Pre-trained Language Models for Blocking in Entity Resolution

Duy Tran
University of Toronto

ABSTRACT

Deep learning (DL) has recently gained popularity for the task of Entity Resolution, which aims to efficiently find different data instances that refer to the same real-world entity. The matching step of the process has gathered more attention, in which approaches using Transformer-based pre-trained language models have achieved state-of-the-art results. There have been fewer DL-based solutions developed for the blocking step, and their relative performance against traditional blocking methods is less clear.

This technical report investigates the use of SimCSE and TSDAE, two self-supervised sentence embedding learning methods, in ER blocking. We empirically determine that TSDAE outperforms the current DL-based state-of-the-art blocking methods. Courtesy of a recent benchmark, we make meaningful comparison of our approach with non-DL blocking methods, which reveals a performance gap that should motivate future DL-based methods.

1 INTRODUCTION

Entity Resolution (ER) is a well-studied problem that aims to identify data instances that refer to the same real-world entity [10, 12, 48]. Given two data sources, the number of potential matches grow quadratically with their sizes, so most ER solutions scale to large datasets by performing two steps: *blocking* and *matching*. This technical report focuses on the blocking step, which picks out a relatively small number of *blocks* or *candidate pairs*, reducing the computational cost of record-to-record comparison during the subsequent matching phase.

Recently, deep learning (DL) has been applied to ER and achieved state-of-the-art results for the matching step [6, 29, 35, 36, 41, 64] and promising results for the blocking step [16, 24, 62, 71, 77]. The key idea of these DL-based blocking methods is to create a high-dimensional vector representation for each tuple based on pre-trained embeddings (e.g. fastText [7]), and cast blocking as an (approximate) nearest neighbor search task based on some similarity measure. Among these works, the current state-of-the-art DL-based blocking methods include DeepBlocker [62] and Sudowoodo [71].

For blocking, DeepBlocker explores several self-supervised DL-based architectures, and finds that an Auto-encoder approach that self-produces the original tuple works well [62]. Sudowoodo [71] uses a contrastive learning framework to pre-train blocking models. In the matching step, Ditto achieves state-of-the-art performance by fine-tuning Transformer-based pre-trained language models (pre-trained LMs for short) [35] in a supervised manner that requires a non-trivial amount of labeled training examples. An approach adapting these ideas may potentially achieve better blocking performance.

In this report, we explore the application of two pre-trained LM-based unsupervised sentence embeddings methods into the blocking step, namely the Simple Contrastive Sentence Embedding framework (SimCSE) [17] and the Transformer-based Sequential Denoising Auto-Encoder method (TSDAE) [70]. Our work makes the following contributions:

(1) We show that our TSDAE-based blocking method consistently outperforms DeepBlocker's Auto-encoder.
(2) Along with a concurrent work [71], our report is among the first efforts to apply contrastive learning and self-supervised sentence embedding learning in the ER blocking step.
(3) For textual data, we conduct experiments in a manner that facilitates meaningful comparison of our DL-based blocking methods and thoroughly fine-tuned non-DL approaches, courtesy of a recent benchmark [47].
(4) While our TSDAE approach outperforms DeepBlocker's, there remains significant gaps until DL-based methods are competitive with some traditional approaches in practice.

In the remainder of this report, we introduce relevant background on blocking (§2), give an overview of SimCSE and TSDAE in the context of ER (§3), present experimental results (§4), review related work (§5), and conclude (§6).

2 BACKGROUND

2.1 Preliminaries

Entity Resolution. Given two sets of records A and B with attributes (A_1,A_2,...A_n) and (B_1,B_2,...B_n), their records as tuples $a = (a_1, a_2, ..., a_n) \in A$ and $b = (b_1, b_2, ..., b_n) \in B$ respectively, the goal of ER is to find the largest possible candidate set $C \subseteq A \times B$ such that for all $(a, b) \in C$, a and b refer to the same real-world entity. ER, ironically, has many different names in the literature, which are often variations of *Deduplication* when $A = B$, and *Record Linkage* otherwise [6, 49]. To tackle the inherent quadratic complexity, most ER systems perform *blocking* before *matching*.

Evaluation Metrics Blocking aims to cheaply group likely matches together with high recall (with typically low precision) and reduce the number of candidate pairs for the subsequent matching step (high-precision pairwise comparison and match/non-match classification). Specifically, common blocking performance metrics include [11, 46, 50]:

(1) *Recall* — the number of true matched candidate pairs divided by the ground-truth (total number of true matched pairs). In other words, it measures the method's effectiveness in not removing true matches.
(2) *Precision* — the number of true matched candidate pairs divided by the total number of candidate pairs generated. A high precision means a method efficiently generates mostly true matched record pairs, and vice versa.
(3) *Reduction ratio* — the fraction of candidate pairs that are removed by a blocking method. A high reduction ratio is desirable, as fewer pairwise comparison would be done during matching. The candidate set size (CSS) $|C|$ or the reduction ratio's complement (candidate set size ratio or CSSR in the literature [62]) could also be used to convey the same idea.

Schema Settings. Blocking methods could be characterized by many dimensions, one of which is *schema awareness* [49]. Schema-aware methods use schema knowledge to form blocks with some specific attributes deemed useful for matching (e.g. determined by measures of coverage, noisiness, and distinctiveness [47]), while schema-agnostic methods simply use all attribute values. The latter are suitable for unstructured & heterogeneous data, albeit with lower precision and reduction ratio [49].

Non-DL Blocking Methods. The research on blocking is extensive (see [11, 34, 48, 49] for surveys and taxonomy). Modern *blocking workflows* consist of a multi-step process: block building, block cleaning, and comparison cleaning [49]. Block cleaning and comparison cleaning aim to remove redundant and superfluous comparisons of blocks, and are often orthogonal to block building methods. Each of these steps presents a design choice, as many techniques are available. Extensive experimental studies have verified that there is no clear winner, and performance is often parameter-sensitive [11, 46, 50].

Traditional block building methods employ hashing, sorting, and various types of indexing to group similar entities into clusters or blocks. Blocks are often defined by a blocking key value derived from a blocking function, which is constructed by rule-based techniques based on syntactic features, heuristics, domain knowledge, and manual attribute selection. Machine learning techniques have also been explored to automatically learn good blocking functions [34, 49].

Recent works has extended the blocking solution space to include nearest-neighbor (NN) methods as alternative or complementary techniques [47, 49, 62]. These methods, including DL-based ones, typically index part or all input data, and form clusters of candidate pairs with similarity search, similarity join, locality-sensitive hashing (LSH), or k-NN search algorithms.

DL-based Blocking Methods. There have been a few DL-based proposals for blocking. Using pre-trained word or sub-word embeddings such as fastText [7], DeepER [16], AutoBlock [77], and DeepBlocker [62] put vector representation of tuples into a high-dimensional space and treat blocking as an approximate NN search using cosine similarity or multi-probing LSH.

DeepER presents two methods for computing the tuple vector: (1) computing each attribute's vector by concatenating unweighted average embeddings of its tokens, or (2) passing attribute vectors through a bidirectional long short-term memory (LSTM) network to account for token order.

AutoBlock uses fastText to first compute attribute embeddings (as a sum of weighted token embeddings) by using a neural network (e.g. LSTM) that was trained on labeled data. With the attribute vectors, it trains a blocking model that maximizes the cosine difference between matches and non-matches. The model is then used to produce multiple tuple signatures that are inserted into an LSH data structure for approximate NN search.

DeepBlocker generalizes these approaches and experiments with a wide variety of DL architectures. DeepBlocker defines an architectural template including three modules: *word embedding*, *tuple embedding*, and *vector pairing*. Its most performant architecture is the Auto-encoder: it converts tuples into embedding vectors with fastText, aggregates those vectors with the smooth inverse frequency (SIF) sentence embedding technique [2] and an Auto-encoder setup (one neural network encoding the SIF embeddings and the other reconstructing the original input); finally the FAISS [27] library is used for approximate top-K cosine similarity search.

Most recently, Sudowoodo [71], a concurrent work that outperforms DeepBlocker, is a pre-trained LM-based contrastive representation learning framework for various data integration and preparation tasks, including blocking and matching. Sudowoodo is a more sophisticated application of contrastive learning compared to our SimCSE approach (see §3), as it integrates various optimization techniques (novel cutoff augmentation, negative sample clustering, redundancy regularization) to enhance performance.

2.2 Pre-trained Language Models and Blocking

Earlier DL-based blocking methods use word or sub-word embeddings such as GloVe [54] or fastText [7] as the starting point; these methods produce context-free representations that have been outperformed in a wide variety of tasks by pre-trained LMs such as the BERT [15]. Pre-trained LMs are deep neural networks with multiple Transformer layers, trained with self-supervised tasks such as masked language modeling (MLM) or next sentence prediction. By paying attention to all tokens in the input sequence, pre-trained LMs produce highly contextualized embeddings that have been shown to better capture syntactic and semantic understanding of natural language [56]. Therefore, pre-trained LMs could potentially generate better tuple embeddings for blocking.

There are few existing works using pre-trained LMs for blocking. DeepBlocker [62] experimented with Sentence-BERT (SBERT) [55] and a "standard transformer" as tuple embedding modules, but both trail behind the Auto-encoder in performance, and their implementation not publicly available. Ditto [35] implements a similar SBERT-based blocking method, albeit in a supervised manner. Most recently, Sudowoodo [71] uses DistilBERT-based [57] contrastive learning, which is similar in principle to our SimCSE approach (see §3).

2.3 Problem Setting

Motivated by the existing design space, we seek to develop high-recall, practical, and hands-off DL-based blocking methods based on pre-trained LMs.

Our ideal methods should require no labeled training data, few parameters, and do not need much tuning. We consider both schema-aware and schema-agnostic settings, which correspond to the structured and textual datasets popular in the literature [41, 62, 71]. Our serialization method (§4) also does not require the two schemas to be aligned or identical. Our focus is on Record Linkage, and while we do not directly consider Deduplication, the methods should easily be generalized to such use cases.

3 SIMCSE & TSDAE

Our general approach is similar to that of existing DL-based works. We pre-train our LM with an self-supervised sentence embedding technique, producing a model to vectorize all data records. For each record, we then use top-K cosine similarity search to find K nearest items, all of which constitute the set of candidate pairs. The two self-supervised sentence embedding methods we investigate are SimCSE and TSDAE.

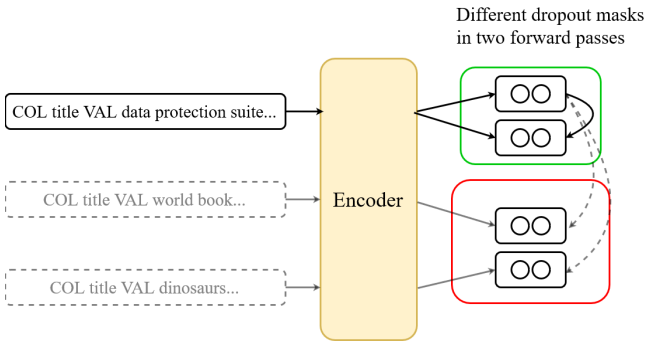

Figure 1: Unsupervised SimCSE uses two dropout masks on the same input to create a "positive" pair, and other in-batch examples are considered "negatives" (dotted lines).

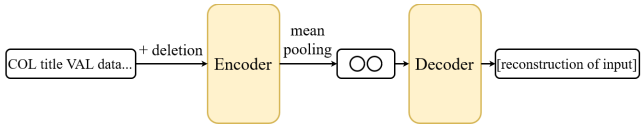

Figure 2: TSDAE resembles a typical auto-encoder setup, but with the addition of deletion noise, and the usage of pre-trained LMs for encoding.

3.1 Similar tuples as similar sentences

Within DeepBlocker's design space, a tuple embedding approach relying on word or sub-word embeddings requires an *aggregator* module to combine word or character vectors into one vector representing the entire original tuple. With pre-trained LMs, one could simply concatenate all attribute values of a tuple into a string, and directly use it as an input sequence for the model to produce the tuple embedding.

A tuple could effectively be viewed as a natural language sentence, and potential matches are sentences that carry similar semantic content. Therefore, the extensive literature on sentence embeddings methods could be leveraged to encode sentences into a fixed-size, dense vector space such that semantically similar sentences are close. We thus proceed with SimCSE and TSDAE, two self-supervised sentence embedding learning methods that are simple to implement and perform well on a variety of sentence embedding tasks.

3.2 SimCSE

SimCSE [17] follows a contrastive learning framework, which aims to pull semantically close neighbors together and push apart non-neighbors [19]. Using a pre-trained LM (e.g. BERT) as an encoder, the unsupervised SimCSE method predicts the input sentence itself with only standard dropout [59] used as noise.

Illustrated by Fig. 1, a sentence input is passed to the encoder twice, each time with an independently sampled dropout mask [65]. The output consisting of two slightly different embeddings is considered a "positive pair", and other sentences in the same mini-batch are "negatives". All parameters are fine-tuned with a cross-entropy objective with in-batch negatives, and the model predicts the positive one among the negatives. This objective is equivalent to the Multiple Negatives Ranking Loss [20], which could be implemented trivially in the SBERT framework [55] (see §4).

Despite being surprisingly simple, the unsupervised SimCSE method matches or exceeds the performance of many supervised embedding methods in semantic textual similarity (STS) [17] and other text embedding tasks [42].

The supervised SimCSE method requires labeled training data that typically need human annotators. Each training example requires at least two labels, one to indicate an *entailment* example (syntactically different but semantically similar) for a positive pair, and the other a contradiction example (syntactically similar but semantically contradicting) for a hard negative pair. As labeled data are typically not available in our problem setting, we only consider the unsupervised variant.

3.3 TSDAE

TSDAE [70] combines pre-trained LMs with the popular Sequential Denoising Auto-Encoder architecture [66]. As shown in Fig. 2, TSDAE introduces noise to input sequences by deleting tokens, and these damaged sentences are encoded by the transformer model with mean pooling into sentence vectors. Another decoder network then attempts to reconstruct the original input from the damaged vectors.

One key difference between TSDAE and the original Transformer (pre-trained with MLM task) [65] is that the TSDAE decoder only has access to the mean-pooled fixed-sized sentence embeddings produced by the encoder, not all contextualized token embeddings as in the original Transformer. This modification potentially forces the encoder to produce better embeddings.

Even though TSDAE does not perform as well as SimCSE on STS tasks, as an unsupervised training method, it has been shown to achieve state-of-the-art results on realistic downstream in-domain tasks. As DeepBlocker's paper has demonstrated the performance of a similar Auto-encoder architecture, we expect TSDAE to also be suitable for ER blocking.

4 EVALUATION

We first describe the implementation, experimental setup, and then present the performance comparisons of SimCSE and TSDAE with other baselines on a variety of datasets in three settings (structured, dirty, and textual). Overall, this evaluation shows that:

(1) TSDAE outperforms DeepBlocker on all datasets in all settings, and is at least competitive with Sudowoodo on datasets we can compare (§4.3).
(2) SimCSE achieves mixed results, generally has higher variance compared to TSDAE and DeepBlocker, and thus is not a robust solution (§4.3).
(3) Compared to a recent benchmark's results [47], DL-based methods like TSDAE do have some advantages over non-DL methods, but remain generally uncompetitive (§4.4).

4.1 Datasets

Our datasets (Tab. 1) are frequently used in the literature, and vary in domains & sizes. For SimCSE and TSDAE experiments with structured and dirty datasets, we modified the datasets used by [41, 62] with the serialization method from Ditto [35]. For a record r =

Type	Dataset	Table A	Table B	Matches
Structured	Amazon-Google	1,363	3,226	1,300
	Beer	4,345	3,000	68
	DBLP-ACM	2,616	2,294	2,224
	Walmart-Amazon	2,554	22,074	962
Dirty	DBLP-ACM	2,616	2,294	2,224
	Walmart-Amazon	2,554	22,074	962
Textual	Abt-Buy	1,076	1,076	1,076
	Amazon-Google	1,354	3,039	1,103
	IMDB-TVDB	5,118	7,810	1,072

Table 1: Datasets

$(r_1,r_2,...,r_n) \in A$ with attributes $(A_1,A_2,...A_n)$:

$$\text{serialize}(r) := \text{[COL]}\ A_1\ \text{[VAL]}\ r_1\ \ldots\ \text{[COL]}\ A_n\ \text{[VAL]}\ r_n$$

where [COL] and [VAL] are special tokens indicating the start of attribute names and values respectively. The dirty datasets are derived from their structured counterparts by a simple modification: each record's attribute is randomly moved to one designated attribute with 50% probability.

For textual datasets, we conduct experiments on the datasets used for schema-agnostic setting from a recent blocking benchmark [47], so that our results are directly comparable with those of thoroughly fine-tuned non-DL blocking and filtering methods from that study (see §4.4). The three datasets chosen only has one "aggregate value" attribute, which concatenates multiple attributes including at least one containing textual blobs. IMDB-TVDB, in particular, exemplifies a hybrid textual dataset: IMDB's records have short textual values that resemble values of structured attributes, while TVDB contains long paragraphs (e.g. reviews, description).

In preliminary experiments, we applied Ditto's summarization technique [35] to textual datasets, which tries to retain the most informative tokens while staying within BERT's limited input length (512 sub-word tokens). For each record, we retain non-stopword tokens with the highest TF-IDF scores. We did *not* find this particular technique to be helpful for blocking, as we observed consistently lower recall for all three textual datasets. Therefore, we simply use the datasets from [47] as provided.

4.2 Implementation and Setup

SimCSE. We use the sentence-transformers framework [55] to implement our SimCSE method, as it provides the equivalent Multiple Negatives Ranking Loss objective [20]. We do not observe significant difference in preliminary experiments with BERT [15], RoBERTa [38], and their distilled versions [57], and so opted for DistilBERT for faster training times.

TSDAE. We use sentence-transformers's out-of-the-box support for adding TSDAE noise (i.e. deletion at a fine-tuned ratio) and TSDAE's objective, as well as DistilBERT for the pre-trained LM.

DeepBlocker. We compare our methods to DeepBlocker's original implementation, which has consistently outperformed earlier DL-based methods [62]. The publicly available implementation of DeepBlocker uses exact top-K cosine similarity search to generate candidate pairs (not FAISS), so we implement the same method for SimCSE and TSDAE for fair comparison.

Dataset	Method	Recall@5 (s.d.)
Amazon-Google (Structured)	DeepBlocker	0.738 (0.004)
	TSDAE	0.860 (0.010)
	SimCSE	0.810 (0.034)
Walmart-Amazon (Structured)	DeepBlocker	0.850 (0.006)
	TSDAE	0.944 (0.008)
	SimCSE	0.686 (0.077)
IMDB-TVDB (Textual)	DeepBlocker	0.764 (0.009)
	TSDAE	0.792 (0.006)
	SimCSE	0.768 (0.022)

Table 2: Representative Recall@5 and standard deviations of DL-based methods

For SimCSE and TSDAE, we do not fine-tune training parameters and simply use the same values of mini-batch size of 8, constant learning rate of $2e-5$, and 5 training epochs. All experiments are run on an Ubuntu Linux machine with 8 cores of an AMD Ryzen Threadripper 2990WX CPU, 64GB RAM, and one AMD Instinct MI100 GPU, except for those of DeepBlocker's for which we use CPU training instead due to an unresolved bug when training on GPU. All results are averages of 5 runs with 5 random seeds.

We vary the value of top-K to change the CSS (e.g. $K=5$ means each tuple in the smaller table is paired with 5 tuples in the larger table with the highest cosine similarity scores). For presentation purposes, we plot results with K instead of reduction ratio (or its complement CSSR), as we aim to only generate candidate set sizes comparable to those of non-DL methods on the same datasets. Using results from [47], we experimentally find K = [5, 200] to be a good range for performance to converge and to produce candidate sets on the same orders of magnitude with non-DL methods (see §4.4). Ideally, we want high recall and low K (i.e. low CSS).

4.3 Comparison with DL-based methods

Structured Data: Fig. 3 shows the value of K on a x-axis, and the recall on the y-axis, and methods closer to the top left of the plot are better, as they achieve higher recall for smaller K. Overall, TSDAE consistently achieves the best performance, reaching high recall (above 90%) for low K values (5-10) on all structured datasets. Compared to DeepBlocker, TSDAE achieves significant improvement on Amazon-Google, Beer, and Walmart-Amazon, as well as comparable performance on DBLP-ACM. SimCSE's performance is mixed, performing significantly worse in Walmart-Amazon while outperforming DeepBlocker in others. In line with the observation made in [44, 70], SimCSE's good performance on STS tasks could poorly correlate with other downstream tasks. Unlike DeepBlocker and TSDAE, SimCSE also displays significantly higher variance among random seeds (see Tab. 2).

Dirty Data: As dirty datasets are derived by moving around attribute values, we expect that a method would have similar performance to structured data if it were robust. DeepBlocker's Auto-encoder [62] does not rely on sequential information as it relies on a bag-of-word aggregation module (SIF), and hence its performance not affected. Fig. 4 shows similar results for TSDAE, as its curves in dirty DBLP-ACM and Walmart-Amazon closely resemble its structured counterparts. SimCSE suffers a 5% reduction in recall on average at all values of K in Walmart-Amazon. Pre-trained LMs with MLM task like BERT do in theory take into account sequential information [15], and this

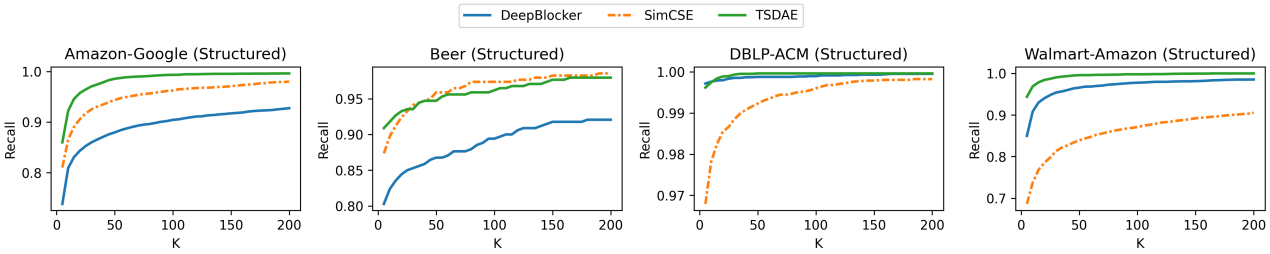

Figure 3: Recall - K comparison of DL-based blocking methods on structured datasets

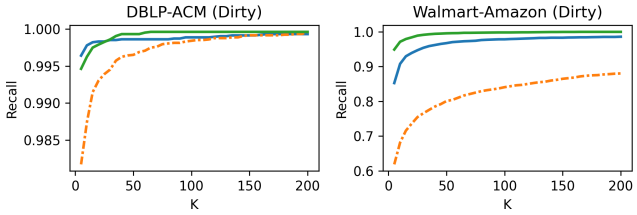

Figure 4: Recall - K comparison of DL-based blocking methods on dirty datasets

Dataset	Method	Recall	CSS
Amazon-Google	DeepBlocker	0.871	54,520
	Sudowoodo	0.973	48,390
	SimCSE	0.935	54,520
	TSDAE	0.974	47,705
DBLP-ACM	DeepBlocker	0.997	13,080
	Sudowoodo	0.996	11,470
	SimCSE	0.968	11,470
	TSDAE	0.996	11,470
Walmart-Amazon	DeepBlocker	0.940	51,080
	Sudowoodo	0.950	44,148
	SimCSE	0.785	51,080
	TSDAE	0.979	44,148

Table 3: Recall comparison with Sudowoodo on some structured datasets

result suggests that the Autoencoder approach in DeepBlocker and TSDAE are more robust against this kind of dirtiness.

Textual Data: Fig. 5 shows that in schema-agnostic textual settings, TSDAE is the best performer. It outperforms DeepBlocker slightly in the Amazon-Google dataset, has decent performance gain for IMDB-TVDB, and is significantly ahead in the Abt-Buy dataset. SimCSE remains inconsistent, performing well only in two out of three datasets.

Comparison with Sudowoodo: Sudowoodo's implementation [71] is not publicly available. It does, however, make some comparison with DeepBlocker, and from their reported candidate set sizes, we report each method's respective recall performance for approximately equivalent CSS in Tab. 3. Slightly smaller or larger values of K (which affects CSS) are chosen to demonstrate clearly the superior or inferior performance of a method. Sudowoodo does not conduct extensive evaluation for blocking, and some dataset characteristics are missing from the paper, so we only include the *structured* datasets that are certainly identical to the ones we use.

While more experiments are warranted for a fair comparison with Sudowoodo, Tab. 3 shows that between the two contrastive learning approaches, Sudowoodo is more performant and robust than SimCSE, while our TSDAE still maintains a slight edge in recall for smaller or equivalent CSS.

4.4 Comparison with non-DL methods

After observing the good performance and robustness of our TSDAE-based blocking method, we now compare it with existing state-of-the-art non-DL solutions, which entails many challenges.

Many traditional blocking methods exist, some requiring domain knowledge or prior attribute selection, so we narrow down our focus to only unsupervised schema-agnostic approaches. Second, our DL-based methods could simply control the CSS by varying the value of K, whereas each non-DL method has their own set of parameters, and it is often difficult to vary these parameters generate a desired CSS. Lastly, as traditional block building methods can be paired with a variety of block cleaning and comparison cleaning methods, the configuration space is large and often necessary, as traditional methods are typically parameter-sensitive and usually benefit from heavy fine-tuning [11, 46, 50].

A recent benchmark study [47] thoroughly conducts exhaustive grid search to fine-tune performance of 14 state-of-the-art blocking methods on 10 datasets. It is also the first work to apply many similarity join and NN search methods into blocking (e.g. kNN-Join [32, 58]). For each method and each dataset, the study optimizes parameters to maximize precision for recall ≥ 0.9, which is a slightly different objective but still produces results that can be compared to ours.

Evaluation Method: To facilitate evaluation, for our textual experiments in §4.3, we use the same datasets used by the benchmark's schema-agnostic settings. Therefore, our results on textual datasets are directly comparable with the benchmark's results. We compare our best-performing TSDAE solution to their two best schema-agnostic solutions, Standard Blocking Workflow (SBW) and kNN-Join. As their grid search optimizes parameters given a recall

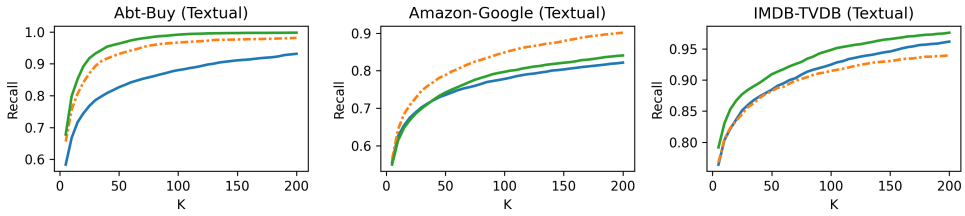

Figure 5: Recall - K comparison of DL-based blocking methods on textual datasets

Dataset	Method	Recall	Precision	CSS
Abt-Buy	TSDAE (K = 25)	0.918	0.136	26,900
	SBW	0.902	0.216	4,493
	kNN-Join	0.924	0.229	4,345
Amazon-Google	TSDAE (K = 545)	0.900	0.001	737,930
	SBW	0.901	0.017	59,685
	kNN-Join	0.900	0.028	35,333
IMDB-TVDB	TSDAE (K = 25)	0.902	0.005	230,310
	SBW	0.924	0.189	5,238
	kNN-Join	0.910	0.122	8,013

Table 4: Comparison with non-DL solutions on schema-agnostic textual datasets

threshold ≥ 0.9, we identify the value of K for which TSDAE achieves recall ≥ 0.9, and compare the methods' precision and CSS.

Comparison Results: Tab. 4 shows the corresponding precision and CSS of TSDAE, SBW, and kNN-Join when achieving high recall on schema-agnostic textual datasets. Even though TSDAE displays good results compared to other DL-based methods, traditional blocking workflows and other NN search methods could perform significantly better when their parameters are fine-tuned (the benchmark optimizes SBW over a space of 3,440 theoretical number of configurations, and 12,000 for kNN-Join). Depending on the use case, TSDAE is a good hands-off solution with only the top-K value as a parameter. Nevertheless, since fine-tuned non-DL blocking methods can achieve high recall with better precision and orders-of-magnitude smaller CSS, they remain the better choice in practice for ER blocking.

5 RELATED WORK

An overview of ER can be found in [10, 12, 48]. There have been multiple efforts spent on building scalable end-to-end ER systems, such as JedAI [40], Magellan [33], pyJedAI [45], and Machop [69]. A recent survey [6] focuses on the application of DL for ER, and many surveys of blocking are available [11, 34, 46, 49, 50] (also see §2 for a short review of non-DL and DL-based blocking approaches).

[36] summarizes ER matching's state of the art and open problems. Earlier DL-based ER matching systems include DeepER [16] and DeepMatcher [41], and approaches applying pre-trained LMs have achieved state-of-the-art results [8, 35, 51, 53]. There have also been attempts to reduce the amount of labeled data required for training of DL-based matching systems by transfer learning [26, 31, 63, 64, 79] and active learning [21, 23, 43]. The rise of DL-based methods have also motivated research into explainable AI for ER models [4, 5, 67], which attempts to interpret and explain matching decisions, as well as efforts to build better benchmarking frameworks [42, 68, 72].

An overview of pre-trained LMs used in NLP tasks can be found in [28, 80], and [22] surveys general text embedding approaches. Pre-trained LM-based unsupervised contrastive learning methods have been popular for learning sentence embeddings, and such methods [9, 13, 18, 25, 30, 37, 61, 74–76, 78] can be further investigated in place of SimCSE for blocking purposes. TSDAE could be characterized as an autoencoder-based pretraining method, of which many alternatives can be found in [80]. Another line of work focuses on learning relational or tabular data representation [3, 14, 60], which could potentially benefit the ER task for structured data.

Some contrastive objectives have been explored for the ER matching step [1, 39, 52, 73], typically in a supervised framework. To the best of our knowledge, besides our report, Sudowoodo [71] is the only work that had applied contrastive learning to blocking and DeepBlocker [62] the only one with an Auto-encoder approach.

6 CONCLUSION

This technical report has demonstrated the effectiveness of unsupervised sentence embedding methods in the task of ER blocking, opening up the design space of DL solutions to pre-trained LMs, contrastive learning, and denoising autoencoder frameworks. Our TSDAE-based blocking approach outperforms DeepBlocker and Sudowoodo, both state-of-the-art DL-based methods, although more future work is required to improve the performance of DL-based blocking methods as compared to that of fine-tuned non-DL methods.

REFERENCES

[1] Mario Almagro, David Jiménez, Diego Ortego, Emilio J. Almazán, and Eva Martínez. 2022. Block-SCL: Blocking Matters for Supervised Contrastive Learning in Product Matching. *CoRR* abs/2207.02008 (2022). https://doi.org/10.48550/arXiv.2207.02008 arXiv:2207.02008

[2] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=SyK00v5xx

[3] Gilbert Badaro and Paolo Papotti. 2022. Transformers for Tabular Data Representation: A Tutorial on Models and Applications. *Proc. VLDB Endow.* 15, 12 (2022), 3746–3749. https://www.vldb.org/pvldb/vol15/p3746-badaro.pdf

[4] Andrea Baraldi, Francesco Del Buono, Matteo Paganelli, and Francesco Guerra. 2021. Landmark Explanation: An Explainer for Entity Matching Models. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*. ACM, 4680–4684. https://doi.org/10.1145/3459637.3481981

[5] Nils Barlaug. 2021. LEMON: Explainable Entity Matching. *CoRR* abs/2110.00516 (2021). arXiv:2110.00516 https://arxiv.org/abs/2110.00516

[6] Nils Barlaug and Jon Atle Gulla. 2021. Neural Networks for Entity Matching: A Survey. *ACM Trans. Knowl. Discov. Data* 15, 3 (2021), 52:1–52:37. https://doi.org/10.1145/3442200

[7] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2017. Enriching Word Vectors with Subword Information. *Trans. Assoc. Comput. Linguistics* 5 (2017), 135–146. https://doi.org/10.1162/tacl_a_00051

[8] Ursin Brunner and Kurt Stockinger. 2020. Entity Matching with Transformer Architectures - A Step Forward in Data Integration. In *Proceedings of the 23rd International Conference on Extending Database Technology, EDBT 2020, Copenhagen, Denmark, March 30 - April 02, 2020*. OpenProceedings.org, 463–473.

https://doi.org/10.5441/002/edbt.2020.58

[9] Fredrik Carlsson, Amaru Cuba Gyllensten, Evangelia Gogoulou, Erik Ylipää Hellqvist, and Magnus Sahlgren. 2021. Semantic Re-tuning with Contrastive Tension. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=Ov_sMNau-PF

[10] Peter Christen. 2012. *Data Matching - Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer. https://doi.org/10.1007/978-3-642-31164-2

[11] Peter Christen. 2012. A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication. *IEEE Trans. Knowl. Data Eng.* 24, 9 (2012), 1537–1555. https://doi.org/10.1109/TKDE.2011.127

[12] Vassilis Christophides, Vasilis Efthymiou, Themis Palpanas, George Papadakis, and Kostas Stefanidis. 2021. An Overview of End-to-End Entity Resolution for Big Data. *ACM Comput. Surv.* 53, 6 (2021), 127:1–127:42. https://doi.org/10.1145/3418896

[13] Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljacic, Shang-Wen Li, Scott Yih, Yoon Kim, and James R. Glass. 2022. DiffCSE: Difference-based Contrastive Learning for Sentence Embeddings. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*. Association for Computational Linguistics, 4207–4218. https://doi.org/10.18653/v1/2022.naacl-main.311

[14] Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2022. TURL: Table Understanding through Representation Learning. *SIGMOD Rec.* 51, 1 (2022), 33–40. https://doi.org/10.1145/3542700.3542709

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 4171–4186. https://doi.org/10.18653/v1/n19-1423

[16] Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq R. Joty, Mourad Ouzzani, and Nan Tang. 2018. Distributed Representations of Tuples for Entity Resolution. *Proc. VLDB Endow.* 11, 11 (2018), 1454–1467. https://doi.org/10.14778/3236187.3236198

[17] Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*. Association for Computational Linguistics, 6894–6910. https://doi.org/10.18653/v1/2021.emnlp-main.552

[18] John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Online, 879–895. https://doi.org/10.18653/v1/2021.acl-long.72

[19] Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality Reduction by Learning an Invariant Mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), 17-22 June 2006, New York, NY, USA*. IEEE Computer Society, 1735–1742. https://doi.org/10.1109/CVPR.2006.100

[20] Matthew L. Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient Natural Language Response Suggestion for Smart Reply. *CoRR* abs/1705.00652 (2017). arXiv:1705.00652 http://arxiv.org/abs/1705.00652

[21] Jiacheng Huang, Wei Hu, Zhifeng Bao, Qijin Chen, and Yuzhong Qu. 2022. Deep entity matching with adversarial active learning. *The VLDB Journal* (28 Apr 2022). https://doi.org/10.1007/s00778-022-00745-1

[22] Francesca Incitti, Federico Urli, and Lauro Snidaro. 2023. Beyond word embeddings: A survey. *Information Fusion* 89 (2023), 418–436. https://doi.org/10.1016/j.inffus.2022.08.024

[23] Arjit Jain, Sunita Sarawagi, and Prithviraj Sen. 2022. Deep Indexed Active Learning for Matching Heterogeneous Entity Representations. *Proc. VLDB Endow.* 15, 1 (jan 2022), 31–45. https://doi.org/10.14778/3485450.3485455

[24] Delaram Javdani, Hossein Rahmani, Milad Allahgholi, and Fatemeh Karimkhani. 2019. DeepBlock: A Novel Blocking Approach for Entity Resolution using Deep Learning. In *2019 5th International Conference on Web Research (ICWR)*. 41–44. https://doi.org/10.1109/ICWR.2019.8765267

[25] Yuxin Jiang, Linhan Zhang, and Wei Wang. 2022. Improved Universal Sentence Embeddings with Prompt-based Contrastive Learning and Energy-based Learning. https://doi.org/10.48550/ARXIV.2203.06875

[26] Di Jin, Bunyamin Sisman, Hao Wei, Xin Luna Dong, and Danai Koutra. 2021. Deep Transfer Learning for Multi-source Entity Linkage via Domain Adaptation. *Proc. VLDB Endow.* 15, 3 (2021), 465–477. https://doi.org/10.14778/3494124.3494131

[27] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-Scale Similarity Search with GPUs. *IEEE Trans. Big Data* 7, 3 (2021), 535–547. https://doi.org/10.1109/TBDATA.2019.2921572

[28] Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. 2021. AMMUS : A Survey of Transformer-based Pretrained Models in Natural Language Processing. *CoRR* abs/2108.05542 (2021). arXiv:2108.05542 https://arxiv.org/abs/2108.05542

[29] Jungo Kasai, Kun Qian, Sairam Gurajada, Yunyao Li, and Lucian Popa. 2019. Low-resource Deep Entity Resolution with Transfer and Active Learning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, 5851–5861. https://doi.org/10.18653/v1/p19-1586

[30] Taeuk Kim, Kang Min Yoo, and Sang-goo Lee. 2021. Self-Guided Contrastive Learning for BERT Sentence Representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*. Association for Computational Linguistics, 2528–2540. https://doi.org/10.18653/v1/2021.acl-long.197

[31] Nishadi Kirielle, Peter Christen, and Thilina Ranbaduge. 2022. TransER: Homogeneous Transfer Learning for Entity Resolution. In *Proceedings of the 25th International Conference on Extending Database Technology, EDBT 2022, Edinburgh, UK, March 29 - April 1, 2022*. OpenProceedings.org, 2:118–2:130. https://doi.org/10.48786/edbt.2022.03

[32] Daniel Kocher and Nikolaus Augsten. 2019. A Scalable Index for Top-k Subtree Similarity Queries. In *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*. ACM, 1624–1641. https://doi.org/10.1145/3299869.3319892

[33] Pradap Konda, Sanjib Das, Paul Suganthan G. C., AnHai Doan, Adel Ardalan, Jeffrey R. Ballard, Han Li, Fatemah Panahi, Haojun Zhang, Jeffrey F. Naughton, Shishir Prasad, Ganesh Krishnan, Rohit Deep, and Vijay Raghavendra. 2016. Magellan: Toward Building Entity Matching Management Systems. *Proc. VLDB Endow.* 9, 12 (2016), 1197–1208. https://doi.org/10.14778/2994509.2994535

[34] Bohan Li, Yi Liu, Anman Zhang, Wen-Huan Wang, and Shuo Wan. 2020. A Survey on Blocking Technology of Entity Resolution. *J. Comput. Sci. Technol.* 35, 4 (2020), 769–793. https://doi.org/10.1007/s11390-020-0350-4

[35] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. 2020. Deep Entity Matching with Pre-Trained Language Models. *Proc. VLDB Endow.* 14, 1 (2020), 50–60. https://doi.org/10.14778/3421424.3421431

[36] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, Jin Wang, Wataru Hirota, and Wang-Chiew Tan. 2021. Deep Entity Matching: Challenges and Opportunities. *ACM J. Data Inf. Qual.* 13, 1 (2021), 1:1–1:17. https://doi.org/10.1145/3431816

[37] Fangyu Liu, Ivan Vulic, Anna Korhonen, and Nigel Collier. 2021. Fast, Effective, and Self-Supervised: Transforming Masked Language Models into Universal Lexical and Sentence Encoders. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*. Association for Computational Linguistics, 1442–1459. https://doi.org/10.18653/v1/2021.emnlp-main.109

[38] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019). arXiv:1907.11692 http://arxiv.org/abs/1907.11692

[39] Michael Loster, Ioannis K. Koumarelas, and Felix Naumann. 2021. Knowledge Transfer for Entity Resolution with Siamese Neural Networks. *ACM J. Data Inf. Qual.* 13, 1 (2021), 2:1–2:25. https://doi.org/10.1145/3410157

[40] Georgios M. Mandilaras, George Papadakis, Luca Gagliardelli, Giovanni Simonini, Emmanouil Thanos, George Giannakopoulos, Sonia Bergamaschi, Themis Palpanas, Manolis Koubarakis, Alicia Lara-Clares, and Antonio Fariña. 2021. Reproducible experiments on Three-Dimensional Entity Resolution with JedAI. *Inf. Syst.* 102 (2021), 101830. https://doi.org/10.1016/j.is.2021.101830

[41] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. 2018. Deep Learning for Entity Matching: A Design Space Exploration. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*. ACM, 19–34. https://doi.org/10.1145/3183713.3196926

[42] Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. MTEB: Massive Text Embedding Benchmark. *arXiv preprint arXiv:2210.07316* (2022). https://doi.org/10.48550/ARXIV.2210.07316

[43] Youcef Nafa, Qun Chen, Zhaoqiang Chen, Xingyu Lu, Haiyang He, Tianyi Duan, and Zhanhuai Li. 2022. Active deep learning on entity resolution by risk sampling. *Knowl. Based Syst.* 236 (2022), 107729. https://doi.org/10.1016/j.knosys.2021.107729

[44] Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, Johannes Heidecke, Pranav Shyam, Boris Power, Tyna Eloundou Nekoul, Girish Sastry, Gretchen Krueger, David Schnurr, Felipe Petroski Such, Kenny Hsu, Madeleine Thompson, Tabarak Khan, Toki Sherbakov, Joanne Jang, Peter Welinder, and Lilian Weng. 2022. Text and Code Embeddings by Contrastive Pre-Training. *CoRR* abs/2201.10005 (2022). arXiv:2201.10005 https://arxiv.org/abs/2201.10005

[45] Konstantinos Nikoletos, George Papadakis, and Manolis Koubarakis. 2022. pyJedAI: a Lightsaber for Link Discovery. In *Proceedings of the ISWC 2022 Posters,*

Demos and Industry Tracks: From Novel Ideas to Industrial Practice co-located with 21st International Semantic Web Conference (ISWC 2022), Virtual Conference, Hangzhou, China, October 23-27, 2022 (CEUR Workshop Proceedings), Vol. 3254. CEUR-WS.org. http://ceur-ws.org/Vol-3254/paper366.pdf

[46] George Papadakis, George Alexiou, George Papastefanatos, and Georgia Koutrika. 2015. Schema-agnostic vs Schema-based Configurations for Blocking Methods on Homogeneous Data. *Proc. VLDB Endow.* 9, 4 (2015), 312–323. https://doi.org/10.14778/2856318.2856326

[47] George Papadakis, Marco Fisichella, Franziska Schoger, George Mandilaras, Nikolaus Augsten, and Wolfgang Nejdl. 2022. How to reduce the search space of Entity Resolution: with Blocking or Nearest Neighbor search? https://doi.org/10.48550/ARXIV.2202.12521

[48] George Papadakis, Ekaterini Ioannou, Emanouil Thanos, and Themis Palpanas. 2021. *The Four Generations of Entity Resolution.* Morgan & Claypool Publishers. https://doi.org/10.2200/S01067ED1V01Y202012DTM064

[49] George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, and Themis Palpanas. 2021. Blocking and Filtering Techniques for Entity Resolution: A Survey. *ACM Comput. Surv.* 53, 2 (2021), 31:1–31:42. https://doi.org/10.1145/3377455

[50] George Papadakis, Jonathan Svirsky, Avigdor Gal, and Themis Palpanas. 2016. Comparative Analysis of Approximate Blocking Techniques for Entity Resolution. *Proc. VLDB Endow.* 9, 9 (2016), 684–695. https://doi.org/10.14778/2947618.2947624

[51] Ralph Peeters and Christian Bizer. 2021. Dual-Objective Fine-Tuning of BERT for Entity Matching. *Proc. VLDB Endow.* 14, 10 (2021), 1913–1921. https://doi.org/10.14778/3467861.3467878

[52] Ralph Peeters and Christian Bizer. 2022. Supervised Contrastive Learning for Product Matching. In *Companion of The Web Conference 2022, Virtual Event / Lyon, France, April 25 - 29, 2022*. ACM, 248–251. https://doi.org/10.1145/3487553.3524254

[53] Ralph Peeters, Christian Bizer, and Goran Glavas. 2020. Intermediate Training of BERT for Product Matching. In *Proceedings of the 2nd International Workshop on Challenges and Experiences from Data Integration to Knowledge Graphs co-located with 46th International Conference on Very Large Data Bases, DI2KG@VLDB 2020, Tokyo, Japan, August 31, 2020 (CEUR Workshop Proceedings)*, Vol. 2726. CEUR-WS.org. http://ceur-ws.org/Vol-2726/paper1.pdf

[54] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL, 1532–1543. https://doi.org/10.3115/v1/d14-1162

[55] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 3980–3990. https://doi.org/10.18653/v1/D19-1410

[56] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2021. A Primer in BERTology: What We Know About How BERT Works. *Transactions of the Association for Computational Linguistics* 8 (01 2021), 842–866. https://doi.org/10.1162/tacl_a_00349 arXiv:https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00349/1923281/tacl_a_00349.pdf

[57] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR* abs/1910.01108 (2019). arXiv:1910.01108 http://arxiv.org/abs/1910.01108

[58] Yasin N. Silva, Walid G. Aref, Per-Åke Larson, Spencer Pearson, and Mohamed H. Ali. 2013. Similarity queries: their conceptual evaluation, transformations, and processing. *VLDB J.* 22, 3 (2013), 395–420. https://doi.org/10.1007/s00778-012-0296-4

[59] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15, 1 (2014), 1929–1958. https://doi.org/10.5555/2627435.2670313

[60] Nan Tang, Ju Fan, Fangyi Li, Jianhong Tu, Xiaoyong Du, Guoliang Li, Samuel Madden, and Mourad Ouzzani. 2021. RPT: Relational Pre-trained Transformer Is Almost All You Need towards Democratizing Data Preparation. *Proc. VLDB Endow.* 14, 8 (2021), 1254–1261. https://doi.org/10.14778/3457390.3457391

[61] Zilu Tang, Muhammed Yusuf Kocyigit, and Derry Tanti Wijaya. 2022. AugCSE: Contrastive Sentence Embedding with Diverse Augmentations. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2022 - Volume 1: Long Papers, Online Only, November 20-23, 2022*. Association for Computational Linguistics, 375–398. https://aclanthology.org/2022.aacl-main.30

[62] Saravanan Thirumuruganathan, Han Li, Nan Tang, Mourad Ouzzani, Yash Govind, Derek Paulsen, Glenn Fung, and AnHai Doan. 2021. Deep Learning for Blocking in Entity Matching: A Design Space Exploration. *Proc. VLDB Endow.* 14, 11 (2021), 2459–2472. https://doi.org/10.14778/3476249.3476294

[63] Saravanan Thirumuruganathan, Shameem Ahamed Puthiya Parambath, Mourad Ouzzani, Nan Tang, and Shafiq R. Joty. 2018. Reuse and Adaptation for Entity Resolution through Transfer Learning. *CoRR* abs/1809.11084 (2018). arXiv:1809.11084 http://arxiv.org/abs/1809.11084

[64] Jianhong Tu, Ju Fan, Nan Tang, Peng Wang, Chengliang Chai, Guoliang Li, Ruixue Fan, and Xiaoyong Du. 2022. Domain Adaptation for Deep Entity Resolution. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*. ACM, 443–457. https://doi.org/10.1145/3514221.3517870

[65] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[66] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *J. Mach. Learn. Res.* 11 (2010), 3371–3408. https://doi.org/10.5555/1756006.1953039

[67] Jin Wang and Yuliang Li. 2022. Minun: evaluating counterfactual explanations for entity matching. In *DEEM '22: Proceedings of the Sixth Workshop on Data Management for End-To-End Machine Learning Philadelphia, PA, USA, 12 June 2022*. ACM, 7:1–7:11. https://doi.org/10.1145/3533028.3533304

[68] Jin Wang, Yuliang Li, and Wataru Hirota. 2021. Machamp: A Generalized Entity Matching Benchmark. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*. ACM, 4633–4642. https://doi.org/10.1145/3459637.3482008

[69] Jin Wang, Yuliang Li, Wataru Hirota, and Eser Kandogan. 2022. Machop: an end-to-end generalized entity matching framework. In *aiDM '22: Proceedings of the Fifth International Workshop on Exploiting Artificial Intelligence Techniques for Data Management, Philadelphia, Pennsylvania, USA, 17 June 2022*. ACM, 2:1–2:10. https://doi.org/10.1145/3533702.3534910

[70] Kexin Wang, Nils Reimers, and Iryna Gurevych. 2021. TSDAE: Using Transformer-based Sequential Denoising Auto-Encoder for Unsupervised Sentence Embedding Learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*. Association for Computational Linguistics, 671–688. https://doi.org/10.18653/v1/2021.findings-emnlp.59

[71] Runhui Wang, Yuliang Li, and Jin Wang. 2022. Sudowoodo: Contrastive Self-supervised Learning for Multi-purpose Data Integration and Preparation. *CoRR* abs/2207.04122 (2022). https://doi.org/10.48550/arXiv.2207.04122 arXiv:2207.04122

[72] Tianshu Wang, Hongyu Lin, Cheng Fu, Xianpei Han, Le Sun, Feiyu Xiong, Hui Chen, Minlong Lu, and Xiuwen Zhu. 2022. Bridging the Gap between Reality and Ideality of Entity Matching: A Revisiting and Benchmark Re-Construction. *CoRR* abs/2205.05889 (2022). https://doi.org/10.48550/arXiv.2205.05889 arXiv:2205.05889

[73] Zhengyang Wang, Bunyamin Sisman, Hao Wei, Xin Luna Dong, and Shuiwang Ji. 2020. CorDEL: A Contrastive Deep Learning Approach for Entity Linkage. In *20th IEEE International Conference on Data Mining, ICDM 2020, Sorrento, Italy, November 17-20, 2020*. IEEE, 1322–1327. https://doi.org/10.1109/ICDM50108.2020.00171

[74] Xing Wu, Chaochen Gao, Zijia Lin, Jizhong Han, Zhongyuan Wang, and Songlin Hu. 2022. InfoCSE: Information-aggregated Contrastive Learning of Sentence Embeddings. *CoRR* abs/2210.06432 (2022). https://doi.org/10.48550/arXiv.2210.06432 arXiv:2210.06432

[75] Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020. CLEAR: Contrastive Learning for Sentence Representation. *CoRR* abs/2012.15466 (2020). arXiv:2012.15466 https://arxiv.org/abs/2012.15466

[76] Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. ConSERT: A Contrastive Framework for Self-Supervised Sentence Representation Transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*. Association for Computational Linguistics, 5065–5075. https://doi.org/10.18653/v1/2021.acl-long.393

[77] Wei Zhang, Hao Wei, Bunyamin Sisman, Xin Luna Dong, Christos Faloutsos, and David Page. 2020. AutoBlock: A Hands-off Blocking Framework for Entity Matching. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*. ACM, 744–752. https://doi.org/10.1145/3336191.3371813

[78] Yan Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. 2020. An Unsupervised Sentence Embedding Method by Mutual Information Maximization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics, 1601–1610. https://doi.org/10.18653/v1/2020.emnlp-main.124

[79] Chen Zhao and Yeye He. 2019. Auto-EM: End-to-end Fuzzy Entity-Matching using Pre-trained Deep Models and Transfer Learning. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*. ACM, 2413–2424. https://doi.org/10.1145/3308558.3313578

[80] Wayne Xin Zhao, Jing Liu, Ruiyang Ren, and Ji-Rong Wen. 2022. Dense Text Retrieval based on Pretrained Language Models: A Survey. *CoRR* abs/2211.14876 (2022). https://doi.org/10.48550/arXiv.2211.14876 arXiv:2211.14876