Unsupervised Bootstrapping of Active Learning for Entity Resolution

Anna Primpeli, Christian Bizer, and Margret Keuper

Data and Web Science Group, University of Mannheim, Mannheim, Germany
{anna,chris}@informatik.uni-mannheim.de
keuper@uni-mannheim.de

Abstract. Entity resolution is one of the central challenges when integrating data from large numbers of data sources. Active learning for entity resolution aims to learn high-quality matching models while minimizing the human labeling effort by selecting only the most informative record pairs for labeling. Most active learning methods proposed so far, start with an empty set of labeled record pairs and iteratively improve the prediction quality of a classification model by asking for new labels. The absence of adequate labeled data in the early active learning iterations leads to unstable models of low quality which is known as the cold start problem. In our work we solve the cold start problem using an unsupervised matching method to bootstrap active learning. We implement a thresholding heuristic that considers pre-calculated similarity scores and assigns matching labels with some degree of noise at no manual labeling cost. The noisy labels are used for initializing the active learning process and throughout the whole active learning cycle for model learning and query selection. We evaluate our pipeline with six datasets from three different entity resolution settings using active learning with a committee-based query strategy and show it successfully deals with the cold start problem. Comparing our method against two active learning baselines without bootstrapping, we show that it can additionally lead to overall improved learned models in terms of F_1 score and stability.

Keywords: Active Learning · Unsupervised Matching · Entity Resolution

1 Introduction

Entity resolution methods often rely on supervised learning for matching entity descriptions from different data sources [3, 5]. This means that a specific set of training record pairs is required for each pair of sources to be matched. The required amount of training data thus grows quickly with the number of data sources to be integrated. Labeling large amounts of data is a tedious task. Active learning for entity resolution aims at minimizing the human labeling effort by iteratively selecting only an informative subset of record pairs for labeling. In each active learning iteration, one or more informative record pairs are selected and provided to a human annotator for labeling. One way to measure the degree

of informativeness is to calculate the disagreement among the predictions of a classifier ensemble known as the Query-by-Committee strategy [2, 19, 21]. The most informative record pairs are the ones that cause the highest disagreement among the members of the committee.

A problem which frequently arises in active learning, is the lack of labeled data in the early iterations, known as the cold start problem [9]. In these cases, the model does not have adequate data to learn from and is therefore of low predictive quality. To circumvent the cold start problem, existing active learning methods for entity resolution require the human annotator to label a small set of record pairs. This set is either selected randomly [9, 18] or based on the distribution of pre-calculated similarity scores [2, 19] before the active learning starts. However, solving the cold start problem by manually annotating a subset of the data is contradicting the main principle of active learning, that of minimizing the human labeling effort.

We propose an alternative method for dealing with the cold start problem. Our method uses unsupervised matching to bootstrap active learning and therefore comes at no additional labeling cost. More concretely, we use datatype specific similarity metrics and assign a similarity score to all record pairs. The similarity score distribution is accounted for setting a suitable threshold value. Considering the threshold boundary, we assign binary labels *match* or *non-match* and confidence weights to the record pairs with some degree of noise. The noisy set of unsupervised weighted labeled record pairs is used to bootstrap active learning. In addition, it is part of the complete active learning cycle as it is used for model training and record pair selection. We show that our thresholding heuristic gives better unsupervised matching results in comparison to commonly used thresholding methods, independently of the underlying similarity distribution. Bootstrapping active learning with unsupervised labeled pairs guarantees high anytime performance and stability. Our experiments show that our proposed method can improve the model quality in terms of absolute F_1 score by 86% in the cold start phase and up to 3% after the cold start phase in comparison to baseline active learning methods that do not use unsupervised bootstrapping. The contributions of our work are summarized as follows:

- We propose a thresholding heuristic that uses a domain independent scoring function and outperforms existing thresholding methods.
- We propose a method for warm-starting active learning that comes at no additional labeling cost and guarantees high anytime performance.
- We perform an extensive evaluation on three types of entity resolution problems: structured, textual, and dirty data.

This paper is organized as follows: Section 2 explains our methodology for bootstrapping active learning with unsupervised matching. Section 3 presents the evaluation of our pipeline and comparison to related work and baseline methods. Section 4 discusses the related work in the areas of unsupervised matching and active learning. Section 5 concludes the paper and summarizes our main findings. The code and data used for the evaluation are publicly available[1].

[1] https://github.com/aprimpeli/UnsupervisedBootAL

2 Methodology

This section gives first an overview of our proposed method and then all methodological steps are explained in detail. Our proposed methodology starts with the unsupervised matching of record pairs which includes the feature vector creation, the feature values aggregation to one similarity score per record pair and a thresholding function. We compare the aggregated similarity scores to the threshold value and assign labels and weights to the record pairs. The unsupervised labeled and weighted record pairs are added in the noisy weighted pool and used for training a Random Forest classifier [1]. In every active learning iteration, a committee of models selects the most informative record pair from the noisy pool to be manually labeled which is then removed from the noisy pool and added in the labeled set. The labeled set is used to expand the Random Forest classifier by incrementally training new trees which are added to the existing forest. Figure 1 presents our complete proposed method.

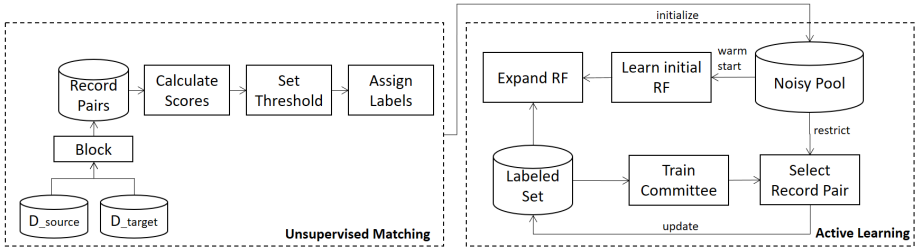


Fig. 1: Unsupervised Bootstrapping for Entity Matching with Active Learning.

2.1 Unsupervised Matching

Feature vector creation We consider the matching problem between two datasets, source and target, with aligned schemata. In order to reduce the number of calculations, we filter out obvious non-matches by blocking using the most identifying domain-specific attribute which we manually define. The blocks are created using Relaxed Jaccard with inner Levenshtein distance and a threshold of 0.2. We create pairwise features from the individual attributes of each record for all remaining record pairs after blocking. Each feature corresponds to a datatype specific similarity score, similarly to the Magellan entity matching system [8]. Feature values of datatype string are compared using the following similarity metrics: Levenshtein, Jaccard, Jaccard with inner Levenshtein, overlap and containment. String attributes with an average length larger than six tokens are considered long strings and the cosine similarity score with tfidf weighting per feature is additionally computed. For numeric attributes the absolute difference is calculated. For date attributes, the day difference, month difference, and year

difference are computed. Finally, we calculate the cosine score with tfidf weighting over the concatenated values of all attributes. We rescale all scores to the range of $[0, 1]$ and convert the difference based features to similarity ones. In the case that the similarity score cannot be computed for an attribute combination because either the source, the target or both values are missing, we assign the out of range score -1. This allows any classifier to consider the relevant record pairs without dropping or replacing the missing values. Figure 2 shows an example of the created feature vector considering a source record S and a target record T describing the same author entity.

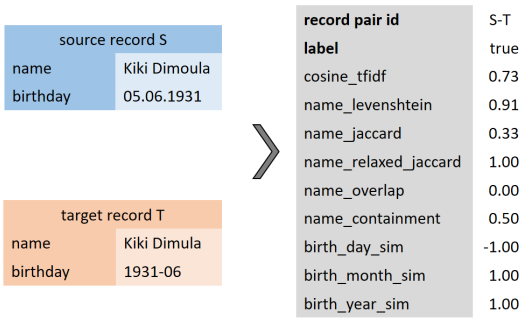


Fig. 2: Feature vector creation example.

Similarity-score aggregation We summarize the feature vector values into one value per record pair and assign this score as its aggregated similarity score. A similarity score close to 1 gives a strong signal that the record pair matches whereas a similarity score close to 0 indicates that it does not match. We calculate the aggregated similarity score per record pair as a weighted linear combination of all its non-missing feature values. The overall cosine similarity receives a weight of 0.5 while all other features share equally a weight of 0.5. We additionally weight every feature value with the overall density of the corresponding feature as dense features tend to be more important than non-dense ones. The aggregated similarity score of a record pair p with n features with density d is calculated using equation (1).

$$s_p = 0.5 \times \text{cosine_tfidf} + 0.5 \times \frac{\sum_{i=1, \mathrm{f}_{ip} \neq -1}^{n} \mathrm{f}_{ip} \times d_i}{|i, \mathrm{f}_{ip} \neq -1|} \tag{1}$$

Thresholding After the feature values have been aggregated to one similarity score per record pair, a threshold value t needs to be defined for assigning matching labels. We propose a thresholding method which determines the threshold

value t as the *elbow point* (also denoted knee or point of maximum curvature [20]) of the cumulative histogram of the similarity scores of all record pairs after blocking. The elbow value can be approximated as the point with the maximum perpendicular distance to the vector between the first and the last point of the cumulative histogram. Figure 3 shows the elbow point of the cumulative histogram of similarity scores for author record pairs. From the histogram we can see that 12.8K pairs in this dataset have a similarity score below the elbow point.

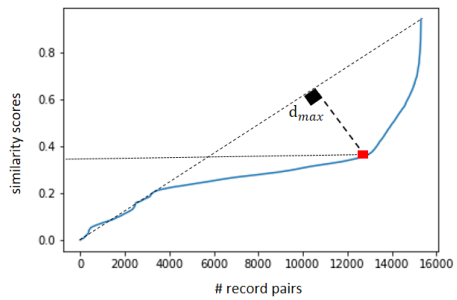


Fig. 3: Elbow point at 0.368 of the cumulative histogram of similarity scores for the dbpedia_dnb record pairs.

In our experiments, we will compare the elbow method to static thresholding which sets the threshold to the middle value of the similarity score range [15, 7] and to Otsu's threshoding method [13]. Otsu's method selects as threshold the value that maximizes the variance between the two classes and therefore expects that the distribution of values is bimodal, i.e. two clear peaks appear in the histograms of similarity scores without any long-tail values.

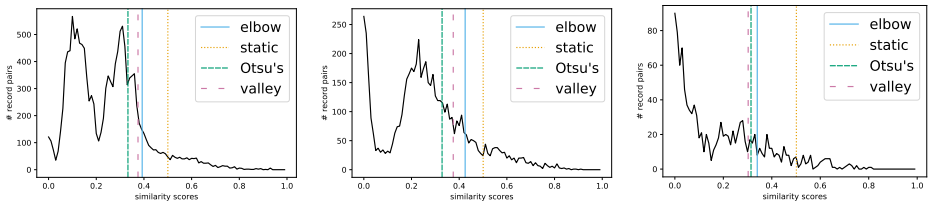
As an additional thresholding baseline, we consider a variation of Otsu's method, known as the *valley-emphasis* threshold which aims to bypass the bimodality assumption [12]. The valley-emphasis threshold, which has also been used in the area of image segmentation, is calculated using equation (2), where p_t is the relative frequency of gray scale t, ω is the probability of each class and μ is the mean gray-level value of each class.

$$t_valley = ArgMax\left\{(1-p_t)\left(\omega_1(t)\mu_1^2(t)+\omega_2(t)\mu_2^2(t)\right)\right\} \tag{2}$$

For the task of image segmentation the relative frequency is calculated as $p_t = n_i/n$, where n_i is the number of occurrences of gray level i and n is the total number of pixels. We adjust the valley-emphasis method to fit the matching task by performing the following two adaptations: (1) We round the similarity scores to the second decimal and calculate n_i as the frequency of the rounded similarity score. In this way we aggregate the occurrences of infrequent values which can allow for reasonable n_i, as the similarity scores can have an arbitrary

number of decimal digits. (2) We set n as the number of occurrences of the most frequent similarity score and not to the total number of record pairs which would be the direct equivalent to the number of pixels. The reason for this adaptation is to allow the valley-emphasis method to have an effect over Otsu's method as otherwise the weighting factor $(1-p_t)$ will always be very close to 1. To the best of our knowledge, we are the first ones to explore and adapt image segmentation thresholding methods to the matching task.

Figure 4 presents the histograms of the similarity scores for three datasets and the threshold boundaries considering the four discussed thresholding methods. It becomes obvious that the similarity distributions among different datasets can vary significantly and therefore a static threshold value cannot fit any distribution. Additionally, Otsu's threshold even when bimodality appears is moved towards the long-tail of the distribution (Figure 4(a)). Finally, the adjusted valley and the proposed elbow method produce similar threshold values.



(a) dbpedia_dnb (b) abt_buy (c) wdc_headphones

Fig. 4: Histograms of similarity scores and threshold boundaries per method.

Confidence weights Apart from their unsupervised matching labels, the record pairs are assigned weights which indicate how confident our unsupervised method is for the predicted label. This is necessary as the record pairs that are expected to be more noisy should affect less the warm start of active learning in comparison to more confident pairs. The confidence weight of a record pair is calculated as the normalized distance of its aggregated similarity score s_p to the threshold value t. Therefore, record pairs close to the decision boundary t will receive a confidence weight close to 0 while record pairs whose similarity scores are the highest or the lowest in the similarity score distribution, will receive a confidence weight close to 1. We use equation (3) for calculating the confidence weight c_p of a record pair p given a threshold value t and a similarity score distribution S.

$$c_p = \begin{cases} \frac{|s_p - t|}{t - \min(S)} & \text{, if } s_p < t \\ \frac{|s_p - t|}{\max(S) - t} & \text{, if } s_p > t \\ 0 & \text{, if } s_p = t \end{cases} \tag{3}$$

2.2 Active Learning

Warm start The unsupervised labeled and weighted record pairs are added in the noisy pool. In a typical pool-based active learning setting the pool contains unlabeled data. In our proposed method the noisy pool contains unsupervised labeled data subject to some degree of noise. The existence of labels in the pool allows us to bootstrap the active learning model as well as the committee of models used as part of the query strategy at no manual labeling cost.

Bootstrapping the active learning model Before starting the active learning process, we use the record pairs of the noisy pool and train a Random Forest classifier with 10 estimators, a minimum split size of 2 while allowing sample replacement and maximum depth. The confidence weights of the record pairs are considered as training weights upon learning. This allows near to zero weighted leaf nodes of the individual trees of the Random Forest classifier to be ignored. In this way, we avoid the over fitting of the initial Random Forest classifier to the most unconfident positive and negative record pairs of the pool.

Bootstrapping the committee We bootstrap the models of the committee for our query strategy by adding one most confident positive and one most confident negative pair of the noisy pool to the labeled set. Thus we can ensure that the labeled set is initialized with record pairs of both classes, match and non-match. The initialized labeled set can be used for training the committee models even at the first active learning iteration where no manual labels are provided.

Modified Query-by-Committee strategy We use a heterogeneous committee for selecting the most informative record pair for labeling. This has been shown to perform better than committees of the same classification model with different model parameterizations [2]. Our committee comprises out of five linear and non-linear classification models: Logistic Regression, Linear SVM, Decision Tree, XGBoost, and Random Forest. The first four classifiers have been shown to achieve good accuracy with little training data [2]. As we apply a Random Forest classifier for model learning, we add this classifier to the committee. In every query iteration, each classification model in the committee is trained on the current labeled set. Next, it votes its predictions on all record pairs of the noisy pool, i.e. every record pair receives five votes. The record pairs with the maximum disagreement are considered to be the most informative. We measure disagreement using vote entropy. We restrict the number of most informative pairs to the ones whose majority vote disagrees with the unsupervised label of the record pair. From this restricted set, one pair is randomly selected for labeling. In this way we aim to select pairs to query whose unsupervised label might be wrong and can therefore lead to the addition of new information to the Random Forest model learned in the warm start phase.

Model learning We propose the incremental training of a classification model per query iteration to allow for a gradual *fading away* effect of the initial model learned in the warm start phase. As this cannot be achieved with the ensemble of committee models designed for the query strategy, we use a Random Forest classifier to which we gradually add more estimators, i.e. trees. Therefore, the model learned in the previous query iterations is not overwritten, a common practice in active learning settings, but expanded. We start our training (active learning iteration 0) with the bootstrapping of the active learning model, as explained previously, by fitting an initial number of trees on the noisy pool of record pairs. Each query iteration adds a small number of new trees to the model of the previous iteration. The added trees are trained on the current labeled set. In the early training iterations, we expect the added estimators to be of low quality and high disagreement on their predictions as they are trained on small amounts of clean data. Therefore, in the early iterations the initial ensemble trained in the warm start phase dominates its predictions over the ones of the added estimators. Once the added estimators become of better quality given the expansion of the labeled set, their prediction agreements will increase, dominate the ones of the initial model and lead to model correction. We set the number of trees learned in the warm start phase to 10 and the increment size of estimators per iteration to 2.

3 Experimental Evaluation

In this section we present the experimental results of our proposed method. First, a detailed description of the datasets used throughout the experimental phase is given. Next, we present and discuss the results of our proposed unsupervised matching method. Finally, the combination of unsupervised matching with active learning is evaluated and compared to two baseline methods.

3.1 Datasets

We use six pairs of datasets for our experimental evaluation from the author and the product domains. The six datasets cover three types of entity resolution problems: structured, textual and dirty, a distinction set by Mudgal et al. [11].

We retrieve two pairs of structured author datasets by exploiting the *owl: sameas* links provided in DBpedia. We use the *owl:sameas* links that link DBpedia author entities to author entities of the DNB[2] (Deutsche Nationalbibliothek) knowledge base and create the *dbpedia_dnb* dataset as well as the VIAF[3] (Virtual International Authority File) knowledge base and create the *dbpedia_viaf* dataset. In total we extract 2887 matching record pairs between DBpedia and DNB and 3353 matching record pairs between DBpedia and VIAF. The DBpedia and DNB datasets have the following attributes in common: *author_name*,

[2] https://www.dnb.de/wir
[3] http://viaf.org/

birthdate, *deathdate*, and *gender*. Between DBpedia and VIAF the attributes *author_name*, *birthdate*, *deathdate*, *gender* and a list of *works* are provided.

We use two pairs of benchmark e-commerce datasets that contain textual attributes. The *abt_buy* datasets [10] derive from two online retailers. A ground truth of 1097 positive correspondences between product entities of the two datasets is provided[4]. The common attributes are: *product name*, *product description* and *product price*. The *product price* attribute has a low density of 18%. The *amazon_google* [10] e-commerce datasets also include product entities described with the following four common attributes: *product name*, *product description*, *manufacturer*, and *price*. The provided ground truth contains 1300 positive correspondences between the amazon and the google product entities.

For the dirty active learning setting we use two e-commerce datasets describing *phones* and *headphones* from the Web Data Commons Project[5]. The datasets have many missing attribute values while existing values are neither normalized nor necessarily correct and therefore fall under the dirty entity matching setting. For our experiments, we disregard all attributes that have a density lower than 0.10. After this filtering, the *wdc_phones* dataset has attributes with densities that range between 11% and 93%. The densities of the attributes for the *wdc_headphones* range between 10% and 91%. The gold standard contains 257 matches for phone products and 225 matches for headphone products [17].

The provided ground truth for all pairs of datasets is complete which allows us to easily create non-matching record pairs. We split the ground truth record pairs after blocking into training (80%) and test (20%). We use the training subset for experimenting with our unsupervised and active learning methods while all results are reported using the test set. Table 1 summarizes the profiling information of the datasets we use for experimentation in terms of initial attributes as well as number of matching and non-matching training and test record pairs. We provide all original and transformed datasets, after feature engineering, for public download[6].

Table 1: Datasets profiling information.

	Dataset	# Aligned Attributes	Train		Test	
			# pos.	# neg.	# pos.	# neg.
Structured	dbpedia_dnb	4	2,310	11,554	577	2,888
	dbpedia_viaf	5	2,552	12,764	801	4,006
Textual	abt_buy	3	878	4,854	219	1,213
	amazon_google	4	1,041	5,714	259	1,428
Noisy	wdc_phones	18	206	1,556	51	389
	wdc_headphones	14	180	983	45	245

[4] https://dbs.uni-leipzig.de/research/projects/object_matching/benchmark_datasets_for_entity_resolution

[5] http://webdatacommons.org/productcorpus

[6] https://github.com/aprimpeli/UnsupervisedBootAL/tree/master/datasets

3.2 Experimental Setup

We run two sets of experiments. First, we evaluate the proposed elbow thresholding method and compare it to the other thresholding methods presented in section 2.1. Next we compare the model performance and stability per active iteration of our proposed bootstrapping method against two baseline methods that do not use unsupervised bootstrapping.

3.3 Thresholding Method Results

We evaluate the elbow point thresholding method proposed for unsupervised matching and compare it to static thresholding, for which the threshold is set to 0.5. Additionally, we perform a comparison to two thresholding methods for binary problems from the field of image segmentation, Otsu's method [13] and the valley-emphasis method [12] after the adjustments explained in section 2.1.

Table 2 presents the results of the four compared thresholding methods in terms of sample correctness (accuracy) and F_1 score. To put the unsupervised matching results into context, we present the difference Δ to the F_1 score achieved in a passive supervised learning scenario, in which all training record pairs are manually labeled and used to train a Random Forest classifier. Additionally, for the four product datasets we compare the results of our passive learning setting to the results reported by state-of-the-art matching systems [11, 16]. The reason for this comparison is two-fold: first, to show that our passive learning setting achieves comparable or better results to state-of-the art matchers and second, to indicate that our passive learning baseline sets a competitive upper boundary for comparing our proposed active learning method.

Comparing the results of the four thresholding methods, we can observe that our proposed elbow point method achieves better results in terms of F_1 score for five of the six datasets in comparison to static thresholding, with the exception of wdc_phones where it underperforms by 2%. However, for the rest of the datasets the elbow method significantly dominates static thresholding by an absolute F_1 margin varying from 1% to 20%.

Otsu's thresholding method underperforms the adjusted valley method by a maximum absolute margin of 32%. It is interesting to observe that the valley method achieves very similar results to our proposed elbow method. However, the elbow method significantly outperforms the valley method for the wdc_phones dataset by 8%. Therefore, we consider the elbow method to generalize the best over all other compared thresholding methods despite the underlying similarity score distribution which can greatly vary among the different datasets as shown in Figure 4. Finally, the elbow thresholding method achieves 11% - 32% lower results in terms of F_1 in comparison to the results achieved with full supervision and has an accuracy of 88% or higher. For the rest of the experimental evaluation we consider the elbow thresholding method.

Table 2: Unsupervised matching results. Comparison of thresholding methods and difference to Supervised Learning.

Dataset	Thresholding Method	Unsupervised Accuracy	Unsupervised F_1	Supervised F_1	Δ to Supervised F_1
dbpedia_dnb	elbow	0.918	0.722	0.976	**-0.254**
	static	0.894	0.538		-0.438
	Otsu's	0.833	0.602		-0.374
	valley	0.906	0.707		-0.269
dbpedia_viaf	elbow	0.956	0.862	0.983	**-0.121**
	static	0.915	0.663		-0.320
	Otsu's	0.743	0.542		-0.441
	valley	0.958	0.861		-0.122
amazon_google	elbow	0.892	0.588	0.699 (0.693 [11])	-0.111
	static	0.882	0.441		-0.258
	Otsu's	0.825	0.600		-0.099
	valley	0.827	0.602		**-0.097**
abt_buy	elbow	0.896	0.674	0.818 (0.628 [11])	**-0.144**
	static	0.912	0.660		-0.158
	Otsu's	0.794	0.562		-0.256
	valley	0.857	0.630		-0.188
wdc_phones	elbow	0.881	0.523	0.851 (0.849 [16])	-0.328
	static	0.881	0.544		**-0.307**
	Otsu's	0.759	0.438		-0.413
	valley	0.757	0.438		-0.413
wdc_headphones	elbow	0.907	0.734	0.966 (0.940 [16])	-0.232
	static	0.898	0.539		-0.427
	Otsu's	0.877	0.682		-0.284
	valley	0.910	0.738		**-0.228**

3.4 Active Learning Results

We run each active learning experiment 5 times and allow 100 iterations for each run. Each iteration corresponds to exactly one manual annotation. We report the average F_1 scores per iteration and the standard deviation (σ) to account for model stability using a separate test set. We compare our proposed method, which we abbreviate with *boot*, to two baseline methods:

Baseline 1: As the first baseline, abbreviated with *no_boot*, we consider an active learning setting with a pool containing all record pairs without labels or weights and a labeled set which is initially empty. As a query strategy, we apply initially random selection until at least one positive and one negative pair is included in the labeled set. After that, we apply a query-by-committee strategy while considering the same model types and disagreement measure explained in section 2.2. A Random Forest classifier is trained in every iteration with the pairs of the labeled set using 10 estimators.

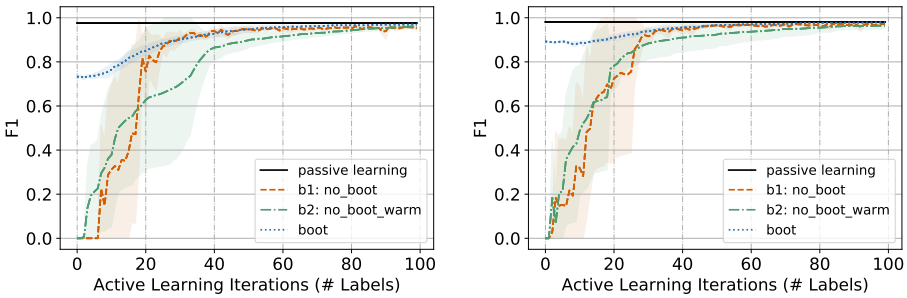
Baseline 2: The second baseline, abbreviated with *no_boot_warm*, is designed in the same way like the first one apart from the model training step. In this

case, we use a warm start setting like in our proposed method with a Random Forest classifier being incrementally expanded. Once the labeled set includes at least one positive and one negative pairs, an initial Random Forest is learned using 10 estimators. Similarly to our *boot* approach, in every iteration two new estimators, i.e. trees, are trained using the labeled set and are added to the initial Random Forest classifier. This baseline guarantees that in every iteration the same number of trees like in the *boot* setting are retrained. In addition, it ensures that the total number of estimators of the Random Forest classifier in every iteration is the same as the one used for our method.

Figures 5, 6, and 7 show the F_1 scores per iteration for the *boot* method in comparison to the two baseline methods and the upper learning bound of passive learning in which all available training data is used. Additionally, we plot the standard deviation σ for every iteration of each active learning setting using the light coloured area around the plotted F1 curves. We observe that for all datasets our method manages to solve the cold start problem while producing stable models at any iteration. In the first active learning iterations (1-40 depending on the dataset) *boot* produces training models of better quality in comparison to the two baselines for all datasets. Considering an active learning setting with a limited budget in terms of manual annotations, our method is preferable as stopping at any iteration produces acceptable results, which is not the case when unsupervised bootstrapping is not applied.

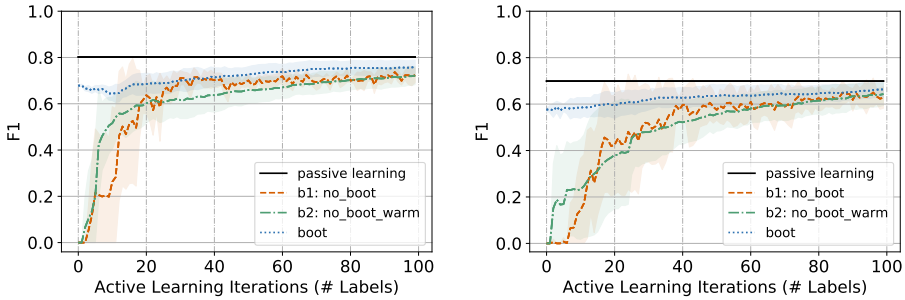
Once the baseline methods go through the cold start phase, their F_1 curves approach the one of the *boot* method and stability increases. In the case of structured datasets, the curves overlap after 30 iterations, signifying that the bootstrapping does not contribute to learning a better model in terms of quality and stability anymore. However, this is not the case for the textual and dirty datasets, for which the *boot* F_1 curve dominates the baseline F_1 curves until the final iteration or until the model converges to the upper learning bound of passive learning, a situation that happens for the wdc_headphones dataset. Therefore, bootstrapping continues to help learning models of better quality even after the cold start phase has passed for the used textual and dirty datasets. A final observation that can be drawn from the three figures, is that the *no_boot_warm* baseline underperforms the *no_boot* baseline in every active learning iteration. This shows that the warm start setting can perform well only when the initially learned model is of an acceptable quality which is guaranteed when it is bootstrapped with unsupervised labeled data but not otherwise.

Table 3 presents the average F_1 scores and standard deviation (σ) for each dataset and method for three snapshots on the 20th, 60th and final iteration. Already in the 20th iteration the *boot* method gives very stable results as the standard deviation ranges from 0.01 to 0.05. At the same iteration point, both baseline methods are significantly more unstable independently from the dataset with the standard deviation ranging from 0.08 to 0.38. This shows that in the cold start phase our proposed *boot* method does not only perform better in terms of F_1 score in comparison to the baseline methods but also produces more stable models. On the 60th iteration all models have recovered from the cold start



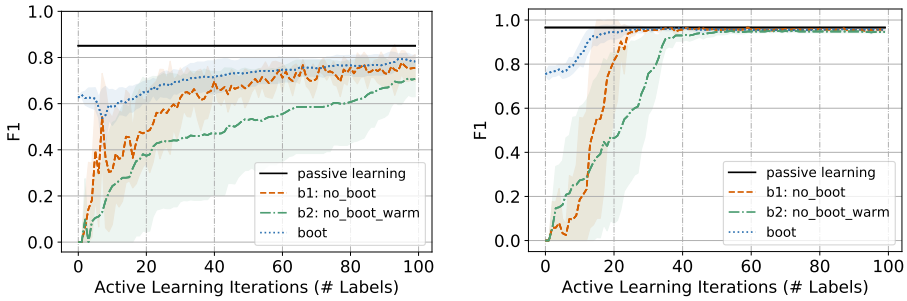
(a) dbpedia_dnb (b) dbpedia_viaf

Fig. 5: F1 and σ per Active Learning Iteration - Structured Datasets.



(a) abt_buy (b) amazon_google

Fig. 6: F1 and σ per Active Learning Iteration - Textual Datasets.



(a) wdc_phones (b) wdc_headphones

Fig. 7: F1 and σ per Active Learning Iteration - Dirty Datasets.

phase and are therefore more stable for all methods with the exception of the *no_boot_warm* baseline which remains highly unstable for the wdc_phones dataset ($\sigma = 0.206$). This extends our previous observation concerning the warm start setting as it becomes obvious that without a good initial model the warm start setting performs weakly in terms of both quality and stability. Finally, on the last iteration, the *boot* method produces the most stable models in comparison to both baselines, which is more profound for the textual and the wdc_phones datasets.

Table 3: Comparison of F_1 and σ.

Dataset	AL method	F1(s)		
		20th iter.	60th iter.	100 iter.
dbpedia_dnb	no_boot	0.756(0.197)	0.953(0.007)	0.952(0.009)
	no_boot_warm	0.628(0.317)	0.916(0.022)	0.961(0.009)
	boot	0.850(0.022)	0.958(0.008)	**0.969(0.001)**
dbpedia_viaf	no_boot	0.725(0.363)	0.967(0.005)	0.972(0.005)
	no_boot_warm	0.782(0.108)	0.937(0.043)	0.964(0.014)
	boot	0.909(0.014)	0.970(0.006)	**0.979(0.002)**
abt_buy	no_boot	0.637(0.080)	0.719(0.034)	0.723(0.031)
	no_boot_warm	0.602(0.086)	0.671(0.046)	0.722(0.039)
	boot	0.685(0.048)	0.738(0.033)	**0.759(0.029)**
amazon_google	no_boot	0.425(0.214)	0.610(0.063)	0.628(0.046)
	no_boot_warm	0.381(0.231)	0.581(0.063)	0.643(0.049)
	boot	0.594(0.055)	0.636(0.041)	**0.663(0.034)**
wdc_phones	no_boot	0.480(0.137)	0.712(0.063)	0.755(0.058)
	no_boot_warm	0.374(0.330)	0.555(0.206)	0.707(0.077)
	boot	0.649(0.050)	0.747(0.053)	**0.783(0.027)**
wdc_headphones	no_boot	0.816(0.242)	0.957(0.008)	0.957(0.008)
	no_boot_warm	0.464(0.386)	0.948(0.006)	0.946(0.004)
	boot	0.945(0.033)	0.955(0.007)	**0.957(0.005)**

4 Related Work

Entity resolution, also referred as record deduplication and entity matching, aims to identify records in one or more datasets that refer to the same real-world entity [3, 5]. Depending on the availability of pre-labeled data, matching methods are divided into unsupervised, weakly supervised and supervised methods [3].

Feature engineering: In order to calculate the similarity between two entities, it is necessary to create features from the combinations of their attributes. Traditionally, features can be created using different similarity functions [3]. The Magellan entity matching system [8] uses data type specific similarity functions per attribute combination. For table row similarity Oulabi et al. [14] suggest the addition of one overall similarity feature using the similarity of the concatenated

attribute values. Recent methods propose the use of word embeddings [4] which however lack interpretability and can perform poorly when the attribute values lack semantic meaning, e.g. in the case of person names or birth dates.

Unsupervised matching: In an unsupervised matching scenario the matching decision for a record pair can be drawn by implementing a set of boolean heuristics [15]. Designing such heuristics requires manual effort. Alternatively, a global threshold can be pre-defined or calculated against which the aggregated pair similarities will be compared [15, 7]. We showed that our elbow thresholding method performs consistently better in comparison to these methods.

Active learning: In the area of supervised matching, many methods have been proposed that aim to reduce the human labeling effort by applying active learning [18, 19, 21]. Typically, committee-based query strategies are applied for selecting informative pairs for labeling [19, 21, 6]. Committee members are usually different parametrizations of the same classification model [19, 21]. Recent work, has shown that having a committee of different classification models is more efficient [2]. Starting active learning pre-assumes the existence of both matching and non-matching pairs in the labeled set. In existing works, the labeled set is initialized with randomly selected labeled pairs [18, 9] or by selecting and labeling record pairs from different areas of the similarity score distribution [2, 19]. Both lines of work increase the human labeling effort in contrast to our method which solely relies on unsupervised matching. Additionally, the methods that rely on selecting and labeling pairs from the similarity score distribution need to predefine a number of similarity groups and the number of pairs from each group that needs to be labeled. However, in our experiments we showed that the similarity score distributions of different datasets may significantly vary and therefore the amount of similarity groups would need to be individually defined for each dataset. Additionally, fixing the labeled set size e.g. to ten pairs, before active learning starts is not optimal as different matching settings can behave very differently in terms of convergence which we showed in our experimental evaluation. Our proposed method achieves good performance, even at the very early iterations while we showed that depending on the difficulty of the matching setting ten pairs can even be enough for reaching maximum performance.

5 Conclusion

We presented a method for bootstrapping active learning for entity resolution using unsupervised matching. In the context of unsupervised matching, we assign labels based on thresholding pre-calculated similarity scores. Our proposed thresholding relies on the elbow point of the cumulative histogram of similarity scores. We showed that the elbow thresholding method performs better in comparison to static thresholding and two thresholding methods from the area of image segmentation. Using the unsupervised labeled data to bootstrap active learning and incrementally expanding a Random Forest classifier after every query iteration, leads to the elimination of the cold start problem appearing in active learning settings that do not apply bootstrapping. Our method guarantees

high anytime performance as it produces even at early active learning iterations better models in terms of quality and stability, compared to methods that do not apply bootstrapping. On top of the improved high anytime performance, our approach continues showing higher stability and F_1 score after 100 iterations especially for datasets containing many missing values as well as rather textual data.

References

1. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
2. Chen, X., et al.: Heterogeneous Committee-Based Active Learning for Entity Resolution (HeALER). In: Proc. of ADBIS. pp. 69–85 (2019)
3. Christophides, V., et al.: End-to-End Entity Resolution for Big Data: A Survey. arXiv:1905.06397 [cs] (2019)
4. Ebraheem, M., et al.: Distributed Representations of Tuples for Entity Resolution. In: Proc. of VLDB. pp. 1454–1467 (2018)
5. Halevy, A., Rajaraman, A., Ordille, J.: Data Integration: the Teenage Years. In: Proc. of VLDB. pp. 9–16 (2006)
6. Isele, R., Bizer, C.: Learning linkage rules using genetic programming. In: Proc. of Ontology Matching. pp. 13–24 (2011)
7. Kejriwal, M., Miranker, D.P.: An Unsupervised Instance Matcher for Schema-free RDF Data. Web Semantics **35**(P2), 102–123 (2015)
8. Konda, P., et al.: Magellan: Toward building entity matching management systems over data science stacks. PVLDB (13), 1581–1584 (2016)
9. Konyushkova, K., Sznitman, R., Fua, P.: Learning Active Learning from Data. In: In Proc. of NIPS. pp. 4225–4235 (2017)
10. Köpcke, H., Rahm, E.: Training Selection for Tuning Entity Matching. In: Proc. of QDB/MUD. pp. 3–12 (2008)
11. Mudgal, S., et al.: Deep learning for entity matching: A design space exploration. In: Proc. of SIGMOD. pp. 19–34 (2018)
12. Ng, H.F.: Automatic thresholding for defect detection. Pattern Recognition Letters **27**(14), 1644–1649 (2006)
13. Otsu, N.: A threshold selection method from gray-level histograms. IEEE transactions on systems, man, and cybernetics **9**(1), 62–66 (1979)
14. Oulabi, Y., Bizer, C.: Extending cross-domain knowledge bases with long tail entities using web table data. In: Proc. of EDBT. pp. 385–396 (2019)
15. Oulabi, Y., Bizer, C.: Using Weak Supervision to Identify Long-Tail Entities for Knowledge Base Completion. In: Proc. of SEMANTiCS. pp. 83–98 (2019)
16. Petrovski, P., Bizer, C.: Learning expressive linkage rules from sparse data. Semantic Web (Preprint), 1–19 (2019)
17. Petrovski, P., et al.: The WDC gold standards for product feature extraction and product matching. In: Proc. of EC-Web. pp. 73–86 (2016)
18. Qian, K., Popa, L., Sen, P.: Active learning for large-scale entity resolution. In: Proc. of CIKM. pp. 1379–1388 (2017)
19. Sarawagi, S., Bhamidipaty, A., Kirpal, A., Mouli, C.: ALIAS: An Active Learning led Interactive Deduplication System. In: Proc. of VLDB. p. 1103–1106 (2002)
20. Satopaa, V., et al.: Finding a" kneedle" in a haystack: Detecting knee points in system behavior. In: Proc. of ICDCS-Workshops. pp. 166–171. IEEE (2011)
21. Tejada, S., Knoblock, C.A., Minton, S.: Learning object identification rules for information integration. Information Systems **26**(8), 607–633 (2001)