Bull. Nov. Comp. Center, Comp. Science, 49 (2025), 1-12

Entity alignment experiments on Russian-English dataset with unmatchable entities

Z.V. Apanovich, D.G. Kernogo

Abstract. In recent years, interest in knowledge graphs (KG) has increased exponentially in both scientific and industrial communities. The KGs play an important role in the AI applications such as natural language processing including question-answering systems, recommender systems, and search engines. Integration of different KGs is one of the most pressing problems and is used, for example, to develop complex digital twins of industrial systems. One of the components of the KG integration problem is the entity alignment (EA) problem, which attempts to identify entities in different KGs describing the same real-world object. A special case of this problem is the problem of cross-language entity alignment, which is closely related to the problem of import substitution, such as finding equivalent drugs, spare parts, or devices for the Internet of Things. Unfortunately, in real KGs, many entities may have no equivalents in other KGs. This paper describes entity alignment experiments using the example of a Russian-English dataset with unmatchable entities.

Keywords: knowledge graph, entity alignment, unmatchable entities

1. Introduction

Knowledge graphs store facts about real-world objects as relational and literal triples. A relational triple represents a relationship between two entities (that is, real-world objects with unique Internationalized Resource Identifiers, IRIs) and has the form *tr_r* = (*subject entity, relation, object entity).* A literal triple stores information about entity attributes and has the form *tr_l* = (*subject entity, attribute, literal value*).

Figure 1 shows fragments from the English-language and Russian-language versions of DBpedia describing Novosibirsk State University. An example of a relational triple is (*Novosibirsk_State_University, rector, Fedoruk,_Mikhail_Petrovich*), and an example of a literal triple is (*Novosibirsk_State_University, established, 1959*). In Figure 1, red lines indicate the *owl:sameAs* equivalence relations linking entities between the Russian and English knowledge graphs. It can be seen that the English entity *dbr:Fedoruk,_Mikhail_Petrovich* in the English knowledge graph corresponds to the Russian entity *М._П._Федорук*, the English predicate *dbo:rector* corresponds to the Russian predicate *ректор*, and the English entity *dbr:Novosibirsk_State_University* corresponds to the Russian entity *Новосибирский государственный университет*. Of course, in ideal scenario, the English-language and Russian-language lists of staff, rectors, and students of Novosibirsk State University should coincide across the two knowledge graphs. In practice, however, these lists are often incomplete and may differ substantially, with some discrepancies between the descriptions of equivalent entities.

First of all, note the different spellings of names in the Russian-language and English-language versions of the knowledge graphs. Also, the two versions specify different

founding years for the university. In addition, the versions differ in the lists of people associated with the NSU (the people who studied or worked there). For example, both versions mention the NSU graduate Efim Zelmanov, a Fields Medal recipient in mathematics. However, only the English version includes information about the NSU graduate Andrei Voronkov, recipient of the Herbrand Award (*dbr:Herbrand_Award*) and a developer of the EasyChair system (dbr: EasyChair), which is widely used by researchers worldwide for conference paper submissions. Since descriptions of real-world objects such as *Andrei Voronkov* and the *Herbrand Award* were missing from the Russian DBpedia at the time of the writing, attempts to align the English and Russian versions produce so-called *dangling* or *unmatchable* entities *dbr:Andrei_Voronkov* и *dbr:Herbrand_Award*.

It can be seen that the most complete description of an entity can be obtained by merging all the triples that describe the same entity across different knowledge graphs. However, to accomplish this, one should establish correctly links between the equivalent entities.

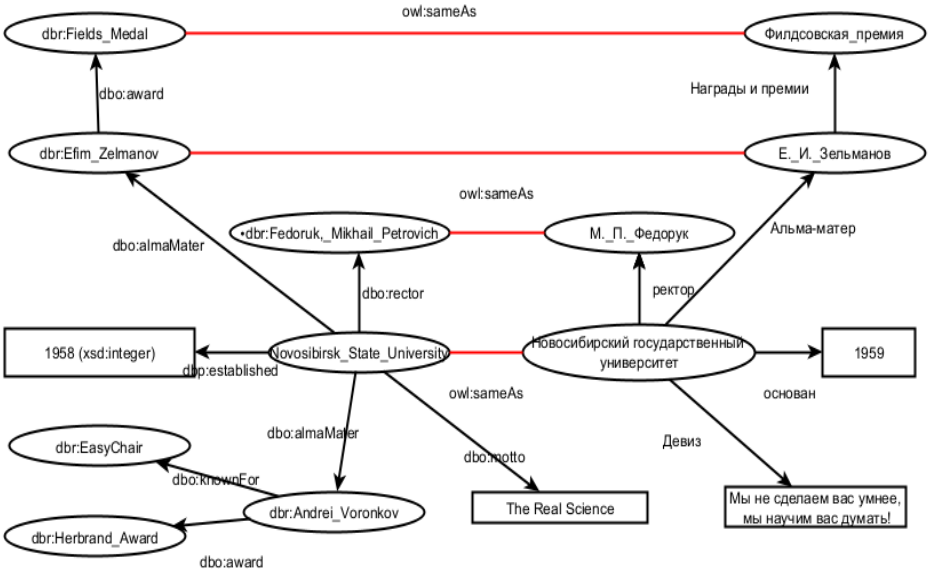

Figure 1. Correspondence between the equivalent entities in the English-language and Russian-language knowledge graphs

Modern entity alignment (EA) methods are based on the assumption that equivalent entities have similar neighborhoods, which explains a wide use of *representation learning* methods. These methods generate so-called *embeddings* of a given dimensionality for entities and relations in knowledge graphs. The advantages of the embedding-based methods are their high scalability and minimal effort required to prepare training datasets. In the following, the embedding of an entity *e* will be represented as **e.**

Clearly, Russian-speaking users are primarily interested in experiments with the Russian-language data. A Russian-English dataset for experiments with cross-language

entity alignment algorithms is described in [1]. The features of the Russian-English dataset are shown in Table 1.

Table 1. Features of the Russian-English dataset

KG language	Number of entities	Number of relations	Number of attributes	Number of relational triples	Number of attributive triples
Ru	15000	66	15018	30489	54499
En	15000	163	15106	43796	76852

Paper [2] shows that the quality of entity alignment can be improved significantly by improving the quality of the embeddings of entity names. Additionally, optimal combinations of methods for generating entity name embeddings were found.

2. Entity alignment with unmachable entities

An embedding-based entity alignment method typically consists of two parts: representation learning, which computes embeddings for the entities belonging to different knowledge graphs, and the matching of these embeddings. The computation of embeddings for different knowledge graph entities is done separately, so these embeddings may fall into different vector spaces. This is why it is necessary to map them into a common vector space, which is done using the so-called «seed alignments» containing the pairs of equivalent entities from the two knowledge graphs. The distance between aligned entity pairs is measured using distance metrics, such as cosine similarity, Manhattan distance, Euclidean distance, and others.

Until recently, EA solutions assumed that every entity in the source knowledge graph (KG) has an equivalent entity in the target KG. Hence each entity equivalent to the source entity was searched as the nearest neighbor of the source entity in the embedding space. In practice, the nearest-neighbor-based methods can result in the entity alignment graph having hub nodes with too many "equivalent" entities, as well as isolated nodes with no "equivalent" entities. Some methods, such as [3, 4], leverage bi-directional alignment integrating a source-to-target with a target-to source alignment. In addition, methods such as Hungarian algorithm and Stable matching algorithm [5-7] are looking for 1-to-1 correspondence between the aligned entities. However, unmatchable entities always exist. For example, one of the largest knowledge graphs wikidata.org contains equivalent entities from many different datasets, such as viaf.org, ror.org, imdb.com, and others, while each of these datasets contains some entities absent in other knowledge graphs. This is why an ideal entity alignment system should be able to detect and handle unmatchable entities.

To address this problem, special datasets are required. The main challenge is to ensure that «dangling» entities actually have no equivalent entities in the second dataset. First, two subgraphs are created, in which every entity has an equivalent in the other graph. Then, two random non-overlapping subsets of entities are removed from both knowledge graphs. The corresponding entities of the removed entities then become unmatchable. The structure of a dataset with unmatchable entities is shown in Figure 2. The Russian-English dataset from [1] was used as the initial set of triples.

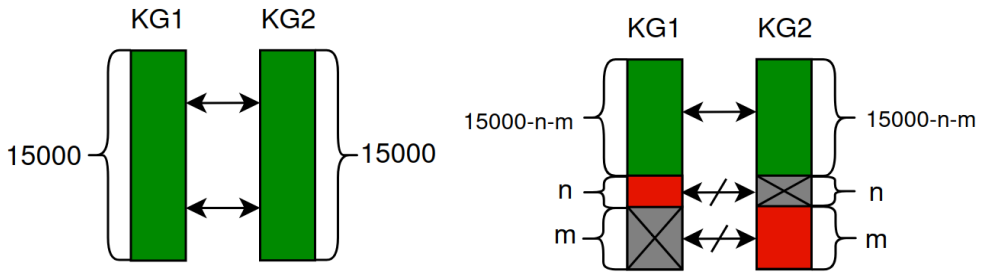

Figure 2. Construction of a dataset with unmatchable entities

3. Algorithms used in the entity alignment experiments

3.1. Algorithm RREA

The RREA algorithm [8] was used for the generation of entity embeddings, as it demonstrates the EA quality superior, for instance, to the RDGCN algorithm. This method uses a Graph Neural Network-based (GNN-based) approach to entity alignment. Its key feature is the use of an orthogonal transformation matrix. To this end, a *Relational Reflection Transformation* is introduced, which satisfies the following two conditions:

1. *Relational Differentiation*. For any two relations r_1 and r_2 and entity e, the embeddings of entity **e** should be mapped into different spaces.
2. *Dimensional Isometry*. The norms of entity embeddings should remain unchanged after the transformation. The distances between entity embeddings should be preserved.

For any relation embedding **r,** this operation constructs a reflection matrix and uses it as a transformation matrix.

3.2. Data set used by the Hungarian algorithm

The entity matching problem can be solved as a problem where entities from one knowledge graph are assigned to the entities in another graph. This requires finding the minimum total sum of pairwise distances between entities. The Hungarian algorithm matches all entities to all entities, enforcing a one-to-one correspondence between them.

The Hungarian algorithm operates on a square distance matrix between the entities of two knowledge graphs, as it constructs a matching with a minimal cost. This is why fictitious entities are added to the graph with fewer entities. The final aligned entities are obtained by removing all pairs including fictitious entities.

3.3. Algorithm TBNNS and C-TBNNS

To identify unmatchable entities, the *Thresholded Bi-directional Nearest Neighbor Search* (TBNNS) was used [9].

The TBNNS method defines three conditions under which a pair of entities u from KG1 and v from KG2 is considered as matchable:

- **v** is the nearest neighbor of **u** in KG2 amongst all other entities in KG2;
- **u** is the nearest neighbor of **v** in KG1 amongst all other entities in KG1;
- the distance between **u** and **v** is less than a given threshold θ.

Entities that do not satisfy these conditions are considered unmatchable.

This constraint is fairly strict and requires careful tuning of the threshold value θ. A high threshold can produce more aligned entity pairs while some of them might be incorrect. A low threshold will result in fewer aligned entity pairs, and almost all of them will be correct.

The C-TBNNS [10] method measures the confidence of an entity pair to be correctly aligned. The confidence score $C(u, v)$ is estimated as **Dist**(**u**, **v**') – **Dist**(**u**, **v**) + **Dist**(**v**, **u**') - **Dis**t(**v**, **u**).

This calculation is based on the idea that if the distance between the embeddings of the entities *u* and *v* is less than the distance between the top-2 closest candidates for alignment, it is possible to assume that these two entities are correctly aligned. The confidence score is used to calculate the loss function.

3.4. EntMatcher and CUEA frameworks

The two frameworks used for experiments with unmatchable entities were EntMatcher [10] and CUEA [11]. The advantage of the EntMatcher is a vast set of representation learning algorithms and strategies for finding corresponding entities between two knowledge graphs.

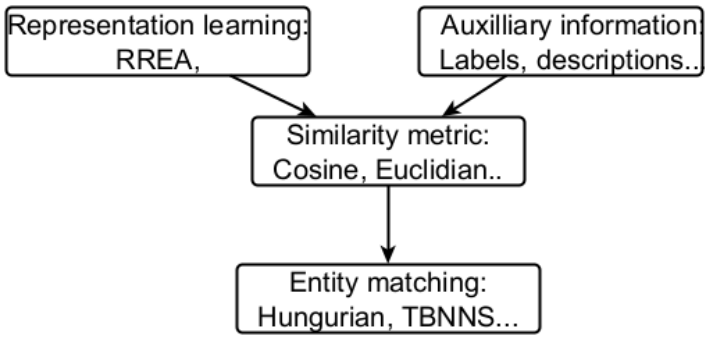

Figure 3. The architecture of the EntMatcher framework

However, this framework is not designed to handle unmatchable entities. The framework developed specifically for this purpose is the CUEA, but its set of embedding methods and entity matching strategies is very limited. To remedy the situation, both frameworks have been modified. The EntMatcher was extended to handle unmatchable entities by adding an implementation of the TBNNS [9]. The architecture of the EntMatcher framework is shown in Figure 3.

A distinctive feature of the CUEA (Confidence-based Unsupervised Entity Alignment) method is its ability to work without a set of pre-aligned entities. It can use external information, such as entity labels, to construct an initial alignment. The CUEA algorithm takes the initial structural and textual embeddings of entities as an input, sums them up together using a weighting coefficient α, and saves the resulting embeddings. In this case, the initial structural embeddings were computed using the RREA method, and the initial textual embeddings were computed using the LaBSE [11] language model.

The current embeddings are then used in subsequent iterative training. A list of aligned entity pairs is also stored in memory, where newly aligned pairs are added, but none are removed (this list will be the final output of the algorithm).

Next, the TBNNS is applied with a given alignment distance threshold θ. The TBNNS takes as an input the current embeddings and the set of the yet-to-be-aligned entities in KG1 and KG2 and adds new pairs to the list of the aligned entities.

Next, the embeddings are retrained using the current list of aligned pairs. The new current embeddings are obtained by summing the existing embeddings and the retrained embeddings with the weighting coefficient β.

Then, the threshold θ is increased by a certain value, and C_TBNNS is applied again, and the loop starts over and continues until the algorithm no longer adds new aligned entities. The final list of aligned entities is used to compute precision, recall, and F1-score.

The CUEA framework was extended by adding an implementation of the RREA entity embedding method [8]. The architecture of the CUEA is shown in Figure 4.

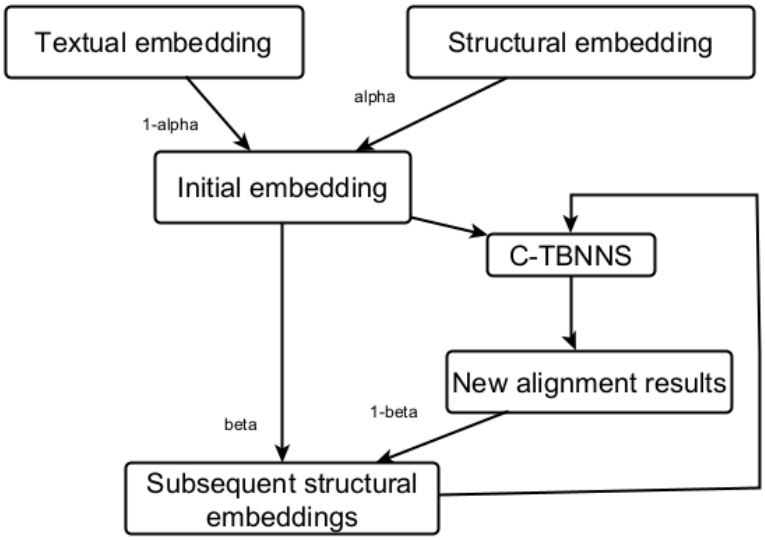

Figure 4. The architecture of the CUEA framework

Then both frameworks were used to conduct experiments on the Russian-English dataset with unmatchable entities, testing a wide range of parameters.

Three groups of experiments have been conducted:

1. Computing of structural embeddings using the EntMatcher Framework and matching the entities using the Hungarian algorithm [6];
2. Computing of structural embeddings using the EntMatcher Framework method and matching the entities using the TBNNBS;
3. Computing of structural embeddings using the CUEA framework and textual embeddings using the LaBSE language model, computing their fusion and matching of entities using the UEA.

Since the RREA algorithm was used for embedding generation in all the three groups of experiments, this will not be mentioned further and we will focus on the different methods of establishing entity correspondences.

4. Results of experiments with unmatchable entities

We used a modified EntMatcher, capable of aligning knowledge graphs with different numbers of unmatchable entities, and the ru_en dataset. The embedding computation algorithm was trained on two complete knowledge graphs and 4,500 known equivalent entity pairs. *Only relational triples were used.*

Testing was done on the remaining pairs (up to 10500 pairs of entities, or 70% of the full dataset). Entities from the left knowledge graph (KG1) were the input of the model. The output of the model was the set of predicted aligned pairs. The following notations are used in the table:

KG1 and KG2 are the numbers of entities in each of the KGs;

UKG1 and UKG2 are the numbers of unmatchable entities in each of the KGs;

Corr. is the number of pairs aligned correctly; and

P, R, F1 are precision, recall, and F1-score.

4.1. Distance matching using the Hungarian algorithm

The results of applying the Hungarian algorithm for entity alignment are presented in Table 2. It can be seen that as the number of unmatchable entities increases, all the three metrics decrease, which is an expected result since the Hungarian algorithm aligns all available entities with each other.

If unmatchable entities are present in only one knowledge graph, then recall is equal to precision.

When the unmatchable entities are present only in the left or only in the right knowledge graph, the results are approximately the same.

When the unmatchable entities are present in both knowledge graphs, the F1-score is slightly lower when there are many unmatchable entities.

Table 2. Entity alignment using the Hungarian algorithm with different numbers of matchable entities in KG1 and KG2

KG1	KG2	UKG1	UKG2	Corr.	P	R	F1
10500	10500	0	0	5424	0.517	0.517	0.517
10250	10250	250	250	4783	0.467	0.478	0.472
10000	10000	500	500	4395	0.440	0.463	0.451
10250	10500	0	250	5134	0.501	0.501	0.501
10000	10500	0	500	4832	0.483	0.483	00.483
9500	10500	0	1000	4344	0.375	0.457	0.457
10500	10250	250	0	5120	0.500	0.500	0.500
10500	10000	500	0	4879	0.488	0.488	00.488
10500	9500	1000	0	4329	0.456	0.456	0.456

4.2. Entity alignment using the TBNNS

The results of runs with the different values of the entity distance threshold θ, including an infinite threshold denoted in the table as INF, are shown in the table below.

Table 3. Entity alignment with the TBNNS using the distance threshold and different numbers of matchable entities in KG1 and KG2

KG1	KG2	UKG1	UKG2	Corr.	θ	P	R	F1
8000	8000	2500	2500	208	0.025	0.756	0.038	0.072
8000	8000	2500	2500	643	0.05	0.703	0.117	0.200
8000	8000	2500	2500	869	0.075	0.675	0.158	0.256
8000	8000	2500	2500	1068	0.1	0.657	0.158	0.256
8000	8000	2500	2500	1260	0.15	0.657	0.194	0.300
8000	8000	2500	2500	1360	0.2	0.622	0.229	0.335
8000	8000	2500	2500	1387	0.25	0.607	0.247	0.351
8000	8000	2500	2500	1408	0.3	0.591	0.252	0.353
8000	8000	2500	2500	1406	0.35	0.576	0.256	0.354
8000	8000	2500	2500	1384	0.4	0.557	0.256	0.350
8000	8000	2500	2500	1414	0.45	0.542	0.257	0.349
8000	8000	2500	2500	1397	0.5	0.535	0.254	0.345
8000	8000	2500	2500	1386	INF	0.539	0.252	0.434

Not too large and not too small value of the threshold θ gives a slight improvement of the result across all the three metrics compared to an infinite threshold. A very small threshold reduces the recall (R) and F1-score but increases the precision (P).

4.3. Experiments with the unmatchable entities using the CUEA

Only the initial structural embeddings trained on the training pairs (using only relational triples) were used as an input into the CUEA algorithm. The results are shown in the table below. The "Pred." column shows the number of predicted aligned entity pairs. The "Mat." shows the number of matchable entities amongst the predicted ("Pred."). The "Corr." column shows the total number of correctly aligned entities. The P, R and F1 columns show the precision, recall and F1-score, respectively. As expected, all metrics decrease as the number of unmatchable entities increases.

Table 4. Results of CUEA running on unmatchable entities

KG1	KG2	UKG1	UKG2	Pred.	Mat.	Corr.	P	R	F1
10500	8500	2000	0	8188	7029	3931	0.462	0.480	0.471
10500	7500	3000	0	7074	5576	3230	0.431	0.457	0.443
10500	6500	4000	0	5978	4305	2501	0.385	0.418	0.401
10500	5500	5000	0	4928	3265	1956	0.356	0.397	0.375
10500	4500	6000	0	3857	2297	1453	0.323	0.377	0.348

4.4. Comparison of results across all methods

Below is a summary table comparing the results of the three methods using the RREA embeddings on the *en_ru* dataset with unmatchable entities. An infinite TBNNS parameter θ was used.

Table 5. Summary of results of different entity alignment algorithms

KG1	KG2	UKG1	UKG2	Method	P	R	F1
10500	8500	2000	0	CUEA	0.462	0.480	0.471
				EntMat+TBNNS	0.708	0.319	0.439
				EntMat+Hungarian	0.416	0.416	0.416
10500	7500	3000	0	CUEA	0.431	0.457	0.443
				EntMat+TBNNS	0.683	0.294	0.411
				EntMat+Hungarian	0.376	0.376	0.376
10500	6500	4000	0	CUEA	0.385	0.418	0.401
				EntMat+TBNNS	0.630	0.271	0.379
				EntMat+Hungarian	0.343	0.343	0.343
10500	5500	5000	0	CUEA	0.356	0.397	0.375
				EntMat+TBNNS	0.598	0.264	0.366
				EntMat+Hungarian	0.320	0.320	0.320
10500	4500	6000	0	CUEA	0.323	0.377	0.348
				EntMat+TBNNS	0.534	0.236	0.328
				EntMat+Hungarian	0.285	0.285	0.285

The results show that the iterative CUEA algorithm achieved the highest F1-score and recall, while EntMatcher combined with the TBNNS achieved the highest precision with low recall.

Finally, Table 6 compares the CUEA alignment results obtained using only structural embeddings with those obtained by combining information on structural embeddings and entity name embeddings. Entity name embeddings were calculated using the LaBSE model. Significant improvements in all parameters are evident.

Table 6. Comparison the CUEA alignment results

KG1	KG2	UKG1	UKG2	Method	P	R	F1
				CUEA Structural	0.462	0.480	0.471
10500	8500	2000	0	CUEA Structural +Name Embed.	**0.972**	**0.610**	**0.750**
				CUEA Structural	0.431	0.457	0.443
10500	7500	3000	0	CUEA Structural +Name Embed.	**0.954**	**0.475**	**0.634**
				CUEA Structural	0.385	0.418	0.401
10500	6500	4000	0	CUEA Structural +Name Embed.	**0.946**	**0.421**	**0.582**
				CUEA Structural	0.356	0.397	0.375
10500	5500	5000	0	CUEA Structural +Name Embed.	**0.930**	**0.340**	**0.497**
				CUEA Structural	0.323	0.377	0.348
10500	4500	6000	0	CUEA Structural +Name Embed.	**0.911**	**0.281**	**0.430**

5. Conclusion

The problem of knowledge graph alignment with unmatchable entities reflects the real-world scenarios and is therefore highly relevant. This work presents the results of entity alignment experiments on a Russian-English dataset with unmatchable entities. The confidence-based unsupervised entity alignment (CUEA) performed fairly well, by adding progressively the pairs of aligned entities. Future work will examine the variations of this approach using various language models for creating embeddings for the various attributes of entities and, in particular, the multi-modal knowledge graphs.

It should be noted that the results of entity alignment methods vary depending on the pairs of the languages of the knowledge graphs. In general, the best results are achieved in the English-French and English-German alignments. The lower alignment quality on the English-Russian dataset is, on the one hand, due to the linguistic specifics. On the other hand, alignment quality is influenced not only by the algorithm itself, but also by the structure of the dataset, particularly the density and the distribution of the degrees of vertices in the knowledge graphs [7].

References

[1] Gnezdilova V.A., Apanovich Z.V. Russian-English dataset and comparative analysis of algorithms for cross-language embedding-based entity alignment // Journal of Physics: Conference Series. – 2021. – Vol. 2099.

[2] Gusev D., Apanovich Z. Methods of processing textual information in entity alignment algorithms // Bull. Novosibirsk Comp. Center. Ser. Computer Science. – Novosibirsk, 2021. – Iss. 45. – P. 49-58.

[3] Lample, G., Conneau, A., Ranzato, M., Denoyer, L., Jégou, H. Word translation without parallel data // Proc. ICLR. – 2018. DOI: 10.48550/arXiv.1710.04087.

[4] Zeng W., Zhao X., Li X., Tang J., Wang W. On entity alignment at scale // VLDB Journal. – 2022. – Vol.31. – P.1009-1033.

[5] Kuhn H.W. The Hungarian method for the assignment problem. – https://web.eecs.umich.edu/~pettie/matching/Kuhn-hungarian-assignment.pdf .

[6] Zeng W., Zhao X., Tang J., Lin X. Collective entity alignment via adaptive features // Proc. ICDE. – 2020. – P. 1870–1873. DOI: 10.1109/ICDE48307.2020.00191.

[7] Zhu R., Ma M., Wang P. RAGA: relation-aware graph attention networks for global entity alignment // Proc. PAKDD. – 2021. – Vol. 12712. – P. 501–513.

[8] Xin M., Wenting W., Huimin X. et al. Relational reflection entity alignment. DOI: 10.1145/3340531.3412001.

[9] Zhao X., Zeng W., Tang J. Recent advance of alignment inference stage // Entity Alignment. – Singapore: Springer Nature, 2023. – P. 77-112.

[10] Zhao X., Zeng W., Tang J. *et al.* Toward Entity Alignment in the Open World: An Unsupervised Approach with Confidence Modeling // Data Science and Engineering. – 2022. – Vol. **7**. – P. 16–29. DOI: 10.1007/s41019-022-00178-4.

[11] Feng F., Yang Y., Cer D., Arivazhagan N., Wang W. Language-agnostic BERT Sentence Embedding. 2020. DOI:10.48550/arXiv.2007.01852.