KCNet: Kernel-based Canonicalization Network for entities in Recruitment Domain

Nidhi Goyal[1], Niharika Sachdeva[2], Anmol Goel[3], Jushaan Singh Kalra[4], and Ponnurangam Kumaraguru[5]*

[1] Indraprastha Institute of Information Technology, New Delhi, India
nidhig@iiitd.ac.in
[2] InfoEdge India Limited, Noida, India
niharika.sachdeva@infoedge.com
[3] Guru Gobind Singh Indraprastha University, Delhi, India
agoel00@gmail.com
[4] Delhi Technological University, Delhi, India
jushaan18@gmail.com
[5] International Institute of Information Technology, Hyderabad, India
pk.guru@iiit.ac.in

Abstract. Online recruitment platforms have abundant user-generated content in the form of job postings, candidate, and company profiles. This content when ingested into Knowledge bases causes redundant, ambiguous, and noisy entities. These multiple (non-standardized) representation of the entities deteriorates the performance of downstream tasks such as job recommender systems, search systems, and question answering. Therefore, making it imperative to canonicalize the entities to improve the performance of such tasks. Recent research discusses either statistical similarity measures or deep learning methods like word-embedding or siamese network-based representations for canonicalization. In this paper, we propose a Kernel-based Canonicalization Network (KCNet) that outperforms all the known statistical and deep learning methods. We also show that the use of side information such as industry type, url of websites, etc. further enhances the performance of the proposed method. Our experiments on 351,600 entities (companies, institutes, skills, and designations) from a popular online recruitment platform demonstrate that the proposed method improves the overall F1-score by 23% compared to the previous baselines, which results in coherent clusters of unique entities.

Keywords: Entity Canonicalization · Recruitment Domain · Entity Normalization

1 Introduction

Recruitment platforms such as LinkedIn, Indeed.com ingest an enormous amount of user-generated content in form of job postings, CVs, and company profiles.

* Major part of this work was done while Ponnurangam Kumaraguru was a faculty at IIIT-Delhi.

This content includes diverse set of recruitment domain entities (company names, institute names, skills, designations) that become part of Knowledge base. As the content is user-generated, multiple variations (e.g., *'economictimes.com'; 'eco. times'; 'the economic times'; 'economic times'; 'ET'*) of each entity name also come up into the KBs. Employing these noisy, redundant, and ambiguous variations directly into downstream applications such as semantic search, question answering, and recommender systems results in poor system performance. Therefore, canonicalization of the entities i.e., mapping various references of a unique entity into a representative cluster, is imperative for recruitment platforms.

Canonicalizing named entities involves various challenges including spelling mistakes and variations (*java developer* & *java deveoper*), overlapping but different entities (*Emerald Bikes pvt limited* & *Emerald Jewellery Retail Limited*), hierarchical variations (*Oracle Financial Services Software* & *Oracle Corporation*), domain-specific concepts (*SOAP* & *REST*), short forms (*umbc* & *University of Maryland, Baltimore*), and semantically similar variations (*Accel Frontline* & *Inspirisys*).

Previous approaches focus on statistical methods [5] for entity canonicalization. However, these methods use handcrafted features that are unable to scale well for advanced (semantic, domain-specific) variations of entities. Fatma et al. [4] employ a deep learning method that overcomes challenges of statistical methods by eliminating the need for explicit feature engineering and using character-based word-embeddings for unknown and emerging entities. Recent literature [9] shows that deep learning methods are often very good at minimizing the training errors but fail to generalize. Literature [9] suggests the introduction of learnable kernels in deep neural networks often improves generalizability.

Therefore, we study a kernel-based neural network designed for entity canonicalization in the recruitment domain. Our proposed method outperforms all the known statistical and deep learning methods on canonicalization tasks. We further enhance the performance of the kernel-based network using side-information which is underexplored in the literature. This literature suggests the use of external side information (morphological, IDF token overlap, PPDB [17]) which is rudimentary [21] and has limited utility in domain-specific settings. We leverage more prosperous meta and semantic side information from external sources (Wikipedia, Google KG) [22,10] to improve the entity canonicalization.

In this paper, we propose a novel multi-tier framework using a learnable kernel network [7,9] which implicitly maps the data into high-dimensional feature space. Our framework captures the non-linear mapping between contextual, meta, and semantic representations through learning objective to output the pairwise similarity between recruitment domain entities. Furthermore, we generate the canonicalized clusters for each entity. We demonstrate and validate the efficacy of our approach on proprietary as well as open source datasets including DBpedia and ESCO [1,3] for generalizability of our solution. We summarize the main contributions of *this* paper as follows:

- We propose a Kernel-based Canonicalization Network (KCNet), which induces a non-linear mapping between the contextual vector representations

while capturing fine-granular and high-dimensional relationships among vectors. To the best of our knowledge, this is the first approach towards exploring kernel features for canonicalizing Knowledge Base entities in the recruitment domain.
- KCNet efficiently models more prosperous semantic and meta side information from external knowledge sources to canonicalize domain-specific entities.
- We perform extensive experiments on real-world proprietary and publicly available datasets in the recruitment domain to show the effectiveness of our proposed approach as compared to baselines.

The organization of the rest of the paper is as follows: Section 2 contains related works; Section 3 elaborates our proposed framework KCNet; Section 4 reports the datasets. Section 5 describes the experimental setup, Section 6 has results and discussion followed by conclusion in Section 7.

2 Related Works

This section briefly describes some of the related works on KB Canonicalization, domain-specific methods, kernel methods, and clustering.
KB Canonicalization. Existing work [5] use manually defined feature spaces to perform the canonicalization task. This approach encodes limited similarity between different semantic representations. Hence, it results in degradation of performance for real-time applications. Vashishth et al. [21] jointly handle noun and relation phrases using knowledge graph embedding models [16] by optimizing its objective function along with using information from external sources called *'side information'*. However, these state-of-the-art knowledge graph embedding methods [2] achieve below par performance for real-world recruitment domain datasets due to noisy, sparseness [6], and context-sensitive information present in triples. Additionally, the side information methods used in literature [21] is rudimentary and lack domain-specific information. Considering these limitations, we leverage external knowledge sources such as Wikipedia Infobox and Google search API which provides additional knowledge for noisy entities.
Domain-Specific methods. Despite the importance of named-entity canonicalization in the recruitment domain, only a few recent studies have explored the problem with respect to unique domain challenges [12,13]. Yan et al. [24] propose a company name normalization system that employs LinkedIn social graphs and a binary classification approach. In this work, the authors use complete profile information as the context. However, this information will be hard to get for new and emerging entities. Lin et al. [11] uses side information and learns domain knowledge from the source text based on the type of entities. Popular state-of-the-art entity linking tools [14] are probabilistic and requires sufficient contextual information to connect to candidate entity and perform well when standard surface forms are available. For example, recruitment domain-specific documents may contain *'Python'* or *'Python Programming'* while, the former is linked to a different type of entity with a high confidence score using these tools. Similarly, Fatma et al. [4] utilizes word and character-based representations based

similarity model along with the attention mechanism to cluster similar entities. However, these works fail to generalize and have limited understanding for more complex and emerging entities.

Kernel-based architectures. Kernel methods have proven effective in exploring larger feature space implicitly in deep learning architectures [23]. Customized kernel [8] based deep learning architectures enhance the performance of the model and map data to an optimized high-level feature space where data may have desirable features toward the application. Recent works utilize deep embedding kernel architectures for identity detection, transfer learning, classification and other tasks [9]. We use kernel infused neural networks to capture the latent semantic relationships and non-linearity between different pair of entities in KBs. These kernel methods are robust for collaboration with neural networks and less expensive than training deep learning architectures.

Clustering methods. Research works have used various clustering techniques for the canonicalization task. Among these methods, Hierarchical Agglomerative Clustering (HAC) is the most extensively used in the literature [5,21].

Our research is uniquely placed at the intersection of the vast literature on kernel-based neural network learning, and clustering approaches for the canonicalization of domain-specific KBs.

3 Kernel-based Canonicalization Network (KCNet)

In this section, we introduce the proposed KCNet approach. We elaborate on the problem definition and each component of our network architecture in detail.

3.1 Problem Definition

Consider $\mathcal{E}$ be the set of entities extracted from job postings, CVs, and company profiles. For each entity x_i, we consider its side information $s_i \in \mathcal{S}\ \forall x_i \in \mathcal{E}$ acquired from heterogeneous sources (elaborated in detail in section 4.2). Given a pair of entities x_1 and x_2 and their corresponding side information s_1 and s_2 where $x_1, x_2 \in \mathcal{E}$ and $s_1, s_2 \in \mathcal{S}$, the main objective is to find a function $f(x_1, s_1, x_2, s_2) \rightarrow sim(x_1, x_2)$. A pairwise similarity matrix ($\mathcal{M}_{sim}$) is formed by applying f over the set of all entity pairs and then a clustering algorithm is used to form unique canonical clusters of similar entities.

3.2 Network Architecture

We propose a multi-tier novel architecture consisting of three modules: Entity embedding generation, Side Information embedding generation, and Kernel network. We apply clustering technique after obtaining output on our proposed architecture. Fig. 1 shows the overall architecture of our proposed approach (KCNet).

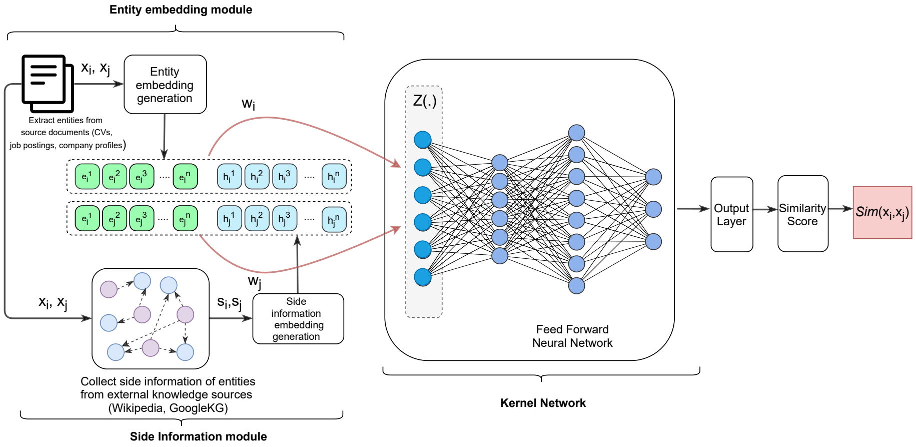
Entity embedding module
x_i, x_j
Entity embedding generation
Extract entities from source documents (CVs, job postings, company profiles)
w_i
e_i^1 e_i^2 e_i^3 .... e_i^n h_i^1 h_i^2 h_i^3 .... h_i^n
e_j^1 e_j^2 e_j^3 .... e_j^n h_j^1 h_j^2 h_j^3 .... h_j^n
w_j
x_i, x_j
s_i,s_j
Side information embedding generation
Collect side information of entities from external knowledge sources (Wikipedia, GoogleKG)
Side Information module
Z(.)
Feed Forward Neural Network
Kernel Network
Output Layer
Similarity Score
$Sim(x_i,x_j)$

Fig. 1. Kernel-based Canonicalization Network for entities in recruitment domain. We first extract entities and combine it with side information. The combination (concat) is passed through the Kernel network. The output is a pairwise similarity matrix.

- ***Entity embedding generation.*** We obtain an m-dimensional (m=768) vector for each entity pair (x_i, x_j) producing (e_i, e_j) in the space $C \in \mathcal{R}^m$. We use a pre-trained distilled version (fewer parameters, less space and time complexity) of S-BERT [19] to generate initial contextual† [6] embeddings for all entities.
- ***Side Information embedding generation.*** We represent (h_i, h_j) as n-dimensional vector (n=768) for the side information acquired for each entity pair (x_i, x_j) in the space $H \in \mathcal{R}^n$. The formation of side information vector is described in Section 4.2. These representations h_i and h_j are concatenated with the corresponding entity representations e_i and e_j to obtain infused vector representations w_i and w_j for the input pair (x_i, x_j). Here, $w_i = e_i \odot h_i$ and $w_j = e_j \odot h_j$, Note that, $\odot$ is the concatenation function of two vectors producing (w_i,w_j) in the space $W \in \mathcal{R}^{m+n}$. The (m+n) -dimensional vector representation is fed into the kernel network to learn the similarity function, $sim(x_i,x_j)$.
- ***Kernel Network.*** We introduce a kernel network to learn the similarity function f to model complex relationships between the data representations of input pairs in an optimized space. The input to this network is denoted as Z, formed by the combined representation w in the equation (1).

$$Z = (w_i \circ w_j) \odot |w_i - w_j| \tag{1}$$

Here, $\circ$ is a Hadamard (element-wise) product which exploits interactions between two vectors at each dimension. We also determine the $\mathcal{L}_1$ distance

[6] $\dagger$ specifies that an entity name such as '*University of Maryland, Baltimore*' contains the location specific context i.e. '*Baltimore*'. The representation of the entire entity is termed as contextual embedding.

for each dimension w_i and w_j and concatenate the interactions of both the components as shown in equation (2).

$$Z = \Big\{ w_i^1 * w_j^1, \ldots, w_i^{m+n} * w_j^{m+n}, |w_i^1 - w_j^1|, \ldots, |w_i^{m+n} - w_j^{m+n}| \Big\} \qquad (2)$$

where w_i^k represents the k^{th} dimension of w_i. The dimensionality of Z is $2*(m+n)$. Kernel function takes in account both element-wise product (design of polynomial kernel) and differences (design of RBF kernel) over each dimension of original entity. This configuration of inputs allows the network to learn a non-linear relationship between the input pairs and symmetric representations at a fine granular level over each input dimension. Therefore, the learned kernel can map a more robust similarity function over the input space in comparison to traditional methods such as RBF and polynomial [8]. Similar observations for the customized kernel have been made in [9]. The newly obtained vector Z captures the latent semantic relationships between the two input entities. This vector is fed into a multi-layer feed-forward neural network with *sigmoid* output, facilitating the learning of a highly non-linear mapping f to predict similarity over entity pairs. The size of hidden layers (number of neurons) in the kernel network is chosen using a hyperparameter k. We define $k = \alpha * d$ where d is the dimensionality of Z and typically, $\alpha = \big\{1, 2, 3\big\}$, say, $f(x_i, x_j) = f(x_j, x_i) > 0$. The kernel network outputs the probability that input pair (x_i, x_j) belong to the same cluster. Therefore,

$$f(x_i, x_j) = P(y^i = y^j | x_i, x_j) \qquad (3)$$

- ***Clustering using pairwise similarity scores.*** We compute the pairwise similarity matrix $\mathcal{M}_{sim}$ for all the entity pairs (x_i, x_j) using probability obtained from previous step and apply Hierarchical Agglomerative Clustering (HAC) to form a unique cluster of entities. HAC is popular technique used in literature [5,21] for canonicalization tasks. Each entity is finally mapped to a unique cluster. We choose the number of clusters k using the silhouette index [20]. We repeat the same process for all the datasets (skills, designations, institutes, companies).

4 Datasets

In this section, we describe our datasets and side information collection process in detail.

4.1 Dataset Description

Proprietary Datasets. We use real-world datasets from one of the largest recruitment platforms in India. The dataset i.e., Recruitment Domain Entities (RDE) consists of 25,602 company clusters (RDE (C)), 23,690 institute clusters (RDE (I)), 607 skill clusters (RDE (S)), and 3,894 designation clusters (RDE (D)). The ground truth clusters are manually annotated by domain experts with a kappa agreement of 0.83. Next, we generate the variation pairs- positive and negative samples. Each name variation of entity $e_x \in \mathcal{E}$ is defined as $\{e_x^1, e_x^2, e_x^3, \dots e_x^n\}$, which belong to same annotated cluster. We remove Non-ASCII characters to form a set of all unique name variations of e_x. For each entity pair, (e_x^i, e_y^j), training data is prepared using the mapping function g, such that, $g(e_x^i, e_y^j) = 1, \forall(x, y)$ where $i \neq j$ and $x = y$ belongs to same annotated cluster (positive pairs). Similarly, $g(e_x^i, e_y^j) = 0$ where $x \neq y$ belongs to different clusters (negative pairs) using a random sampling approach [15].

Open Datasets. We test the effectiveness of our framework (KCNet) using open-source datasets i.e., DBpedia(C) and ESCO. DBpedia(C) [4] dataset is prepared by querying DBpedia for company names to extract *dbo:Company* which contains 2,944 entity clusters and 22,829 variation pairs. ESCO [3] i.e., ESCO(S) and ESCO(D) are open-source recruitment domain datasets for ESCO-skills and ESCO-designations. ESCO(S) has 35,554 variation pairs and 2,644 clusters of ESCO-skills. ESCO(D) has 62,969 variation pairs and 2,903 clusters. Authors [4] prepared and released these datasets for research community.

4.2 Side information Collection

We leverage two sources for side information extraction, *Wikipedia infoboxes* and *Google KG*.

Wikipedia infobox. We query Wikipedia using its advanced search options [7] and extracted knowledge from Wikipedia infoboxes for different datasets such as {*'title_wikis', 'websites', 'types'*} for RDE(S); {*'Names', 'websites', 'title_wikis'*} for RDE(D); {*'Names', 'websites', 'affiliation'*} for RDE(I); {*'Names', 'websites', 'title_wikis','types'*} for ESCO(S); {*'Names', 'websites', 'title_wikis'*} for ESCO(D); {*'types', 'industries', 'websites', 'native names', 'title_wikis'*} for DBpedia(C).

Google KG. For some entities with short forms, noisy variations, etc. we are unable to fetch knowledge using Wikipedia search; therefore, we leverage Google KG Search API [8] to extract rich semantic textual descriptions of entities to supplement the model with semantic knowledge. Other attributes such as {*location*, *type*, *established*} are extracted that form a part of meta knowledge. Finally, we combine the side information extracted from Google KG and Wikipedia infoboxes. For example, an entity *'vb script'* and its combined side information is defined as *'descriptions'*:'VBScript is an Active Scripting language . . . advanced programming constructs'; *'title_wikis'*:'VBScript';*'websites'* docs.microsoft.com/en-us/previous-versions/t0aew7h6}. We create side information embeddings s_i, a concatenated sequence of side information vector representations $\{s_i^1, s_i^2, \dots s_i^p\}$, where p is the number of attributes obtained from external sources. We generate the side information embeddings using a pre-trained distilled version of S-BERT [19] model. We follow the same process across all the entity types.

[7] https://www.mediawiki.org/wiki/MediaWiki

[8] https://serpapi.com/

5 Experimental Setup

In this section, we describe our baselines, model configurations, and evaluation metrics.
Baselines. We compare our approach against the following methods:
Galarraga-IDF. Authors [5] uses AMIE algorithm and handcrafted features to find the similarity between entity e_x and entity e_y. We utilize the weighted word overlap approach as a baseline method.
Entity embeddings (Distilled S-BERT(*)) +cosine. We generate our entity embeddings (see Section 3.2) to obtain the vector representation for the entity pair (x_i, x_j). Instead of using the next module, i.e. Kernel Network, we apply cosine similarity measure to get a pairwise similarity matrix.
Entity and Side information embeddings (Distilled S-BERT(**)) +cosine. We obtain entity embedding of (x_i, x_j) and side information embedding of (s_i, s_j) to get (w_i, w_j) (see Section 3.2). The pairwise similarity matrix is generated using cosine similarity.
Char-BiLSTM+A. Fatma et al. [4] describe the architecture which utilize a siamese network that takes characters as input and passes it through the pair of BiLSTM layers enhanced by the attention layer.
Word-BiLSTM+A. This baseline modifies the previous method (Char-BiLSTM+A) [4] by replacing character-based representations with word-based representations followed by attention layer.
Char-BiLSTM+A+Word+A. Authors [4] combine Char-BiLSTM+A and Word-BiLSTM+A architectures combining word and character representations followed by attention mechanism.
Model Configurations. We learn pairwise similarity models using the proposed architecture for different datasets (companies, institutes, designations, skills). The training and testing dataset split is taken as *(80, 20)*. The optimal value of hyperparameters (*size of hidden layer*, α) for companies, designations, and skills is (*1536, 2*) whereas for institutes, (*768, 2*). Batch-size is *512* and the number of fully connected layers are *3*. Rectified linear units ($ReLU$) is used as activation function and dropout rate is 0.3. *Binary cross-entropy loss* and *Adam* as an optimizer is used to train the kernel network and learn the pairwise similarity matrix ($\mathcal{M}_{sim}$).
Evaluation metrics. For pairwise similarity results (Table 1), we use *Precision* (P), *Recall* (R) and *F1-score* (F) [18]. We evaluate clusters (see Table 2) using *Micro Precision*, *Macro Precision*, *Micro Recall*, and *Macro Recall* used in the literature [21].

6 Results and Discussion

Pairwise Similarity. Table 1 summarizes the test results of the pairwise similarity of KCNet along with other baseline approaches. We observe that *Galarraga IDF* (a weighted word overlap similarity measure) and the entity embeddings generated using *Distilled S-BERT(*)+cosine* results in low pairwise similarity

Table 1. Test Results of pairwise similarity using our proposed model in comparison with different baselines. Here S, D, I, C refers to Skills, Designations, Institutes, and Companies datasets respectively. Results of † are taken from [4]. P and F refers to Precision and F1-scores. Distilled S-BERT (*, **) refers to (entity, entity ⊙ side information) embedding using distilled S-BERT model.

Model	**Performance**													
	Proprietary								**Open**					
	S		**D**		**I**		**C**		**ESCO(S)**		**ESCO(D)**		**DBpedia(C)**	
	P	F	P	F	P	F	P	F	P	F	P	F	P	F
Galarraga-IDF†	33.2	12.5	63.0	60.3	64.3	66.5	75.8	71.2	50.8	32.8	61.7	38.9	22.6	23.6
Distilled S-BERT(*)+cosine	47.8	47.5	49.7	48.8	49.7	49.1	49.2	49.1	49.3	44.4	49.3	39.0	49.6	45.3
Distilled S-BERT(**)+ cosine	47.5	48.8	49.8	49.9	34.6	41.5	56.2	48.4	49.5	50.0	49.4	49.7	50.0	49.8
CharBiLSTM+A†	81.8	86.9	72.6	77.2	84.5	84.8	99.3	98.9	85.9	86.9	76.3	75.1	72.1	59.7
WordBiLSTM+A†	80.1	86.5	90.5	94.8	80.6	83.3	95.3	95.6	85.6	89.6	83.1	83.7	77.6	70.7
CharBiLSTM+A+Word+A†	82.7	88.5	94.4	96.3	86.7	86.7	99.5	99.2	87.3	90.7	84.2	85.4	78.0	71.3
KCNet (without sideinfo)	**96.7**	**90.6**	**99.6**	90.9	**92.4**	**89.3**	99.4	98.8	**99.0**	**95.1**	**98.8**	**86.9**	**99.0**	**92.5**
KCNet (with sideinfo)	**99.5**	**99.4**	**99.7**	**99.6**	**99.5**	**99.5**	**99.5**	**99.3**	**99.2**	**98.3**	**98.8**	**89.4**	**99.1**	**97.0**

for non-overlapping variations and different surface forms of entities. For eg. (*'mdx'*,*'MultiDimensional eXpressions'*) has similarity of 0.73 using Distilled S-BERT(*)+cosine. *KCNet* gives the similarity of 0.84 as it learns the structure and non-linear mapping in latent space, even in the absence of side information. With side information, *KCNet* learns the latent semantic relationships between these two entities and returns a high similarity score of 0.99. Another example is overlapping variations (*uplholstery fillings, upholstery paddings*); *Distilled S-BERT(*)+cosine* returns a pairwise similarity score of 0.86 for these same entities, whereas *KCNet* learns a better representation and gives a pairwise similarity score of 0.99. *KCNet* generalizes well across all entity types, it gives higher P and F even for all open datasets where it outperforms with 21% F1-score as compared to best baseline.

Clustering results. Test results after applying the clustering approach is reported in Table 2. Overall, *KCNet* significantly outperforms the best baseline [4] by an improved micro F1-score by 23% and macro F1-score by 25%. A one-way repeated measures ANOVA test was conducted to determine significance for all evaluation metrics ($p < 0.00003$).

Side information for KCNet. We evaluate the performances of different versions of *KCNet* (with and without side info). From Table 1, we observe that P and F performance benefits from increased performance in the presence of side information. *Char-BiLSTM+A+Word+A* captures limited patterns and unable to model similar semantic variations (*mycology, fungi studies*) for which *KCNet* gives a pairwise similarity score of 0.98. This shows that *KCNet* is able to model these variations well when supplemented with side-information. Even though side information might be unavailable for a some entities, the proposed framework results in overall better entity canonicalization.

Error analysis: Although KCNet gives promising results across all datasets, it wrongly clusters some entities; for example, some skills such as *bees wax* and *natural wax* signify same concept but occur in the different cluster. One possible reason could be that the representation of words *bees* and *natural* are far

Table 2. Test Results over HAC using pairwise similarity. Here, β: baseline (*Char-BiLSTM+A+Word+A*) and γ: proposed model (*KCNet*) with sideinfo.

Dataset	**Model**	**Metrics**					
		Micro			**Macro**		
		P	R	F	P	R	F
S	β	0.71	0.64	0.67	0.94	0.31	0.47
	γ	**0.99**	**0.97**	**0.98**	**0.96**	**0.97**	**0.96**
D	β	0.95	0.53	0.67	0.83	0.15	0.24
	γ	0.86	**0.78**	**0.82**	**0.85**	**0.54**	**0.66**
I	β	0.84	0.75	0.79	0.96	0.48	0.64
	γ	0.83	**0.85**	**0.84**	0.74	**0.71**	**0.72**
C	β	0.98	0.99	0.98	0.97	0.96	0.96
	γ	0.98	0.97	**0.98**	**0.98**	**0.97**	**0.98**
ESCO(S)	β	0.84	0.82	0.83	0.65	0.49	0.55
	γ	**0.93**	**0.92**	**0.92**	**0.89**	**0.75**	**0.81**
ESCO(D)	β	0.49	0.79	0.61	0.21	0.32	0.25
	γ	**0.91**	0.61	**0.73**	**0.81**	0.22	**0.34**
DBpedia(C)	β	0.88	0.52	0.65	0.92	0.25	0.39
	γ	**0.93**	**0.75**	**0.83**	0.86	**0.60**	**0.70**

apart in the contextual vector representation space, so the model assigns a lower similarity score and hence, incorrectly classifies it. Similarly, *'packager'* is incorrectly placed in cluster of [*'dozer driver'*, *'dozer/crawler driver'*, *'packager'*]. The possible reason could be the complete absence of side information for three entities confuses KCNet with closer contextual vector representations. Despite this, KCNet addresses the challenge of handling abbreviations, short forms, and non-overlapping entities by learning vector representations of these entities in the kernel space.

7 Conclusion

Our research focused upon canonicalizing real-world entities from the recruitment domain such as companies, designations, institutes, and skills by designing a novel multi-tier framework Kernel-based Canonicalization Network (KCNet). KCNet induces a non-linear mapping between the contextual vector representations while capturing fine-granular and high-dimensional relationships among vectors. KCNet efficiently models more prosperous semantic and meta side information from external knowledge towards exploring kernel features for canonicalizing entities in the recruitment domain. Furthermore, we applied Hierarchical Agglomerative Clustering (HAC) using the pairwise similarity matrix $\mathcal{M}_{sim}$ to create unique clusters of entities. Experiments revealed that the Kernel-based neural network approach achieves significantly higher performance on both proprietary and open datasets. We demonstrate that our proposed methods are also generalizable to domain-specific entities in similar scenarios.

Acknowledgements.

We would like to acknowledge the support from SERB, InfoEdge India Limited, and FICCI. We are grateful to PreCog Research Group and Dr. Siddartha Asthana for critically reviewing the manuscript and stimulating discussions.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: Dbpedia: A nucleus for a web of open data. In: The semantic web, pp. 722–735. Springer (2007)
2. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in neural information processing systems (2013)
3. European Commission: ESCO handbook. EU publications (2019)
4. Fatma, N., Choudhary, V., Sachdeva, N., Rajput, N.: Canonicalizing knowledge bases for recruitment domain. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 500–513. Springer (2020)
5. Galárraga, L., Heitz, G., Murphy, K., Suchanek, F.M.: Canonicalizing open knowledge bases. In: Proceedings of the 23rd acm international conference on conference on information and knowledge management. pp. 1679–1688 (2014)
6. Gupta, S., Kenkre, S., Talukdar, P.: Care: Open knowledge graph embeddings. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 378–388 (2019)
7. Hofmann, T., Schölkopf, B., Smola, A.J.: Kernel methods in machine learning. The annals of statistics pp. 1171–1220 (2008)
8. Kuo, B.C., Ho, H.H., Li, C.H., Hung, C.C., Taur, J.S.: A kernel-based feature selection method for svm with rbf kernel for hyperspectral image classification. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing **7**(1), 317–326 (2013)
9. Le, L., Xie, Y.: Deep embedding kernel. Neurocomputing **339**, 292–302 (2019)
10. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al.: Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. Semantic Web **6**(2), 167–195 (2015)
11. Lin, X., Chen, L.: Canonicalization of open knowledge bases with side information from the source text. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE). pp. 950–961. IEEE (2019)
12. Liu, Q., Javed, F., Dave, V.S., Joshi, A.: Supporting employer name normalization at both entity and cluster level. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1883–1892 (2017)
13. Liu, Q., Javed, F., Mcnair, M.: Companydepot: Employer name normalization in the online recruitment industry. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 521–530 (2016)
14. Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: Dbpedia spotlight: shedding light on the web of documents. In: Proceedings of the 7th international conference on semantic systems. pp. 1–8 (2011)

15. Neculoiu, P., Versteegh, M., Rotaru, M.: Learning text similarity with siamese recurrent networks. In: Proceedings of the 1st Workshop on Representation Learning for NLP. pp. 148–157 (2016)
16. Nickel, M., Rosasco, L., Poggio, T.A., et al.: Holographic embeddings of knowledge graphs. In: AAAI. vol. 2, pp. 3–2 (2016)
17. Pavlick, E., Rastogi, P., Ganitkevitch, J., Van Durme, B., Callison-Burch, C.: Ppdb 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers). pp. 425–430 (2015)
18. Raghavan, V., Bollmann, P., Jung, G.S.: A critical investigation of recall and precision as measures of retrieval system performance. ACM Transactions on Information Systems (TOIS) **7**(3), 205–229 (1989)
19. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 3982–3992. Association for Computational Linguistics, Hong Kong, China (Nov 2019)
20. Starczewski, A., Krzyżak, A.: Performance evaluation of the silhouette index. In: International Conference on Artificial Intelligence and Soft Computing. pp. 49–58. Springer (2015)
21. Vashishth, S., Jain, P., Talukdar, P.: CESI: Canonicalizing open knowledge bases using embeddings and side information. In: Proceedings of the 2018 World Wide Web Conference. pp. 1317–1327. WWW '18, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland (2018)
22. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Communications of the ACM **57**(10), 78–85 (2014)
23. Weston, J., Ratle, F., Mobahi, H., Collobert, R.: Deep learning via semi-supervised embedding. In: Neural networks: Tricks of the trade, pp. 639–655. Springer (2012)
24. Yan, B., Bajaj, L., Bhasin, A.: Entity resolution using social graphs for business applications. In: 2011 International Conference on Advances in Social Networks Analysis and Mining. pp. 220–227. IEEE (2011)