electronics

MDPI

Article

Entity Matching by Pool-Based Active Learning

Youfang Han and Chunping Li *

School of Software, Tsinghua University, Beijing 100084, China; fred.hanyf@goertek.com
* Correspondence: cli@tsinghua.edu.cn; Tel.: +86-10-6279-5437

Abstract: The goal of entity matching is to find the corresponding records representing the same entity from different data sources. At present, in the mainstream methods, rule-based entity matching methods need tremendous domain knowledge. Machine-learning-based or deep-learning-based entity matching methods need a large number of labeled samples to build the model, which is difficult to achieve in some applications. In addition, learning-based methods are more likely to overfit, so the quality requirements of training samples are very high. In this paper, we present an active learning method for entity matching tasks. This method needs to manually label only a small number of valuable samples, and use these labeled samples to build a model with high quality. This paper proposes hybrid uncertainty as a query strategy to find those valuable samples for labeling, which can minimize the number of labeled training samples and at the same time meet the requirements of entity matching tasks. The proposed method is validated on seven data sets in different fields. The experiments show that the proposed method uses only a small number of labeled samples and achieves better effects compared to current existing approaches.

Keywords: active learning; entity matching; machine learning; query strategy

Citation: Han, Y.; Li, C. Entity Matching by Pool-Based Active Learning. *Electronics* **2024**, *13*, 559. https://doi.org/10.3390/electronics13030559

Academic Editor: Hideaki IIduka

Received: 18 December 2023
Revised: 23 January 2024
Accepted: 29 January 2024
Published: 30 January 2024

1. Introduction

Entity matching aims to determine whether the corresponding data records in different data sources represent the same entity. For example, there are two data sets of resident information in Figure 1, the goal is to judge whether these two records represent the same resident from different data sources by comparing the attributes (e.g., name and age). Entity matching has a broad range of applications and there is a lot of work devoted to the development of the entity matching system [1,2]. For instance, Mellgan [3] is the most advanced open-source entity matching solution in recent years. It constructs a complete entity matching system, which can be directly used by users for data cleaning and data integration.

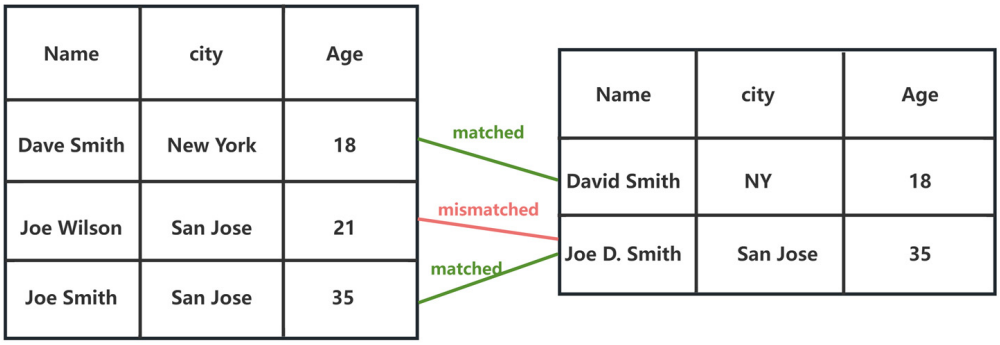

Figure 1. An example of entity matching.

At present, mainstream entity matching methods are mostly rule-based or learning-based [4]. Rule-based entity matching methods usually require users to have a certain

understanding of the data set, or need experts in the related domains to design the rules for better matching effects. These methods are sensitive to their selected similarity measurement method and the threshold, which usually needs specific design by domain experts [5–7]. Learning-based entity matching methods include machine-learning-based (ML-based) and deep-learning-based (DL-based) entity matching. The ML-based methods use the existing powerful machine learning algorithms to automatically learn the characteristics of matched entities [8,9]. DL-based methods [10,11] usually need pre-trained language models (PLM) and domain-related texts for fine-tuning.

Although machine learning and deep learning techniques have achieved good performance in entity matching tasks, they still have some limitations in practical application, as follows.

1. In many scenarios, it is difficult to obtain a larger number of labeled samples. Manual labeling requires a lot of labor-costing and time-costing, and usually difficult to obtain, adequate effective labels in a short time.
2. Entity matching tasks usually have extremely imbalanced data samples. Generally, the number of mismatched samples is much larger than that of matched samples. The binary classification with imbalanced label distribution may lead to insufficient training of matched samples.
3. For entity matching, there are less benefits in labeling too many entity pairs. For example, "journey to the west" and "pilgrimage to the west" represent the same entity, "dream of the Red Mansions" and "story of the stone" also represent the same entity. The former can be judged to be the same entity through simple character-level comparison, while the latter can be judged only through labeling by relevant experts. This phenomenon is very common for entity matching tasks. Many entity pairs can determine them to be matched or mismatched easily by comparison, so the benefit of labeling these data is very low. Therefore, if the records in different data sets are directly handed over to experts for pairwise labeling, there will be a lot of extraordinary workloads.
4. Although the deep learning method based on language models can achieve a good matching result, it usually needs suitable pre-trained and domain-related texts for fine-tuning. When encountering new domain problems, this method is difficult to achieve good results without pre-trained language models and domain-related knowledge.

In the paper, we propose an active learning method training the entity matching model to solve the above problems. Our method aims to label the least number but most meaningful samples while achieving higher accuracy. Active learning is a sub-field of machine learning. An active learning system attempts to label some unlabeled entities by asking Oracle (such as manual annotators) to overcome the bottleneck of scarce labels [12]. An effective active learning algorithm can achieve exponential acceleration of labeling efficiency [13,14]. Some application scenarios using this technology to solve the problem of lack of labeled samples or low quality samples, such as information classification [15] and medical analysis [16], etc. The active learning method puts forward some labeling requests and submits selected small number of samples to experts for labeling, which can not only greatly reduce the labeling-cost, but also improve the quality of labeled samples, reduce the impact of noise data, and promote the generalization ability of the learning model.

Here we present a pool-based active learning method integrating with query strategies, which can efficiently deal with the entity matching task on less labeled training samples without domain knowledge. We design a hybrid uncertainty as query strategy, which can select the most valuable samples from the unlabeled data pool for labeling. The proposed method does not need pre-trained language models or complicated pre-processing, and has good performance with less labeled training samples by traditional classifiers. In the small scale of data sets, the proposed method is even superior to the state-of-the-art deep learning methods.

The contributions of the paper are summarized as follows:

1. We propose a pool-based active learning method for entity matching tasks, which can find the most valuable labeled samples to build the learning model using only a small number of labeled samples and achieve good performance compared to existing methods. This work can effectively solve the problems in the acquisition of labeled samples and lack of domain knowledge in entity matching.
2. Our method integrates with query strategies to select valuable samples effectively from unlabeled samples for labeling. Experiment results show that the selected samples are highly representative and the method can effectively reduce the labeling workload.
3. We verified the performance of our method on seven public datasets. Compared to existing ML-based and DL-based methods, the proposed method can reach a similar F1 score while using only a small number of labeled samples. In the small scale of data sets, the proposed method is even superior to the state-of-the-art deep learning methods.

The remainder of this paper is organized as follows. Section 2 presents the related works for entity matching tasks. Section 3 introduces the proposed pool-based active learning method for entity matching, including data preprocessing, generating the initial labeled pool, query strategy, and stop criterion. We verify the performance of our method through numerous experiments in Section 4. Section 5 discusses the advantages, characteristics and limitations of our methods. We have the concluding remarks in Section 6.

2. Related Works

This section surveys the related works for entity matching tasks, including rule-based, ML-based, and DL-based entity matching.

2.1. Rule-Based Entity Matching

Rule-based entity matching methods usually need a lot of domain knowledge. With the support of domain experts, rule-based methods can perform well on most of entity matching tasks. For example, on secure data sharing [17], mobile edge search [18], and some other fields, the rule-based entity matching methods can well meet the task requirements. Moreover, Rohit Singh [19] proposed a powerful tool to automatically generate rules that satisfy a given high level specification, which makes rules easy to design.

When we cannot obtain complicated rules designed by experts, the method of Linearly Weighted Combination Rules (LWCR) [4] can be used. LWCR is a useful baseline method for Entity Matching, and our proposed method will utilize LWCR as an auxiliary for measuring the similarity of entity pairs. LWCR is used to weight the sum of the similarity values of each attribute, as shown in Formula (1).

$$\mathrm{Sim}(x, y) = \sum_{i=1}^{n} \alpha_i \cdot s_i(x, y) \quad (1)$$

In Formula (1), x and y represent the records from two different data sets, and n represents the number of attributes. α_i represents the preset weight, i.e., the importance of the i-th attribute. The sum of a_1 to a_n is 1. $s_i(x, y)$ represents the similarity of the i-th attribute of x and y, whose value is between 0 and 1. $s_i(x, y)$ usually adopts existing string-similarity measure methods, such as Levenshtein method [20], Jaro–Winkler method [21], and Jaccard method [22], etc.

The entity matching method based on LWCR has some limitations. First, it is difficult to set the weight of each attribute, which needs the user to know the importance of each attribute. Second, the relationship between attributes and similarity may not be linear. However, LWCR can obtain an approximate similarity of each record pair quickly without any domain knowledge. Although the similarity calculated by LWCR may not be accurate, it can provide an approximate result to judge whether the record pair is matched. This

approximate result will be used as the standard for pruning the training set and the standard for selecting the initial labeled data pool in the subsequent experiments.

2.2. Machine Learning-Based Entity Matching

At present, many works adopt machine-learning-based methods for entity matching. ML-based methods usually create the record pair from two different sources, and predict whether the record pair is matched. This subsection mainly introduces the general process of ML-based methods.

The ML-based methods can be used to predict whether the record pair is matched. We can train a model for prediction through the training set, and then apply the model to predict on the test set to judge whether the record pair is matched. Let the training set $T = \{(x_1, y_1, l_1), \ldots, (x_n, y_n, l_n)\}$. Both x_i and y_i represent a record pair, l_i represents the label in which "yes" is matched and "no" is mismatched.

To facilitate the training of the ML model, it is necessary to pre-process the record pair (x_i, y_i). Define a set of features f to quantify the record pairs, and then T can be transformed into $T' = \{(<f_1(x_1, y_1), \ldots, f_m(x_1, y_1)>, c_1), \ldots, (<f_1(x_n, y_n), \ldots, f_m(x_n, y_n)>, c_n)\}$. m indicates the number of extracted features, n indicates the number of groups of record pairs. c indicates the label, i.e., $c_i = 1$ indicates l_i = "yes", $c_i = 0$ indicates l_i = "no". The feature f is usually defined by the similarity of each attribute. After obtaining the transformed data set T′, we can apply existing ML models for training, such as SVM, random forest, and so on. The model M is obtained from the training set, and then input $<f_1(x_i, y_i), \ldots, f_m(x_i, y_i)>$ into M to obtain the result c_i, which is the result of predicting whether x_i and y_i are matched.

At present, a number of works are based on ML methods for entity matching tasks [23,24]. Ref. [25] compares the performance of different models on entity matching tasks. Enrico et al. [26] propose a stacking approach for threshold-based ML models, i.e., using integrated method to improve the prediction effect. Mugeni et al. [27] propose a k-nearest neighbor graph-based blocking approach for entity matching and the performance is even better than many DL-based methods. These works make the ML-based methods play an important role in entity matching tasks. The ML methods are suitable for most of application scenarios, but highly dependable on data quality. When the imbalanced sample distribution, noise samples, and less labeled samples occur in real applications, the effect of ML methods will be greatly degraded.

2.3. Deep Learning-Based Entity Matching

With the development of deep learning technology, the DL-based entity matching methods gradually show their advantages. At present, some works explore the applications of deep learning techniques in entity matching tasks, such as deep neural networks, large language models, and transfer learning, etc. This section will introduce some representative works.

DeepER [28] and DeepMatcher [29] are state-of-the-art DL methods that have achieved good performance on entity matching tasks. DeepER uses LSTM model with the Siamese architecture. DeepER is the first method that tokenizes each record pair using embedding technology, such as Glove [30] and fastText [31], and then aggregates token-level embeddings into an entity representation. DeepMatcher uses recurrent neural networks (RNN) to build a hybrid sequence-aware model with the attention mechanism. Unlike DeepER, DeepMatcher calculates the similarity of attributes from inputted record pairs to capture the similarity at the attribute level. Meanwhile, Sidharth [29] compares the results of different neural networks, such as SIF [32], RNN [33], and Attention [34]. Huang et al. [35] proposes a method combining deep learning and adversarial active learning for entity matching. These deep learning methods have achieved good performance in entity matching tasks. Although the DL-based methods can obtain better prediction results than traditional ML-methods, they need a lot of labeled data to construct the model. DL-based methods are difficult to achieve higher performance on small-scale, less-label data sets.

Moreover, some innovative methods use pre-trained language models (ΠLM) and domain-related texts for entity matching tasks. Ref. [36] proposes a novel entity matching system Ditto, which is based on pre-trained transformer. This method fine-tunes pre-trained LM Sentence-BERT [37] for entity matching tasks [38], which allows domain knowledge to be added by highlighting important pieces of the input that may be useful for matching decisions. Li et al. [39] propose a pre-trained Transformer language model for effective entity matching and obtaining higher F1 score on two large datasets. Refs. [40,41] also apply pre-trained language models to solve the entity matching task, which shows that a pre-trained language model can further improve the performance. In addition, Ref. [42] adopts transfer learning on entity matching. This method uses pre-trained entity matching models from large-scale knowledge bases (KB), and then fine-tunes a small number of samples to obtain a good prediction result. Although these kinds of methods can achieve better results than traditional ML methods, they all need the assistance of a pre-trained language model and domain-related knowledge. When the application scenario without domain knowledge or the pre-trained language model is not suitable, it is hard to obtain better effectiveness in real applications.

3. The Pool-Based Active Learning Method for Entity Matching

In this section, we present the proposed active learning method for entity matching, the purpose of which is to use a smaller number of labeled record pairs for prediction and to obtain good results and meet the requirements. We will introduce the overall process of the algorithm in Section 3.1, the pre-processing method in Section 3.2, the method of generating the initial labeled pool in Section 3.3, the query strategies in Section 3.4, and the stop criterion in Section 3.5.

Some of the symbols used in the subsequent sections are defined in Table 1. S represents the labeled data pool, i.e., the set of labeled record pairs, which is used to construct classifiers, and its size is $|S|$. M represents the maximum number of iterations, i.e., M labeling queries. Q represents all record pairs in the training set, and $|Q|$ is its size. Q_n indicates that a group of record pairs $\{q_1, q_2, \ldots, q_n\}$ are labeled in an iteration. $L(q_1)$, $L(q_2)$, …, $L(q_n)$ indicates that the labels of the record pairs in Q_n. Formula $L: Q \rightarrow \{0,1\}$ is regarded as the ground truth, i.e., the result labeled by experts, where 0 indicates mismatched and 1 indicates matched. V represents all record pairs in the validation set. The record pairs in V have been labeled. The data in V are used to evaluate the performance of the model in the iteration process of active learning and fine-tune the hyperparameters. T represents all record pairs in the test set, and the data in T are used for the evaluation of the final model. $F1(V, C_i)$ represents the F1 score calculated by the labeled data in V using the classifier C_i. The F1 score is used to measure the performance of classifier C_i on the test set and validation set. After m times of iterations, the prediction result of the best classifier can be expressed as $F1(T, C_i, m)$.

Table 1. Some essential symbols.

Symbol	Definition
M	Iteration Number
S	Set of labeled record pairs on training set (labeled data pool)
Q	Set of all record pairs on training set
$Qn = \{q1, q2, \ldots, qn\}$	Set of n queried record pairs
$L(q1), L(q2), \ldots, L(qn)$	Labels provided by experts
$C = \{C1, C2, \ldots, Cn\}$	Set of classifiers
T	Set of labeled record pairs on the test set
V	Set of labeled record pairs on validation set

Table 1. *Cont.*

Symbol	Definition
F1(V, Ci)	F1 score w.r.t. ground truth of a set V and classifier C_i predicts labels for elements in V
Ci,j	Classifier Ci in the j-th iteration

3.1. The Framework of Pool-Based Active Learning

The active learning method will select the most valuable samples from the unlabeled samples continuously and submit them to experts for labeling, and then verify whether the model performance meets the requirements after adding new labeled data. If it does not meet the requirements, continue to select the most valuable samples and submit them to the experts for labeling. The value of samples is usually judged by uncertainty. If the uncertainty of a sample is higher, it means that the sample needs to be judged by experts manually, and these samples have a higher probability to improve the performance of the model. The sample query strategies of active learning are broadly divided into two categories: stream-based strategies and pool-based strategies [43,44]. For stream-based strategies, unlabeled samples will be handed over to the selection-engine in order, and the selection-engine will decide whether to submit the samples for labeling. The sample will be discarded if it is not selected. The pool-based strategies will maintain a set of unlabeled data, and in each time the selection-engine will select the most valuable samples from the unlabeled data pool for labeling.

The proposed active learning method for entity matching adopts the pool-based query strategy, the process of which is shown in Figure 2. Firstly, select a small number of valuable samples from the unlabeled training samples for labeling, i.e., construct the initial labeled pool. Then, construct the initial classifiers by labeled samples. If the performance of classifiers meets the requirements of the stop criterion, output the optimal classifier, otherwise, calculate the uncertainty of each sample from the unlabeled training set, select n samples with the most uncertainty for labeling according to the query strategies, and add these labeled samples to the labeled pool. Repeat the above steps until the classifier meets the requirements of the stop criterion. In each iteration, the label-request will be put forward actively, and the most valuable samples will be selected and handed over to experts for labeling to maximize the benefit of the experts' work. We can assume that the labels given by the experts are ground truth.

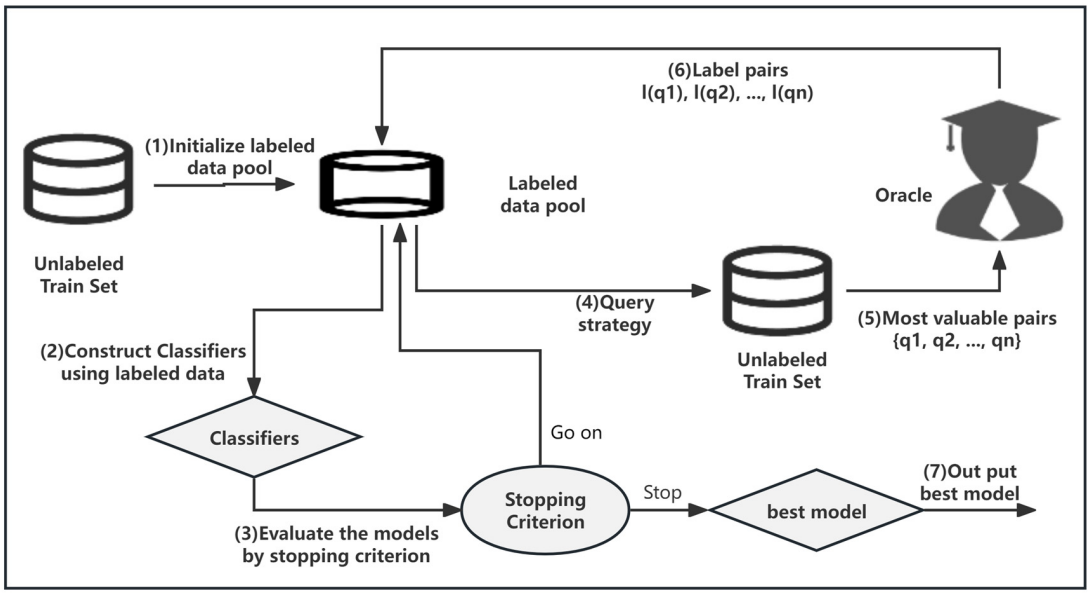

Figure 2. The process of active learning on entity matching.

The active learning algorithm for entity matching is described in Figure 3. Data preprocessing includes structuring the data set and pruning the training set (lines 1–2). The stage of active learning includes first generating the initial labeled pool (line 3) and then starting the iteration of active learning. In each iteration, ML-based models are built with the data in the labeled pool. When the stop criterion is met, exit the iteration (lines 4–7). Otherwise, the most valuable samples from the unlabeled training set are selected for labeling according to the query strategies, and these labeled samples are added to the labeled pool (lines 8–9). When the iteration is terminated, output the optimal model in the process of active learning (line 10).

1. Generate data set
2. Pruning
3. Generate Initial labeled pool S from training data Q
4. While True:
5. Construct classifiers C using labeled data pool S
6. if satisfy stop criterion:
7. break
8. Get $Q_n = \{q_1, q_2, \ldots, q_n\}$ by query strategy
9. Add $<q_1, L(q_1)>, <q_2, L(q_2)>, \ldots, <q_n, L(q_n)>$ to S
10. Return C

Figure 3. Algorithm of active learning for entity matching.

3.2. Data Preprocessing

3.2.1. Generating Data Set

The original data set is usually unstructured. It is necessary to standardize the original data into structured data before active learning. As mentioned in Section 2.2, each pair of records (x_i, y_i) needs to be quantified with a set of features f to convert the data(x_1, y_1, l_1) into $(<f_1 (x, y), \ldots, f_m(x, y)>, c)$. String-similarity measure methods mentioned in Section 2.1 are used to construct feature f. One or more methods can be used to construct features for the same attribute.

Each attribute adopts one or more string-similarity measure methods to construct a feature, and all features are spliced together to form the feature vector $<f_1(x_1, y_1), \ldots, f_m(x_1,y_1)>$. Different attributes need to be used in different string-similarity measure methods to construct the feature vector according to their characteristics. The standardization rules of the training set, validation set, and test set are the same. Taking Figure 4 as an example, the Levenshtein method, Jaro–Winkler method, and Jaccard method are used to calculate the similarity of attribute "title" and "author", the Jaccard method is used to calculate the similarity of attribute "venue", and the exact matching method is used to calculate the similarity of attribute "year". Then, these eight similarity values are spliced into feature vectors $v = <s_1, s_2, \ldots, s_8>$ for model training.

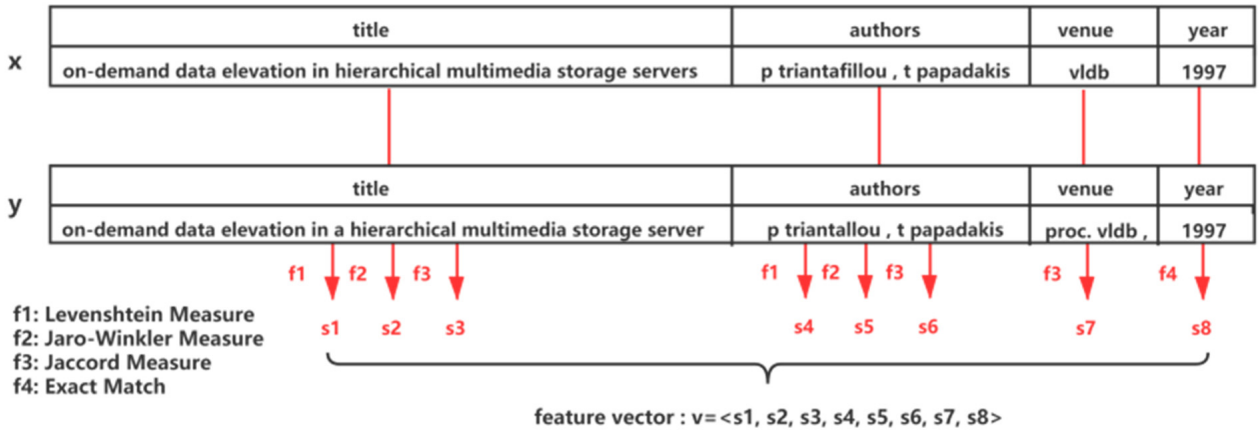

Figure 4. Construction of feature vector.

3.2.2. Pruning

Entity matching is viewed as a binary classification problem, in which input is record pairs and output is "match" or "mismatch". Unlike most binary classification tasks, the sample distribution of entity matching is extremely imbalanced. Usually, a record in data source A can just match with one record in data source B. In large-scale data sets, every time a matched record pair is added, N * M mismatched record pairs will be added at the same time. N and M are the sizes of the two data sources, respectively. In other words, the number of record pairs increases exponentially with the size of the data source, but the number of matched record pairs only increases linearly, which makes the number of negative samples far more than positive samples.

The imbalanced distribution of record pairs will lead to poor performance of matched samples prediction and over-fitting of mismatched samples when using model training directly. The prediction of matched samples is key to entity matching, so it is necessary to reduce the number of mismatched samples on the training set as much as possible, meanwhile retaining as many matched samples as possible. At present, there are a series of works on blocking technology [45]. Blocking is an effective method to cut a larger number of mismatched record pairs, and the experimental datasets are processed by the blocking technique.

In Section 2.1, the LWCR method can calculate an approximate similarity for the record pairs. The accuracy of this method is low, but if the similarity of a record pair is lower than the threshold, these record pairs have a high probability be mismatched. Therefore, the pruning strategy in our work is to calculate a similarity for all record pairs in the training set by LWCR, sort all record pairs according to the similarity value, and prune the pairs whose similarity is lower than a preset threshold.

This pruning strategy can not only cut a larger number of mismatched samples but also reduce the data noise. Some record pairs represent the same entity in the real world but have extremely low similarity. Although these record pairs are matched, their low similarity may lead to deviation in model training. The pruning strategy cuts such low similarity matched pairs at the same time, which can improve the generalization ability of learning models to some extent.

3.3. Generating the Initial Labeled Pool

At the beginning of active learning, we need to create an initial labeled pool, and the subsequent operations will maintain this labeled pool. The quality of initial labeled pool will directly affect the performance of initial classifiers and have a great impact on the whole process of active learning. If the performance of initial classifiers is very poor, it is difficult to screen the valuable samples for labeling in the following iterations, which will lead to slow down the improvement of the model.

The simplest method of generating the initial labeled pool is to select n record pairs randomly, but this method is very unstable. In the case of an imbalanced distribution of samples, the probability of selecting mismatched pairs is higher, which leads to a poor effect of fitting. Meanwhile, this method may select many fuzzy samples, so that initial classifiers cannot learn the features of matched samples and mismatched samples effectively.

Here we use the LWCR method to generate the initial labeled pool. First, calculate an approximate similarity of each record pair in Q according to the LWCR, and then sort all record pairs according to this approximate similarity value, and take N/2 samples from the head and the end, respectively for constructing the initial labeled pool. The advantage of this method is that it can ensure the positive and negative samples of initial data are equal. Because the data with the highest similarity are matched, while the data with the lowest similarity are mismatched. M, the data at the head and end are the most representative, which can enable initial classifiers to learn basic features and make a preliminary judgment on the matched and mismatched record pairs effectively.

3.4. Query Strategy

The key of active learning is to select N samples with the highest value for labeling in each iteration. Samples with the highest value are those that have the greatest impact on model improvement. If the label of a record pair can be calculated easily, experts do not need to spend time labeling it, and adding it to the labeled pool cannot improve the model to the great extent. Conversely, if it is difficult for classifiers to judge the label of a record pair, e.g., the probability judges as matched is close to probability judges as mismatched, it is needed to obtain an accurate label by experts, and then add it to the labeled pool for a great improvement on classifiers. Therefore, the record pairs, which are difficult for classifiers to judge and have the highest uncertainty, are the most valuable.

The goal of the query strategy is to find the unlabeled record pairs with the highest uncertainty to improve the predictive model greatly. The classifiers in our work are probability models, which can calculate the probability of matched and mismatched annotations respectively. Entropy is one of the most common methods for calculating uncertainty on probability models [46]. This method only considers the prediction results of one classifier. The entropy uncertainty calculated from one classifier is usually inaccurate. Therefore, it is necessary to consider the prediction results of different classifiers comprehensively for more accurate uncertainty. We further propose a hybrid uncertainty as query strategy, which includes three integrated parts, e.g., entropy-average, entropy-variance, and probability-variance uncertainty.

3.4.1. Entropy-Average Uncertainty

Each iteration of active learning will generate multiple classifiers to predict each record pair. Each classifier will generate the probability of "matched" and "mismatched" and calculate the entropy of the samples. $H_1(e)$, $H_2(e)$, ..., $H_n(e)$ represents the information entropy of n classifiers C_1, C_2, ..., C_n. The entropy-average uncertainty can consider the results of all classifiers comprehensively, which is defined shown in Formula (2):

$$\text{Ave\_Entropy}(e) = \frac{\sum_{i=1}^{n} H_i(e)}{n} \quad (2)$$

Intuitively, this method considers the entropy calculated by different classifiers comprehensively, i.e., the higher the average of entropy, the more difficult it is for the model to predict. The query strategy by entropy-average uncertainty selects n samples with the highest value of Ave_Entropy(e) for labeling.

3.4.2. Entropy-Variance Uncertainty

In some cases, the entropy-average uncertainty method cannot select the most uncertain samples. During the experiment, we found that some record pairs have high entropy calculated in one classifier and low entropy calculated in another classifier. Comparing

with the record pairs with moderate entropy calculated in both classifiers, the uncertainty of the previous record pairs should be higher, even if the average entropy of the two record pairs is same. Therefore, we propose another method to calculate uncertainty, namely entropy-variance uncertainty, as shown in Formula (3).

$$\text{var\_Entropy}(e) = \frac{\sum_{i=1}^{n} \left(H_i(e) - \text{Ave\_Entropy}(e)\right)^2}{n} \tag{3}$$

This method can be used to evaluate the variance of entropy on different classifiers. If the entropy on one classifier is high and on the other classifier is low, the entropy variance of the record pair will be high and more uncertain. This method can solve the limitation on entropy-average uncertainty.

3.4.3. Probability-Variance Uncertainty

There are still some limitations on the query strategy of entropy-variance uncertainty. Entropy can describe the uncertainty of samples. When one model predicts the sample as matched and another model predicts the sample as mismatched, they still may obtain the same entropy. These samples have low entropy-average uncertainty and entropy-variance uncertainty, but the uncertainty of them should be high, because the prediction results are totally different in these two models.

This phenomenon rarely occurs in most application scenarios. Because the results of record pairs predicted by different models are usually similar. However, the number of samples is relatively small in active learning. Adding samples with high uncertainty in each iteration may lead to deviation in the prediction of other samples by different models, which leads to some samples being predicted as different labels. Since entity matching is a binary classification task, we can define the variance of the probability of matched samples predicted by different models as uncertainty, i.e., probability-variance uncertainty, as shown in Formula (4).

$$\text{var\_Prob}(e) = \frac{\sum_{i=1}^{n} \left(p_{i,\text{match}} - \overline{P_{\text{match}}}\right)^2}{n} \tag{4}$$

where $p_{i,\text{match}}$ represents the probability of classifier C_i predicting on the sample e is matched, and $\overline{P_{\text{match}}}$ represents the average probability of all classifiers predicting on the sample e is matched.

3.4.4. Hybrid Uncertainty

The above-mentioned uncertainty calculation methods have their advantages and limitations as query strategies. Therefore, we here adopt a hybrid query strategy that comprehensively combines the results of these three uncertainty calculation methods.

First, calculate the entropy-average uncertainty, entropy-variance uncertainty, and probability-variance uncertainty of all unlabeled samples in Q, and sort these three uncertainties, respectively. Then, the index sorted by these three methods is weighted sum up as hybrid uncertainty. The range of each index is from 1 to $|Q|$. If the weight of the three methods is 1, the lowest uncertainty of each instance is 3 and the highest uncertainty is $3|Q|$. The query strategy by hybrid uncertainty is selecting samples with the highest hybrid uncertainty. If the hybrid uncertainty of two samples is same, we compare the entropy-average uncertainty, i.e., selecting the samples with the highest entropy-average uncertainty. The weights of three uncertainties can be adjusted according to data characteristics. Generally, the weight of the entropy-average uncertainty is the highest. When there is a great dispute on the result of most samples by different models, the weight of entropy-variance uncertainty and probability-variance uncertainty can be increased appropriately. The hybrid uncertainty can evaluate the uncertainty of samples comprehensively. The subsequent experiments will compare the performance of hybrid uncertainty and the other three uncertainties.

3.5. Stop Criterion

In active learning tasks, the effective stop criterion is important. Stop criterion determines when the query process can be terminated. A good stop criterion should prevent the iteration from stopping prematurely while the performance of classifiers is still improved, and stop labeling samples when models are stable.

The stop criterion is shown in Figure 5. We use each intermediate classifier to predict the samples on validation set V. When the F1 (V, C_i) calculated by the classifier Ci in the i-th iteration on validation set V does not increase or even decrease, the iteration will be terminated. Due to certain deviations in each intermediate classifier, we usually continue the active learning process for more stable classifiers, that is, if the F1 score does not increase after successive iterations, the classifier C_i with the highest F1 score will be outputted. In addition, we need to set a minimum threshold L according to the result of other baseline methods. If the F1 score predicted by classifiers on the validation set V is lower than L, we will continue the iteration regardless of the stop criterion. This strategy can ensure that the result of active learning will not be too poor, and prevent the classifiers from stopping halfway before learning enough characteristics. Moreover, the F1 score of intermediate classifiers may increase constantly when the labeled data are added continuously. We set an upper limitation M of iteration to avoid the iteration being terminated too late. When the number of iterations reaches M, the iteration will be terminated regardless of the stop criterion, and the classifier C_i with the highest F1 (T, C_i) will be outputted. If the size $|S|$ of the labeled data pool is too large, it does not attain the purpose of active learning, i.e., reducing the labeling workload. Therefore, the upper limitation M of iteration is usually not too large.

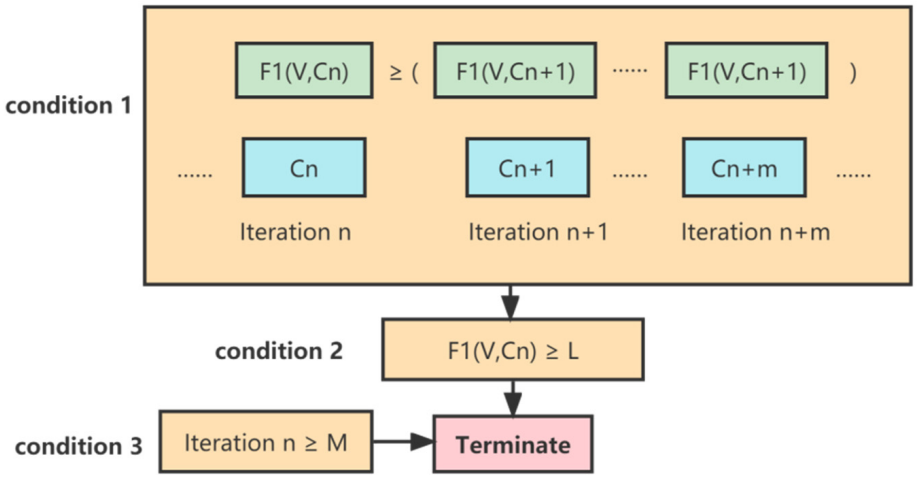

Figure 5. Stop criterion.

4. Experiment and Evaluation

In this section, we will verify the effectiveness and applicability of the proposed active learning method through experiments. Section 4.1 introduces the experiment setup. Section 4.2 introduces the evaluation metrics, and Section 4.3 shows the effectiveness of the proposed method through multiple experiments, including the effectiveness of the query strategies, the method of generating the initial labeled pool, and the pruning method. We further compare the performance of the proposed method with existing DL-based methods and AL-based methods in Section 4.3.

4.1. Experiment Setup

4.1.1. Data Sets

The data used in experiments are from seven public data sets for entity matching tasks provided by [11]. The data sets are from different domains, as shown in Table 2. The column "pairs" lists the number of labeled samples in each data set. Each sample is a

record pair composed of two records, labeled 1 is for "matched" and 0 for "mismatched". The column "match" lists the number of matched samples. The column "# Attr." shows the number of attributes of the data set. All attributes are atomic, i.e., not a composite form of multiple values.

Table 2. Experimental data sets.

Data Set	Domain	Pairs	Match	# Attr.
Amazon-Google	Software	11,460	1167	3
BeerAdvo-RateBeer	Beer	450	68	4
DBLP-ACM	Citation	12,363	2220	4
DBLP-Scholar	Citation	28,707	5347	4
Fodors-Zagats	Restaurant	946	110	6
iTunes-Amazon	Music	539	132	8
Walmart-Amazon	Electronics	10,242	962	5

These data sets have been divided into training set, validation set, and test set according to the ratio of 6:2:2, and the labels have the same distribution. We adopt the partition scheme of public datasets directly to ensure fairness in subsequent experiments with other related works. We use the training set for constructing models by active learning, use the validation set to verify the performance of intermediate classifiers and fine-tune the hyperparameters of classifiers, and finally use the test set to verify the performance of the final classifier. Due to a large amount of data in the training set and the extreme imbalance in the distribution of matched samples and mismatched samples, we use the pruning strategy described in Section 3.2 to cut down the most mismatched samples in the training set. The sample distribution of the pruned data sets is shown in Table 3. The left side of '/' in the training set represents the number of samples after pruning, and the right side of '/' represents the number of samples before pruning.

Table 3. Training set, test set and validation set.

DataSet	TrainSet		TestSet		ValidSet	
	Match	Pairs	Match	Pairs	Match	Pairs
Amazon-Google	589/699	5077/6874	234	2293	234	2293
BeerAdvo-RateBeer	29/40	103/268	14	91	14	91
DBLP-ACM	1323/1332	2602/7417	444	2473	444	2473
DBLP-Scholar	2915/3207	6231/17,223	1070	5742	1070	5742
Fodors-Zagats	62/66	227/567	22	190	22	189
iTunes-Amazon	74/78	166/321	27	109	27	109
Walmart-Amazon	536/576	5379/6144	193	2049	193	2049

4.1.2. Classifiers

In the experiment, we use the mainstream machine learning algorithms to construct the classifiers, including SVM, random forest, KNN, and naive Bayesian, which are widely used in active learning tasks [12]. Record pairs are inputted into the classifiers and return "match" or "mismatch". The main function of classifiers is to find the samples with the highest uncertainty for experts to label.

These four algorithms have different mathematical principles, and they can used for classification based on entropy. The hyperparameters and random seed of classifiers are consistent to reduce the variance in the active learning process. In SVM, the penalty

parameter C is 1 and the kernel is Gaussian kernel. In Random forest, max depth is 3, the number of trees is 50, and criterion is Gini. In KNN, the number of neighbors is 5. Other hyperparameters adopt default in Python sklearn package (Python version 3.7.0, sklearn package version 0.20.1). These classifiers can be substituted by other entropy-based algorithms.

4.1.3. Other Hyper Parameters

In the experiment, the maximum number of iterations of active learning Mmax is 20. The seven data sets can be divided into small-scale data sets (BeerAdvo-RateBeer, iTunes-Amazon, and Fodors-Zagats) and large-scale datasets (DBLP-ACM, DBLP-Scholar, Amazon-Google, Walmart-Amazon). For small-scale data sets, the size of the initial labeled pool $|S|$ is 6, and each iteration add 4 high-uncertainty samples for experts to obtain labels and 4 low-uncertainty samples with their predicted labels into the labeled pool. For large-scale data sets, the size of the initial labeled pool $|S|$ is 50, and each iteration adds 20 high-uncertainty and 20 low-uncertainty samples into the labeled pool. In hybrid uncertainty method, weights of all classifiers are set to 1.

4.2. Evaluation Metrics

Due to the imbalanced distribution of samples for entity matching tasks, we use the F1 score to measure the performance of models. The F1 score makes a comprehensive judgment on the predicted results by calculating the harmonic average of precision and recall. Menestrina et al. [47] demonstrated that the F1 score is the most suitable metric for entity matching tasks. Most related works use the F1 score as the preferred metric for entity matching.

In the experiment, we first compare the F1 score of different methods. The higher the F1 score, the better the performance of the method. When comparing the active learning methods of different query strategies, it is possible that different methods have the same F1 score. When the F1 score is same, we compare the size of labeled pool N. If N is small, it means that less labeled training samples are used, i.e., less labeling workload is required to achieve the same prediction effect.

4.3. Experiment Results

We verify the effectiveness of the query strategies and the strategy of initializing the labeled pool in the active learning stage, and verify the effectiveness of the pruning strategy on the train set in the preprocessing stage.

4.3.1. Performance of Query Strategies

This subsection compares the performance of proposed query strategies and the baseline strategy, the entropy-based query strategy. In the experiment, we use the method of initializing the labeled pool proposed in Section 3.3, and the stop criterion proposed in Section 3.5. The integrated query strategies use multiple classifiers and the entropy-based query strategy uses the random forest as the classifier. The parameters of the classifiers are consistent throughout the experiment.

Table 4 compares the performance of proposed integrated query strategies and the baseline strategy based on entropy. The performance of the hybrid uncertainty is better than other uncertainties on seven data sets, and the optimal F1 score is also obtained in dataset iTunes-Amazon, in which the amount of labeled data is slightly higher than the entropy-variance uncertainty. The entropy-average uncertainty performs best on one data set, the entropy-variance uncertainty performs best on two data sets, and the probability-variance uncertainty performs best on one data set.

Table 4. Comparing the performance of five uncertainty methods.

DataSet	Entropy		Ave_Entropy		Var_Entropy		Var_Prob		Hybrid	
	F1	N	F1	N	F1	N	F1	N	F1	N
Amazon-Google	0.337	450	0.348	430	0.376	450	0.209	450	**0.424**	**370**
BeerAdvo-RateBeer	0.839	30	0.828	22	0.839	22	**0.867**	**22**	**0.867**	**22**
DBLP-ACM	0.982	310	0.982	330	0.979	210	0.976	440	**0.984**	**210**
DBLP-Scholar	0.903	520	0.896	600	0.909	510	0.900	630	**0.909**	**360**
Fodors-Zagats	1.000	22	**1.000**	**10**	**1.000**	**10**	1.000	18	**1.000**	**10**
iTunes-Amazon	0.982	38	0.982	50	**1.000**	**22**	0.982	58	1.000	46
Walmart-Amazon	0.696	70	0.696	120	0.707	210	0.700	240	**0.721**	**80**

The bold entries show the best result of all methods.

In conclusion, among these five query strategies, the hybrid uncertainty has the best performance overall. The entropy-average uncertainty, entropy-variance uncertainty and probability-variance uncertainty can obtain good results on some data sets, but they are not as stable as the hybrid uncertainty. The performance of entropy-based uncertainty is poorer than the above four integrated strategies.

4.3.2. Comparing with Deep Learning Based Methods

To verify the effectiveness of the proposed method, we compare the proposed method with the mainstream methods, including SIF [32], RNN [33], Attention [34], DeepER [28], DeepMatcher [29], Magellan [3], and AutoML-EM [48]. The results of the hybrid uncertainty in Table 4 are compared with the above seven methods. The compared results are shown in Table 5. Table 5 records the F1 score of each method in the seven data sets. The last column "ΔF1" represents the difference in the F1 score between the proposed method and the optimal results of these seven methods. Table 6 shows the number of labeled training samples. The column "ratio" shows the ratio of labeled training samples used by the proposed method to other methods.

Table 5. Comparing the proposed method with baseline methods.

DataSet	SIF	RNN	Attention	Magellan	DeepER	DeepMatcher	AutoML-EM	Ours	ΔF1
Amazon-Google	0.606	0.599	0.611	0.491	0.561	**0.693**	0.664	0.424	−0.269
BeerAdvo-RateBeer	0.581	0.722	0.808	0.788	0.5	0.727	0.823	**0.867**	0.044
DBLP-ACM	0.975	0.983	**0.984**	**0.984**	0.976	**0.984**	**0.984**	**0.984**	0
DBLP-Scholar	0.909	0.93	0.933	0.923	0.908	**0.947**	0.946	0.909	−0.038
Fodors-Zagats	**1**	**1**	0.821	**1**	0.977	**1**	**1**	**1**	0
iTunes-Amazon	0.814	0.885	0.808	0.912	0.88	0.88	0.963	**1**	0.037
Walmart-Amazon	0.651	0.676	0.5	0.719	0.506	0.669	**0.785**	0.721	−0.064

The bold entries show the best result of all methods, and the underline ones mean the suboptimal result.

Table 6. Comparing the number of labeled samples.

DataSet	Proposed Method	Other Methods	Ratio
Amazon-Google	370	5077	0.073
BeerAdvo-RateBeer	22	103	0.216
DBLP-ACM	210	2602	0.081
DBLP-Scholar	360	6231	0.058
Fodors-Zagats	10	277	0.036

Table 6. *Cont.*

DataSet	Proposed Method	Other Methods	Ratio
iTunes-Amazon	46	166	0.277
Walmart-Amazon	80	5379	0.015

From the comparison of experiment results, the performance of the proposed method is slightly lower than the optimal one on datasets DBLP-Scholar and Walmart-Amazon, and the F1 score is lower than the optimal model of 0.05 on average. Meanwhile, the amount of labeled training samples of the proposed method in all data sets has decreased significantly, and the average amount of labeled training samples is only about 10% of other methods. In DBLP-Scholar and Walmart-Amazon, the proposed method can greatly reduce the number of labeled samples while maintaining certain prediction accuracy. On datasets BeerAdvo-RateBeer, DBLP-ACM, Fodors-Zagats, and iTunes-Amazon, the proposed method obtains the best F1 score. On these four datasets, our method can not only reduce the number of labeled training samples but also improve the F1 score.

Comparing Tables 5 and 6, it can be found that the proposed method performs better in small-scale data sets, which can even obtain better results than the DL-based methods. This is probably because the labeled data in small-scale data sets has a greater probability to represent the characteristics of the whole data set. In large-scale data sets, the amount of labeled training samples is too small compared with the whole data set so that a small amount of labeled data may not fit the characteristics of the whole data set, which makes the model perform worse than the DL-based methods. Nevertheless, the proposed method is also applicable in large-scale data sets, which can greatly reduce the workload of labeling with losing little accuracy.

In the experiments, we also found that the proposed method has poor performance on Amazon-Google. Amazon-Google records software-related data, which has three attributes: "title", "manufacturer" and "price". Among these attributes, the missing rate of "manufacturing" in one data source is nearly 90%, and the "price" has no obvious matching law. Therefore, most of classifiers can only be constructed by the "title". In this case, only a single attribute is used to judge whether the record pair is matched or not, so it is difficult to obtain good results by the method of string similarity. The proposed method only considers the structural similarity of the attribute, but for Amazon-Google, we need to analyze the semantic similarity to obtain more accurate prediction results. Therefore, the F1 score of Magellan and the proposed method are lower than DL-based methods.

4.3.3. Comparing with Existing Active Learning Based Methods

With the advantages of less labeled samples and human–machine collaboration, active learning can be well applied in entity matching tasks. This subsection mainly compares our method with existing AL-based methods. DBLP-Scholar and DBLP-ACM are the datasets selected for most AL-based works, so we compare the F1 score and number of labeled samples on these two datasets. The compared methods include ERLEARN [49], ALGPR [50], ALIAS [51], DTAL [52], and DAL [52]. The results are shown in Table 7. Compared with baseline methods, our method has the best F1 value.

Table 7. Comparing the proposed method with existing AL-based methods.

Method	DBLP-Scholar		DBLP-ACM	
	F1	N	F1	N
ERLEARN	0.87	163	N/A	N/A
ALGPR	0.80	210	N/A	N/A
ALIAS	0.78	160	N/A	N/A

Table 7. *Cont.*

Method	DBLP-Scholar		DBLP-ACM	
	F1	**N**	**F1**	**N**
DTAL	0.895	1000	0.979	400
DAL	0.888	1000	0.954	400
Ours	**0.908**	**360**	**0.983**	**210**

The bold entries show the best result of all methods, and the underline ones mean the suboptimal result.

4.3.4. Performance of Initial Labeled Pool

The quality of the initial labeled pool plays an important role in the active learning method. If the samples in the initial labeled pool are low quality, it will lead to the poor effect of the first batch of classifiers, lead to the inaccurate entropy calculated by these classifiers, and the selected samples are not with the highest uncertainty.

Figure 6 compares the performance of the random selection method and the rule-based method proposed in Section 3.3 for initializing the labeled pool on Walmart-Amazon. Without adding any new labeled samples, the F1 score of the rule-based method on the validation set has reached 0.6, while the random selection method is only 0.45. The random selection method requires about 17 iterations to achieve the same F1 score as the rule-based method. We can see that the proposed method of initializing the labeled pool is effective, which can reduce the number of iterations and make the intermediate model fit quickly.

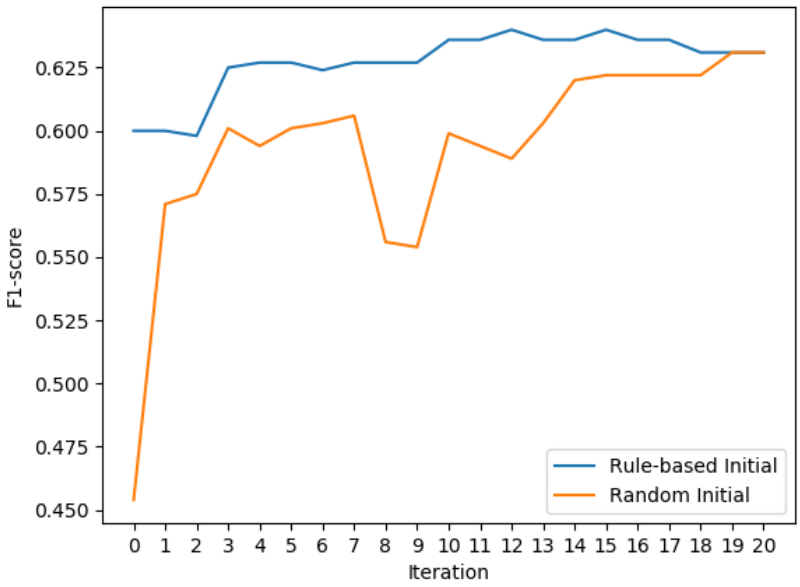

Figure 6. F1-iteration curve on initialization strategies.

4.3.5. Performance of Pruning Method

Table 8 compares the results of the proposed method using hybrid uncertainty as a query strategy before and after pruning. * indicates pruning operation. The results show that the F1 score of the method after pruning is better than that without pruning. For some data sets, such as Amazon-Google and BeerAdvo-RateBeer, the F1 score has been significantly improved.

Table 8. Evaluate the performance of the pruning strategy.

DataSet	Hybrid		Hybrid *	
	F1	**N**	**F1**	**N**
Amazon-Google	0.282	450	0.424	370
BeerAdvo-RateBeer	0.763	46	0.867	22
DBLP-ACM	0.973	250	0.984	210
DBLP-Scholar	0.865	470	0.909	360
Fodors-Zagats	1	50	1	10
iTunes-Amazon	0.912	62	1	46
Walmart-Amazon	0.651	150	0.721	80

* indicates the hybrid method with pruning operation.

From the experiment results, we can see that the proposed pruning method is effective. Although the result of entity matching using the LWCR method is poor, this method can give an approximate similarity for all record pairs. If the approximate similarity of a record pair is too low, it means that this record pair has a high probability of being mismatched. Even if a small number of record pairs are matched with low similarity, they have a bad impact on constructing classifiers because their attributes have a low similarity. Therefore, using this pruning method can not only reduce the number of mismatched samples but also remove some matched pairs with low similarity, which prevents the query strategies from selecting "noise samples" to the labeled pool, to improve the generalization ability of classifiers.

4.3.6. Stability

To further verify the stability of the proposed active learning algorithm, we shuffled the training set, validation set, and test set in the public datasets and randomly divided them in the same proportion. We implemented ten repeated experiments and the results are shown in Figure 7. From the box-plot figure, we can see that the results are relatively stable. The average standard deviation is approximately 0.02 in seven datasets. This experiment can demonstrate that the proposed active learning algorithm has good stability.

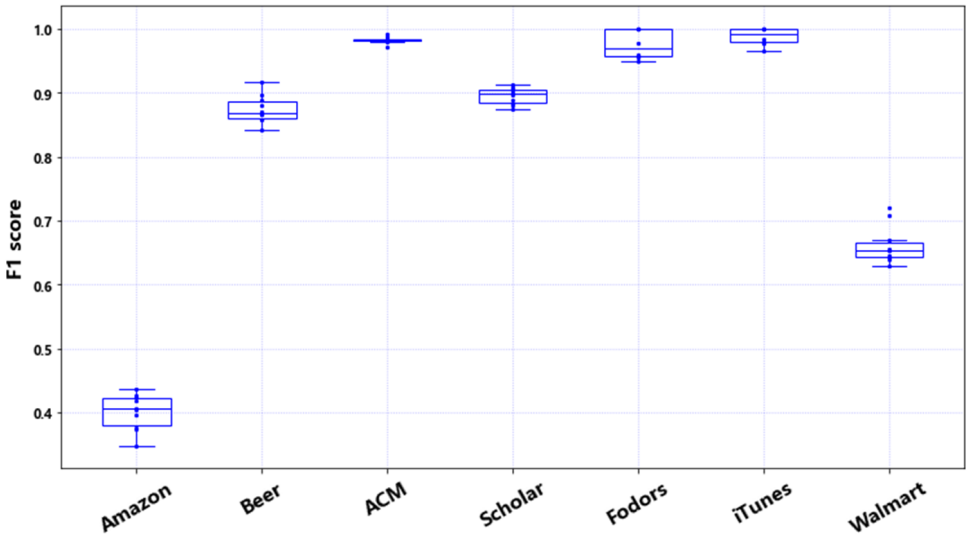

Figure 7. Box-plot of ten repeated experiments on seven datasets.

5. Discussion

From the above experiments, we have verified the effectiveness of the proposed active learning algorithm. In the absence of labeled samples and domain knowledge, our

method can provide a fast solution to entity matching tasks. From the experiment results, the proposed method is more suitable for small-scale datasets. This is probably because the labeled data in small-scale data sets have a greater probability of representing the characteristics in the whole data set. In large-scale datasets, there will be more proprietary terms and more semantic information, which makes it difficult for AL-based methods to capture the matching patterns with a small number of labeled samples. Moreover, our method is suitable for single matching and multiple matching tasks, but not suitable for partly matching due to the limitation of similarity measures.

There are still some limitations on our work. First, our stop criterion relies on the validation set, and sometimes the iteration is stopped prematurely in large-scale data sets. Second, the pruning strategy is relatively simple. The method of linear weighted combination rules may cut off some matched samples, which leads to losing some valuable samples. In addition, the proposed method only uses the structure similarity method for comparing record pairs, which does not consider the semantic similarity of attributes. In the future work, we will further optimize the method of attribute similarity comparison and pruning strategy. We will also further explore appropriate query strategies to make the active learning method more suitable for entity matching tasks.

6. Conclusions

This paper proposes an active learning method with an integrated query strategy, that can be well applied to entity matching tasks. This method can construct the machine learning model through a very small number of labeled samples, and achieve higher F1 score. In some small-scale data sets, the effect is even better than the DL-based methods. The proposed active learning method can effectively reduce the workload of labeling samples, and select the most valuable samples to experts for labeling. We further propose a hybrid uncertainty as query strategy, which is suitable for most data sets. The proposed method of initializing the labeled pool also has a good effect, which makes the classifiers fit fast.

The proposed method can be well applied for entity matching tasks, especially in lacking labeled samples. The ML-based entity matching methods rely on a larger number of labeled samples for training. The DL-based methods usually need the pre-trained language model or domain-related data for fine-tuning. Our proposed method does not rely on domain knowledge or language models, and just uses less labeled samples to obtain good performance. The proposed method can greatly reduce the workload of labeling, and solve the limitations of traditional learning-based methods in entity matching tasks, which makes the entity matching tasks in lacking domain knowledge have an efficient and relatively accurate solution.

Author Contributions: Conceptualization, Y.H. and C.L.; methodology, Y.H. and C.L.; validation, Y.H. and C.L.; writing—original draft preparation, Y.H.; writing—review and editing, C.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by NSFC grant number 61672309.

Data Availability Statement: The data sets used in this work are freely available and can be accessed in public open websites.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Tan, W.C. Technical Perspective: Toward Building Entity Matching Management Systems. *SIGMOD Rec.* **2018**, *47*, 32. [CrossRef]
2. Koepcke, H.; Rahm, E. Frameworks for Entity Matching: A Comparison. *Data Knowl. Eng.* **2010**, *69*, 197–210. [CrossRef]
3. Konda, P.; Das, S.; Doan, A.; Ardalan, A.; Ballard, J.R.; Li, H.; Panahi, F.; Zhang, H.; Naughton, J.; Prasad, S.; et al. Magellan: Toward Building Entity Matching Management Systems. *VLDB Endow.* **2016**, *9*, 1197–1208. [CrossRef]
4. Christen, P. *Data Matching*; Springer: Berlin/Heidelberg, Germany, 2012.

5. Singh, R.; Meduri, V.; Elmagarmid, A.; Madden, S.; Papotti, P.; Quiané-Ruiz, J.-A.; Solar-Lezama, A.; Tang, N. Generating Concise Entity Matching Rules. In Proceedings of the ACM International Conference on Management of Data, Chicago, IL, USA, 14–19 May 2017.
6. Shen, W.; Li, X.; Doan, A.H. Constraint-based Entity Matching. In Proceedings of the AAAI Conference on Artificial Intelligence, Pittsburgh, PA, USA, 9–13 July 2005.
7. Whang, S.E.; Benjelloun, O.; Garcia-Molina, H. Generic Entity Resolution with Negative Rules. *VLDB J.* **2009**, *18*, 1261–1277. [CrossRef]
8. Singla, P.; Domingos, P. Entity Resolution with Markov Logic. In Proceedings of the Sixth International Conference on Data Mining, Hong Kong, China, 18–22 December 2006.
9. Chaudhuri, S.; Chen, B.C.; Ganti, V.; Kaushik, R. Example-Driven Design of Efficient Record Matching Queries. In Proceedings of the 33rd International Conference on Very Large Data Bases, Vienna, Austria, 23–28 September 2007.
10. Schmidhuber, J. Deep Learning in Neural Networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef]
11. Barlaug, N.; Atle Gulla, J. Neural Networks for Entity Matching: A Survey. *ACM Trans. Knowl. Discov. Data* **2021**, *15*, 37. [CrossRef]
12. Settles, B. *Active Learning Literature Survey*; Technical Report; University of Wisconsin: Madison, WI, USA, 2010.
13. Balcan, M.; Beygelzimer, A.; Langford, J. Agnostic Active Learning. *J. Comput. Syst. Sci.* **2009**, *75*, 78–89. [CrossRef]
14. Attenberg, J.; Provost, F. Inactive learning? Difficulties Employing Active Learning in Practice. *SIGKDD Explor. Newsl.* **2010**, *12*, 36–41. [CrossRef]
15. Chen, Z.; Tao, R.; Wu, X.; Wei, Z.; Luo, X. Active Learning for Spam Email Classification. In Proceedings of the 2nd International Conference on Algorithms, Computing and Artificial Intelligence, Sanya, China, 20–22 December 2019.
16. Samuel, B.; Robinson Emma, C.; Bernhard, K. A Survey on Active Learning and Human-in-the-Loop Deep Learning for Medical Image Analysis. *Med. Image Anal.* **2021**, *71*, 102062.
17. Agoun, J.; Hacid, M. Access Control based on Entity Matching for Secure Data Sharing. *Serv. Oriented Comput. Appl.* **2022**, *16*, 31–44. [CrossRef]
18. Zhang, P.; Kang, X. Similar Physical Entity Matching Strategy for Mobile Edge Search. *Digit. Commun. Netw.* **2020**, *6*, 203–209. [CrossRef]
19. Singh, R.; Meduri, V.V.; Elmagarmid, A.; Madden, S.; Papotti, P.; Quiané-Ruiz, J.-A.; Solar-Lezama, A.; Tang, N. Synthesizing Entity Matching Rules by Examples. *VLDB Endow.* **2017**, *11*, 189–202. [CrossRef]
20. Ngomo, A.C.N. Link Discovery with Guaranteed Reduction Ratio in Affine Spaces with Minkowski Measures. In Proceedings of the International Semantic Web Conference, Boston, MA, USA, 11–15 November 2012.
21. Jaro, M.A. Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa. *J. Am. Stat. Assoc.* **1989**, *84*, 414–420. [CrossRef]
22. Rodrigues, E.O.; Casanova, D.; Teixeira, M.; Pegorini, V.; Favarim, F.; Clua, E.; Conci, A.; Liatsis, P. Proposal and Study of Statistical Features for String Similarity Computation and Classification. *Int. J. Data Min. Model. Manag.* **2020**, *12*, 277–280. [CrossRef]
23. Verykios, V.S.; Moustakides, G.V.; Elfeky, M.G. A Bayesian Decision Model for Cost Optimal Record Matching. *VLDB J.* **2002**, *12*, 28–40. [CrossRef]
24. Dey, D. Entity Matching in Heterogeneous Databases: A Logistic Regression Approach. *Decis. Support Syst.* **2007**, *44*, 740–747. [CrossRef]
25. Primpeli, A.; Bizer, C. Profiling Entity Matching Benchmark Tasks. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, Online, 19–23 October 2020.
26. Palumbo, E.; Rizzo, G.; Troncy, R. STEM: Stacked Threshold-based Entity Matching for Knowledge Base Generation. *Semant. Web.* **2018**, *10*, 117–137. [CrossRef]
27. Mugeni, J.B.; Amagasa, T. A Graph-Based Blocking Approach for Entity Matching Using Contrastively Learned Embeddings. *ACM SIGAPP Appl. Comput. Rev.* **2023**, *22*, 37–46. [CrossRef]
28. Ebraheem, M.; Thirumuruganathan, S.; Joty, S.; Ouzzani, M.; Tang, N. DeepER–Deep Entity Resolution. *arXiv* **2017**, arXiv:1710.00597.
29. Mudgal, S.; Li, H.; Rekatsinas, T.; Doan, A.; Park, Y.; Krishnan, G.; Deep, R.; Arcaute, E.; Raghavendra, V. Deep Learning for Entity Matching: A Design Space Exploration. In Proceedings of the International Conference on Management of Data, Houston, TX, USA, 10–15 June 2018.
30. Pennington, J.; Socher, R.; Manning, C.D. GloVe: Global Vectors for Word Representation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), Doha, Qatar, 25–29 October 2014.
31. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching Word Vectors with Subword Information. *Trans. Assoc. Comput. Linguist.* **2017**, *5*, 135–146. [CrossRef]
32. Arora, S.; Liang, Y.; Ma, T. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. In Proceedings of the 5th International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
33. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder Decoder for Statistical Machine Translation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), Doha, Qatar, 25–29 October 2014.

34. Parikh, A.P.; Täckström, O.; Das, D.; Uszkoreit, J. A Decomposable Attention Model for Natural Language Inference. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2016), Austin, TX, USA, 1–4 November 2016.
35. Huang, J.; Hu, W.; Bao, Z.; Chen, Q.; Qu, Y. Deep Entity Matching with Adversarial Active Learning. *VLDB J.* **2023**, *32*, 229–255. [CrossRef]
36. Li, Y.; Li, J.; Suhara, Y.; Doan, A.H.; Tan, W.C. Deep Entity Matching with Pre-trained Language Models. *VLDB Endow.* **2020**, *14*, 50–60. [CrossRef]
37. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2019), Hong Kong, China, 3–7 November 2019.
38. Li, Y.; Li, J.; Suhara, Y.; Wang, J.; Hirota, W.; Tan, W.C. Deep Entity Matching: Challenges and Opportunities. *J. Data Inf. Qual.* **2021**, *13*, 1–17. [CrossRef]
39. Li, Y.; Li, J.; Suhara, Y.; Doan, A.; Tan, W.-C. Effective Entity Matching with Transformers. *VLDB J.* **2023**, *32*, 1215–1235. [CrossRef]
40. Brunner, U.; Stockinger, K. Entity Matching with Transformer Architectures- A Step Forward in Data Integration. In Proceedings of the 23rd International Conference on Extending Database Technology, Copenhagen, Denmark, 30 March–2 April 2020.
41. Peeters, R.; Bizer, C.; Glavaš, G. Intermediate Training of BERT for Product Matching. In Proceedings of the DI2KG Workshop at VLDB, Tokyo, Japan, 31 August 2020.
42. Zhao, C.; He, Y. Auto-EM: End-to-end Fuzzy Entity-Matching using Pre-trained Deep Models and Transfer Learning. In Proceedings of the World Wide Web Conference (WWW 2019), San Francisco, CA, USA, 13–17 May 2019.
43. Dagan, I.; Engelson, S.P. Committee-based Sampling for Training Probabilistic Classifiers. In Proceedings of the 12th International Conference on Machine Learning, Tahoe City, CA, USA, 9–12 July 1995.
44. Lewis, D.D.; Gale, W.A. A Sequential Algorithm for Training Text Classifiers. In Proceedings of the 17th Annual International ACM SIGIR Conference on Research and development in Information Retrieval, Dublin, Ireland, 3–6 July 1994.
45. Baxter, R.; Christen, P.; Churches, T. A Comparison of Fast Blocking Methods for Record Linkage. In Proceedings of the ACM SIGKDD Workshop on Data Cleaning, Record Linkage, and Object Consolidation, Washington, DC, USA, 24 August 2003.
46. Shannon, C.E. A Mathematical Theory of Communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423. [CrossRef]
47. Menestrina, D.; Whang, S.E.; Garcia-Molina, H. Evaluating Entity Resolution Results. *VLDB Endow.* **2010**, *3*, 208–219. [CrossRef]
48. Wang, P.; Zheng, W.; Wang, J.; Pei, J. Automating Entity Matching Model Development. In Proceedings of the IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 19–22 April 2021.
49. Qian, K.; Popa, L.; Sen, P. Active Learning for Large-scale Entity Resolution. In Proceedings of the ACM on Conference on Information and Knowledge Management, Singapore, 6–10 November 2017.
50. Arasu, A.; Götz, M.; Kaushik, R. On Active Learning of Record Matching Packages. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Chicago, IL, USA, 14–19 May 2010.
51. Sarawagi, S.; Bhamidipaty, A. Interactive Deduplication Using Active Learning. In Proceedings of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, AB, Canada, 23–26 July 2002.
52. Kasai, J.; Qian, K.; Gurajada, S.; Li, Y.; Popa, L. Low-resource Deep Entity Resolution with Transfer and Active Learning. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019.