Explaining Entity Matching Models with CREW

Riccardo Benassi, Michele Luca Contalbo, Francesco Del Buono, Francesco Guerra, Giacomo Guiduzzi, Matteo Paganelli*, Sara Pederzoli, Donato Tiano and Maurizio Vincini

University of Modena and Reggio Emilia, Via P. Vivarelli 10, Modena, Italy

Abstract

Deep learning models achieve high performance in Entity Matching tasks, but lack interpretability, limiting user understanding of their decision-making process. Several explainers, such as LIME, Mojito, Landmark, LEMON, and CERTA, have been proposed in the literature to address this issue. However, these methods primarily focus on model fidelity without prioritizing comprehensibility, resulting in explanations difficult to interpret. This extended abstract introduce CREW, a system designed to explain matching decisions. CREW enhances both interpretability and fidelity by grouping words from EM records based on semantic similarity, dataset structure, and their importance to the model. Experimental results demonstrate that CREW produces explanations that are both more interpretable for users and more faithful to the model compared to existing methods.

Keywords

Entity Matching, Data Integration, Explainable AI, Interpretability, Explainer systems

1. Introduction

As data continues to grow and become more distributed across multiple sources, ensuring its consistency and reliability becomes increasingly challenging. Entity Matching (EM) addresses this challenge by detecting duplicate records, which improves data quality and make the data more effective for downstream applications. Current state-of-the-art EM methods rely on deep learning, in particular transformer-based architectures, to achieve high performance [1, 2]. However, these models suffer from a lack of interpretability, making it challenging to understand the rationale behind their decisions and limiting their applicability in real-world scenarios [3, 4, 5, 6]. Explanation systems aim to shed light on these complex models, fostering trust and enabling their use in sensitive domains [7].

The typical approach for explaining EM models is to use local post-hoc methods that build surrogate models to simulate the decision-making process of the model around a specific data

SEBD 2025: 33rd Symposium on Advanced Database Systems, June 16-19, 2025, Ischia, Italy

*Corresponding author.

riccardo.benassi@unimore.it (R. Benassi); micheleluca.contalbo@unimore.it (M. L. Contalbo); francesco.delbuono@unimore.it (F. D. Buono); francesco.guerra@unimore.it (F. Guerra); giacomo.guiduzzi@unimore.it (G. Guiduzzi); matteo.paganelli@unimore.it (M. Paganelli); sara.pederzoli@unimore.it (S. Pederzoli); donato.tiano@unimore.it (D. Tiano); maurizio.vincini@unimore.it (M. Vincini)

0009-0007-4819-259X (R. Benassi); 0009-0008-7526-8534 (M. L. Contalbo); 0000-0003-0024-2563 (F. D. Buono); 0000-0001-6864-568x (F. Guerra); 0000-0003-0819-405X (G. Guiduzzi); 0000-0001-8119-895X (M. Paganelli); 0009-0000-7659-662X (S. Pederzoli); 0000-0003-0605-4184 (D. Tiano); 0000-0001-9262-2939 (M. Vincini)

Approach	Level	Feature corr.	Text	Focus
LIME [8] SHAP [9] Landmark [11] Mojito [4]	Token	✗	✓	Model fidelity
Mojito [4] CERTA [12]	Attr.	Within attribute	✗	
GMASK [13]	Cluster	✓	✓	
CREW (our)	Cluster	✓	✓	Model fidelity Understability

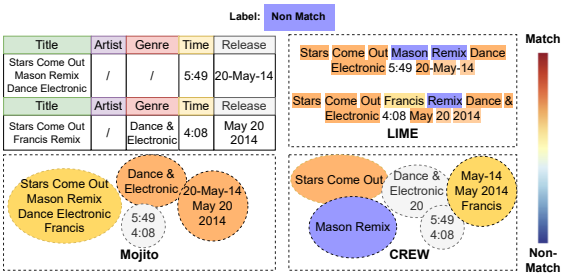


Figure 1: Comparison of existing explanation methods with CREW.

point. Notable examples include LIME [8], SHAP [9], ExplainER [5], Mojito [4], LEMON [10], Landmark Explanation [11], and CERTA [12]. Additionally, other post-hoc methods not originally designed for EM can also be applied. For instance, GMASK [13], which explains NLI models by identifying clusters of correlated words, can be adapted to explain EM models. These methods, as shown in Figure 1, assign feature importances at varying granularities: LIME, SHAP, and Landmark focus on individual tokens, Mojito and CERTA evaluate the impact of attributes, while GMASK computes impacts at the cluster-level.

Despite their usefulness, existing explanation techniques suffer from several limitations. Token-level explanations help identify the most influential tokens in a prediction, however, they tend to be verbose and may oversimplify the EM task, failing to capture complex patterns and dependencies between words within records. In contrast, attribute-level explanations offer a more compact representation but 1) they are unsuitable for textual data, which is prevalent in real-world product description datasets, 2) struggle with noisy data, where misplaced attribute values may cause the contribution of an attribute to rely on semantically unrelated information, and 3) provide only an approximate representation of the model's behavior, where dominant impacts might obscure less obvious but still important impacts within the same attribute. Cluster-level explanations address some of these issues by modeling relationships between words when attributing importance, rather than treating them independently. However, they can still suffer from poor interpretability, as most existing methods focus solely on fidelity to the model rather than ensuring that the explanations are easily understandable for users.

In this paper, an extended abstract of our previous work [14], we explore the use of CREW (Cluster of RElated Words), an explanation technique for entity matching that overcomes the previous limitations by prioritizing the usability and interpretability of explanations. More specifically, this approach generates explanations by grouping words into clusters which are designed to satisfy three key properties: they should contain semantically similar words, align with the structure of entity descriptions when attributes are present, and exhibit distinct contributions to the model's decision. Ensuring semantic coherence within clusters allows users to associate them with specific entity characteristics, while preserving attribute structure maintains the logical organization of the dataset. Additionally, differentiating clusters based on their influence helps users identify the most critical factors driving the model's predictions.

2. Related work

Entity Matching (EM) is predominantly addressed using deep learning models, which have proven highly effective even in noisy scenarios. Early approaches like DeepER [15] and DeepMatcher [16] pioneered the use of deep learning for EM, while state-of-the-art methods such as Ditto [1] and R-SupCon [17] leverage transformer architectures to automatically extract meaningful record representation supporting the matching task. Other EM approaches based on transformer architectures are discussed in [18], and a recent survey on this topic is available in [19]. Despite their high accuracy, these models operate as black boxes, making it difficult to understand the factors influencing their predictions [2, 20, 21].

Interpreting the behavior of EM models is an emerging research problem that is addressed through intrinsically explainable models or post-hoc methods. Intrinsically explainable models generate matching predictions through self-explanatory structures that inherently provide insight into their decision process. To our knowledge, WYM [22] is the only approach in this category. This model adopts a fully interpretable workflow, which involves three main steps: extracting decision units from the input pair of records, assigning relevance scores to each decision unit, and combining these decision units with their associated relevance scores using a white-box machine learning classifier. In contrast, most existing methods for EM explanation rely on post-hoc analysis. Mojito [4], Landmark [11], and LEMON [10] use LIME [8] to estimate token impact but differ in granularity: Mojito provides attribute-level explanations, while Landmark generates dual token-level explanations. LEMON further extends Landmark by producing counterfactual explanations, which provide examples of values that can flip the prediction and whose granularity is automatically determined. CERTA [12] is another post-hoc technique that does not rely on LIME, but instead uses a probabilistic approach to estimate the impact of each token. The method links a pair of records to a third reference record, from which it extracts a minimal set of values. These values are then incorporated into the original record pair to assess how the prediction would change with the added information.

All of the previous methods prioritize model fidelity, aiming to generate explanations that faithfully reflect the underlying model's decision process. In contrast, CREW focuses on enhancing explanations comprehensibility by selecting relevant information in a way that is easy for users to understand matching decisions while maintaining fidelity to the model.

3. The CREW's approach

CREW generates explanations for matching decisions through a two-step approach. First, it clusters words from the target record pair into semantically meaningful groups to enhance user interpretation. Then, it quantifies each cluster's contribution to the model's decision, highlighting those that support a match versus those that indicate a non-match. More specifically, given an EM model $\mathscr{M}$ that labels a word sequence representing a pair of records $p = (w_1, w_2, \dots, w_p)$ as 1 (match) or 0 (non-match), CREW first organizes the words of the pair p into a knowledge graph, which better models the correlations between words. A correlation clustering algorithm is then applied to this graph, producing word clusters $C_1, \dots, C_k$. Each group is then scored based on its impact on the matching decision, i.e., $i_{C_1}, \dots, i_{C_k}$. This operation is performed using a standard

post-hoc local explainer (e.g., LIME). Finally, the clusters and their respective impacts are paired together to generate the final explanation of the target pair p, i.e., $EX_p = \{(C_j, i_j) | j = 1, \dots, k\}$.

3.1. Clustering the words

CREW implements the clustering operation by running an instance of the correlation clustering algorithm [23]. The input of this algorithm is a knowledge graph, which we define as follows.

Definition 1 (Knowledge graph). *We define the knowledge graph as a fully connected, weighted graph $G = (V, E)$, where the vertices are the words of an EM pair, and the edges represent the level of relatedness between the corresponding words. Each edge (i, j) is associated with a label $L_{i,j}$, which can take values $+$ or $-$, depending on whether the connection is a positive or negative evidence of relatedness. We define E^+ and E^- as the sets of edges labeled $+$ and $-$, respectively, i.e., $E^+ = \{(i, j) \mid L_{i,j} = +\}$, and $E^- = \{(i, j) \mid L_{i,j} = -\}$.*

The algorithm aims to identify a partitioning of the words that maximizes the agreement of correlations and minimizes the disagreement of correlations for words belonging to the same cluster. Formally, the problem translates into the identification of a valid assignment of the variable $x_{i,j} \in [0, 1]$ (where 0 indicates that two vertices are included in the same cluster and 1 the opposite case) that minimizes the sum of the negative edges included in a cluster and maximizes the sum of positive edges.

$$\begin{aligned} &\text{minimize} && \sum_{(i.j)\in E^-} E_{w_{i,j}}(1 - x_{i,j}) + \sum_{(i.j)\in E^+} E_{w_{i,j}} x_{i,j} \\ &\text{subject to} && x_{i,j} \in [0, 1], \quad x_{i,j} + x_{j,k} \geq x_{i,k}, \quad x_{i,j} = x_{j,i} \end{aligned}$$

To generate user-friendly explanations, CREW clusters words based on three relationships: semantic relatedness, grouping related terms to highlight key entity aspects; schema relatedness, organizing words according to the dataset schema to reflect the data provider's perspective; and importance relatedness, prioritizing words based on their influence on the model's decision. As specified in These relationships are combined linearly to determine edge weights.

Semantic relatedness. CREW measures semantic relatedness between words using the cosine similarity of their BERT embeddings[1]. These similarity scores are zero-centered (i.e., adjusted by subtracting the mean similarity) and used as the first component in determining edge weights in the graph.

Schema relatedness. CREW incorporates the knowledge of the attribute-based structure of entity descriptions into the graph's edge weights by applying a constant penalty ρ to edges connecting words from different attributes.

Importance relatedness. To encourage clustering of words that contribute similarly to the model's decision, CREW first uses LIME to obtain word-level impact scores. These scores are positive for words that support a matching prediction and negative for those that push toward a non-matching prediction. CREW then compares these impact scores pairwise to derive a measure of importance relatedness. The goal is to assign scores close to 1 for words with

[1] https://huggingface.co/bert-base-uncased

similar impacts reinforcing the same class and close to -1 for words with opposing impacts (i.e., contributing to different classes). Equation 1 defines how the importance relatedness between two words w_1 and w_2 with impact scores i_{w_1} and i_{w_2} is computed in CREW. The sign of their ratio determines whether the words support the same or opposing predictions, ensuring that words with conflicting impacts are not clustered together. The ratio of the smaller to the larger absolute impact quantifies similarity on a scale from 0 to 1. Finally, the average absolute impact scales the correlation, weighting stronger influences more heavily.

$$ImpRel(i_{w_1}, i_{w_2}) = \frac{\text{sign}(i_{w_1})}{\text{sign}(i_{w_2})} \cdot \frac{\min(|i_{w_1}|, |i_{w_2}|)}{\max(|i_{w_1}|, |i_{w_2}|)} \cdot \frac{|i_{w_1}| + |i_{w_2}|}{2} \quad (1)$$

3.2. Weighting the clusters

After generating word clusters, CREW evaluates their impact on the model's decision. To achieve this, CREW relies on LIME, though other explainability methods can be easily adapted. Unlike its previous use for computing importance relatedness, LIME is now applied at the cluster level rather than individual words. To enable this, we introduce two key modifications to LIME's standard workflow. First, all words within a cluster are concatenated using a special character, preventing LIME's tokenizer from splitting them and ensuring they are treated as a single unit during perturbation. Second, once LIME generates perturbed samples, the words are split back before being fed into the EM model, restoring their original format. These adjustments allow LIME to assess cluster-level impact while keeping both LIME itself and the EM model unchanged, requiring only minimal string processing to assign cluster weights.

4. Experimental evaluation

The experimental evaluation aims to demonstrate three complementary properties of CREW: 1) the ability to generate explanations that are easily understandable and intuitive for the users (Section 4.1); 2) the fidelity in explaining the decisions made by a black-box EM model (Section 4.2); and 3) the efficiency in generating explanations (Section 4.3). For additional experiments and the full results of certain evaluations summarized here due to space constraints, we refer readers to our original paper [14].

Datasets. We conducted experiments using the Magellan benchmark datasets[2], a widely recognized reference for Entity Matching. These datasets, organized as entity pairs with shared attributes, fall into three categories: structured, textual, and dirty. Following standard practice in explainable AI, we sampled 100 pairs per dataset, evenly split between matching and non-matching pairs.

Baselines and settings. We compare CREW with LIME [8], Mojito [4], and GMASK [13], each representing a different family of explainability techniques. LIME provides token-level explanations, Mojito generates attribute-level explanations for EM, and GMASK groups correlated words, originally for NLI tasks. Mojito is the most directly comparable, as its weighted attributes align with our weighted word clusters. To align LIME and GMASK with CREW's output, we

[2]https://github.com/anhaidgroup/deepmatcher/blob/master/Datasets.md

applied the OPTICS algorithm [24] to cluster words by importance scores. All explanation systems were tested on the same BERT-based EM models, fine-tuned on the benchmark datasets using a classification layer on top [1, 18]. However, the methods remain model-agnostic and can be applied to other EM models like Ditto [1] or DeepMatcher [16]. We refer interested readers to the original paper [14] for more information about the approach configurations.

4.1. Explanation comprehensibility

This section evaluates whether the explanations generated by CREW and competing approaches are comprehensible and usable for users. To conduct this evaluation, we adopt an LLM-based approach, leveraging the LLM-as-a-judge paradigm to evaluate the quality of the explanations. We refer interested readers to the original paper [14] for a complementary assessment where human-defined key properties are used to assess the explanation comprehensibility.

To perform this experiment, we employ the Pairwise Ranking Prompting (PRP) approach [25] applied to *gpt-4o-mini*, which provides an effective and stable procedure for leveraging an LLM to perform ranking tasks. Additionally, to mitigate the well-documented sensitivity of LLMs to text order [26, 27], each comparison is performed twice, swapping the order of the two elements being compared. In our experiment, this framework was used to identify the explanation system that produces the most effective explanations. Specifically, for each of the 100 record pairs in a given dataset, we prompted the LLM to compare the explanations generated by CREW against those produced by other explanation systems, considering both comparison directions. We instructed the model to penalize explanations that focus on irrelevant attributes, overlook key matching features, or provide inconsistent and unreliable justifications, while prioritizing those that correctly identify key words that explain the matching decision. In total, 12k pairwise comparisons were conducted, and each explanation was assigned a score using the following formula, as defined in the original PRP framework:

$$s_i = 1 \cdot \sum_{j \neq i} \mathbb{I}_{e_i > e_j} + 0.5 \cdot \sum_{j \neq i} \mathbb{I}_{e_i = e_j} \tag{2}$$

where $\mathbb{I}_{e_i > e_j}$ is an indicator function that is 1 if the LLM prefers the first explanation (i.e., e_i) over the second one (i.e., e_j), and $\mathbb{I}_{e_i = e_j}$ is 1 when the LLM provides conflicting outputs.

The results in Figure 2 clearly show that CREW outperforms competing methods, winning an average of approximately 450 comparisons. The second-best performing approach is LIME, which generates explanations preferred, on average, over competing methods about 370 times. Mojito and GMASK are the least performing methods, with an average of 210 and 150 comparisons won, respectively. It is interesting to note that CREW achieves excellent performance on the only textual dataset considered in the evaluation, but performs the worst on the S-FZ and S-BR datasets, which have descriptions with a relatively low average word count (about 20 words). Finally, it is worth highlighting that LIME demonstrates consistent results across all datasets, with the lowest standard deviation.

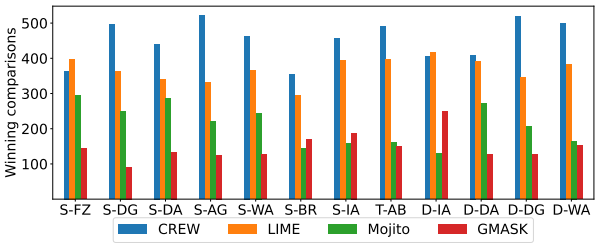


Figure 2: LLM-based evaluation of explanation comprehensibility conducted using *gpt-4o-mini* with the Pairwise Ranking Prompting approach [25].

Table 1
Degradation Test. The method to weight the contribution of each word group is indicated within brackets: Avg=average of token-level impacts, LIME=the impact of the groups in computed with LIME.

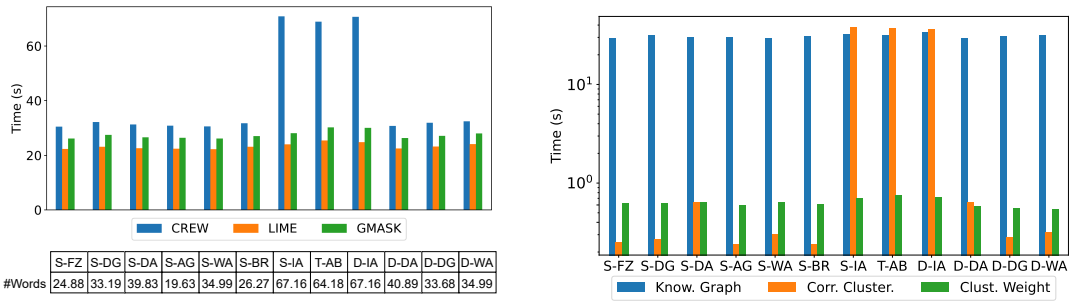
Dataset	CREW (LIME)	CREW (Avg)	LIME (LIME)	LIME (Avg)	Mojito (LIME)	Mojito (Avg)	GMASK (LIME)	GMASK (Avg)
AVG	**0.573**	0.481	0.444	0.397	0.238	0.197	0.314	0.206

4.2. Fidelity of the explanations

This section evaluates how well explanations align with the underlying model using the degradation score, a standard metric that measures how much model accuracy decreases when features, deemed important by the explainer, are removed. A significant accuracy drop indicates that the explanation is faithful. To compute the score, features are removed in order of their impact, both from most to least relevant and vice versa. The area between the two resulting curves, *MoRF* and *LeRF* (Most Relevant / Least Relevant features removed First), represents the degradation score, with a larger area indicating higher trust in the explanation. In our experiments, we implemented this metric using groups of words as features and the F1 score to measure the model's accuracy. The results are shown in columns "LIME" of Table 1, which reports the degradation score for each dataset and explanation system. Table 1 reports a second experiment (columns "Avg") where the impact of a group is computed by averaging the token-level impacts of the words in the group, instead of using LIME. Regardless of the configuration adopted, CREW achieves the best performance, thus demonstrating that it generates explanations more faithful to the underlying model than the competing approaches. LIME achieves the second-best results, obtaining an average degradation score in the range of 0.4-0.44, while Mojito and GMASK generate an average degradation score in the ranges of 0.2-0.24 and 0.2-0.3, respectively. For CREW there is a clear difference in performance between using the "Avg" group weighting technique and using LIME. The latter implementation is preferable regarding model fidelity: it achieves a degradation score of 0.57 compared to 0.48 for the "Avg" configuration.

4.3. Efficiency

We assess the CREW's efficiency by measuring the average explanation generation time, as shown in Figure 3a. We did not report the performance of Mojito, because it is the same as



(a) Overall efficiency (in seconds).

(b) CREW runtime breakdown (log scale).

Figure 3: Comparison of average explanation generation times across explainers, with CREW's performance detailed by knowledge graph creation, correlation clustering, and clustering weighting.

LIME. The results show that CREW is, on average, 10 seconds slower than the other approaches. However, the average time is strongly influenced by three datasets (i.e., S-IA, T-AB, and D-IA) where CREW takes more than double the time required by the other approaches. This is attributed to the length of entity descriptions, which exceeds 60 words on average, as shown at the bottom of Figure 3a. To better understand the reasons for this variation, Figure 3b shows the time breakdown along its three main steps: graph creation, correlation clustering computation and clustering weighting. When the descriptions contain many words, the correlation clustering step has the greatest impact on execution time. If performance is a concern, optimized clustering techniques can be used without altering the rest of the pipeline. We recall that GMASK computes clusters with only 10 words instead of considering all words. However, entity descriptions in our datasets often exceed this threshold, reaching up to 90 words. In [14], we tested CREW with a 40-word limit, significantly reducing execution time for S-IA, T-AB, and D-IA to 26.53, 26.65, and 26.61 seconds, respectively, making it more efficient than GMASK.

5. Conclusion

In this paper, we introduced CREW, a cluster-based explainer for Entity Matching that generates user-interpretable and faithful explanations. CREW follows a two-step approach. First, it clusters words based on three complementary types of knowledge: semantic, schema, and importance relatedness. This selection ensures semantically meaningful clusters with diverse importance levels, helping users identify key information in record matching. Second, CREW quantifies each cluster's contribution to the EM model's prediction using a local post-hoc explainer. Experimental results show that CREW provides more interpretable and faithful explanations than state-of-the-art alternatives.

Declaration on Generative AI

The authors used GPT-4 for translation, grammar and spell checking. The AI-generated content served only as a starting point, with substantial additional work contributed by the authors.

References

[1] Y. Li, J. Li, Y. Suhara, A. Doan, W. Tan, Deep entity matching with pre-trained language models, Proc. VLDB Endow. 14 (2020) 50–60.

[2] M. Paganelli, F. D. Buono, A. Baraldi, F. Guerra, Analyzing how BERT performs entity matching, Proc. VLDB Endow. 15 (2022) 1726–1738.

[3] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, N. Elhadad, Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission, in: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, 2015, pp. 1721–1730.

[4] V. D. Cicco, D. Firmani, N. Koudas, P. Merialdo, D. Srivastava, Interpreting deep learning models for entity resolution: an experience report using LIME, in: aiDM@SIGMOD, ACM, 2019, pp. 8:1–8:4.

[5] A. Ebaid, S. Thirumuruganathan, W. G. Aref, A. Elmagarmid, M. Ouzzani, Explainer: Entity resolution explanations, in: 2019 IEEE 35th International Conference on Data Engineering (ICDE), IEEE, 2019, pp. 2000–2003.

[6] S. Thirumuruganathan, M. Ouzzani, N. Tang, Explaining entity resolution predictions: Where are we and what needs to be done?, in: Proceedings of the Workshop on Human-In-the-Loop Data Analytics, 2019, pp. 1–6.

[7] C. Molnar, Interpretable Machine Learning, 2019. https://christophm.github.io/interpretable-ml-book/.

[8] M. T. Ribeiro, S. Singh, C. Guestrin, " why should i trust you?" explaining the predictions of any classifier, in: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, 2016, pp. 1135–1144.

[9] S. M. Lundberg, S. Lee, A unified approach to interpreting model predictions, in: NIPS, 2017, pp. 4765–4774.

[10] N. Barlaug, Lemon: Explainable entity matching, IEEE Transactions on Knowledge and Data Engineering (2022) 1–16. doi:10.1109/TKDE.2022.3200644.

[11] A. Baraldi, F. D. Buono, M. Paganelli, F. Guerra, Landmark explanation: An explainer for entity matching models, in: CIKM, ACM, 2021, pp. 4680–4684.

[12] T. Teofili, D. Firmani, N. Koudas, V. Martello, P. Merialdo, D. Srivastava, Effective explanations for entity resolution models, in: ICDE, IEEE, 2022, pp. 2709–2721.

[13] H. Chen, S. Feng, J. Ganhotra, H. Wan, C. Gunasekara, S. Joshi, Y. Ji, Explaining neural network predictions on sentence pairs via learning word-group masks, ArXiv abs/2104.04488 (2021). URL: https://api.semanticscholar.org/CorpusID:233204288.

[14] R. Benassi, F. Guerra, M. Paganelli, D. Tiano, Explaining entity matching with clusters of words, in: ICDE, IEEE, 2024, pp. 2325–2337.

[15] M. Ebraheem, S. Thirumuruganathan, S. R. Joty, M. Ouzzani, N. Tang, Distributed representations of tuples for entity resolution, Proc. VLDB Endow. 11 (2018) 1454–1467.

[16] S. Mudgal, H. Li, T. Rekatsinas, A. Doan, Y. Park, G. Krishnan, R. Deep, E. Arcaute, V. Raghavendra, Deep learning for entity matching: A design space exploration, in: Proceedings of the 2018 International Conference on Management of Data, 2018, pp. 19–34.

[17] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, D. Krishnan,

Supervised contrastive learning, Advances in neural information processing systems 33 (2020) 18661–18673.
[18] U. Brunner, K. Stockinger, Entity matching with transformer architectures - A step forward in data integration, in: EDBT, OpenProceedings.org, 2020, pp. 463–473.
[19] N. Barlaug, J. A. Gulla, Neural networks for entity matching: A survey, ACM Trans. Knowl. Discov. Data 15 (2021) 52:1–52:37.
[20] M. Paganelli, D. Tiano, F. Guerra, A multi-facet analysis of bert-based entity matching models, The VLDB Journal (2023) 1–26. doi:10.1007/s00778-023-00824-x.
[21] M. Paganelli, P. Sottovia, A. Maccioni, M. Interlandi, F. Guerra, Explaining data with descriptions, Inf. Syst. 92 (2020) 101549.
[22] A. Baraldi, F. D. Buono, F. Guerra, M. Paganelli, M. Vincini, An intrinsically interpretable entity matching system, in: EDBT, OpenProceedings.org, 2023, pp. 645–657.
[23] E. D. Demaine, D. Emanuel, A. Fiat, N. Immorlica, Correlation clustering in general weighted graphs, Theor. Comput. Sci. 361 (2006) 172–187.
[24] M. Ankerst, M. M. Breunig, H. Kriegel, J. Sander, OPTICS: ordering points to identify the clustering structure, in: SIGMOD Conference, ACM Press, 1999, pp. 49–60.
[25] Z. Qin, R. Jagerman, K. Hui, H. Zhuang, J. Wu, L. Yan, J. Shen, T. Liu, J. Liu, D. Metzler, X. Wang, M. Bendersky, Large language models are effective text rankers with pairwise ranking prompting, in: K. Duh, H. Gomez, S. Bethard (Eds.), Findings of the Association for Computational Linguistics: NAACL 2024, Association for Computational Linguistics, Mexico City, Mexico, 2024, pp. 1504–1518. URL: https://aclanthology.org/2024.findings-naacl.97/. doi:10.18653/v1/2024.findings-naacl.97.
[26] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, P. Liang, Lost in the middle: How language models use long contexts, Transactions of the Association for Computational Linguistics 12 (2024) 157–173. URL: https://aclanthology.org/2024.tacl-1.9/. doi:10.1162/tacl_a_00638.
[27] Y. Lu, M. Bartolo, A. Moore, S. Riedel, P. Stenetorp, Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity, in: S. Muresan, P. Nakov, A. Villavicencio (Eds.), Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 8086–8098. URL: https://aclanthology.org/2022.acl-long.556/. doi:10.18653/v1/2022.acl-long.556.