Structure-Guided Entity Resolution: Fine-Tuning LLMs for Robust Name Matching in Complex Linguistic Contexts

Shivam Chourasia
Dream Sports
chshivam@utexas.edu

Hitesh Kapoor
Dream Sports
hitesh.p.kapoor@gmail.com

Nilesh Patil
Dream Sports
nilesh@urgrad.rochester.edu

Abstract

Matching person names across heterogeneous records is a core challenge in entity resolution, especially within linguistically and culturally complex environments. Variations in naming conventions, inconsistent transliteration across scripts, and frequent data entry errors make it difficult to unify user identities, an essential requirement for Know Your Customer (KYC) compliance. While Large Language Models have shown promise in understanding natural language, they often struggle with the structured ambiguity present in such domain-specific settings. This paper introduces **Structure-Guided Entity Resolution (SGER)**, a novel framework that fine-tunes an LLM through a two-phase curriculum. The model is first trained to parse the grammatical and semantic structure of personal names, then optimized for the downstream task of binary entity matching. We evaluate SGER in the challenging context of Indian identity data, one of the most linguistically diverse and noisy environments globally. SGER achieves **99.02% accuracy** and an **F1 of 0.994** on a held-out set of 50,000 real-world pairs, outperforming GPT-4o few-shot prompting and single-stage fine-tuning baselines. The system is fully deployed in production at Dream11, the world's largest fantasy sports platform, serving 250M+ users. Our results demonstrate that curriculum-guided training enables robust, high-precision entity resolution in real-world multilingual systems at scale.

1 Introduction

Entity Resolution (ER) is the task of identifying records that refer to the same real-world entity (Elmagarmid et al., 2007; Fellegi and Sunter, 1969). In compliance-heavy domains such as KYC and AML, the quality of ER has direct operational and regulatory impact. This challenge is especially pronounced in culturally and linguistically diverse contexts, where names do not behave like stable keys. They vary with cultural convention, writing system, data entry practice, and the quirks of digitization.

The difficulties are especially visible in India. Naming conventions differ across regions and communities, and there is no single standard for which components count as first, middle, and last. A single individual's name may include patronymics, caste names, or village names as integral components, and these conventions vary by region and community. Honorifics and social suffixes such as `-bhai` and `-ji` appear in daily use and vanish in formal records. Scripts differ, and transliteration into Latin letters is not uniform across Devanagari, Bengali, Tamil, and other scripts (International Organization for Standardization, 2001; Steinberger et al., 2013). Operational pipelines add their own sources of variation: manual entry introduces typographical errors, and OCR blends tokens and drops whitespace. The same person may appear as "Shubham Kumar Singh" on a PAN card, "Shubham K Singh" in a bank record, and "ShubhamKumarSingh" in a utility bill scan. Table 1 lists concrete examples.

Our platform, Dream11, the world's largest fantasy sports platform serving over 250 million users, faces these variations daily. Missed links block legitimate users from completing verification, while false links can fail to flag fraudulent actors creating multiple accounts. Such fragmentation is a daily, high-volume challenge for platforms at scale. The limitations of traditional methods lead to high rates of both false positives and false negatives, carrying significant operational costs and regulatory risks. Classic techniques such as edit distance, Jaro-Winkler, and phonetic encodings provide useful signals but struggle with the full structure of the problem (Cohen et al., 2003; Christophides et al., 2020).

Large Language Models change the landscape because they capture regularities beyond local character edits. Prior work shows that fine-tuned LLMs

can outperform conventional ER systems (Peeters et al., 2024; Steiner et al., 2024; Li et al., 2024; Fan et al., 2024; Xin et al., 2024). However, training directly on the binary decision forces the model to learn name structure and the decision boundary simultaneously, leaving performance untapped in domains where the structure is strong but implicit.

We propose **Structure-Guided Entity Resolution (SGER)**. We take Llama 3 8B and fine-tune it in two phases. Phase 1 teaches the model to map a raw name to a JSON schema with fields `first_name`, `middle_name`, and `last_name`. Phase 2 starts from the Phase 1 checkpoint and trains a binary classifier. On a disjoint validation set of 50,000 pairs from Indian KYC workflows, SGER reaches 99.02% accuracy and an F1 of 0.994. We describe our deployed system that serves this model in production, processing identity verification requests at scale with measurable business impact.

This paper's primary contribution is a highly accurate and deployed system for name-based entity matching in linguistically diverse environments that leverages a structured, curriculum-based fine-tuning strategy. We demonstrate that this approach allows the model to internalize the linguistic and cultural nuances of Indian names, achieving state-of-the-art accuracy and setting a new standard for entity resolution in complex, multilingual domains.

2 Related Work

The literature on name matching spans basic string comparison, phonetic heuristics, classical supervised learning, and modern neural approaches. We summarize relevant strands and place SGER within that landscape.

Heuristics and rules: Edit distance, Jaro-Winkler, and phonetic encodings capture surface similarity but struggle with token reordering, merged whitespace, and cross-script transliteration (Cohen et al., 2003; Christen, 2012). Phonetic algorithms such as Soundex and Metaphone are tuned for English pronunciation and are ill-suited for Indian languages, where morphological and cross-script variation dominates (Mhaske et al., 2022).

Classical ML: Before deep learning, supervised models using string similarity features and handcrafted rules outperformed individual heuristics but required extensive feature engineering (Christophides et al., 2020).

Deep learning and PLMs: Pre-trained language models like BERT (Devlin et al., 2019), when fine-tuned for ER, significantly improve over previous methods by capturing richer semantic information (Li et al., 2023).

LLMs for entity matching: The survey by Peeters and Bizer (2023) maps the landscape, while Steiner et al. (2024) demonstrate that fine-tuning is highly effective. Huang (2024) proposes a relation-based approach for high-stakes tasks. Our SGER methodology implicitly addresses this concern: by first training the model to understand structural relationships between name components (Phase 1), we equip it to disambiguate difficult cases in the downstream binary setting. We are the first to empirically validate a two-phase curriculum learning framework (Feng et al., 2023; Soviany et al., 2022) tailored to name-based entity resolution in complex real-world settings.

We argue that in domains with implicit "grammar" (e.g., culturally diverse names), standard fine-tuning is suboptimal, as it forces the model to simultaneously learn structural patterns and perform classification. Our curriculum learning strategy separates these concerns, leading to improved performance. While prior work has explored aspects of name processing in Indic languages (Bahad et al., 2024), none have adopted a structured, multi-phase training paradigm that incrementally builds linguistic understanding before downstream decision-making.

3 Methodology: Curriculum-Based Entity Resolution

To tackle the challenges outlined above, we developed the SGER system. The core of our methodology is a two-phase, curriculum-based fine-tuning approach that equips a pre-trained LLM with specialized knowledge of name structures before training it on the matching task. Our central hypothesis is that by decoupling structural understanding from the matching task, the model develops a more robust and generalizable internal representation of names, providing a powerful inductive bias for the downstream classification task.

An overview of the two-phase training pipeline is illustrated in Figure 1. Phase 1 focuses on name structure understanding, where the model is fine-tuned to output structured JSON representations from full name strings. In Phase 2, the fine-tuned model is used as a starting point for binary name

Table 1: Examples of Name Matching Challenges in Multilingual Contexts (Indian Case Study)

Challenge Category	**Example 1**	**Example 2**	**Match**	**Explanation**
Inverted or Merged Names	Kirtan Singh	SinghKirtan	Yes	Order of first and last names is swapped or concatenated.
Spelling Inconsistencies	Subham	Shubham	Yes	Transliteration from regional scripts leads to inconsistent English spellings.
Phonetic Ambiguity	Vipin	Bipin	No	Similar sounding names that refer to different individuals.
Truncations & Abbreviations	Rajeshk	Rajesh Kumar	Yes	Informal records often use shortened or abbreviated names.
Patronymic Variants	Rajesh So Hari	Hari	No	Father's name appears as a relational suffix (son of).
Honorifics & Postfixes	Rameshbhai Patel	Ramesh Patel	Yes	Suffixes like `-bhai`, `-ji`, or honorifics are inconsistently applied.

matching, where it classifies whether two name variants refer to the same individual.

3.1 System Architecture

SGER uses Meta's Llama 3 8B as the base (Grattafiori et al., 2024). The choice balances accuracy with cost: the model is large enough to capture regularities in Indian names and small enough to fine-tune and serve efficiently. Llama 3 8B is open-source, well-documented, and widely adopted in both academic and industrial contexts, making it a natural and reproducible choice. Our task is linguistically complex but narrow in scope and highly structured. This allows us to use a model of modest size without compromising on accuracy. As our results show, the curriculum-based fine-tuning pipeline enables Llama 3 8B to achieve near-perfect accuracy, demonstrating that larger models are not necessarily required for specialized entity resolution tasks. The input at inference is a pair of name strings, and the output is a single token: "`Yes`" or "`No`". The novelty lies in the training path rather than in a new network component.

3.2 Phase 1: Name Structure Understanding

Phase 1 teaches the model the internal grammar of names through a supervised mapping from a single name string to a JSON object listing its main components.

Input: A single name string, e.g., "Kirtan Singh Rathore". Inputs are space-trimmed and lowercased so that the model encounters the same quirks that appear in real data.

Output: A structured JSON object:

```
{
  "first_name": "Kirtan",
  "middle_name": "Singh",
  "last_name": "Rathore"
}
```

Training Data: We constructed a dataset of approximately 10,000 Indian names, manually annotated with their structural components. The names were sampled from anonymized identity records and augmented with synthetically generated names to improve coverage and diversity. Particular attention was given to capturing a broad spectrum of regional, linguistic, and cultural naming patterns, including non-standard spellings, varied name orders, and merged or abbreviated forms. This dataset is distinct from the binary name-matching pairs described in Section 4.1 and serves as an upstream pretraining corpus that equips the model with structural awareness prior to fine-tuning on the matching task.

Optimization: We apply supervised fine-tuning with LoRA on Llama 3 8B. The task encourages the model to build a stable internal representation of how names are composed, lowering the difficulty of Phase 2. We use mixed precision training and early stopping.

By completing this task, the model learns to iden-

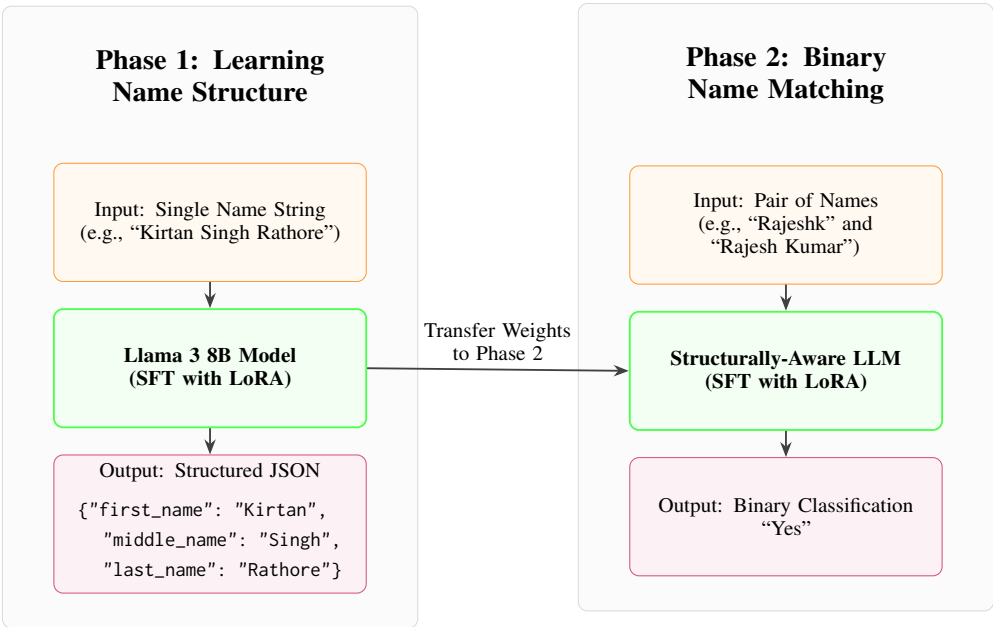

Figure 1: Structure-Guided Entity Resolution (SGER) methodology. Phase 1 fine-tunes Llama 3 8B to parse names into structured JSON using SFT with LoRA. The learned weights transfer to Phase 2, where the model performs binary name matching.

tify common first names, family names, and middle names (like "Kumar" or "Lal"). It also learns to handle structural variations, such as recognizing that "SinghKirtan" is likely a merged form of a first and last name. This phase acts as a form of domain-specific pre-training, equipping the model with an internal representation of name semantics before the binary matching task.

3.3 Phase 2: Binary Name Matching

Phase 2 continues from the Phase 1 checkpoint and trains a classifier over pairs of names.

Input: Each instance contains a brief instruction, a few-shot block with examples, and the target pair:

```
[Instruction]
[Few-Shot Examples]
Name 1: "A" | Name 2: "B" ->
Match?
```

Prompt Design: The instruction and few-shot prompts were crafted with the risk operations team to reflect the full range of real-world name ambiguities seen in production. This includes handling of abbreviations, merged tokens, transliteration inconsistencies from regional scripts, reordered components, and honorific or suffix variations. The same instruction template was used across all few-shot and fine-tuned model setups, ensuring consistency during training and inference.

Output: A single classification token: either "`Yes`" or "`No`", indicating whether the two input names refer to the same individual. At inference, we extract the model's scores for the answer tokens "`Yes`" and "`No`", apply a softmax to obtain the probability of a match, and classify a pair as positive when this probability exceeds a decision threshold selected to maximize F1 on the validation set. This yields deterministic, calibrated binary decisions.

Training Data: SFT with LoRA on labeled name pairs (Section 4.1). The prompt template is used during training to match the evaluation setup.

4 Experimental Setup

4.1 Data

The evaluation corpus is proprietary and drawn from our platform's historical KYC verification workflows. Records contain name pairs only; no other PII fields are present.

We begin with 20,000 labeled name pairs curated from our past operations and expand to more than 50,000 examples with three domain-informed augmentation strategies.

Our augmentation philosophy draws from computer vision, where transforms such as horizontal flips, geometric rotations, and noise injection encode known invariances of the visual domain (Shorten and Khoshgoftaar, 2019). We apply the same principle to names: each augmentation strat-

egy encodes a specific invariance relevant to the name-matching domain.

- **Pair Swapping**, analogous to mirroring in CV, adds for each pair (name1, name2) the swapped pair (name2, name1) to ensure the model is order-invariant.
- **Component Permutation**, analogous to geometric transforms, permutes the constituent parts (first, middle, last name) of names to create structurally valid but novel examples.
- **Random Space Removal**, analogous to noise injection, randomly removes spaces between name parts (e.g., "Kirtan Singh" becomes "KirtanSingh") to simulate common data entry errors.

Evaluation uses a disjoint held-out set of 50,000 real-world pairs drawn from our historical production data and manually labeled by our in-house risk operations team. Disjointness is enforced at the individual name level: no name appearing in any test pair occurs in any training pair, whether original or augmented, preventing data leakage across splits. This held-out set is deliberately 2.5× the base training corpus to provide high confidence in production performance, as the system operates in a compliance-critical identity verification pipeline. The impact of each augmentation strategy is quantified in our ablation study (Section 5.2).

4.2 Baselines

We compare SGER with five systems:

- **Levenshtein Similarity:** Fuzzy string matching with threshold 0.8 tuned for F1 (Cohen et al., 2003; Christen, 2012).
- **BERT (Fine-Tuned):** Multilingual `bert-base-multilingual-cased` with a classification head over [CLS] on concatenated names.
- **GPT-4o (Few-Shot):** Prompted with the same instruction template and curated examples as SGER, without fine-tuning.
- **LLM (Few-Shot, Llama 3 8B):** The same few-shot setup on the base Llama 3 8B model.
- **LLM (SFT, Llama 3 8B):** A one-stage fine-tune directly on the binary task. We report variants with and without data augmentation to isolate its contribution.

4.3 Training Details

All experiments use Llama 3 8B Instruct as the base model. Both phases rely on SFT with LoRA (Hu et al., 2021): rank 4, scaling factor `alpha` 8, mixed precision (`fp16`). We adopt default hyperparameters from PEFT and Hugging Face Transformers with early stopping on validation F1. Training is conducted on a single NVIDIA A100 GPU with 80GB memory. Inputs are space-trimmed and lowercased.

4.4 Evaluation Metrics

We report accuracy, precision, recall, and F1. High precision reduces false links (compliance risk); high recall reduces friction for legitimate users. We use F1 for early stopping.

5 Results and Discussion

5.1 Quantitative Analysis

Table 2: Performance Comparison of Name Matching Models

Model / Method	Accuracy	Precision	Recall	F1-Score
Levenshtein (Th=0.8)	57.4%	75.1%	70.2%	0.726
BERT (Fine-Tuned)	69.1%	81.2%	80.1%	0.806
GPT-4o (Few-Shot)	85.4%	90.1%	92.1%	0.911
LLM (Few-Shot, Llama 3 8B)	74.3%	85.2%	82.6%	0.839
LLM (SFT, Llama 3 8B)	91.2%	93.3%	95.9%	0.946
LLM (SFT + Aug., Llama 3 8B)	95.7%	97.6%	97.0%	0.973
SGER (Our Method)	**99.02%**	**99.95%**	**98.9%**	**0.994**

SGER outperforms all baselines by a significant margin. Classical methods like Levenshtein struggle with structural and phonetic variations. GPT-4o performs strongly in the few-shot setting but still falls short, highlighting the value of domain-specific fine-tuning. The few-shot Llama 3 model underperforms its fine-tuned variants, confirming that in-domain supervision is essential. SGER's curriculum combined with data augmentation pushes F1 from 0.946 to 0.994 over the single-stage baseline, demonstrating that curriculum learning enables production-grade entity matching.

5.2 Ablation Study

The rows in Table 2 are ordered to isolate the contribution of each pipeline component, starting from a vanilla single-stage SFT baseline and adding one component at a time.

Effect of Data Augmentation. Training on the original 20K curated pairs alone leaves the model

under-exposed to structural variants that appear frequently in production data. Expanding to 50K+ examples with the augmentation strategies described in Section 4.1 raises the F1 from 0.946 to 0.973. Without these augmentations, the model memorizes surface patterns of the curated set but fails to generalize to the long tail of distortions encountered in production.

Effect of Curriculum Learning. Adding the Phase 1 structural pre-training step (Section 3.2) before Phase 2 matching yields a further gain. The curriculum helps most on pairs combining multiple distortions, for example an abbreviated and reordered name, where Phase 1 stabilizes the model's internal representation of name components before the matching decision. Together, data augmentation and curriculum learning are complementary and their combined effect brings SGER to an F1 of 0.994.

5.3 Error Analysis

Manual review of misclassified pairs revealed two primary failure modes:

- **Ambiguous Phonetic Cases:** Names phonetically similar but spelled differently (e.g., `"Saurabh"` vs. `"Sorab"`) were sometimes classified as non-matches due to inconsistent transliteration or regional pronunciation norms.
- **Complex Compound Errors:** Cases combining multiple simultaneous distortions (missing whitespace, inverted order, and OCR corruption) remain challenging. For instance, "`SubhamKshr`" vs. "`KishoreShubham`" was misclassified due to concurrent whitespace, ordering, and character-level errors.

6 Production Deployment and Impact

SGER is fully deployed in our production identity verification pipeline, serving over 250 million users on Dream11.

6.1 Deployment Architecture

We deploy the model on a Kubernetes-based architecture providing horizontal scalability, high availability, and fault tolerance. Inference runs on 3× NVIDIA L4 GPUs. We serve the model using the vLLM framework (Kwon et al., 2023), which enables efficient token-level scheduling and high-throughput batching.

6.2 System Performance

Our deployed system handles up to **10,000 requests per minute** (RPM) with a **P99 latency of 120 milliseconds**. This level of throughput is critical during major sporting events, when our user onboarding traffic spikes significantly.

6.3 Business Impact

Prior to SGER, heuristic-based matching routed ambiguous cases to a manual review queue.

Elimination of manual review. With 99.95% precision, SGER enables fully automated match/no-match decisions. The rare false negatives ($<$1% of users) are resolved through customer service.

Cost savings. Prior to SGER, ambiguous cases were routed to a third-party vendor for manual verification. Eliminating this dependency translates to over **$500K in annual cost savings** from reduced operational overhead.

7 Conclusion

We presented SGER, a curriculum-based approach to name-only entity resolution that first trains an LLM to understand name structure before fine-tuning it on binary matching. On Indian KYC data, SGER achieves a state-of-the-art F1 of 0.994, eliminating manual review and delivering over $500K in annual savings. Its successful deployment in our platform's production pipeline underscores its robustness, scalability, and real-world value.

While our experiments focus on Indian KYC data, SGER is broadly applicable to other settings with rich linguistic diversity, irregular naming, or noisy inputs. Applications such as multilingual identity systems, global e-commerce, and cross-border fraud detection can benefit from this approach. Our error analysis suggests several promising directions: combining SGER with phonetic algorithms for ambiguous transliteration cases and extending to a multimodal system that processes identity document images to resolve OCR-related noise at their origin (Ai et al., 2024).

Ethical Considerations

The dataset used in this work is proprietary and drawn from our platform's anonymized KYC records. No personally identifiable information was used in model evaluation or is reported in this paper. All data processing was conducted under our internal data protection agreements. The deployed

system is designed to reduce bias in identity verification by handling diverse naming conventions more accurately than rule-based approaches.

References

Chaoyi Ai, Yong Jiang, Shen Huang, Pengjun Xie, and Kewei Tu. 2024. Learning robust named entity recognizers from noisy data with retrieval augmentation. *Computing Research Repository*, arXiv:2407.18562.

Sankalp Bahad, Pruthwik Mishra, Karunesh Arora, Rakesh Chandra Balabantaray, Dipti Misra Sharma, and Parameswari Krishnamurthy. 2024. Fine-tuning pre-trained named entity recognition models for indian languages. *Computing Research Repository*, arXiv:2405.04829.

Peter Christen. 2012. *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer.

Vassilis Christophides, Vasilis Efthymiou, Themis Palpanas, George Papadakis, and Kostas Stefanidis. 2020. An overview of end-to-end entity resolution for big data. *ACM Computing Surveys*, 53(6).

William W. Cohen, Pradeep Ravikumar, and Stephen E. Fienberg. 2003. A comparison of string distance metrics for name-matching tasks. In *IJCAI Workshop on Information Integration on the Web (IIWeb)*, pages 73–78.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis, and Vassilios S. Verykios. 2007. Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 19(1):1–16.

Meihao Fan, Xiaoyue Han, Ju Fan, Chengliang Chai, Nan Tang, Guoliang Li, and Xiaoyong Du. 2024. Cost-effective in-context learning for entity resolution: A design space exploration. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, pages 3696–3709.

Ivan P. Fellegi and Alan B. Sunter. 1969. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210.

Tao Feng, Zifeng Wang, and Jimeng Sun. 2023. CITING: Large language models create curriculum for instruction tuning. *Computing Research Repository*, arXiv:2310.02527.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aieleen Lakber, Amit Sangani, Amir Barati Farimani, and Ana Bellova. 2024. The llama 3 herd of models. *Computing Research Repository*, arXiv:2407.21783.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of large language models. *Computing Research Repository*, arXiv:2106.09685.

Zezhou Huang. 2024. Disambiguate entity matching using large language models through relation discovery. *Computing Research Repository*, arXiv:2403.17344.

International Organization for Standardization. 2001. Iso 15919:2001—information and documentation: Transliteration of devanagari and related indic scripts into latin characters. https://cdn.standards.iteh.ai/samples/28333/a0a778b7b2034a91aab1e97b3a34d125/ISO-15919-2001.pdf.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. *Computing Research Repository*, arXiv:2309.06180.

Huahang Li, Longyu Feng, Shuangyin Li, Fei Hao, Chen Jason Zhang, and Yuanfeng Song. 2024. On leveraging large language models for enhancing entity resolution: A cost-efficient approach. *Computing Research Repository*, arXiv:2401.03426.

Yifan Li, Cen Qu, Chao Li, and Jia Wang. 2023. Deep learning for entity matching: A survey. *ACM Transactions on Knowledge Discovery from Data*, 17(3).

Arnav Mhaske, Harshit Kedia, Sumanth Doddapaneni, Mitesh M. Khapra, Pratyush Kumar, Rudra Murthy V, and Anoop Kunchukuttan. 2022. Naamapadam: A large-scale named entity annotated data for indic languages. *Computing Research Repository*, arXiv:2212.10168.

Ralph Peeters and Christian Bizer. 2023. Pre-trained language models for entity matching: A survey. *Computing Research Repository*, arXiv:2305.10618.

Ralph Peeters, Aaron Steiner, and Christian Bizer. 2024. Entity matching using large language models. *Computing Research Repository*, arXiv:2310.11244.

Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48.

Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. 2022. Curriculum learning: A survey. *International Journal of Computer Vision*, 130:1526–1565.

Ralf Steinberger, Bruno Pouliquen, Mijail Kabadjov, and Erik van der Goot. 2013. Jrc-names: A freely available, highly multilingual named entity resource. *Computing Research Repository*, arXiv:1309.6162.

Aaron Steiner, Ralph Peeters, and Christian Bizer. 2024. Fine-tuning large language models for entity matching. *Computing Research Repository*, arXiv:2409.08185.

Chunlei Xin, Yaojie Lu, Hongyu Lin, Shuheng Zhou, Huijia Zhu, Weiqiang Wang, Zhongyi Liu, Xianpei Han, and Le Sun. 2024. Beyond full fine-tuning: Harnessing the power of LoRA for multi-task instruction tuning. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2307–2317, Torino, Italia. ELRA and ICCL.