BEACON: Budget-Aware Entity Matching Across Domains

NICHOLAS PULSONE, Worcester Polytechnic Institute, USA
ROEE SHRAGA, Worcester Polytechnic Institute, USA
GREGORY GOREN, eBay Research, Israel

Entity Matching (EM)–the task of determining whether two data records refer to the same real-world entity–is a core task in data integration. Recent advances in deep learning have set a new standard for EM, particularly through fine-tuning Pretrained Language Models (PLMs) and, more recently, Large Language Models (LLMs). However, fine-tuning typically requires large amounts of labeled data, which are expensive and time-consuming to obtain. In the context of e-commerce matching, labeling scarcity varies widely across domains, raising the question of how to intelligently train accurate domain-specific EM models with limited labeled data. In this work we assume users have only a limited amount of labels for a specific target domain but have access to labeled data from other domains. We introduce BEACON, a distribution-aware, budget-aware framework for low-resource EM across domains. BEACON leverages the insight that embedding representations of pairwise candidate matches can guide the effective selection of out-of-domain samples under limited in-domain supervision. We conduct extensive experiments across multiple domain-partitioned datasets derived from established EM benchmarks, demonstrating that BEACON consistently outperforms state-of-the-art methods under different training budgets.

CCS Concepts: • **Computing methodologies** → **Machine learning**; • **Information systems** → **Entity resolution**.

Additional Key Words and Phrases: entity matching, pretrained language models, machine learning

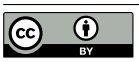
ACM Reference Format:
Nicholas Pulsone, Roee Shraga, and Gregory Goren. 2026. BEACON: Budget-Aware Entity Matching Across Domains. *Proc. ACM Manag. Data* 4, 3 (SIGMOD), Article 144 (June 2026), 26 pages. https://doi.org/10.1145/3802021

1 Introduction

Entity Matching (EM)–also known as Entity Resolution or Record Linkage–is a fundamental task in the data integration pipeline. At its core, EM involves determining whether two data records refer to the same real-world entity. It is widely applied in scenarios such as de-duplication, where the goal is to group samples in a single dataset into clusters of distinct entities. EM is a longstanding problem in the data management and data science communities and has been studied extensively over the past several decades [4, 5, 7, 8, 17, 20].

Historically, EM was addressed using string similarity metrics and probabilistic models [17, 57], followed by the development of rule-based systems [20, 30]. With the advent of learning methods, more flexible and adaptive models emerged [7, 8], ultimately leading to the adoption of deep learning methods based on neural networks [16, 31]. While recent studies have employed Large Language Models (LLMs) for EM [38], the fine-tuning of Pretrained Language Models (PLMs)–such as BERT and RoBERTa–on task-specific EM datasets [10, 36] remains a common paradigm [4]. This

Authors' Contact Information: Nicholas Pulsone, Worcester Polytechnic Institute, Worcester, Massachusetts, USA, nbpulsone@wpi.edu; Roee Shraga, Worcester Polytechnic Institute, Worcester, Massachusetts, USA; Gregory Goren, eBay Research, Netanya, Israel.



ACM 2836-6573/2026/6-ART144
https://doi.org/10.1145/3802021

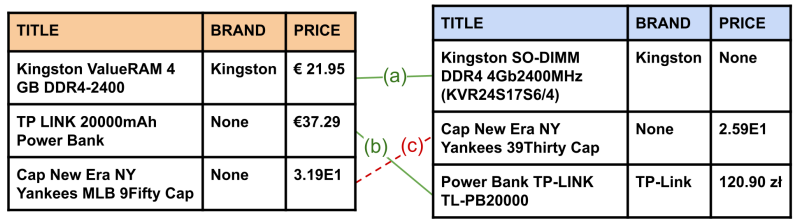


Fig. 1. Performing EM between two e-commerce datasets. Each sample compares two product records and is derived from the Web Data Commons Multi-Dimensional Entity Matching Benchmark (WDC) [37].

is largely because LLMs impose significant computational overhead during inference [62]–even in zero-shot settings–and incur high monetary costs when accessed via hosted APIs [38]. Among PLM-based approaches, DITTO [27] represents the state-of-the-art (SOTA) EM framework. This work builds on the DITTO pipeline and focuses on optimizing EM in settings with limited training data.

Many real-world EM applications involve heterogeneous data spanning multiple *domains* [18, 39]. For example, the WDC benchmark [37] partitions data by product category, and other standard EM datasets–such as Abt-Buy and Amazon–Google Products–exhibit similar domain structure [25]. Consequently, it is often intuitive to train and evaluate separate models for each domain to optimize performance [46], to-be-defined in Section 3.3. This setup enables a more informative evaluation protocol than traditional EM, as it reveals how model performance varies across domains.

Processing diverse domains often coincides with substantial variation in data characteristics, including the availability of labeled examples. As machine learning methods, particularly deep learning-based language models, have become increasingly prevalent in EM, there is a growing reliance on large labeled datasets. However, curating such datasets at scale is expensive and labor-intensive [19, 47]. To capture this constraint, we consider an *annotation budget* [19, 24, 44], which we define as a hard limit on the number of labeled samples available for training an EM model (e.g., 5,000 samples). This budget reflects the finite labeling resources available to a practitioner.

In realistic deployment settings, practitioners often possess labeled data for a single domain of interest, alongside abundant unlabeled data from related domains [45]. As a result, the primary bottleneck becomes the cost of acquiring additional labels across domains. In e-commerce for example, organizations may have extensive historical or cross-category product data but have annotated only the specific domain they seek to improve. Another common scenario is that an organization already has labeled data spanning multiple categories, but must support a new domain with limited or no labeled data and only a small additional annotation budget. In such cases, a fundamental question arises: how can we intelligently select which samples to label in order to train effective EM models under a fixed annotation budget? This paper aims to address this question by *exploring how EM can be performed effectively across domains under a fixed annotation budget, and introducing* ***BEACON****—a novel budget-aware framework designed to address this challenge.*

To illustrate the motivation for our setting and approach, we now turn to a concrete example.

Example 1.1. A common EM process involves forming pairs of entities that could potentially refer to the same real-world object. This is typically done using a procedure known as *blocking* [35, 49, 51]. Blocking considers all possible combinations of records and eliminates unlikely matches early in the pipeline using computationally inexpensive rules and heuristics. The result is a reduced search

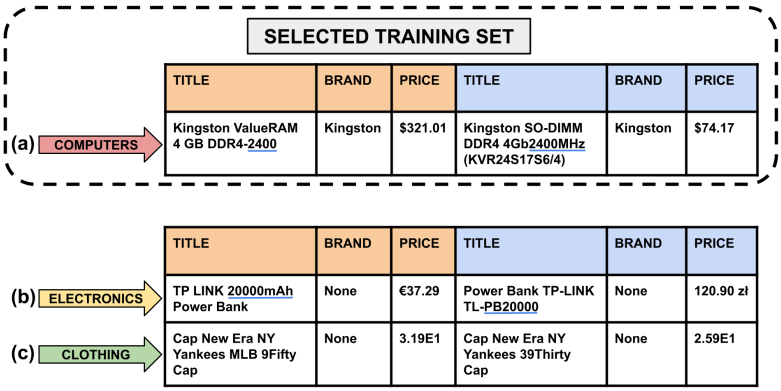


Fig. 2. Entity matching samples from Figure 1 grouped by the product category domain. The goal is to select a pairwise sample to complement the selected training set for EM in the "Computers" product category.

space that enables a more fine-grained matching procedure. For this example, we assume the output of blocking is already given.

Figure 1 shows three entity pairs produced by blocking, with each pair comparing two product records from e-commerce datasets. The samples span multiple product categories and illustrate the types of candidate pairs that serve as inputs to EM models. A solid green line between records indicates a match, while a red dotted line indicates a non-match. Samples (a) and (c) represent matches, where both entities in the pair refer to the same product (though this is not known during matching). For example, sample (a) compares the same RAM stick, and sample (c) compares the same power supply. Conversely, sample (b) is a non-match: although both products are baseball caps from the same brand, they are distinct items. The record data is incomplete–as can be seen with missing "brand" values–highlighting the challenges of real-world data systems and EM in practice.

Building on the previous example, we now examine how EM can be approached *across domains*–at the level of individual product categories. Such category-level partitioning is common in e-commerce EM benchmarks [37, 39], which explicitly organize data by product type to enable domain-specific evaluation. This approach requires a careful selection of samples to train an effective EM model, especially under a limited training budget.

Example 1.2. Suppose we aim to train a model for the "Computers" category. Figure 2 shows the same three pairwise samples from Figure 1, now labeled by product category. Note that any pairs spanning different product categories are automatically removed during blocking, allowing the remaining data to be partitioned into distinct category-specific groups. To build a training set for "Computers," we first include all in-domain pairs, as they are most likely be relevant to our model. In our toy example, only pair (a) belongs to the domain and is selected first. If room remains in the budget after including all in-domain samples, we need to decide which out-of-domain examples are most "beneficial."

Sample (b), from "Electronics," might be a natural candidate. Intuitively, "Electronics" is ontologically closer to "Computers" than "Clothing" is, meaning that, in an abstract product taxonomy or to a human judge, the computers and electronics categories have the same semantic neighborhood, or at least more so than clothing does. More importantly however, both domains exhibit similar

structures, such as numerical specifications that may vary in format or include/exclude units. Sample (c), from "Clothing," may also be useful but is clearly not as relevant. Given a tight training budget, we would prioritize the "Electronics" sample over the "Clothing" sample.

It should be noted that simply including samples from categories that are ontologically closest to "Computers" often underperforms in practice, as our offline experiments have shown. This example illustrates the intuition behind domain-aware sampling, yet high-level domain knowledge alone is insufficient to quantify the informativeness of out-of-domain data. This motivates the use of embedding-based and language-model–driven techniques discussed further in Section 4.2.

Example 1.2 highlights the central question addressed in this work: how to select the *best* out-of-domain training samples under a budget constraint in order to optimize EM model performance for each domain. More generally, the structural similarities we intuitively observe between domains–such as between computers and electronics–can be captured by comparing the distributions of their data in a shared high-dimensional space. Distribution-aware methods make it possible to formalize these similarities, guiding the selection of out-of-domain samples that most effectively complement the in-domain data.

Our work makes the following contributions:

- *Problem Formulation.* We formalize *Budget-Aware Entity Matching Across Domains* (EMAD), extending EM to account for specialized per-domain models and a training annotation budget that reflects realistic labeling constraints.
- *Distribution-aware methods.* We propose novel sampling strategies that leverage embedding-level representations of pairwise EM data to guide out-of-domain selection and improve performance across domains.
- *The BEACON framework.* We introduce BEACON, a novel budget-aware ensemble framework that employs a dynamic training loop and a dual-PLM architecture, periodically regenerating embeddings and resampling training data to optimize model performance.
- *Experimental Evaluation.*[1] We adapt EM PLMs (e.g., DITTO [27]), recent EM domain adaptation frameworks (e.g., MFSN [50]), recent low-resource EM methods (e.g., Battleship [19] and PromptEM [55]), and recent LLM approaches (e.g., Jellyfish [60, 61]) to establish strong baselines for EMAD, over which we conduct extensive experiments showing that BEACON outperforms approaches across different budget constraints. For this evaluation, we derive nine domain-partitioned datasets from WDC [37], as well as two additional datasets derived from the WDC Product Corpus [40] and Abt-Buy [25], which we also release to support future research on EMAD. We further provide *Open-source Code and Data.*[2]

The remainder of the paper is organized as follows. Section 2 reviews related work and positions our study within the broader context of existing research. Section 3 introduces our new problem formulation—*Budgeted Entity Matching Across Domains* (EMAD)–along with key preliminaries. In Section 4, we describe methods for solving EMAD, including both basic and distribution-aware approaches. Section 4 also presents the BEACON framework, which optimizes EM performance under the proposed setting. We present our experimental setup, including the datasets and baselines used in Section 5. We report experimental results and analyses in Section 6. Finally, Section 7 concludes the paper and outlines directions for future work.

[1] An extended technical report containing additional experimental results is available on arXiv [41].
[2] https://github.com/nbpulsone/BEACON

2 Related Work

Our work builds on recent advances in learning-based EM, particularly those leveraging deep learning and PLMs. Both the blocking and matching phases of EM have received attention in the deep learning literature. For example, several recent works [9, 22, 51] have explored deep learning approaches for the blocking phase. Since our focus is on the matching phase, we assume that our models operate on the output of any effective blocking method–whether traditional or deep learning-based.

Deep learning for matching was first introduced by DeepER [16] and later extended to DeepMatcher [31]. These works framed EM as a pairwise classification task using deep neural networks and achieved strong performance. More recently, PLM-based matching has become the new standard, with DITTO [27] demonstrating that fine-tuning PLMs such as BERT can significantly improve accuracy. Rastaghi et al. [1] adds robustness-focused optimizations to the DITTO pipeline and Adapterem [32] enhances computational efficiency. Our work also builds on the DITTO architecture, but operates under a different setting: EM under limited data availability and across multiple domains. While this problem has received comparatively little attention, recent works have explored related challenges in cross-domain and low-resource EM.

Cross-Domain and Domain-Aware EM. Recent work has explored domain-aware EM approaches that explicitly account for domain information during model training, though these settings differ from BEACON. For example, Unicorn [54] proposes a unified mixture-of-experts architecture that supports multiple data-matching tasks by learning shared representations across domains. While Unicorn emphasizes cross-task generalization and zero-shot prediction, BEACON instead focuses on *budget-aware selection of cross-domain training data* for a single EM task. HierGAT [58] models domain-specific attribute, token, and entity interactions using hierarchical graph attention mechanisms. Whereas HierGAT learns explicit domain-conditioned relational structure, BEACON operates in a PLM embedding space and leverages distribution-aware sampling to allocate a fixed annotation budget. Related survey work (e.g., Paganelli et al. [33, 34]) further analyzes how PLM representations adapt to EM-specific patterns, providing valuable architectural insights into cross-domain EM that complements BEACON's domain-aware data selection.

A related line of work focuses on domain adaptation for EM, where labeled data from a source domain is used to improve performance on a different target domain, often with limited or no labeled examples. The idea was first introduced by DAME [52], which employs a mixture-of-experts approach to capture task-specific information from source domains and transfer it to a separate target domain. DADER [53] similarly explores domain adaptation for EM, but uses feature alignment strategies to make the source domain more informative for the target. Most recently, MFSN [50] introduced a framework that explicitly models private and common matching features across domains using multiple encoders, enabling more effective knowledge transfer. We use the enhanced variant, MFSN-FRSE, as a baseline in our work.

While our work shares the high-level goal of improving EM under domain shifts, there are two important distinctions that highlight the novelty of our problem setting. First, we do not assume access to a separate source-domain dataset; instead, we operate within a single large dataset that contains multiple related domains of interest. These domains can be viewed as *micro-domains*–related subdomains (e.g., "Computers" and "Clothing") within a larger *macro-domain* (e.g., "e-commerce Products"). In contrast, existing work on cross-domain EM [50, 52, 53] primarily focuses on transferring knowledge across distinct macro-domains. Second, we assume access to a limited amount of in-domain labeled data and study how to augment it under a constrained sample budget. Rather than transferring representations or model parameters between datasets, our formulation emphasizes *selecting samples across domains within the same dataset.*

Table 1. Summary of baseline methods and experimental configurations. ✓ and ✗ indicate whether a method uses labeled in-domain data, incorporates out-of-domain samples, or requires model training in our experimental setting.

Method	Uses In-Domain Labels	Uses Out-of-Domain Data	Requires Training	Selection Strategy
GEN	✓	✓	✓	Random sampling across all domains
SPEC	✓	✗	✓	Random sampling within target domain
MFSN [50]	✓	✓	✓	Feature-space domain transfer
Battleship [19]	✓	✓	✓	Graph-based uncertainty and centrality
PromptEM [55]	✓	✗	✓	Prompt tuning with pseudo-label generation
LLM-based (LLaMA [15], Jellyfish [60])	✗	✗	✗	N/A (Zero-shot inference)
BEACON (ours)	✓	✓	✓	Distribution-aware dynamic resampling

Low-Resource and Budget-Constrained EM. Some works have devised methods to perform EM in settings with limited labeled data. PromptEM [55] applies prompt tuning and pseudo-labeling to improve performance for low-resource EM. Other works have applied active learning techniques to EM under limited supervision [6, 21, 24, 42, 44]. The idea was first introduced by Sarawagi and Bhamidipaty [44], who demonstrated that a matcher can be effectively trained using an intelligently selected subset of training data. Building on this foundation, Qian et al. [42] proposed an active learning framework that learns a diverse set of rules to improve large-scale matching performance. More recent approaches, such as those of Kasai et al. [24] and Huang et al. [21], guide the training of deep learning–based EM models by selecting informative examples. Other works [11, 19] explicitly explore fine-tuning PLMs under fixed labeling budgets, but this remains an under-explored area despite its practical relevance. Among these approaches, we adapt a SOTA active learning method for EM, Battleship [19], and low-resource EM method, PromptEM [55], to the EMAD setting and use them as baselines in our experimental evaluation.

Our work contributes to this line of research by introducing novel sampling strategies designed to maximize performance under strict labeling constraints. However, our approach is not a direct application of active learning or low-resource EM techniques. Instead, we operate in a distinct problem setting that explicitly considers cross-domain structure: given limited supervision in a target domain, we study how to select additional samples to label from *other* domains (and, symmetrically, how out-of-domain supervision can complement scarce in-domain labels) to improve target-domain performance. In contrast, active learning methods typically select the most informative unlabeled samples *within* a single domain for annotation, largely ignoring domain boundaries altogether.

It is important to recognize the strong connection between cross-domain and low-resource EM–the intersection in which our work resides. In practice, low-resource settings can exploit domain distinctions by selecting samples that best align with existing data distributions, focusing labeling on the most informative samples. Conversely, domain variation often creates low-resource challenges–some domains have ample labeled data while others do not–motivating approaches that transfer knowledge across domains. This natural interplay between limited supervision and domain variation underscores the motivation for our formulation of the EMAD problem, discussed further in Section 3.3.

LLM and Prompt-based EM. Recent work [38, 56] has begun to explore the use of LLMs and prompt-based techniques for EM. A representative example is ComEM [56], which ensembles multiple LLMs and prompting strategies to generate globally consistent EM predictions. These approaches primarily focus on adapting large pretrained models to the EM task, whereas BEACON provides a data selection strategy that can be paired with any trained PLM or LLM-based classifier, including prompt-based systems. In this work, however, we focus on low-resource, budget-aware EM, and leave the integration of BEACON with LLM-based EM approaches for future work.

Table 1 situates our methods within the broader EM literature and clarifies how they differ from the baseline approaches in terms of supervision, domain usage, and training requirements.

3 Preliminaries and Problem Definition

We now introduce the notation used in the paper (Section 3.1), the standard EM task (Section 3.2), and the extended formulation of Budget-Aware Entity Matching Across Domains (Section 3.3).

3.1 Notations and Blocking

We follow Li et al. [27] and others in defining EM as the task of matching two distinct datasets. Other formulations also exist, where matching is performed within a single dataset [44] (i.e., de-duplication). Let D_{left} and D_{right} denote the two datasets to be matched, with $|D_{\text{left}}| = l$ and $|D_{\text{right}}| = m$. Each dataset is tabular, consisting of *entities* (or records), where an entity r can be represented as a row of structured information in a table (dataset). In this work, we focus on the *clean-clean* EM setting, which assumes that neither D_{left} nor D_{right} contains duplicates internally. In what follows, the set of all possible pairs of entities is given by the Cartesian product $D_{\text{left}} \times D_{\text{right}}$. We refer to these pairs as *candidates*, and the complete set of such pairs as the *candidate set*.

The EM process typically proceeds in two stages: a blocking phase followed by a matching phase. In the blocking phase, a set of lightweight heuristics is applied to eliminate unlikely matches in the *candidate set* and reduce the search space. This yields a subset of candidates $N \subset D_{\text{left}} \times D_{\text{right}}$ such that $|N| \approx \max(O(l), O(m))$, which avoids the $O(l \times m)$ complexity of exhaustive pairwise comparison. While blocking is essential to scalability, this paper focuses on improving the downstream matching phase.

We assume that the set N can be partitioned into j disjoint subsets–each corresponding to a different domain–such that:

$$N = n_1 \cup n_2 \cup \cdots \cup n_j, \quad n_i \cap n_k = \emptyset \text{ for } i \neq k$$

In this work, we focus on product data, where the domain corresponds to product categories (e.g., "Computers", "Clothing", etc.) or brands (e.g., "Apple", "Samsung", etc.).

3.2 Entity Matching

Given a filtered candidate set N, the goal of the matching phase is to determine which candidates contain matching entities. This is formulated as a binary classification task: for each candidate $(r_1, r_2)_i \in N$, we assign a label $y_i \in \{0, 1\}$, where $y_i = 1$ if r_1 and r_2 are matching entities, and 0 otherwise. We refer to a candidate together with its corresponding label as a *sample*.

We adopt PLMs such as BERT [12] (or RoBERTa [28] and DistilBERT [43]) to perform the classification. A PLM refers to a transformer-based model trained on a large corpus of text, which can be fine-tuned for downstream tasks such as EM. Following the DITTO framework [27], we serialize an entity r with p attributes as:

$$\texttt{[COL]}\ \texttt{attr}_1\ \texttt{[VAL]}\ \texttt{val}_1 \cdots\ \texttt{[COL]}\ \texttt{attr}_p\ \texttt{[VAL]}\ \texttt{val}_p$$

and each candidate as:

$$\texttt{[CLS]}\ r_1\ \texttt{[SEP]}\ r_2$$

This serialized string is passed through the PLM. The `[SEP]` and `[CLS]` tokens are special tokens included in the input. The `[SEP]` token indicates the boundary between the left and right entities in the serialized string. The output embedding corresponding to the `[CLS]` token is treated as a summary vector for the candidate. Importantly, the label indicating whether a pair is a match or non-match is not included in the serialized string used to form the embedding representation. This

vector is then passed through a fully connected layer, followed by a softmax or sigmoid activation, to produce the match prediction $\hat{y}_i$.

3.3 Problem Definition

We now extend the formulation to the *Budget-Aware Entity Matching Across Domains* (EMAD) setting. We define the set of domains to be $\mathcal{D}$, where $|\mathcal{D}| = j$. Instead of training a single matcher on N for all domains, we train j domain-specific matchers using the sets $X_1, X_2, \ldots, X_j$, where for $i \in \mathcal{D}$, each X_i is a training set for domain i. The naïve way to address this is use only domain-specific data for each training set, where $X_i = n_i, \forall i$ (see Section 4.1.2).

While it may also be attractive to train a single matcher for all domains (i.e., $X_i = N, \forall i$), there are strong motivations for using domain-specific matchers. We illustrate these motivations with the following example:

Example 3.1. Consider the task of matching entities across two product domains: "Shoes" and "Jewelry." For shoes, EM often depends on a small set of attributes with highly standardized values. For instance, two Nike Air Zoom Pegasus 40 running shoes (US size 10, Style Code `DV3853-401`, retail price $130) can be matched with high confidence if their *Brand*, *Size*, and *Style Code* attributes align across both records.[3]

Conversely, matching jewelry items such as gold rings requires reasoning over more descriptive and less standardized attributes–e.g., *material composition*, *metal*, *gemstone type*, and *carat weight*– where subtle differences in any one of these attributes may alone determine whether two rings refer to distinct products.[4]

Moreover, the interdependence between attributes differs across domains: in footwear, the *Style Code* is meaningful only in conjunction with the *Brand*, whereas in jewelry, *Brand* and *Size* alone do not provide much matching insight without additional context.

Example 3.1 highlights that a single matcher is unlikely to capture the diverse decision patterns observed across domains–a phenomenon also noted in other machine learning tasks such as recommendation systems [23, 48]. By contrast, a matcher trained on a specific domain can effectively capture domain-specific nuances, even when out-of-domain candidates are included in the training set. Prior work in domain adaptation [53] has demonstrated that models trained on domain-relevant data achieve improved performance within their target domain. Unlike these approaches, however, our formulation builds on this insight by explicitly studying how to *borrow out-of-domain signal under a fixed annotation budget* to complement limited in-domain data, with the goal of training specialized domain-specific models rather than assuming that a single unified model will suffice.

We assume that each domain is defined such that both entities in any candidate belong to the same domain (e.g., both records are from the "Computers" category). For each domain i, we are given a *training budget* β that constrains the total number of candidates in X_i. These candidates may be drawn from both in-domain data (n_i) and out-of-domain data ($N \setminus n_i$).

Definition 3.2 (Budget-Aware Entity Matching Across Domains (EMAD)). Given a domain partition of $N = n_1 \cup n_2 \cup \cdots \cup n_j$ over the domains $\mathcal{D}$ of the candidate set and a training budget β, the goal of EMAD is, for each domain $i \in \mathcal{D}$, to construct a budgeted training set

$$X_i = n_i^* \cup S_i^{\text{out}}, \quad \text{where } |X_i| = \beta, \quad S_i^{\text{out}} \subseteq (N \setminus n_i),$$

such that the resulting model achieves high matching performance on domain i.

[3]Please find the full item description [3]
[4]Please find the full item description [2]

Here, n_i^* denotes a subsampled or oversampled version of n_i, and S_i^{out} is a set of out-of-domain samples selected according to a given sampling strategy. This formulation allows us to investigate how different sampling techniques affect domain-specific model performance under realistic budget constraints.

4 Budget-Aware Entity Matching Across Domains

We now introduce a suite of methods to address the EMAD problem, along with complementary techniques to improve PLM-based EM under budget constraints.

Section 4.1 presents foundational baseline models, each motivated by simple but effective heuristics for sample selection. Section 4.2 expands on this by introducing models that perform distribution-aware sample selection, using embedding-based techniques to align out-of-domain data with in-domain distributions. Finally, Section 4.3 introduces BEACON–our central contribution–an ensemble-based framework that incorporates a dynamic resampling procedure to maximize EM performance across domains.

4.1 Basic Models

We begin by introducing basic models for EMAD. Although simple, these models provide strong baselines as they are grounded in natural intuitions that also serve as the foundation for our more advanced models. Throughout this section and Section 4.2, we employ a common operator, $\texttt{RandomSelect}(A, b)$, which randomly selects b items from a set A, where b is an integer such that $b \leq |A|$. This operator defines the random sampling mechanism that is used in several of our formulations.

4.1.1 The General Model (GEN). The General Model (GEN) generates X_i by randomly sampling from the candidate set, without regard to domain. For example, if we are training a model to match WDC "Computers" pairs, we sample across all available domains (e.g., "Electronics," "Clothing," etc.) to fill the budget, including candidates from the target domain itself (see Example 1.1). GEN corresponds to the idea of training a single matcher across all domains, as discussed in Section 3.3. By randomly sampling across all domains, GEN simulates a unified model trained without regard to domain boundaries, while still respecting the budget constraint. GEN is motivated by the hypothesis that broad domain coverage may expose the model to a more diverse range of attribute patterns, potentially improving its generalization to unseen data. In practice, GEN represents the most domain-agnostic baseline.

Formally, for $1 \leq i \leq j$, we have

$$X_i = \texttt{RandomSelect}(N, \beta) \tag{1}$$

Given a budget β, GEN draws β samples from the candidate set at random to construct a training set for each domain. Thus, the training set for each domain becomes a random mixture of in-domain (n_i) and out-of-domain (S_i^{out}) candidate pairs.

4.1.2 The Domain-Specific Model (SPEC). The Domain-Specific Model (SPEC) generates X_i exclusively using samples from the target domain. Given the differences in the EM process across domains, we consider the effect of training solely on domain-specific data–forcing the model to focus on the unique characteristics of matching within a given domain. Not only do domains differ in their matching logic and attribute structure, but they can also vary in the availability of training data. As shown in Table 2, certain domains can contain orders of magnitude more samples than others.

For SPEC, if a domain contains fewer than β samples, we oversample (with replacement) from the available data. As depicted in Figure 3, oversampling–especially for smaller domains–leads to steady

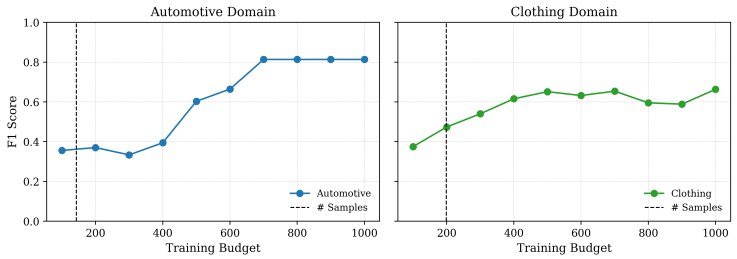


Fig. 3. **Effect of Oversampling on PLM-based Entity Matching.** We fine-tune RoBERTa on two small product domains–*Automotive* and *Clothing*–from the WDC dataset using progressively larger training budgets that permit oversampling.

performance improvements that tend to plateau. Conversely, with smaller budgets, the domain may contain more than β samples. While state-of-the-art active learning methods [19, 21, 24] have shown that fine-tuning on a selected subset of the available training data can improve matching performance, for SPEC and the models that follow, we instead opt to randomly down-sample to keep the focus of this paper on out-of-domain sampling.

Formally, for $1 \leq i \leq j$, we define:

$$X_i = \begin{cases} n_i^{\lfloor \frac{\beta}{|n_i|} \rfloor} \cup \texttt{RandomSelect}(n_i, \beta \bmod |n_i|), & \text{if } |n_i| \leq \beta, \\ \texttt{RandomSelect}(n_i, \beta), & \text{otherwise.} \end{cases} \tag{2}$$

Given a budget β, SPEC randomly draws β samples from n_i (with replacement if necessary) to construct a training set composed entirely of samples from the relevant domain. Intuitively, Equation (2) formalizes the idea that the SPEC model is restricted to in-domain information, serving as the baseline for performance when no cross-domain data is leveraged.

4.2 Distribution-Aware Models

We now introduce models that approach EMAD using distribution-aware techniques. Modern language models enable serialized candidate pairs to be represented as high-dimensional vectors, commonly referred to as *embeddings* [59] (see Section 3.2). These embeddings capture semantic information about the textual content of the records; however, they do not encode label information, ensuring that the selection process is driven by unsupervised characteristics of the data. The closer two embeddings are in this space, the more semantically similar their corresponding candidate pairs.

Because domains differ in vocabulary, attribute composition, and matching logic, their embeddings often form clustered regions. While these regions may overlap–particularly for semantically related domains–the overall structure tends to reflect meaningful domain-level distinctions. We refer to the shape and spread of these clustered regions as the *distribution* of a domain's embeddings–a mathematical characterization of how its candidate pairs are arranged, reflecting attribute correlations, dependencies, and domain-specific terminology. By leveraging these distributions, we can compare domains quantitatively and reason about their similarity. This, in turn, enables *distribution-aware* sampling strategies that selectively incorporate out-of-domain samples whose embeddings align with the in-domain distribution, forming the basis of the novel sampling methods we introduce in this work.

Given a domain $i \in \mathcal{D}$, the goal is to select out-of-domain samples from $N \setminus n_i$ so as to align the distribution of the in-domain samples n_i with a target distribution. Because the distributions

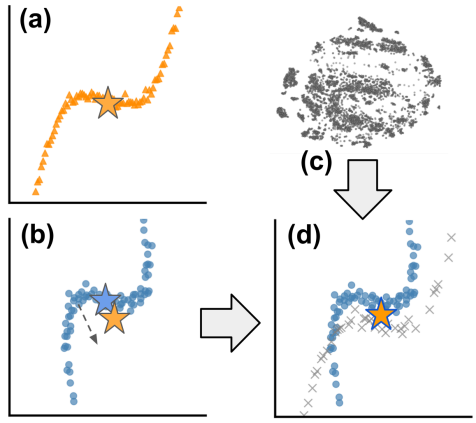


Fig. 4. **Illustration of the Train–Validation Distribution Fitting (TVDF) sampling procedure.** (a) Validation set embeddings with centroid μ_{val} (orange star). (b) In-domain training embeddings for domain $i \in \mathcal{D}$ with centroid μ_i (blue star) and overlaid μ_{val}. (c) Out-of-domain samples are ranked by the TVDF selector based on their contribution to aligning μ_i with μ_{val}. (d) Selected samples are added to the training set, resulting in closer alignment between the training and validation centroids.

vary across domains, the appropriate target distribution may differ from one domain to another. The following models make different assumptions about the target distribution, which guide the selection of out-of-domain samples.

4.2.1 The Nearest Neighbors Model (NN). The Nearest Neighbors Model (NN) first includes all in-domain samples from n_i in X_i, similar to SPEC. In the majority of cases where the budget allows for more samples (e.g., $n_i < \beta$), the model then selects additional samples from other domains whose embeddings are closest to the domain-specific centroid under a distance metric (e.g. cosine distance). Intuitively, these samples are those whose inclusion alters the domain-specific centroid the least. NN assumes that the domain-specific embeddings already provide a good representation of the domain's distribution.

Formally, for $1 \leq i \leq j$, we define:

$$X_i = \begin{cases} \texttt{RandomSelect}(n_i, \beta), & \text{if } |n_i| \geq \beta, \\ n_i \cup \text{NN}_{\beta - |n_i|}(N \setminus n_i, \mu_i), & \text{otherwise,} \end{cases} \tag{3}$$

where μ_i is the centroid embedding of n_i, and $\text{NN}_k(A, \mu_i)$ denotes the set of k samples in A whose embeddings are nearest to μ_i.

Given a set of embeddings A, we define the NN selector as:

$$\text{NN}_k(A, \mu_i) = \underset{x \in A}{\text{Top-k}} \left(\cos(\mu_i, x) \right), \tag{4}$$

which represents the k candidate pairs from A with the highest cosine similarity to μ_i, i.e., the k elements $x \in A$ that maximize $\cos(\mu_i, x)$ in descending order. The Top-k operator returns the k highest-scoring elements, where ties, if any, are broken arbitrarily.

4.2.2 The Train-Validation Distribution Fitting Model (TVDF). The Train-Validation Distribution Fitting Model (TVDF) shares a similar setup with NN. It first includes all available domain-specific samples from the target domain in X_i. When $n_i < \beta$, it selects additional samples from other

domains whose embeddings bring the training distribution closer to that of the target domain's validation set.

More concretely, we compute the centroid of the target domain's training embeddings and the centroid of its validation embeddings. Additional samples are drawn from other domains such that, when added to the training set, they minimize the distance between the updated training centroid and the validation centroid. In this view, TVDF assumes the current in-domain labeled data are insufficient to represent the target domain's embedding distribution, and therefore leverages the validation set as a proxy for the (unseen) test distribution.

Formally, for each domain $i \in \mathcal{D}$, we define the centroids of the training and validation embeddings as:

$$\mu_i = \frac{1}{|n_i|} \sum_{x \in n_i} x, \qquad \mu_{\text{val}_i} = \frac{1}{|v_i|} \sum_{x \in v_i} x, \tag{5}$$

where v_i denotes the validation embeddings for domain i.

Given a total training budget β, the budgeted training set X_i is constructed as:

$$X_i = \begin{cases} \texttt{RandomSelect}(n_i, \beta), & \text{if } |n_i| \geq \beta, \\ n_i \cup \text{TVDF}_{\beta - |n_i|}(N \setminus n_i, \mu_i, \mu_{\text{val}_i}), & \text{otherwise.} \end{cases} \tag{6}$$

For each candidate pair $x \in N \setminus n_i$, let the updated centroid after adding x be:

$$\mu_{n_i \cup x} = \frac{|n_i| \cdot \mu_i + x}{|n_i| + 1}. \tag{7}$$

Given a set of embeddings A, we then define the TVDF selector as:

$$\text{TVDF}_k(A, \mu_i, \mu_{\text{val}_i}) = \underset{x \in A}{\text{Top-k}} \left[\cos(\mu_{n_i \cup x}, \mu_{\text{val}_i}) - \cos(\mu_i, \mu_{\text{val}_i}) \right], \tag{8}$$

which returns the k candidate pairs from A that most increase the cosine similarity between the updated training centroid and the validation centroid. Figure 4 illustrates the TVDF selector in a toy example; for simplicity, candidates are shown in a two-dimensional embedding space with the training and validation centroids overlaid.

4.2.3 K-Center Greedy Sampling (KCG). The K-Center Greedy (KCG) model treats the selection of out-of-domain samples as a k-center optimization problem. Here, the embeddings of the in-domain samples n_i serve as the initial centers, and the goal is to select samples from $N \setminus n_i$ such that the maximum distance from any sample to its nearest center is minimized. This corresponds to a farthest-first traversal strategy that incrementally adds the most distant points to maximize coverage in the embedding space.

KCG assumes that the in-domain data already provides a strong local representation of the domain. Its objective is to improve generalization by incorporating diverse out-of-domain samples that fill gaps in this representation.

Formally, for each domain $i \in \mathcal{D}$, we define the training set as:

$$X_i = \begin{cases} \texttt{RandomSelect}(n_i, \beta), & \text{if } |n_i| \geq \beta, \\ n_i \cup \text{KCG}_{\beta - |n_i|}(n_i, N \setminus n_i), & \text{otherwise.} \end{cases} \tag{9}$$

For each candidate $x \in N \setminus n_i$, its distance to the nearest in-domain center c is defined as:

$$d(x, n_i) = \min_{c \in n_i} \big(1 - \cos(x, c)\big). \tag{10}$$

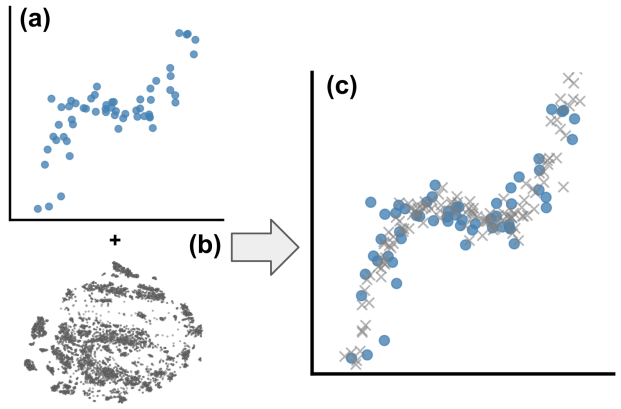


Fig. 5. **Illustration of the K-Center Greedy (KCG) sampling procedure.** (a) In-domain embeddings n_i serve as the initial centers. (b) Out-of-domain samples are ranked by the KCG selector based on their distance to the nearest in-domain center. (c) Selected samples are added to the training set to fill gaps in the embedding space, increasing overall coverage.

The KCG selector then chooses the $k = \beta - |n_i|$ candidates that maximize this distance:

$$\mathrm{KCG}_k(n_i, A) = \underset{x \in A}{\text{Top-k}}\ d(x, n_i), \tag{11}$$

This is solved greedily by repeatedly adding the farthest embedding in A and updating the distance to the nearest center, yielding a 2-approximation to the optimal k-center solution under cosine distance. Figure 5 illustrates the KCG selection process in a simplified 2-dimensional embedding space.

4.3 BEACON

We present **BEACON** (**B**udget-aware **E**ntity matching **ACr**O**ss** domai**N**s), our proposed framework for optimizing EMAD under budget constraints. Figure 6 presents an overview of the BEACON framework, highlighting its three major components: (1) an *Ensemble of Models* (Section 4.3.1), (2-4) a *Dynamic Training Loop* (Section 4.3.2), and (3–4) a *Resampling Mechanism* (Section 4.3.3). Notably, BEACON's dynamic training loop and distribution-aware resampling mechanism constitute novel components of the framework which have not been explored in prior cross-domain EM research. Together, these components enable iterative refinement of the training data and progressively improve EM performance across domains.

4.3.1 The Ensemble of Models. Following prior work on ensemble-based EM systems [14, 29], we propose an ensemble model to improve classification performance under the EMAD setting. Ensemble methods are widely recognized for enhancing predictive robustness, with even simple aggregation strategies such as soft voting proving effective across diverse classification tasks [13, 26].

Once several models are trained, their predictions can be aggregated to form a more accurate consensus. Given k trained models, let $q \in [1, k]$ index a model. We assume access to a validation set v_i for each domain $i \in \mathcal{D}$ and measure the validation F1 performance of each model as $F1_{\mathrm{VAL},q}$. We adopt a *weighted soft voting* scheme, in which the confidence outputs $Conf_q$ from each model are combined using weights proportional to their validation F1 scores. This provides a principled way to emphasize models that perform best during validation. Formally, the final ensemble confidence $Conf_f$ for a given candidate pair is computed as:

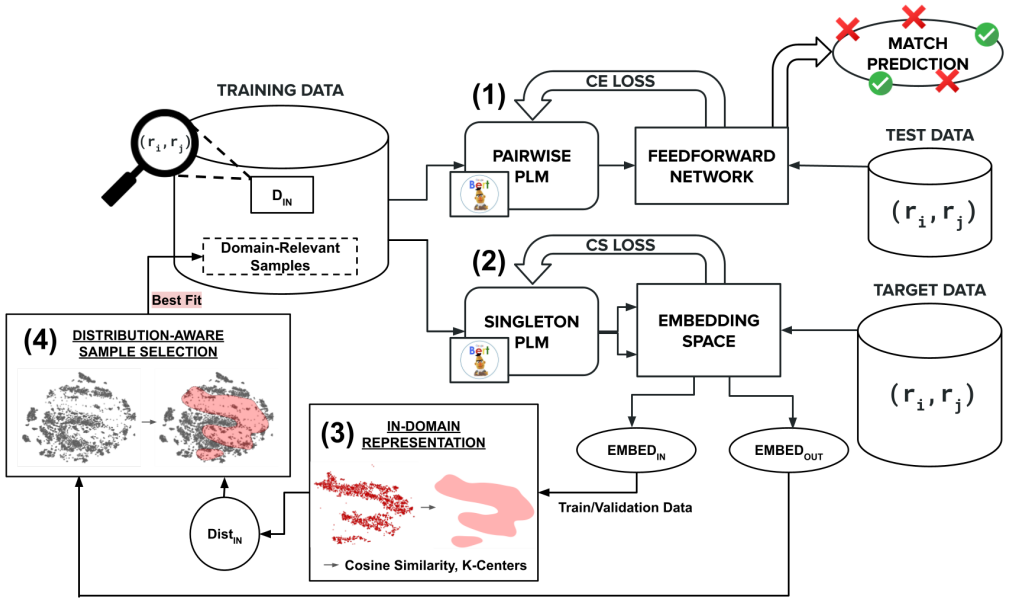


Fig. 6. **An Illustration of the BEACON framework.** (1) A pairwise PLM is fine-tuned on budgeted in- and out-of-domain candidates for EM. (2) A singleton PLM learns embedding representations ($EMBED_{IN}, EMBED_{OUT}$) to guide sample selection. (3) In-domain embeddings form a distribution representation ($Dist_{IN}$) based on cosine similarity or k-centers. (4) Distribution-aware sampling selects out-of-domain examples to update the training set before fine-tuning resumes.

$$Conf_f = \frac{\sum_{q=1}^{k} Conf_q \cdot F1_{\text{VAL},q}}{\sum_{q=1}^{k} F1_{\text{VAL},q}} \tag{12}$$

4.3.2 The Dynamic Training Loop. As mentioned in Section 1, our models build on the state-of-the-art EM framework DITTO [27], which fine-tunes a PLM for EM tasks. Unlike prior frameworks [27, 31, 50] that train on a static dataset, BEACON introduces a *dynamic training loop* that adaptively resamples data using updated embeddings. This mechanism improves embedding quality throughout training–specifically for the NN, TVDF, and KCG models, which rely on meaningful structure in the embedding space. The cycle formed by steps (2-4) in Figure 6 illustrates the training loop.

While PLMs can produce reasonable embeddings out of the box, fine-tuning them on the EM task yields more accurate, domain-specific representations. Accordingly, BEACON pauses training at intermediate checkpoints to regenerate embeddings using the current model state, and then resamples the training data based on the updated embeddings. This enables better alignment between the training distribution and the embedding geometry.

To support this dynamic resampling process, BEACON integrates two coordinated PLMs with complementary objectives:

- A *pairwise* PLM, fine-tuned on labeled candidate pairs using cross-entropy loss for the core EM task.

- A *singleton* PLM, trained on individual entities using cosine similarity loss, whose role is to generate improved embeddings for the resampling procedure.

We also experimented with a simplified setup that used the singleton PLM for both training and embedding generation, but this consistently underperformed relative to the dual-PLM design. The separation allows the pairwise PLM to specialize in the matching task, while the singleton PLM provides embeddings that better capture entity-level similarity to guide resampling. This dual-PLM architecture is novel in the EM literature, explicitly decoupling the supervision and representation roles of PLMs.

4.3.3 Resampling Techniques. We now focus on the resampling stage of the dynamic training loop, which is crucial for enabling our models to maintain a pool of increasingly relevant training samples. At this point in the loop, we assume the model has paused at a checkpoint from which updated embeddings can be extracted for both in-domain and out-of-domain samples. The three methods that utilize this resampling process–NN, TVDF, and KCG–share a common structure and goal: to dynamically select out-of-domain examples that most effectively complement the in-domain data.

Figure 6 (steps 3-4) illustrates the shared workflow. Each method first constructs a representation of the in-domain distribution (e.g., a centroid or a set of embedding centers). Out-of-domain samples are then ranked by how well they align with this representation, using the scoring criteria specific to each method. Finally, the top-k samples are selected to be merged with the in-domain training set, and the training loop resumes. Repeating this process periodically allows the model to gradually refine its understanding of the embedding space and sample distribution during training.

5 Experimental Setup

We conduct a comprehensive set of experiments to evaluate our proposed methods against several baselines, including LLM-based approaches, under the EMAD setting. In Section 5.1, we discuss the datasets we use for our experiments. We follow with implementation details in Section 5.2, and introduce the baselines we use for our experiments in Section 5.3.

5.1 Datasets

To evaluate EMAD, we derive datasets from three widely used EM benchmarks: the WDC Multi-Dimensional Benchmark [37], the WDC Products Corpus [40], and Abt-Buy [25].

Domain Partitioning. Following Example 1.2, we partition each benchmark into multiple domain-specific EM datasets based on a semantic domain attribute (e.g., *product category* or *brand*). We retain only domains containing at least 100 labeled samples to ensure sufficient data for domain-specific fine-tuning and evaluation. When partitioning by domain, we preserve the original train, validation, and test splits provided by each benchmark rather than artificially rebalancing them. These splits reflect the inherent heterogeneity of real-world e-commerce data, including variation in sample availability and class balance across domains. The distribution-aware methods introduced in Section 4.2 aim to mitigate these disparities by adaptively sampling informative out-of-domain samples that can compensate for data scarcity or skewed class ratios.

The WDC Multi-Dimensional Benchmark. WDC is a large collection of product-based datasets that vary along three controlled dimensions: (1) the percentage of corner cases (i.e., difficult-to-match samples), (2) the percentage of unseen entities in the test set, and (3) the size of the development set. Unlike other EM benchmarks, WDC assigns a *gold-standard product category label* to each record, enabling a definitive partition of the data into domains. This property makes WDC uniquely suited for evaluating domain-aware EM frameworks such as BEACON.

Table 2. Average WDC Dataset Statistics Across Domains. Each value is averaged over all domain partitions.

Split	Train		Validation		Test	
Statistic	Samples	% Pos	Samples	% Pos	Samples	% Pos
Domain						
Automotive	120	57.4	32	23.7	23	26.0
Cameras	1317	57.7	204	17.7	172	18.8
Cell Phones	151	69.5	24	36.8	48	24.5
Clothing	219	58.2	28	21.2	37	23.3
Computers	6473	51.2	1174	14.6	790	15.6
Home/Garden	60	47.5	25	23.9	125	20.9
Jewelry	526	42.4	183	17.8	318	17.0
Movies	58	51.7	15	26.7	48	18.9
Instruments	266	53.4	54	20.4	100	19.1
Office	1800	47.2	467	16.4	496	16.5
Electronics	2759	53.5	483	17.0	391	18.0
Sports	1715	59.4	256	18.1	211	19.9
Tools	129	77.9	14	41.4	29	28.7
Toys	139	48.9	27	18.5	25	20.0
Total	15497	52.8	2941	16.6	2724	17.6

We use the *large* versions of the WDC datasets to guarantee a sufficiently large sample pool for studying different budget constraints. Varying the remaining two dimensions of WDC (corner case percentage and unseen-entity ratio) yields a total of **nine datasets** to use throughout our experiments. In our main experiments, we fix the corner case percentage at 50% and vary the unseen-entity percentage in the test set (0%, 50%, and 100%). Partitioning these datasets by *product category* yields between 11 and 13 domain-specific subsets, depending on the variant.

Table 2 reports average statistics across the nine WDC variants after domain partitioning. The results show substantial variation in both sample counts and class balance across domains. Some categories, such as *Computers*, contain thousands of labeled examples, while others have only a few hundred. Likewise, the proportion of positive pairs varies widely–ranging from less than 50% in domains like *Jewelry* to over 70% in *Tools*. These discrepancies introduce significant distribution shifts in both sample availability and label composition, which directly affect the difficulty of training domain-specific models.

The WDC Product Corpus. To avoid overfitting our conclusions to datasets with relatively small annotation pools, we additionally include the *medium* variant of the WDC Products Corpus [40] in our main experimental results. This dataset consists of four product domains ("Cameras", "Computers", "Shoes", and "Watches"), each with substantially larger annotation pools (approximately 4k–6k labeled pairs). These larger domains complement the other WDC variants by enabling an evaluation of EMAD in settings where in-domain supervision is already sufficient for fine-tuning, and where the role of out-of domain selection is more nuanced.

The Abt-Buy Benchmark. The Abt-Buy benchmark [25] is a widely used EM dataset that aligns product listings from Abt.com and Buy.com. While it shares the e-commerce setting of WDC, Abt-Buy differs in that its attributes are predominantly textual and less uniformly structured. This characteristic allows us to evaluate BEACON in scenarios where domain boundaries arise from heterogeneous textual representations rather than uniformly structured attributes. We partition the Abt-Buy dataset by *brand*, again retaining only domains with at least 100 labeled samples.

Scope and Generality. It is worth noting that while our experiments focus on e-commerce product data, the EMAD formulation is applicable to other macro-domains. For example, in restaurant

matching [31], different cuisine types (e.g., Indian, Italian, or vegan) may benefit from domain-specific models, while still sharing structural attributes such as names, descriptions, and locations. This shared structure suggests that distribution-aware cross-category sampling–analogous to BEACON's use of product categories–could also be effective in such settings. At the same time, not all datasets admit a natural or useful domain partition, and some partitions may offer limited benefit due to weak domain boundaries. For instance, defining domains by *price range* yielded unimpressive results in our offline experiments. Nevertheless, we view e-commerce product data as a representative and practically relevant testbed, and contend that the principles underlying EMAD could extend to other domains where meaningful substructure exists.

5.2 Implementation Details

All experiments were conducted on NVIDIA A100 (80GB) GPUs, each part of a high-performance computing cluster running Linux and Python 3.7.

For each experiment, we construct a unique training pool and domain configuration, with total training pools ranging from approximately 5k to 20k candidate pairs and the number of domains ranging from 6 to 13. In our main results (Section 6.1), models are evaluated across ascending training budgets (e.g., 1k, 2k, ..., 10k)

A training-sample budget represents a practical constraint imposed by an end user of an EM system–reflecting the maximum number of labeled candidate pairs available for fine-tuning. To simulate this setting, we evaluate across a diverse range of budgets that represent users with varying levels of labeling capacity, thereby examining how model performance scales with budget size.

For all reported experiments, we fine-tune `RoBERTa` [28] as the PLM backbone, generating 768-dimensional embeddings. We adopt the default hyperparameters from DITTO [27], including its *summarize* optimization for efficient fine-tuning.

Following prior work in EM [16, 27, 31], we evaluate models using the *F1 score*. We report both **macro-average F1** (averaged equally across domains) and **weighted-average F1** (weighted by test set size per domain). The macro F1 emphasizes performance fairness across domains, while the weighted F1 captures overall matching accuracy across all samples. Formally, for F1 scores $F1_i$ computed on the test set t_i of each domain $i \in \mathcal{D}$, we define:

$$\bar{F1}_{macro} = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} F1_i, \quad \bar{F1}_{weighted} = \frac{\sum_{i \in \mathcal{D}} F1_i \cdot |t_i|}{\sum_{i \in \mathcal{D}} |t_i|}, \tag{13}$$

where $\bar{F1}_{macro}$ is the unweighted domain-average F1, and $\bar{F1}_{weighted}$ weights each domain's F1 by the number of test pairs $|t_i|$.

5.3 Baselines

To benchmark our proposed methods, we include a diverse set of baseline models:

- **SPEC** and **GEN**: Two simple baselines for EMAD (see Section 4.1). SPEC trains with only in-domain data, while GEN samples uniformly across all domains without distinction.

- **MFSN** [50]: The state-of-the-art *domain adaptation* framework for EM. MFSN (Matching Feature Separation Network) explicitly models both *domain-common* and *domain-private* matching features to align feature distributions between source and target domains. To enable a fair comparison within the EMAD setting, we adapt MFSN by treating the labeled in-domain data as the target domain and a budgeted random subset of labeled out-of-domain data as the source domain. Under this adaptation, MFSN is trained using *both* in-domain and

Table 3. Macro F1 Performance for 50% CC WDC Datasets

(a) 0% Unseen

Method	5.0k	10.0k	Mean	SD
SPEC	0.763	0.770	0.743	0.045
GEN	0.724	0.738	0.706	0.043
BEACON (ours)	**0.778**	**0.790**	**0.768**	0.034
MFSN [50]	0.710	0.724	0.712	0.013
BATTLESHIP [19]	0.699	0.697	0.676	0.045
PROMPTEM [55]	0.610	0.613	0.609	0.008
LLAMA [15]	0.653	0.653	0.653	0.000
JELLYFISH [60]	0.625	0.625	0.625	0.000

(b) 50% Unseen

Method	5.0k	10.0k	Mean	SD
SPEC	0.655	0.688	0.660	0.034
GEN	0.660	0.709	0.656	0.057
BEACON (ours)	**0.758**	**0.769**	**0.752**	0.025
MFSN [50]	0.658	0.631	0.637	0.020
BATTLESHIP [19]	0.661	0.674	0.643	0.041
PROMPTEM [55]	0.583	0.588	0.584	0.005
LLAMA [15]	0.659	0.659	0.659	0.000
JELLYFISH [60]	0.692	0.692	0.692	0.000

(c) 100% Unseen

Method	5.0k	10.0k	Mean	SD
SPEC	0.584	0.580	0.573	0.020
GEN	0.607	0.622	0.602	0.033
BEACON (ours)	**0.652**	**0.664**	**0.645**	0.024
MFSN [50]	0.553	0.541	0.526	0.022
BATTLESHIP [19]	0.596	0.591	0.577	0.034
PROMPTEM [55]	0.529	0.533	0.527	0.006
LLAMA [15]	0.631	0.631	0.631	0.000
JELLYFISH [60]	0.611	0.611	0.611	0.000

out-of-domain labeled data, allowing it to perform cross-domain adaptation under a fixed annotation budget within our experimental pipeline.

- **Battleship [19]**: A state-of-the-art active learning framework for EM that selects samples using graph-based centrality and uncertainty. We adapt Battleship to the EMAD setting by first training it on in-domain data, and then using the resulting model to actively select a budgeted set of out-of-domain samples to add to the in-domain training pool.

- **PromptEM [55]**: A state-of-the-art low-resource EM baseline that leverages PLM-based prompt tuning and pseudo-label generation to improve performance when annotated data are scarce. We train PromptEM using in-domain labels only in order to isolate the effectiveness of modern low-resource EM techniques without incorporating out-of-domain sample selection.

- **LLaMA-3.1** [15]: An open-source LLM with 8 billion parameters that achieves state-of-the-art performance among publicly available models of comparable scale. We use the `LLaMA-3.1-8B` model, adapting the "general-complex" prompt structure and zero-shot inference approach described by Peeters et. al. [38] to perform EM. This baseline evaluates the general reasoning and semantic understanding capabilities of modern LLMs on EM tasks.

- **Jellyfish** [60]: A fine-tuned version of `LLaMA-3.1-8B` designed for structured data processing tasks, including EM. Jellyfish extends LLaMA through supervised fine-tuning on multiple data-centric tasks such as table understanding, schema matching, and EM. Notably, its fine-tuning corpus includes several EM benchmarks–such as the product-based Amazon–Google dataset [25]–making it a particularly relevant and competitive baseline for our EMAD evaluation. Jellyfish is evaluated without additional training using its recommended EM prompting template.

Note that LLM-based methods do not have an explicit training budget since the size of their training data is not publicly available. Their performance is reported as constant across budgets for comparison.

6 Results and Analysis

In this section, we present the results of evaluating the BEACON framework across our domain-partitioned datasets. Section 6.1 reports the main experimental results. Section 6.2 analyzes how BEACON functions as an ensemble of models. Section 6.3 examines per-category performance under varying budget constraints. Section 6.4 compares the performance of BEACON against baseline methods without a budget.

Table 4. Weighted F1 Performance for 50% CC WDC Datasets

(a) 0% Unseen

Method	5.0k	10.0k	Mean	SD
SPEC	**0.777**	0.782	0.746	0.061
GEN	0.728	0.761	0.717	0.055
BEACON (ours)	**0.777**	**0.783**	**0.763**	0.039
MFSN [50]	0.746	0.750	0.733	0.029
BATTLESHIP [19]	0.678	0.700	0.664	0.047
PROMPTEM [55]	0.718	0.724	0.712	0.019
LLAMA [15]	0.575	0.575	0.575	0.000
JELLYFISH [60]	0.676	0.676	0.676	0.000

(b) 50% Unseen

Method	5.0k	10.0k	Mean	SD
SPEC	0.710	0.731	0.697	0.048
GEN	0.694	0.729	0.682	0.064
BEACON (ours)	**0.758**	**0.763**	**0.739**	0.035
MFSN [50]	0.681	0.680	0.668	0.022
BATTLESHIP [19]	0.647	0.673	0.634	0.047
PROMPTEM [55]	0.669	0.683	0.669	0.014
LLAMA [15]	0.595	0.595	0.595	0.000
JELLYFISH [60]	0.691	0.691	0.691	0.000

(c) 100% Unseen

Method	5.0k	10.0k	Mean	SD
SPEC	0.644	0.646	0.631	0.032
GEN	0.668	0.685	0.657	0.044
BEACON (ours)	**0.701**	**0.717**	**0.697**	0.021
MFSN [50]	0.590	0.588	0.579	0.017
BATTLESHIP [19]	0.632	0.641	0.617	0.038
PROMPTEM [55]	0.597	0.599	0.592	0.008
LLAMA [15]	0.625	0.625	0.625	0.000
JELLYFISH [60]	0.649	0.649	0.649	0.000

6.1 Main Results

We present the results of our extensive experimental evaluation in Tables 3, 4, and 5. While we evaluate model performance across multiple annotation budgets for each dataset (e.g., from 1k to 10k), we report results for two representative budgets (5k and 10k) for brevity.

Performance on the WDC Multi-Dimensional Benchmark. We conduct experiments on the WDC datasets with 50% corner cases across ten annotation budgets ranging from 1k to 10k samples. Macro-averaged results are reported in Table 3, with corresponding weighted results shown in Table 4. In the macro setting, BEACON achieves the highest F1 scores at both β = 5k and β = 10k, as well as the highest mean performance across all ten budgets. Specifically, BEACON improves macro-average F1 by **2.5%**, **6.0%**, and **1.4%** on the seen, half-seen, and unseen test datasets, respectively–yielding an average improvement of **3.3%** over the next-best method. We attribute this gain to BEACON's ability to construct budget-constrained training sets that are both representative and well-aligned in embedding space.

In the weighted setting, BEACON again outperforms all baselines, improving weighted F1 by **2.3%**, **4.2%**, and **4.0%** for the seen, half-seen, and unseen datasets, respectively–an average improvement of **3.5%**. The generally higher weighted F1 values suggest that domains with more training data benefit more directly from effective sample selection.

We observe several additional trends. First, **SPEC** performs best when the test set contains no unseen entities. This behavior is expected, as SPEC is trained exclusively on in-domain data and therefore has direct exposure to the distribution of entities present in the test set. Conversely, the **GEN** model performs better relative to **SPEC** when the test set consists entirely of unseen entities. By sampling broadly across all domains, GEN learns more general representations that transfer better to unfamiliar domains, yielding stronger performance in fully unseen settings.

Interestingly, the performance of the LLM-based models (LLaMA-3.1 and Jellyfish) varies across the different unseen-entity configurations. Since these models are not fine-tuned on WDC data, the notion of "seen" versus "unseen" entities does not directly apply. Instead, the observed variation likely reflects differences in domain composition and difficulty across datasets–for example, some datasets may contain more linguistically complex or attribute-rich samples that better align with the LLMs' pretraining distributions.

PromptEM achieves lower F1 scores than the other methods, particularly in the macro setting. This behavior reflects its design: in our evaluation, PromptEM is trained using in-domain labels only and does not incorporate out-of-domain samples. As a result, its performance highlights that data-scarce domains often benefit substantially from leveraging cross-domain information, and that low-resource techniques alone may be insufficient to consistently optimize performance across a diverse set of domains. We further observe that BEACON consistently outperforms the active learning baseline Battleship. Although Battleship has demonstrated state-of-the-art performance in traditional single-domain active learning settings for EM, its selection strategy does not explicitly

Table 5. Weighted F1 Performance on Additional EMAD Experiments

(a) WDC Product Corpus

Method	10.0k	20.0k	Mean	SD
SPEC	0.871	0.868	0.861	0.017
GEN	0.853	0.883	0.850	0.029
BEACON (ours)	**0.877**	**0.886**	**0.873**	0.022
MFSN [50]	0.870	0.875	0.857	0.026
BATTLESHIP [19]	0.756	0.755	0.748	0.020
PROMPTEM [55]	0.820	0.815	0.814	0.014
LLAMA [15]	0.767	0.767	0.767	0.000
JELLYFISH [60]	0.837	0.837	0.837	0.000

(b) Abt-Buy

Method	2.0k	3.5k	Mean	SD
SPEC	0.836	0.839	0.830	0.014
GEN	0.772	0.870	0.796	0.087
BEACON (ours)	**0.839**	**0.885**	**0.858**	0.024
MFSN [50]	0.714	0.696	0.708	0.006
PROMPTEM [55]	0.692	0.704	0.695	0.008
BATTLESHIP [19]	0.462	0.559	0.487	0.073
LLAMA [15]	0.366	0.366	0.366	0.000
JELLYFISH [60]	0.797	0.797	0.797	0.000

(c) Ensemble Ablation (WDC)

Method	5.0k	10.0k	Mean	SD
SPEC	0.710	0.731	0.697	0.048
GEN	0.694	0.729	0.682	0.064
NN	0.722	0.741	0.705	0.059
TVDF	0.739	0.745	0.717	0.046
KCG	0.718	0.747	0.719	0.032
KCG-TVDF	**0.758**	0.763	**0.739**	0.035
ALL	0.747	**0.764**	**0.739**	0.034

account for domain structure. This suggests that methods designed for single-domain active learning do not directly transfer to budgeted, cross-domain EM scenarios without explicit domain-aware adaptations, as incorporated in BEACON.

Experiments with Larger Annotation Pools. Beyond the nine multi-dimensional WDC datasets, BEACON also achieves superior performance on the WDC Products Corpus. For this dataset, we run 10 budgets ranging from $\beta = 2$k to $\beta = 20$k in increments of 2k samples to account for the larger domains. The corresponding results are reported in Table 5a. These findings suggest that our methods are robust to variation in both the size and distribution of annotation pools across datasets. Moreover, they demonstrate that distribution-aware out-of-domain selection can improve performance even when the number of in-domain samples is already sufficient for fine-tuning. We therefore conclude that BEACON provides a robust solution for EM under tight data budgets and diverse domain conditions.

Experiments with Diverse Schema Structures. To evaluate EMAD on Abt-Buy, we experiment with five annotation budgets ranging from $\beta = 1.5$k to $\beta = 3.5$k in increments of 500 samples, with results shown in Table 5b. BEACON outperforms all baselines on this dataset at the 2k and 3.5k budget levels, achieving the highest mean F1 score across all evaluated budgets. Notably, we observe substantially less performance variation across budgets on Abt-Buy than on the other datasets (i.e., lower standard deviation). We hypothesize that this effect stems primarily from the choice of domain partition: grouping products by *brand* may induce greater homogeneity within domains than partitioning by *product category*, as brands often focus on a narrower range of products with more consistent matching characteristics. Nevertheless, most approaches still benefit from additional out-of-domain samples, with performance generally improving from the 2k to the 3.5k budget. Another point of interest is the substantial performance improvement obtained through fine-tuning for LLM-based methods. While LLaMA achieves an F1 score of 0.366 in a zero-shot setting, fine-tuning via Jellyfish more than doubles performance, reaching an F1 score of 0.797. This suggests that fine-tuning is particularly effective for predominantly textual or unstructured EM tasks, further motivating the need for principled data selection strategies to guide fine-tuning under limited budgets. Overall, these results suggest that BEACON remains effective on textual EM datasets and further highlight its robustness and practical utility across diverse EM settings.

6.2 Ablation Study

To isolate the effect of ensembling, we fix the evaluation dataset to be the 50% corner case, 50% unseen variant of WDC. We also report only weighted F1 to focus on raw predictive performance, without concern for domain size balance. All other experimental settings mirror those described in Section 6.1.

We evaluate a range of model combinations drawn from the methods introduced in Sections 4.1 and 4.2. Table 5c summarizes the results. While we experimented with all possible combinations, we

report only a representative subset to summarize the key outcomes. Specifically, Table 5c includes single-model results for each method, a two-model ensemble (KCG-TVDF), and a five-model ensemble that aggregates all proposed methods. Overall, ensemble models outperform individual ones, with **KCG-TVDF** and **KCG-TVDF-NN-SPEC-GEN** achieving the best or second-best F1 scores across all tested budgets. This confirms that ensembling multiple models has a consistently positive effect on EM performance.

Notably, many ensembles–including those shown in Table 5c–achieve nearly identical mean F1 scores across budgets, suggesting similar levels of overall predictive power. Among them, **KCG-TVDF** stands out as the most practical choice due to its simplicity and robustness: it combines only two models yet reaches top-tier performance with minimal ensemble complexity. We therefore adopt **KCG-TVDF** as the core ensemble underlying BEACON.

6.3 Performance of Individual Domains

We now turn to a per-domain analysis using the WDC dataset with 50% corner cases and 50% unseen entities. Specifically, we examine individual domain performance and analyze how our distribution-aware methods allocate out-of-domain samples when augmenting a single target domain (Computers).

To evaluate the matching performance of individual domains, we examine the 12 product categories that contain at least 100 total samples. The experimental setup is identical to that of Section 6.1, except that we report performance for each individual product category rather than macro or weighted averages. Results are summarized in Table 6, showing F1 scores for $\beta = 5$k and $\beta = 10$k budgets and their mean across budgets. To contextualize BEACON's performance, we also report SPEC–the second-best baseline in this setting.

Interestingly, EM performance does not correlate directly with the number of samples in a given domain. For example, the *Cellphones* category achieves an average F1 score of 0.980 across ten budgets despite having only 197 total samples. In contrast, *Computers*–the largest category–obtains a lower average F1 score (0.787).

This discrepancy likely stems from the inherent variation across domains. As discussed in Section 3.3, differences in vocabulary, attribute composition, and matching criteria make some domains inherently more challenging for EM than others. More difficult domains may require substantially more labeled data to achieve comparable performance, naturally leading to larger variation across categories. In addition, smaller domains tend to be less stable: with limited training and test data, their F1 scores become highly sensitive to the difficulty of individual samples. For instance, *Cellphones* may perform well simply because its limited examples are relatively straightforward, whereas *Tools*–another small domain–may contain a higher proportion of challenging or ambiguous pairs. Moreover, the limited samples in these smaller domains are less likely to be representative of their true underlying data distributions, further contributing to their observed performance instability. Consequently, most large domains exhibit more consistent F1 scores, typically in the 0.7–0.8 range.

Figure 7 illustrates how our distribution-aware methods select source domains when augmenting the *Computers* target category. We conduct a controlled EMAD experiment with an 8k annotation budget and report the proportion of selected samples originating from each source domain for the KCG and TVDF selection strategies. As Figure 7 suggests, KCG selects from a more diverse set of source domains, while TVDF concentrates its selections on fewer domains; however, both methods primarily favor semantically related domains (e.g., *Cameras* and *Electronics*).

Table 6. WDC (50%cc, 50% unseen) Category Analysis.

Category	5.0k	10.0k	Mean	SPEC
Computers (6504)	0.808	0.820	**0.787**	0.778
Electronics (2744)	0.777	0.774	**0.751**	0.736
Office (1782)	0.782	0.789	**0.748**	0.706
Sports (1705)	0.646	0.594	**0.610**	0.589
Cameras (1580)	0.755	0.762	**0.755**	0.728
Jewelry (460)	0.645	0.656	**0.645**	0.604
Automotive (143)	0.950	0.950	**0.948**	0.891
Clothing (199)	0.626	0.742	0.733	**0.735**
Cellphones (139)	0.989	1.000	**0.980**	0.426
Instruments (130)	0.712	0.751	**0.705**	0.675
Tools (102)	0.656	0.646	**0.630**	0.546
Home/Garden (70)	0.745	0.744	**0.735**	0.508
Weighted Average (15558)	0.758	0.763	**0.739**	0.697

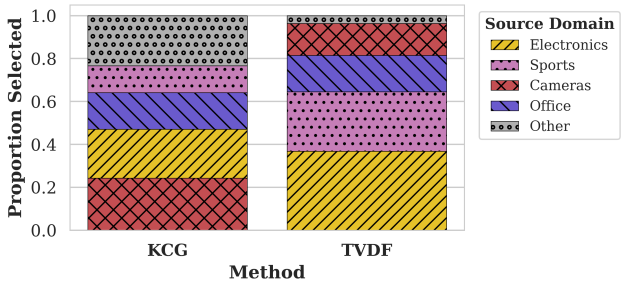


Fig. 7. Analysis of out-of-domain sample selection for the *Computers* target domain from WDC using the KCG and TVDF methods.

6.4 Unbudgeted Analysis

We now perform an analysis parallel to Section 6.3, but without imposing a training budget. This experiment compares BEACON against the GEN and SPEC models when each is allowed to use all available training data. For GEN, this corresponds to training a single model over the entire dataset of approximately 16,000 candidate pairs, while SPEC trains separate models using only domain-specific data for each category. In this context, GEN represents a fully pooled model that exploits all training data across domains, whereas SPEC serves as a purely domain-isolated baseline. We show the capabilities of BEACON (under a 10k budget) to improve the in-domain performance. The results are given in Table 7. For SPEC, the values in parentheses denote the number of samples in each domain, while for BEACON they indicate the absolute improvement in F1 score over SPEC. Overall, BEACON achieves higher F1 scores than both baselines despite using fewer samples than GEN. The largest gains appear in smaller domains such as *Instruments*, where BEACON improves over the next-best baseline by 0.155.

In a few domains, GEN slightly outperforms BEACON–for example, in *Clothing*, where GEN achieves an F1 score 0.10 higher. SPEC performs best in the largest domain, *Computers*, though BEACON's score is only 0.001 lower. On average, BEACON exceeds both unbudgeted baselines, reaching a mean F1 score of 0.763.

Table 7. WDC (50%cc, 50% unseen) Unbudgeted Analysis.

Category	SPEC	+BEACON (→10k)	GEN (~16K)
Computers	**0.821** (6504)	0.820 (-0.001)	0.819
Electronics	0.770 (2744)	**0.774** (+0.004)	0.737
Office	0.714 (1782)	**0.789** (+0.075)	0.786
Sports	0.317 (1705)	0.594 (+0.277)	**0.659**
Cameras	0.702 (1580)	**0.762** (+0.060)	0.698
Jewelry	0.293 (460)	0.656 (+0.363)	**0.683**
Clothing	0.308 (199)	0.742 (+0.434)	**0.842**
Automotive	0.370 (143)	**0.950** (+0.580)	0.625
Cellphones	0.393 (139)	**1.000** (+0.607)	**1.000**
Instruments	0.330 (130)	**0.751** (+0.421)	0.596
Tools	0.438 (102)	**0.646** (+0.208)	0.625
Home/Garden	0.355 (70)	**0.744** (+0.389)	0.705
Weighted Avg	0.627	**0.763** (+0.136)	0.754

Another noteworthy observation from Table 7 is that both BEACON (at the 10k budget) and the unbudgeted GEN model achieve an F1 score of 1.0 on *Cellphones*, whereas SPEC performs substantially worse, with an F1 of 0.393. Although this result partly reflects the volatility of smaller domains–given that *Cellphones* is the fourth smallest domain–it also highlights the substantial benefit of incorporating out-of-domain samples. In this case, the inclusion of cross-domain data enables BEACON and GEN to achieve perfect category-level performance, underscoring the potential of out-of-domain sampling to stabilize small-domain training.

7 Conclusion

This work introduces *Budget-Aware Entity Matching Across Domains*, a new problem formulation aimed at training effective domain-specific matchers under strict sample budgets. To address this challenge, we propose **BEACON**, a distribution-aware sampling framework that selects out-of-domain samples to optimize domain-specific matching performance. We evaluate BEACON on multiple domain-partitioned datasets derived from several Entity Matching benchmarks, comparing it against strong baselines including low-resource, domain adaptation, and LLM-based methods. BEACON consistently outperforms these baselines, achieving higher macro and weighted F1 scores under equal or smaller budgets. While BEACON currently leverages a PLM backbone fine-tuned for EM, its principles extend to other embedding-based approaches. We plan to explore LLM backbones to assess cost–performance trade-offs and hybrid strategies that integrate embedding-based sampling with active learning to further enhance budget-aware EM.

Acknowledgments

Large language models were used to assist with drafting portions of the manuscript, including text, tables, graphs, etc. This material is based upon work supported by the National Science Foundation under Grant NRT-HDR-2021871. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Mehdi Akbarian Rastaghi, Ehsan Kamalloo, and Davood Rafiei. 2022. Probing the robustness of pre-trained language models for entity matching. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 3786–3790.

[2] Amazon. 2025. *Product Title: 14K White Gold 1 Carat Lab Grown 6 Prong Solitaire Round Cut IGI CERTIFIED Diamond Engagement Ring*. Amazon.com. Retrieved October 10, 2025 from https://www.amazon.com/dp/B09X5QL8T4 Accessed on October 10, 2025.

[3] Amazon. 2025. *Product Title: Nike Air Zoom Pegasus 40 - Men's Running Casual Shoes*. Amazon.com. Retrieved October 10, 2025 from https://www.amazon.com/dp/B0C4PK22LB Accessed on October 10, 2025.

[4] Jan-Philipp Awick and Jorge Marx Gómez. 2025. Entity Matching in the Era of Language Models: A Structured Literature Review. In *2025 25th International Conference on Control Systems and Computer Science (CSCS)*. IEEE, 368–375.

[5] Nils Barlaug and Jon Atle Gulla. 2021. Neural networks for entity matching: A survey. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 15, 3 (2021), 1–37.

[6] Kedar Bellare, Suresh Iyengar, Aditya G Parameswaran, and Vibhor Rastogi. 2012. Active sampling for entity matching. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1131–1139.

[7] Indrajit Bhattacharya and Lise Getoor. 2007. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1, 1 (2007), 5–es.

[8] Mikhail Bilenko and Raymond J Mooney. 2003. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. 39–48.

[9] Alexander Brinkmann, Roee Shraga, and Christina Bizer. 2024. Sc-block: Supervised contrastive blocking within entity resolution pipelines. In *European Semantic Web Conference*. Springer, 121–142.

[10] Ursin Brunner and Kurt Stockinger. 2020. Entity matching with transformer architectures-a step forward in data integration. In *23rd International Conference on Extending Database Technology, Copenhagen, 30 March-2 April 2020*. OpenProceedings, 463–473.

[11] Fernando de Meer Pardo, Claude Lehmann, Dennis Gehrig, Andrea Nagy, Stefano Nicoli, Misheva Branka Hadji, Martin Braschler, and Kurt Stockinger. 2025. Gralmatch: matching groups of entities with graphs and language models. In *28th International Conference on Extending Database Technology (EDBT), Barcelona, Spain, 25-28 March 2025*. Open Proceedings, 1–12.

[12] 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, volume 1 (long and short papers)*. 4171–4186.

[13] Thomas G Dietterich. 2000. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*. Springer, 1–15.

[14] Huahua Ding, Chaofan Dai, Yahui Wu, Wubin Ma, and Haohao Zhou. 2024. Setem: Self-ensemble training with pre-trained language models for entity matching. *Knowledge-Based Systems* 293 (2024). doi:10.1016/j.knosys.2024.111708

[15] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv e-prints* (2024), arXiv–2407.

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

[17] Ivan P Fellegi and Alan B Sunter. 1969. A theory for record linkage. *Journal of the American statistical association* 64, 328 (1969), 1183–1210.

[18] Cheng Fu, Xianpei Han, Jiaming He, and Le Sun. 2021. Hierarchical matching network for heterogeneous entity resolution. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*. 3665–3671.

[19] Bar Genossar, Avigdor Gal, and Roee Shraga. 2023. The battleship approach to the low resource entity matching problem. *Proceedings of the ACM on Management of Data* 1, 4 (2023), 1–25.

[20] Mauricio A Hernández and Salvatore J Stolfo. 1995. The merge/purge problem for large databases. *ACM Sigmod Record* 24, 2 (1995), 127–138.

[21] Jiacheng Huang, Wei Hu, Zhifeng Bao, Qijin Chen, and Yuzhong Qu. 2023. Deep entity matching with adversarial active learning. *The VLDB Journal* 32, 1 (2023), 229–255.

[22] 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)*. IEEE, 41–44.

[23] Yuchen Jiang, Qi Li, Han Zhu, Jinbei Yu, Jin Li, Ziru Xu, Huihui Dong, and Bo Zheng. 2022. Adaptive domain interest network for multi-domain recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 3212–3221.

[24] 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 Annual Meeting of the Association for Computational Linguistics*. 5851–5861.

[25] Hanna Köpcke, Andreas Thor, and Erhard Rahm. 2010. Evaluation of entity resolution approaches on real-world match problems. *Proceedings of the VLDB Endowment* 3, 1-2 (2010), 484–493.

[26] Ludmila I Kuncheva. 2014. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons.
[27] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. 2020. Deep entity matching with pre-trained language models. *Proceedings of the VLDB Endowment* 14, 1 (2020), 50–60.
[28] 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. *arXiv preprint arXiv:1907.11692* (2019).
[29] Jwen Fai Low, Benjamin CM Fung, and Pulei Xiong. 2024. Better entity matching with transformers through ensembles. *Knowledge-Based Systems* 293 (2024). doi:10.1016/j.knosys.2024.111678
[30] Alvaro E Monge, Charles Elkan, et al. 1996. The field matching problem: algorithms and applications.. In *Kdd*, Vol. 2. 267–270.
[31] 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*. 19–34.
[32] John Bosco Mugeni, Steven Lynden, Toshiyuki Amagasa, and Akiyoshi Matono. 2023. Adapterem: pre-trained language model adaptation for generalized entity matching using adapter-tuning. In *Proceedings of the 27th International Database Engineered Applications Symposium*. 140–147.
[33] Matteo Paganelli, Francesco Del Buono, Andrea Baraldi, and Francesco Guerra. 2022. Analyzing How BERT Performs Entity Matching. *Proc. VLDB Endow.* 15, 8 (2022), 1726–1738.
[34] Matteo Paganelli, Donato Tiano, and Francesco Guerra. 2024. A multi-facet analysis of BERT-based entity matching models. *VLDB J.* 33, 4 (2024), 1039–1064.
[35] George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, and Themis Palpanas. 2020. Blocking and filtering techniques for entity resolution: A survey. *ACM Computing Surveys (CSUR)* 53, 2 (2020), 1–42.
[36] Ralph Peeters and Christian Bizer. 2021. Dual-objective fine-tuning of BERT for entity matching. *Proceedings of the VLDB Endowment* 14 (2021), 1913–1921.
[37] Ralph Peeters, Reng Chiz Der, and Christian Bizer. 2023. WDC Products: A Multi-Dimensional Entity Matching Benchmark. arXiv:2301.09521 [cs.LG] https://arxiv.org/abs/2301.09521
[38] Ralph Peeters, Aaron Steiner, and Christian Bizer. 2025. Entity matching using large language models. *OpenProceedings* 2 (2025), 529–541.
[39] Anna Primpeli and Christian Bizer. 2020. Profiling entity matching benchmark tasks. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 3101–3108.
[40] Anna Primpeli, Ralph Peeters, and Christian Bizer. 2019. The WDC training dataset and gold standard for large-scale product matching. In *Companion Proceedings of The 2019 World Wide Web Conference*. 381–386.
[41] Nicholas Pulsone, Roee Shraga, and Gregory Goren. 2026. BEACON: Budget-Aware Entity Matching Across Domains (Extended Technical Report). *arXiv preprint* (2026). Technical report, available at arXiv.
[42] Kun Qian, Lucian Popa, and Prithviraj Sen. 2017. Active learning for large-scale entity resolution. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1379–1388.
[43] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).
[44] Sunita Sarawagi and Anuradha Bhamidipaty. 2002. Interactive deduplication using active learning. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. 269–278.
[45] Burr Settles. 2009. Active learning literature survey. (2009).
[46] Kashif Shah, Selcuk Kopru, and Jean David Ruvini. 2018. Neural network based extreme classification and similarity models for product matching. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*. 8–15.
[47] Roee Shraga. 2022. HumanAL: Calibrating human matching beyond a single task. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*. 1–8.
[48] Rickson Simioni Pereira. 2024. Automated Generation and Configuration of Domain-Specific Recommender Systems. In *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems*. 168–173.
[49] Giovanni Simonini, Sonia Bergamaschi, and HV Jagadish. 2016. BLAST: a loosely schema-aware meta-blocking approach for entity resolution. *Proceedings of the VLDB Endowment* 9, 12 (2016), 1173–1184.
[50] Chenchen Sun, Yang Xu, Derong Shen, and Tiezheng Nie. 2024. Matching feature separation network for domain adaptation in entity matching. In *Proceedings of the ACM Web Conference 2024*. 1975–1985.
[51] 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. *Proceedings of the VLDB Endowment* 14, 11 (2021), 2459–2472.

[52] Mohamed Trabelsi, Jeff Heflin, and Jin Cao. 2022. Dame: Domain adaptation for matching entities. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 1016–1024.
[53] 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 *Proceedings of the 2022 international conference on management of data*. 443–457.
[54] Jianhong Tu, Ju Fan, Nan Tang, Peng Wang, Guoliang Li, Xiaoyong Du, Xiaofeng Jia, and Song Gao. 2023. Unicorn: A Unified Multi-tasking Model for Supporting Matching Tasks in Data Integration. *Proc. ACM Manag. Data* 1, 1 (2023), 84:1–84:26.
[55] Pengfei Wang, Xiaocan Zeng, Lu Chen, Fan Ye, Yuren Mao, Junhao Zhu, and Yunjun Gao. 2022. PromptEM: Prompt-tuning for Low-resource Generalized Entity Matching. *Proc. VLDB Endow.* 16, 2 (2022), 369–378.
[56] Tianshu Wang, Xiaoyang Chen, Hongyu Lin, Xuanang Chen, Xianpei Han, Le Sun, Hao Wang, and Zhenyu Zeng. 2025. Match, Compare, or Select? An Investigation of Large Language Models for Entity Matching. In *COLING*. Association for Computational Linguistics, 96–109.
[57] William E Winkler. 1990. String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. (1990).
[58] Dezhong Yao, Yuhong Gu, Gao Cong, Hai Jin, and Xinqiao Lv. 2022. Entity Resolution with Hierarchical Graph Attention Networks. In *SIGMOD Conference*. ACM, 429–442.
[59] Alexandros Zeakis, George Papadakis, Dimitrios Skoutas, and Manolis Koubarakis. 2025. An in-depth analysis of pre-trained embeddings for entity resolution. *The VLDB Journal* 34, 1 (2025), 5.
[60] Haochen Zhang, Yuyang Dong, Chuan Xiao, and Masafumi Oyamada. 2023. Jellyfish: A large language model for data preprocessing. *arXiv preprint arXiv:2312.01678* (2023).
[61] Haochen Zhang, Yuyang Dong, Chuan Xiao, and Masafumi Oyamada. 2024. Jellyfish: Instruction-tuning local large language models for data preprocessing. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. 8754–8782.
[62] Zeyu Zhang, Paul Groth, Iacer Calixto, and Sebastian Schelter. 2025. A Deep Dive Into Cross-Dataset Entity Matching with Large and Small Language Models. In *Proceedings of the 28th International Conference on Extending Database Technology (EDBT)*. OpenProceedings, 922–934. doi:10.48786/edbt.2025.75

Received October 2025; revised February 2026; accepted February 2026