

MateTee: A Semantic Similarity Metric Based on Translation Embeddings for Knowledge Graphs

Camilo Morales^{1,2}, Diego Collarana^{1,2}(✉), Maria-Esther Vidal^{2,3},
and Sören Auer^{1,2}

¹ Enterprise Information Systems (EIS), University of Bonn, Bonn, Germany
{moralesc, collaran, auer}@cs.uni-bonn.de

² Fraunhofer Institute for Intelligent Analysis and Information Systems (IAIS),
Sankt Augustin, Germany
vidal@cs.uni-bonn.de

³ Universidad Simón Bolívar, Caracas, Venezuela

Abstract. Large Knowledge Graphs (KGs), e.g., DBpedia or Wikidata, are created with the goal of providing structure to unstructured or semi-structured data. Having these special datasets constantly evolving, the challenge is to utilize them in a meaningful, accurate, and efficient way. Further, exploiting semantics encoded in KGs, e.g., class and property hierarchies, provides the basis for addressing this challenge and producing a more accurate analysis of KG data. Thus, we focus on the problem of determining relatedness among entities in KGs, which corresponds to a fundamental building block for any semantic data integration task. We devise MateTee, a semantic similarity measure that combines the *gradient descent* optimization method with semantics encoded in ontologies, to precisely compute values of similarity between entities in KGs. We empirically study the accuracy of MateTee with respect to state-of-the-art methods. The observed results show that MateTee is competitive in terms of accuracy with respect to existing methods, with the advantage that background domain knowledge is not required.

1 Introduction

We are living in the Big Data era where a large number of structured and semi-structured datasets are publicly available. Such datasets are collected from different social domains, e.g., government, scientific communities, or social media and social networks. The semantic representation of the data in RDF Graphs helps in the endeavor of automatically solving data-driven oriented tasks, providing as result, more useful and meaningful services from such big and heterogeneous data [2]. Particularly, the tasks affected by a good similarity metric between data entities are: semantic data integration of heterogeneous data, entity linking and clustering, as well as the generation of recommendations. The future of the Web of Data and the Web of Things brings even more heterogeneity and larger datasets. Streaming data coming at high rates need to be processed on-demand,

all of which only increases the need of automation in the process of creation and processing of semantics. Data management and Artificial Intelligence approaches play an important role on the task of KG data analysis. Machine Learning (ML), mostly in its supervised flavor, aims to give machines the capability to learn by examples, essentially, labeled data. ML field has achieved promising results with sophisticated techniques, such as Kernel methods or Deep Learning models. Furthermore, the Semantic Web, and in general, all the available Knowledge Graphs (KGs) such as DBpedia or Yago, have been built with a tremendous effort of the scientific community having the main objective of making the data understandable not only by humans but also by computers. Structured data facilitates the tasks of data integration, relations or associations discovery, as presented by Bordes et al. 2013 with TransE [3].

On one hand, we have an immense amount of available knowledge facts, encoded as structured data in knowledge graphs, and on the other, we have the Machine Learning boom and techniques able to have access to Big Data sets, for two main tasks: classification and link prediction. In the case of KGs, we are referring to classification of entities in a set of classes, and prediction (or discovery) of new relations between entities, i.e., RDF triples. In this paper, we focus on the problem of determining relatedness among entities in KGs. This problem serves as a building block for classification and link prediction, as well as for semantic data integration. Building on results from knowledge graph embedding, we devise MateTee a semantic similarity measure that combines *gradient descent* optimization method with semantics encoded in ontologies, to determine relatedness among entities in KGs.

We conduct an empirical evaluation to assess the quality of MateTee with respect to state-of-the-art similarity measures. Experiments are performed on two benchmarks: (a) The CESSM [15] KGs of proteins annotated with the Gene Ontology¹; (b) A KG of people from DBpedia. Observed results suggest that MateTee is able to outperform existing similarity measures, while it does not require any background knowledge or domain expertise to be configured.

In summary, our main contributions are as follows:

- An *end-to-end* approach named MateTee able compute similarity value between entities in a KG. MateTee is based on TransE, which utilizes the gradient descent optimization method to learn the *features representation* of the entities automatically.
- An extensive empirical evaluation on existing benchmarks and state-of-the-art showing MateTee behavior. Results indicate the benefits of using embeddings for determining relatedness among entities in KGs. MateTee and experimental studies are publicly available².

The remainder of the paper is structured as follows. First, we motivate the problem of determining relatedness among entities in KGs using a practical example. Then, we briefly describe preliminaries and background concepts required to

¹ <http://geneontology.org/>

² <https://github.com/RDF-Molecules/MateTee>

understand the problem treated in the paper, as well as the proposed solution. Section 4 defines the problem, the proposed solution, and MateTee architecture. Section 5 reports on the empirical evaluation, while related work is summarized in Sect. 6. Conclusion and future work are outlined in Sect. 7.

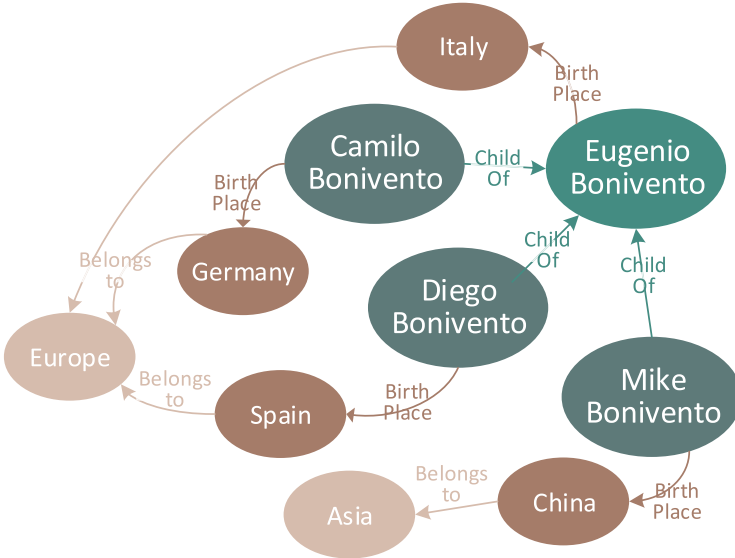


Fig. 1. Motivating Example. A portion of a Knowledge Graph (KG) describing relationships among persons and the places where they have been born. There exist different types of relations and multiple connectivity patterns among entities in KG

2 Motivating Example

Consider a knowledge graph in Fig. 1. Nodes of the same color indicate they share the same properties, while nodes of different colors differ in at least one property. Determining relatedness among same-colored nodes, e.g., Camilo with Diego, requires to compare, in a 1-1 fashion, values of each property of those entities and aggregate the results. This computation can be done as Camilo and Diego have the same set of properties, i.e., *Child_of* and *Birth_Place*. Contrary, if entities have different properties, i.e., they are on different colors, the problem is to measure their relatedness considering the complete set of properties of both nodes while is not possible to use the 1-1 approach, e.g., Germany and Camilo. Moreover, whenever entities are compared in terms of their *neighborhoods* and *reachable* nodes, Camilo should be more similar to Diego than to Mike, as Diego and Camilo are from Europe, while Mike is from China.

These difficulties come inherently with the multi-relational datasets. In relational data tables, all elements have the same properties, i.e., columns, and therefore, the similarity computation is performed aggregating a 1-1 similarity

value between each pair of the properties. With multi-relational data, nodes need to be made *comparable*, which means that they all must have the same set of properties or features. This can be done manually, *handcrafting* the features, and creating a list of them for each node, based on previous knowledge of the specific field or domain of the data. These sets of features will be regarded as a new representation of the nodes in KG. Then, these sets of features can be compared, again, in a 1-1 fashion. The problem is that manual creation of the features requires deep domain knowledge, not to mention it is error-prone and time consuming. Thus, to solve these problems a similarity measurement approach that automatically creates a canonical entity representations is required. In this paper, we present MateTee, a similarity approach that relies on embedding the original KG into a vector space in order to make all entities comparable. Similarity values among embeddings are measured based on any distance metric defined for vector spaces, e.g., Euclidean distance.

3 Preliminaries

MateTee determines relatedness between entities in Knowledge Graphs based on Encoding Generation methods such as TransE [3]. MateTee combines the gradient descent optimization method (used in TransE) with the explicit knowledge encoded in the ontologies of a KG.

3.1 Translation Embeddings

MateTee is based on TransE [3], acronym of *Translation Embeddings*, presented by Bordes et al. 2013. TransE tackles the problem of embedding a Knowledge Graph (KG) into a low dimensional vector space (called embedding space) for subsequent prediction or classification objectives, e.g., predict missing edges. The core of TransE is to learn the embeddings of entities in a way that similar entities in the KG should be also close in the embedding space. Additionally, dissimilar entities in the KG should be also far in the embeddings space. Learning the embeddings is done by analysing the *connectivity patterns* between entities in a KG, and then encoding these patterns into their vector representation, i.e., their embeddings. The optimization technique *Stochastic Gradient Descent* is executed to compute this encoding.

Modeling RDF triples in the embedding space with relations as **translations** is the core contribution of TransE. The basic idea behind translation-based model is the following:

$$Subject + Translation \approx Object$$

TransE aims at minimizing the error when summing up the distance d between the embeddings of the *Subject + Translation* pair and the embedding of the *Object*. Stochastic Gradient Descent (SGD) meta-heuristic allows for learning entity embeddings by minimizing the error defined as the sum of the distances d of all the triples in the KG. A global minimum cannot be ensured because SGD depends on a *randomly* selected start position of the *descent*.

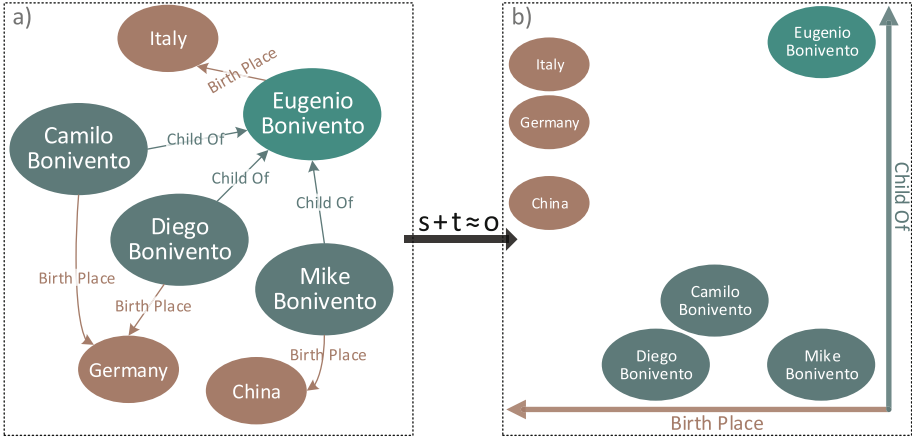


Fig. 2. TransE approach intuition. (a) An RDF Knowledge Graph where similar entities are in the same color; (b) Clusters of entities in the embedding space. Entities of the same color are close to each other in the identified cluster.

The random initialization procedure followed by TransE is presented in detail at [8]. Figure 2 illustrates the intuition of this approach.

4 MateTee: A Semantic Similarity Measure for RDF Knowledge Graphs

MateTee focuses on measuring the similarity between any pair of entities belonging to an input RDF Knowledge Graph. Measuring the similarity between entities is an important phase for any data integration problem, and for most machine learning tasks, e.g., clustering of nodes, or link prediction in KGs.

The main problem for computing the similarity of RDF knowledge graphs is that not all the nodes have the same properties, therefore, a 1-1 comparison at property level cannot be performed. State-of-the-art methods like GADES [20] perform a semantic analysis of the entities based on multiple aspects, i.e., 1-hop neighborhood, class hierarchy of the subjects/objects, class hierarchy of the properties, and mixtures of them. This analysis relies on domain knowledge and user expertise about the provenance of the data, e.g., GADES requires a good design of the hierarchy of classes and properties.

To overcome this problem, MateTee embeds an RDF knowledge graph into a vector space, once all the entities are represented as vectors with same dimensionality, it uses any common distance metric to calculate their similarity values. MateTee relies on finding a vector representation of graph entities to produce the similarity value. For this, MateTee utilizes TransE [3], a method based on Stochastic Gradient Descent that encodes the connectivity patterns of the entities into a low-dimensional embedding space. TransE ensures that similar nodes

in the RDF graph are close in the embedding space, while dissimilar nodes in the graph are distant in the embedding space.

By using TransE approach, MateTee aims to calculate similarity values as close as possible to the *ground truth*: values accepted by the scientific community because they were calculated manually with deep domain expertise, e.g., Sequence Similarity in the Gene Ontology domain. Formally, MateTee can be defined as:

Definition 1 (MateTee Embedding). *Given a knowledge graph $G = (V, E)$ composed by a set T of RDF triples, where $V = \{s \mid (s, p, o) \in T\} \cup \{o \mid (s, p, o) \in T\}$ and $E = \{p \mid (s, p, o) \in T\}$, MateTee aims to find a set M of embeddings of each member of V , such that:*

$$\arg \min_{\mathbf{m}_1, \mathbf{m}_2 \in M} Error(M) = \arg \min_{\mathbf{m}_1, \mathbf{m}_2 \in M} \sum_{\mathbf{m}_1, \mathbf{m}_2 \in M} |S_1(\mathbf{m}_1, \mathbf{m}_2) - S_2(\mathbf{m}_1, \mathbf{m}_2)|$$

where S_1 is a similarity metric computed using any distance measure defined for vector spaces, e.g., Euclidean distance, and S_2 is a similarity value given by the Gold Standards. The Gold Standards are the values considered as ground truth.

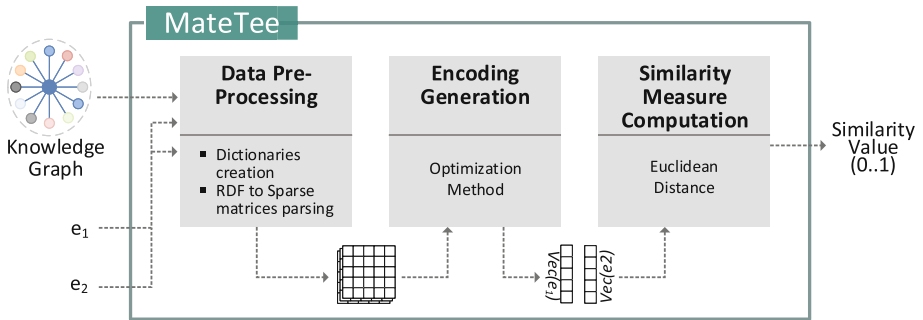


Fig. 3. The MateTee Architecture. MateTee receives as input an RDF Knowledge Graph (KG), and entities e_1 and e_2 from the KG. MateTee outputs a similarity value between e_1 and e_2 according to the connectivity patterns found in KG. A pre-processing step allows for the transformation of a KG into a matrix-based representation. Then, n -dimensional embeddings are generated. Finally, values of similarity are computed

4.1 The MateTee Architecture

Figure 3 depicts the *end-to-end* MateTee architecture. MateTee receives as input an RDF Knowledge Graph (KG), and entities e_1 and e_2 belonging to the KG. The objective of the complete process is to calculate the similarity value between e_1 and e_2 . The first step is to **Pre-Process** the original data in order to transform it into the format required by the optimization method. As the optimization methods are numeric based, we need a numerical representation of the data.

In other words, the string-based triples coming as input must be translated into a numeric format, usually sparse matrices. The implementation of TransE employs three sparse matrices: one representing the Objects, another for the Subjects, and a third one for the Translations. The matrices have as many columns as RDF triples are in the original KG, and as many rows as entities, i.e., number of Subjects + number of Translations + number of Objects. Note that if a Subject appears also as Object in another RDF triple, it is considered as one. Moreover, in order to map the original entities to their respective *encodings*, i.e., embeddings, dictionaries need to be created. Dictionaries map the original URIs of the entities with the ID of their embeddings.

Once the numerical representation and dictionaries of the RDF triples are created, the embeddings of the entities can now be *learned*. Learning embeddings happens at the **Encoding Generation** phase. This numerical representation of the data is now fed to the optimization method. The method aims to update the value of the embeddings in order to minimize an overall error according to a proposed model. MateTee is based on TransE, this method aims at minimizing the distance (in MateTee Euclidean Distance is used) between the sum of the embeddings of the Subject and Translation to the embeddings of the Object. TransE also defines *corrupted triples*, which are triples with either the Subject or Object replaced by another randomly selected resource from the set of entities. This is required because TransE needs not only to ensure that similar entities should be close in the embedding space, but also, that dissimilar entities must be farther than the similar ones. This can be seen in the following *Loss Function* used by TransE:

Definition 2 (Loss function). *Given is a set of RDF triples T and their respective set of corrupted RDF triples (original triples with either the Subject or Object replaced) T' . Embeddings of Subject s , Object o , and Transitions t in T are represented as \mathbf{S} , \mathbf{O} , and \mathbf{T} , respectively. Similarly, embeddings of Subject s' and Object o' in corrupted RDF triples in T' are represented as \mathbf{S}' and \mathbf{O}' , respectively. The loss function can be defined as:*

$$Loss(T, T') = \sum_{(s,t,o) \in T} \sum_{(s',t,o') \in T'} [margin + d(\mathbf{S} + \mathbf{T}, \mathbf{O}) - d(\mathbf{S}' + \mathbf{T}, \mathbf{O}')]_+$$

The key is to notice that the loss function only considers the positive part of the difference of the distances, plus the margin; this is denoted by $[x]_+$ in the loss formula. Considering positive values is crucial because if the distance between entities of the original triple, i.e., $d(\mathbf{S} + \mathbf{T}, \mathbf{O})$, is greater than the distance between the entities of the corrupted triple, i.e., $d(\mathbf{S}' + \mathbf{T}, \mathbf{O}')$, then the difference between the two is positive (regardless of the margin) and this number will increase the overall error. This situation should not occur according to the model $S + T \approx O$ as we want this difference to be as close to zero as possible. On the other hand, if the opposite situation happens, the distance between the entities of the original RDF triple, i.e., $d(\mathbf{S} + \mathbf{T}, \mathbf{O})$, is smaller than the distance between the entities of the corrupted triple, i.e., $d(\mathbf{S}' + \mathbf{T}, \mathbf{O}')$. This state is exactly what the model looks for, and since the difference between

both distances is negative, the overall error is not increased as only the positive part is considered. In the case when the entities of the original RDF triple is smaller than the distance between the entities of the corrupted triple, the margin tightens the model as the negative difference between both distances must be at least as big as the margin, otherwise the overall error will be increased.

TransE - Gradient Descent Algorithm. The core of TransE learning algorithm performs the following steps:

1. **Initialization:** The embedding of each entity (Subject/Object) is initialized uniformly and randomly between $\frac{-6}{\sqrt{k}}$ and $\frac{6}{\sqrt{k}}$ where k is the dimensionality of the embeddings. At this point only the relations are normalized, they will not be normalized again during the optimization. Entities will be normalized at the beginning of each iteration.
2. **Training (loop):**
 - (a) **Entity embeddings normalization:** In each iteration, first current embeddings of the entities are normalized. This is important because it prevents the optimization to minimize the error by artificially increasing the length i.e., norm, of the embeddings.
 - (b) **Creation of mini-batches:** Triples to be used as training examples for each iteration of the GD are selected. First, a random sample of set of triples from the input data set is chosen, and then, for each triple in the sample, a corrupted triple is created.
 - **Corrupted triples:** A corrupted triple is the same as the original but with either its Subject or Object replaced by another randomly selected entity from the data set, always just one, not both at the same time, as show in Fig. 4:
 - (c) **Embeddings update:** Once the training set of examples, i.e., real triples \cup corrupted triples is set, it proceeds with the actual optimization process:
 - For each one of the dimensions of each one of the embeddings in the data set, we calculate the derivative of the overall error with respect to this parameter. This derivative gives the direction on which the overall error is growing with respect to this parameter. Then, to know how to update this parameter so that the overall error decreases, it changes the direction to the opposite of the derivative, and moves one unit of the learning rate (which is also an input hyper-parameter). This process iterates until a maximum number of iterations is reached.

When the optimization reaches the termination condition, e.g., the maximum number of iterations in TransE, the embeddings of the entities have been already learned. Having the embeddings of all the entities in the input KG, including e_1 and e_2 , MateTee can now proceed to the **Similarity Measure Computation** of both entities. Any distance metric for vector spaces can be used to calculate this value, e.g., any Minkowski distance, Euclidean for MateTee. It is important to notice that MateTee calculates the similarity and not the distance.

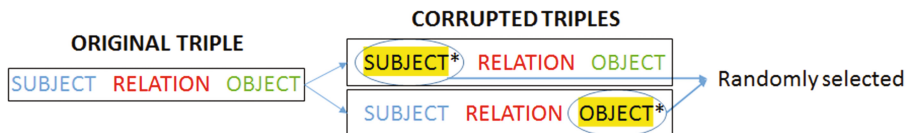


Fig. 4. Corrupted triples. An original RDF triple t and two corrupted versions of t are presented on the left and right hand of the figure, respectively. Corrupted triples have either the Subject or the Object replaced by another randomly selected entity from the set of entities from the input KG

Therefore, using the Euclidean distance MateTee finds a similarity value between 0 and 1, using the following formula:

$$similarity(A, B) = \frac{1}{1 + EuclideanDistance(A, B)}$$

5 Empirical Evaluation

We empirically study the effectiveness of MateTee on solving the problem of measuring the semantic similarity between entities in a KG. We assess the following research questions: **(RQ1)** Does the translations embeddings method used in MateTee improve the accuracy of determining relatedness between entities in a KG? **(RQ2)** Is MateTee able to perform as good as the state-of-the-art similarity measures? **(RQ3)** Does MateTee perform well in Knowledge Graphs from different domains? To answer our research questions, we evaluate MateTee in two different scenarios. In the first evaluation, we compare Proteins annotated with the Gene Ontology³. In the second evaluation, we compare people extracted from DBpedia, we prepare a dataset named *DBpedia People* [5].

Implementation. MateTee is implemented in Python 2.7.10. The experiments were executed on a Ubuntu 14.04 (64 bits) machine with CPU: Intel(R) Xeon(R) E5-2660 2.60GHz (20 physical cores) with 132 GB RAM, and GPU card GeForce GTX TITAN X. Source code and a Docker set up are available in Git⁴.

5.1 Similarity Among Proteins Annotated with the GO Ontology

Datasets. This experiment is conducted on the collections of proteins published at the Collaborative Evaluation of GO-based Semantic Similarity Measures [16] (CESSM) websites 2008⁵ and 2014⁶. The CESSM 2008 collection is composed of 13,430 pairs of proteins from UniProt with 1,039 distinct proteins, while

³ <http://geneontology.org/>

⁴ <https://github.com/RDF-Molecules/MateTee>

⁵ <http://xldb.di.fc.ul.pt/tools/cessm/>

⁶ <http://xldb.di.fc.ul.pt/biotools/cessm2014/>

the CESSM 2014 dataset includes 22,302 pairs of proteins also from UniProt with 1,559 distinct proteins. The sets of annotations of CESSM 2008 and 2014 comprise 1,908 and 3,909 distinct GO terms, respectively. The original CESSM collections are presented in a multi-file fashion, one file per protein. Technical details in Table 1 refer to the unified (single file) dataset, after data transformations are applied. CESSM computes the Pearson’s correlation coefficients with respect to three similarity measures from the genomic domain⁷: *ECC similarity* [7], *Pfam* [16], and the *Sequence Similarity* (SeqSim) [19]. Furthermore, the CESSM evaluation framework makes the results of eleven semantic similarity measures available. These state-of-the-art semantic similarity measures are specific for the genomic domain and exploit the knowledge encoded in the Gene Ontology (GO) to determining relatedness among proteins in the CESSM collections. These semantic similarity measures are extensions of well-known similarity measures to consider GO annotations, Information Content (IC) of these annotations, and pair-wise combinations of common ancestors in GO hierarchy. The extended similarity measures are the following: Resnik (R) [17]; Lin (L) [11]; and Jiang and Conrath (J) [13]. Additionally, the CESSM evaluation framework considers the average of the ICs of pairs of common ancestors during the computation of these measures; this measure is denoted with the label A. Following the approach reported by Sevilla et al. [18], the maximum value of IC of pairs of common ancestors is computed; combined measures are distinguished with the label M. As proposed by Couto et al. [6], the best-match average of the ICs of pairs of disjunctive common ancestors (DCA) is also computed; measures labelled with B or G correspond to combinations with the best-match average of the ICs. Finally, the Jaccard index is applied to sets of annotations together with domain-specific information in the similarity measures simUI (UI) and simGIC (GI) [15].

Table 1. CESSM 2008 and 2014 - Dataset description. Shows dataset size in Megabytes, overall number of triples, number of left entities (Subjects), right entities (Objects), and shared entities (appearing as Subject and as Object), and number of relations, to present a comparison of size between datasets from 2008 and 2014

		CESSM 2008	CESSM 2014
Size (MBs)		1	1
Triples		8,359	20,153
Entities	Left	1,039	1,559
	Shared	0	0
	Right	1,908	3,909
Relations		1	1

⁷ The area in molecular biology and genetics that studies the genetic material of an organism.

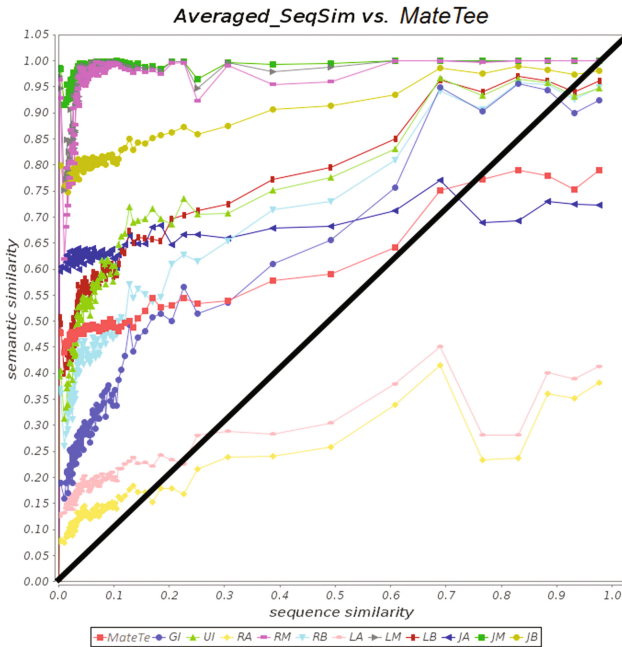


Fig. 5. Results from the CESSM evaluation framework for the CESSM 2008 collection. Results include: average values for MateTee with respect to SeqSim. The black diagonal line represents the values of SeqSim for the different pairs of proteins in the collection. The similarity measures are: simUI (UI), simGIC (GI), Resnik’s Average (RA), Resnik’s Maximum (RM), Resnik’s Best-Match Average (RB/RG), Lin’s Average (LA), Lin’s Maximum (LM), Lin’s Best-Match Average (LB), Jiang & Conrath’s Average (JA), Jiang & Conrath’s Maximum (JM), J. & C.’s Best-Match Average (JB). MateTee outperforms eleven measures and reaches a value of Pearson’s correlation of **0.787**

Results. Figures 5 and 6 report on the comparison of MateTee and the rest of the eleven similarity measures with SeqSim; both plots were generated by the CESSM evaluation framework. The black diagonal lines represent the values assigned by SeqSim. The majority of the studied similarity measures assign high values of similarity to pairs of proteins that SeqSim considers as similar proteins, i.e., in pairs of proteins with high values of SeqSim, the majority of the curves of the similarity measures are close to the black line. Nevertheless, the same behavior is not observed for the pairs of proteins that are not similar according to SeqSim, i.e., the corresponding curves are far from the black line. Contrary to state-of-the-art similarity measures, MateTee is able to compute values of similarity that are more correlated to SeqSim, i.e., the curve of MateTee is close to the black line in both collections. MateTee is able to reach values of the Pearson’s correlation of **0.787** and **0.817** in CESSM 2008 and 2014, respectively.

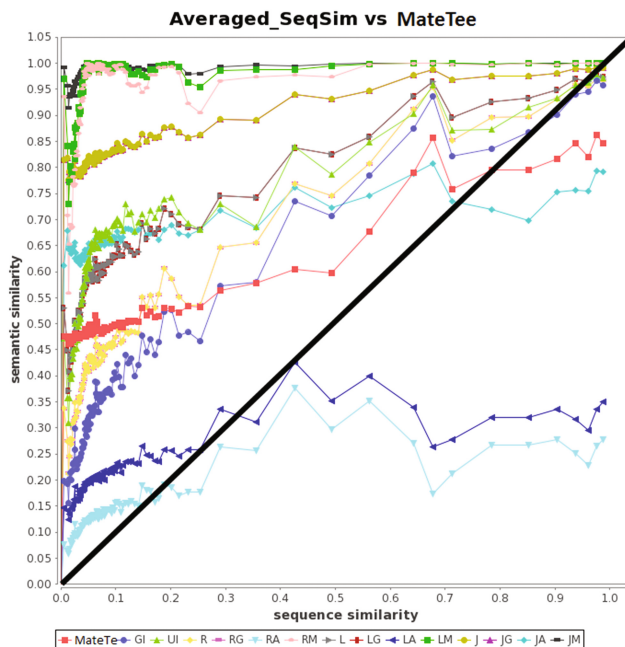


Fig. 6. Results from CESSM evaluation framework for the CESSM 2014 collection. Results include: average values for MateTee with respect to SeqSim. The black diagonal line represents the values of SeqSim for the different pairs of proteins in the collection. The similarity measures are: simUI (UI), simGIC (GI), Resnik’s Average (RA), Resnik’s Maximum (RM), Resnik’s Best-Match Average (RB/RG), Lin’s Average (LA), Lin’s Maximum (LM), Lin’s Best-Match Average (LB), Jiang & Conrath’s Average (JA), Jiang & Conrath’s Maximum (JM), J. & C.’s Best-Match Average (JB). MateTee outperforms eleven measures and reaches a value of Pearson’s correlation of **0.817**

Additionally, we present the results of the comparison of MateTee and eleven similarity measures with respect to the gold standard similarity measures: ECC, Pfam, and SeqSim; Table 2 presents these results. Moreover, additional similarity measures are included in the study: d_{tax} [1], d_{ps} [14], OnSim [22], IC-OnSim [21], and GADES [20]. As before, values of the Pearson’s correlation represent the quality of a measurement of similarity, the higher the correlation with the gold standards, the better the measurement. The top 5 similarity measures (before introducing MateTee) with higher quality are highlighted in gray, and the highest is highlighted in bold.

Discussion: From the results, the following insights can be concluded; MateTee already outperforms the quality of GADES for both collections 2008 and 2014, which is the best-performing measurement before our method, for the Sequence Similarity. In the 2008 collection, MateTee stands at the 5th position against the other two gold standards, only at 0.015 points to the GADES for ECC, and

Table 2. GO - CESSM 2008 and 2014 - Results. Quality in terms of Pearson’s correlation coefficient between three gold standards, i.e., SeqSim (Sequence) similarity, Pfam (Protein Families) similarity and EC (Enzyme Commission) similarity, and thirteen in-house similarity measures of CESSM, plus OnSim, IC-OnSim and GADES. With gray background the best 6 correlations and the best in bold

Similarity measure	2008			2014		
	<i>SeqSim</i>	<i>ECC</i>	<i>Pfam</i>	<i>SeqSim</i>	<i>ECC</i>	<i>Pfam</i>
GI [16]	0.773	0.398	0.454	0.799	0.458	0.421
UI [16]	0.730	0.402	0.450	0.776	0.470	0.436
RA [19]	0.406	0.302	0.323	0.411	0.308	0.264
RM [22]	0.302	0.307	0.262	0.448	0.436	0.297
RB [7]	0.739	0.444	0.458	0.794	0.513	0.424
LA [13]	0.340	0.304	0.286	0.446	0.325	0.263
LM [22]	0.254	0.313	0.206	0.350	0.460	0.252
LB [7]	0.636	0.435	0.372	0.715	0.511	0.364
JA [11]	0.216	0.193	0.173	0.517	0.268	0.261
JM [22]	0.234	0.251	0.164	0.342	0.390	0.214
JB [7]	0.586	0.370	0.331	0.715	0.451	0.355
d_{tax} [1]	0.650	0.388	0.459	0.682	0.434	0.407
d_{ps} [15]	0.714	0.424	0.502	0.750	0.480	0.450
OnSim [26]	0.733	0.378	0.514	0.774	0.455	0.457
IC-OnSim [25]	0.779	0.443	0.539	0.810	0.513	0.489
GADES [24]	0.780	0.446	0.539	0.812	0.515	0.49
MateTee	0.787	0.431	0.496	0.817	0.501	0.461

0.043 for Pfam. While in the 2014 collection, MateTee stands at the 3th position against the Pfam gold standard, only at 0.029 points to the GADES (the best before MateTee), and at the 5th position against the ECC gold standard, only at 0.014 points to the GADES (the best before MateTee).

It can be observed that GADES [20] is the greatest competitor for MateTee. It performs better comparing with the ECC and Pfam gold standards, but it is outperformed against SeqSim. As the results of GADES and MateTee are rather close. For the three gold standards, the advantage of MateTee against GADES is that the former requires domain expertise to define its final similarity measure (GADES defines multiple measures based on: Class hierarchy, Neighborhood, Relation Hierarchy, Attributes, and mixtures of them). While the latter learns the embeddings in an automatic way (through an optimization process called Stochastic Gradient Descent), and then uses any common vector similarity measure, e.g., Euclidean or Cosine, to calculate their similarity.

5.2 Similarity Among People from DBpedia

Dataset: Table 3 shows technical details of the datasets used in the *DBpedia People* experiment. The **Gold Standard (GS)**: the collection was extracted from the live version of DBpedia (July 2016); it contains 20,000 subjects of type

Table 3. Description of the data set *DBpedia People*. Shows the datasets size in Megabytes, overall number of triples, overall number of persons, number of left entities (Subjects), right entities (Objects), and shared entities (appearing as Subject and as Object), and number of relations, to present a comparison of size between three dumps

		Data set 1	Data set 2	Data set 3
Size (MBs)		80	80	80
Triples		552,355	553,232	552,527
Subjects (Persons)		20,000	20,000	20,000
Entities	Left	60,000		
	Shared	0		
	Right	247,465		
Relations		1,981		

Person⁸, i.e., 20,000 subjects with all available properties and their values. The overall number of RDF triples is 829,184. The Gold Standard is used to compute Precision and Recall during the evaluation. The **Test Datasets (TS)**: are created from the Gold Standard with their properties and values were randomly split among three test datasets. Each triple is randomly assigned to one or several test datasets. The selection process takes two steps: (1) a number of test datasets to copy a triple to is chosen randomly under a uniform distribution; (2) the chosen number is used as a sample size to randomly select particular test datasets to write a triple. URIs are generated specifically for each test data set. Eventually, each test dump contains a subset of the properties in the gold standard. Each subset of properties of each person is composed randomly using a uniform distribution.

Metrics: We measure the behavior of MateTee in terms of the following metrics: (a) **Precision** From all matched pairs (pairs with similarity greater than the threshold), percentage of correct matches.

$$Precision = \frac{\text{Number of correctly matched pairs}}{\text{Total number of matched pairs}}$$

(b) **Recall** From all expected matches (all, including below and above the threshold), percentage of correct matches.

$$Recall = \frac{\text{Number of correctly matched pairs}}{\text{Total number of expected matches}}$$

Results: We tested the quality of MateTee by comparing its results with two other similarity measurements: Jaccard [5] and GADES [20]. For each one we calculate the Precision and Recall, considering different values of **Threshold**. The Threshold is the minimum similarity value so that the pair of people is considered in the matched-pairs set. Table 4 show the results obtained using Jaccard, GADES, and MateTee similarity approaches.

⁸ <<http://dbpedia.org/ontology/Person>>.

Table 4. DBpedia People Test Datasets. Results comparison of precision and recall using Jaccard, GADES and MateTee similarity measurements, obtained with different threshold values: 0.6, 0.7, 0.8 and 0.9. In bold the best value for each threshold

	T0.6		T0.7		T0.8		T0.9	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Jaccard	0.36	0.01	0.30	0.01	0.30	0.01	0.30	0.01
GADES	0.87	0.73	0.83	0.43	0.80	0.16	0.63	0.05
MateTee	0.93	0.79	0.99	0.59	1.00	0.10	0.00	0.00

Discussions: From the results we extract the following insights. Regarding Precision, MateTee similarity measurement has the best quality among all the three measurements, and for all the considered thresholds. Regarding Recall, our method is the best up until a threshold of 0.7. For higher thresholds, e.g., 0.8, the recall rapidly goes down to 0.1, and to absolute 0 for 0.9. The explanation for this is that MateTee, being an optimization-based method, will always have an error as small as possible, so even if the neighborhoods of two entities are exactly the same, it is very unlikely to have similarities higher than 0.9 or 1.0, they will for sure be higher than between people which neighborhoods are absolutely different, but very unlikely be equal to 1.0. Then, using a threshold equal to 0.9, very few pairs of people will be considered, and with 1.0, absolutely no pairs are considered to count in the numerator of the Recall formula.

6 Related Work

Griver et al. [9] present **Node2Vec**, the latest method of the *everything-2-vec* saga. **Node2Vec** tackles the problem of giving a vector representation to nodes in graphs. Node2Vec focuses mainly on two common prediction tasks: *Multi-label classification of nodes*, where the objective is to classify new unknown nodes into one of the known classes, and *Link prediction* with the objective of predicting if a link i.e., relation, should be established (or re-established in case of incomplete datasets) between a pair of given nodes. Further, node2vec identifies the type of link. The main contribution and uniqueness of node2vec, compared with similar techniques, is the flexible notion it gives to the meaning of *neighbourhood*. It is based on the idea that nodes, and their connectivity patterns in the network, can be described based on two factors: First, on the *communities* to which they belong, i.e., *homophily* or essentially the set of their 1-hop neighbors, and second, on the role the nodes play in the network, i.e., *structural equivalence* or the type of node they are, e.g., border node, internal node, etc. Therefore, a node could have multiple neighbourhoods, and it can only be considered k of these neighbourhoods, the problem turns into how to sample them.

Based on *Breadth-first Sampling (BFS)* and *Depth-first Sampling (DFS)*, node2vec proposes a new sampling approach called *Random Walks*. It consists on explore the connectivity patterns based on both BFS and DFS manners, interpolating between both approaches based on a bias term. This idea comes from the

fact that the connectivity patterns in real life graphs, e.g., Social Networks, are not exclusively based on structural equivalence and *homophily*, but commonly in a mixtures of both. The bias term aims to accommodate the random walks to the actual structure of the sub-graph being analyzed.

Another publication also aiming on finding vector representations of entities in KGs, and that in fact was the predecessor of TransE, is the **Structured Embeddings** (SE), presented in [4] also by Bordes et al. SE is a method on which the vector representation of the entities is established by a neural network acting like a bridge between the entities in the original data and their feature representation. The fundamental characteristics of their approach includes *Flexibility* and domain independence, meaning that it should work and be easily adaptable for most of available KBs, and *Compactness* in the sense that each entity is assign one low-dimensional vector in the feature space, and only one matrix to each relation. As for TransE, SE considers relations between the entities, i.e., subjects and objects, as operators. If certain operation is performed in the feature space between the subject vector and the relation matrix, the resulting vector must be the object vector (or a nearest in its neighbor). The main difference between SE and further approaches from the same authors, e.g., TransE [3], is that SE models relations as pair of matrices. For TransE the relations are normal embeddings with the special characteristic that they are not normalized after each iteration, as for Subjects and Objects.

7 Conclusions and Future Work

In this paper we presented MateTee, a method to compare entities in knowledge graphs, based on the vectorization of the nodes, and specially without any domain expertise. To test the accuracy of MateTee, we compared its results with the state-of-the-art methods like GADES or OnSim, as well as state-of-the-art similarity measures available in the CESSM evaluation framework. MateTee exhibited high accuracy and competitive results, even outperforming the results of GADES, one of the best-performing similarity metric. This behavior was observed in the collections of proteins for UniProt, and the collection of persons from DBpedia. Therefore, observed results suggest that representing knowledge encoded in KGs in the embedding space and using vector based similarity metrics to compare the embeddings of KG entities, provides an accurate method for determining relatedness among entities in the KG.

In the future work, we plan to extend MateTee, to consider the implicit knowledge encoded in the ontologies used to describe a KG. Furthermore, MateTee will be modified to be able to produce values of similarity on-demand, i.e., MateTee workflow will start to work on a pre-training of the original data set. Thus, the optimization process will converge more rapidly whenever any new entity are added to the KG because all other embeddings are positioned in the vector space in a way that the error is already minimized.

Acknowledgments. This work is supported in part by the European Union under the Horizon 2020 Framework Program for the project BigDataEurope (GA 644564) as well as by the German Ministry of Education and Research with grant no. 13N13627 for the project LiDaKrA. We thank Mikhail Galkin for creating the DBpedia collection used in our experiments, and Ignacio Traverso Ribón for his support on the experimental comparison with GADES.

References

1. Benik, J., Chang, C., Raschid, L., Vidal, M.-E., Palma, G., Thor, A.: Finding cross genome patterns in annotation graphs. In: Bodenreider, O., Rance, B. (eds.) DILS 2012. LNCS, vol. 7348, pp. 21–36. Springer, Heidelberg (2012). doi:[10.1007/978-3-642-31040-9_3](https://doi.org/10.1007/978-3-642-31040-9_3)
2. Bernstein, A., Hendler, J.A., Noy, N.F.: A new look at the semantic web. *Commun. ACM* **59**(9), 35–37 (2016)
3. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) *Advances in Neural Information Processing Systems 26*, pp. 2787–2795. Curran Associates Inc. (2013)
4. Bordes, A., Weston, J., Collobert, R., Bengio, Y.: Learning structured embeddings of knowledge bases (2011)
5. Collarana, D., Galkin, M., Lange, C., Grangel-González, I., Vidal, M., Auer, S.: Fuhsen: a federated hybrid search engine for building a knowledge graph on-demand (short paper). In: Debruyne, C., et al. (eds.) *OTM Conferences - ODBASE*. LNCS, vol. 10033, pp. 752–761. Springer, Heidelberg (2016)
6. Couto, F.M., Silva, M.J., Coutinho, P.: Measuring semantic similarity between gene ontology terms. *Data Knowl. Eng.* **61**(1), 137–152 (2007)
7. Devos, D., Valencia, A.: Practical limits of function prediction. *Proteins: Struct., Funct., Bioinf.* **41**(1), 98–107 (2000)
8. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*. Society for Artificial Intelligence and Statistics (2010)
9. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks (2016). [arXiv:1607.00653](https://arxiv.org/abs/1607.00653). In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016)
10. Jiang, J.J., Conrath, D.W.: Semantic similarity based on corpus statistics and lexical taxonomy. In: *Proceedings of 10th International Conference on Research in Computational Linguistics, ROCLING 1997* (1997)
11. Lin, D.: An information-theoretic definition of similarity. In: *ICML*, vol. 98 (1998)
12. Lin, D.: An information-theoretic definition of similarity. In: *Proceedings of the Fifteenth International Conference on Machine Learning, ICML 1998*, San Francisco, CA, USA, pp. 296–304. Morgan Kaufmann Publishers Inc. (1998)
13. Lord, P., Stevens, R., Brass, A., Goble, C.: Investigating semantic similarity measures across the gene ontology: the relationship between sequence and annotation. *Bioinformatics* **19**, 1275–1283 (2003)
14. Pekar, V., Staab, S.: Taxonomy learning: factoring the structure of a taxonomy into a semantic classification decision. In: *COLING 2002 Proceedings of the 19th International Conference on Computational Linguistics*, vol. 2, pp. 1–7. Association for Computational Linguistics (2002)

15. Pesquita, C., Faria, D., Bastos, H., Falcão, A.O., Couto, F.M.: Evaluating go-based semantic similarity measures. In: Proceedings of the 10th Annual Bio-Ontologies Meeting (BIOONTOLOGIES), pp. 37–40 (2007)
16. Pesquita, C., Pessoa, D., Faria, D., Couto, F.: Cessm: collaborative evaluation of semantic similarity measures. *JB2009: Challenges Bioinform.* **157**, 190 (2009)
17. Resnik, P.: Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language. *J. Artif. Intell. Res.* **11**, 95–130 (1998)
18. Sevilla, J.L., Segura, V., Podhorski, A., Guruceaga, E., Mato, J.M., Martínez-Cruz, L.A., Corrales, F.J., Rubio, A.: Correlation between gene expression and go semantic similarity. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2**(4), 330–338 (2005)
19. Smith, T., Waterman, M.: Identification of common molecular subsequences. *J. Mol. Biol.* **147**(1), 195–197 (1981)
20. Traverso-Ribón, I., Vidal, M.-E., Kämpgen, B., Sure-Vetter, Y.: Exploiting relation and class taxonomy semantics to compute similarity in knowledge graphs. In: SEMANTICS (2016)
21. Traverso-Ribón, I., Vidal, M.-E.: Exploiting information content and semantics to accurately compute similarity of go-based annotated entities. In: IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology, CIBCB (2015)
22. Traverso-Ribón, I., Vidal, M.-E., Palma, G.: OnSim: a similarity measure for determining relatedness between ontology terms. In: Ashish, N., Ambite, J.-L. (eds.) DILS 2015. LNCS, vol. 9162, pp. 70–86. Springer, Cham (2015). doi:[10.1007/978-3-319-21843-4_6](https://doi.org/10.1007/978-3-319-21843-4_6)