.4	64.1	60.9	84.3	89.2	86.6	74.3	82.4	78.0
HierMatcher	64.1	61.8	61.9	41.6	38.9	37.5	72.1	67.2	68.2
Magellan	92.3	92.7	92.3	95.4	92.2	93.7	80.7	**90.2**	85.1
HIF+LN	84.1	73.0	77.9	15.0	97.1	21.0	**96.1**	44.3	54.7
HIF+LR	79.9	89.1	84.2	86.7	95.7	87.1	85.2	84.2	84.6
HIF+DT	**97.1**	**94.9**	**96.0**	**95.9**	97.0	96.4	90.0	85.1	87.5
HIF+KAT_{ID3}	**97.1**	94.7	95.8	95.8	**97.4**	**96.6**	87.8	88.7	**88.2**
HIF+KAT_{XGB}	87.7	94.0	90.6	91.1	95.7	93.3	88.4	87.4	87.9

Methods	I-A_2			D-A_2			D-S_2		
	P	R	F_1	P	R	F_1	P	R	F_1
DM-RNN	43.3	42.4	42.3	39.1	55.5	45.7	31.9	50.7	39.0
DM-ATT	46.4	50.4	46.5	42.5	48.3	45.2	55.5	60.4	57.8
DM-HYB	51.1	54.5	49.5	48.8	44.6	46.2	57.3	65.1	60.4
HierMatcher	41.2	43.9	37.8	48.5	27.8	32.6	50.4	44.1	45.8
Magellan	51.8	49.4	50.6	58.5	74.8	65.6	72.6	69.7	71.1
HIF+LN	54.1	34.0	41.6	-	-	-	73.1	84.7	78.5
HIF+LR	49.5	44.5	46.5	-	-	-	62.1	75.7	68.1
HIF+DT	**55.6**	**54.5**	**54.9**	**75.4**	85.5	80.1	77.8	70.9	74.2
HIF+KAT_{ID3}	50.6	53.4	51.6	73.6	85.4	79.0	81.9	**77.2**	**79.5**
HIF+KAT_{XGB}	35.9	51.0	41.5	**75.4**	**86.1**	**80.3**	**82.1**	77.1	**79.5**

Methods	Phone			Skirt			Toner		
	P	R	F_1	P	R	F_1	P	R	F_1
DM-RNN	88.1	92.1	90.0	62.3	73.8	67.6	60.3	80.8	68.6
DM-ATT	77.1	83.8	80.3	44.5	70.1	54.4	40.6	62.2	48.8
DM-HYB	93.9	90.1	91.9	55.6	76.1	64.2	55.0	87.3	67.4
HierMatcher	83.6	89.2	86.2	51.7	77.0	61.7	46.7	67.9	55.2
Magellan	95.1	92.1	93.6	96.1	**97.2**	96.6	96.7	**97.6**	**97.2**
HIF+LN	80.5	65.5	72.2	93.8	51.5	62.8	88.4	83.8	86.0
HIF+LR	**97.3**	80.0	87.5	**99.9**	26.4	41.7	62.6	89.8	62.0
HIF+DT	93.0	**97.0**	**94.9**	96.7	96.7	96.7	**97.6**	96.7	**97.2**
HIF+KAT_{ID3}	92.2	96.9	94.5	96.9	96.6	**96.7**	**97.6**	96.7	**97.2**
HIF+KAT_{XGB}	92.6	96.1	94.4	99.0	93.5	96.2	**97.6**	96.8	**97.2**

Table 6: Experimental results under low-resource setting with precision, recall, and F_1 measure (%). Dash (-) indicates these methods fail to converge on the datasets.