Knowledge Graph Prediction using Negative Statements: an Approach Based on Entity-nearest Neighbor Count Algorithm

Furel D. Teguimene[1,*,†], AZANZI Jiomekong[1,†], Sanju Tiwari[2] and Gaoussou Camara[3]

[1]Department of Computer Science, University of Yaounde I, Yaounde, Cameroon
[2]Teerthanker Mahaveer University, Universidad Autonoma de Tamaulipas - Matamoros
[3]Alioune Diop University of Bambey

Abstract

This paper presents our contribution to knowledge graph predictions using negative statements (NEGKNOW) challenge. This contribution consists of the definition of the Entity-nearest Neighbor Count (E-NNC) Algorithm. In this algorithm, we consider that if two entities are in relation, then, they linked to least one common entity. The list of common entities between two entities are called their common neighbors. Thus, the algorithm calculates the common neighbor between two entities. The E-NNC algorithm defined in this work was applied for the three tasks of the NEGKNOW challenge. These tasks consists of predicting if there is an interaction between two proteins (Task A), a protein and a disease (Task B) and a gene and a disease (Task C). The organizers of this challenge provided the train and the test set. The algorithm assessed on the train set to evaluate its performance. For task A, the algorithm proves to be powerful because we obtained an accuracy of 0.9. For Task B, an accuracy of 0.9 and 0.5 for task C.

Keywords

Entity-nearest Neighbor Count, Knowledge Graph, Knowledge Graph Completion, Relation Prediction

1. Introduction

Knowledge graphs (KGs) are composed of real world entities and relations between these entities [1, 2]. However, knowledge graphs suffers the problem of completeness [3, 4, 2, 5]. Actually, in many KGs such as Freebase DBpedia, Yago, a considerate number of important information are missing [3]. Therefore, there is an urgent need for methods to automatically complete KGs by inferring missing knowledge such as missing entities or missing relations.

Knowledge graph completion aims to identify missing information, such as missing links between entities or missing entities and use these elements to complete the graph [4, 2]. In the biological domain for instance, it may be interesting to predict if two proteins are in relation [4].

Relation prediction or relation linking [5, 4] aims to learn a relation between two KG entities when the relation itself is not explicitly defined in the KG. The knowledge graph predictions using negative statements challenge[1] (NEGKNOW) challenge aims to evaluate systems handling negative statements in knowledge graphs (KGs) during the relation prediction task.

In this paper, we present our contribution to the NEGKNOW challenge. This contribution consists of a method for predicting relations between two entities by identifying the relations that these entities have in common (or if these entities have the same neighbor). In this approach, when two entities have a common neighbor, then they can be related. This approach was applied to the three prediction tasks proposed by the challenge organizers: (1) Protein-protein interaction prediction, (2) Gene-Disease Association Prediction, (3) Disease Prediction. A set of experiments were done to find out how the parameter can be configured to help to have good predictions.

NEGKNOW@ISWC'24: Challenge presentation during ISWC, November 11-15, 2024, Baltimore, US
*Corresponding author.
†These authors contributed equally.

[1]https://negknow.github.io/NEGKNOW/index.html

In the rest of the paper, we present the methodology (Section 2), the results (Section 3) and the conclusion (Section 4).

2. Entity-Nearest Neighbor Count Algorithm

In this work, we consider that if two entities are related, then they have at least one entity they are linked to (an example is presented by the Fig. 1). We call this entity the common neighbor of the two entities. The equation $CN(E_1, E_2) = |a|$ is used to calculate the number of neighbors the entities E_1 and E_2 have in common. The Entity-Nearest Neighbor Count (E-NNC) algorithm 1 calculates the number of common neighbors between two entities. This algorithm is going to be used for calculating the common neighbor between entities and predict if these entities are related for the different tasks.

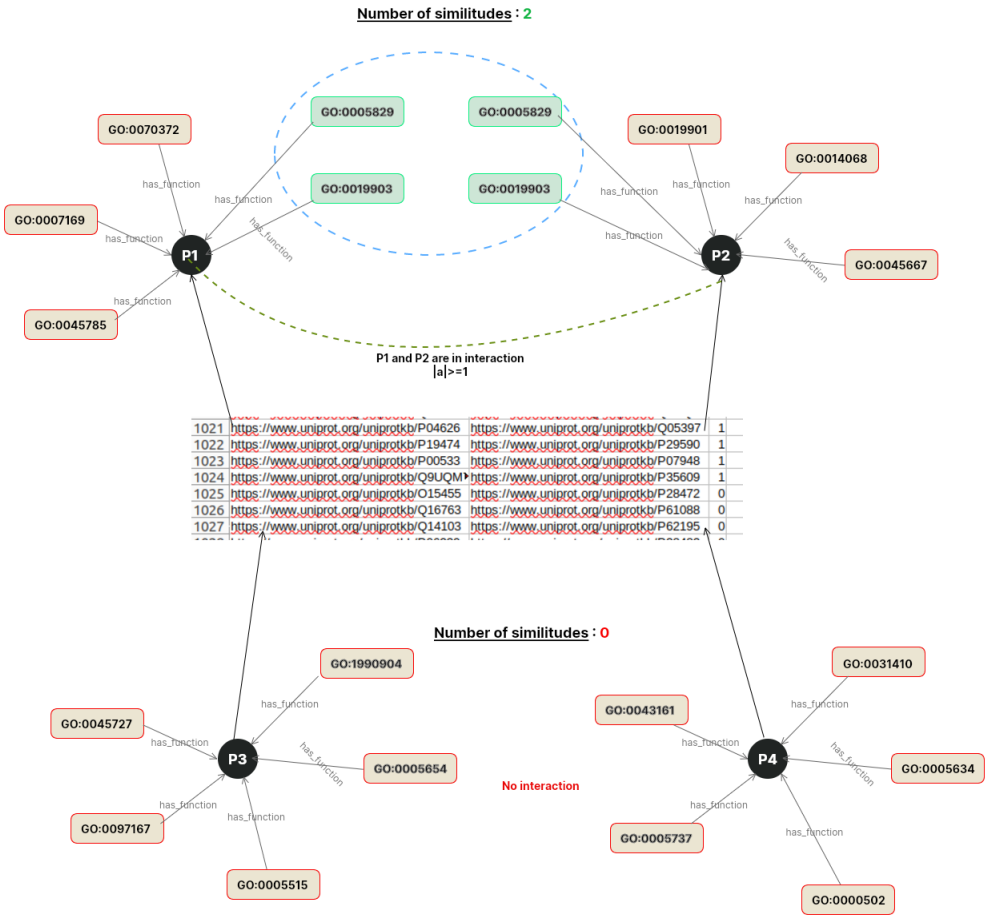

Figure 1: Prediction a relation between entities E_1 and E_2 based on the number of common neighbor

Algorithm 1 E-NNC algorithm

Require: E_1, E_2, KG
Ensure: $nbOfCommonNeighbor$
  $elementsE1 : Table[1...n]$ of entity
  $elementsE2 : Table[1...m]$ of entity
  $i \leftarrow -1$
  $nbOfCommonNeighbor \leftarrow 0$
  while Non $EOF(1)$ **do**
    $i \leftarrow i + 1$;
    if $LireFichier1, element$ is an E1 element **then**
      $elementsE1[i] \leftarrow element$
    else if $LireFichier1, element$ is an E2 element **then**
      $elementsE2[i] \leftarrow element$
    end if
  end while
  for $i \leftarrow$0 to n **do**
    for $j \leftarrow$0 to m **do**
      if $elementsE1[i]$ == $elementsE2[j]$ **then**
        $nbOfCommonNeighbor \leftarrow nbOfCommonNeighbor + 1$
      end if
    end for
  end for

2.1. Hardware, software and programming language

During this challenge, our working environment was as follows:

- A Dell Inc. Latitude 5580 laptop, with an Intel® Core™ i7-7820HQ processor clocked at 2.90 GHz with 8 cores, 16.0 GB of RAM and a disk capacity of 512 GB ;
- The work was carried out on an Ubuntu 22.04.4 LTS operating system ;
- The Java language (jdk-20) is the one used for this challenge executed in the IntelliJ IDEA IDE version 2023.

2.2. Protein-protein interaction prediction

Task A aims to predict whether there is a relationship between two given proteins. In this section, we considered two data sources, the GO KG provided by the organizers and the STRING database. The adaptation of the E-NNC algorithm is presented by the algorithm 2. In this algorithm, we consider two parameters:

- The number of neighbors that two entities have in common in the GO KG which we named $|a|$.
- The score that measures if two entities are similar, produced by the STRING database which we named s. Actually, the interaction between two proteins is described in the STRING database using a confidence score.

This algorithm returns 1 when the two entities are in relation and 0 when it is not the case.

Algorithm 2 Protein-protein interaction prediction using E-NNC algorithm

Require: $P\_1, P\_2, gokg, |a|, s$
Ensure: $prediction$
  $prediction \leftarrow 0$
  $score \leftarrow 0.472$
  $nbOfCommonNeighbor \leftarrow 1(P\_1, P\_2, gokg)$
  if ($nbOfCommonNeighbor \geq |a|$) and ($s \geq score$) **then**
    $prediction \leftarrow 1$
  end if

2.3. Patient-disease Interaction Prediction

Task B consists of predicting if a patient has already been diagnosed with a given disease. Thus, to predict if there is the relation "hasBeendiagnosed" between a patient and a disease. The algorithm **??** presents how this interaction is being predicted. The organizers provided the HP (Human Phenotype Ontology) knowledge base (KB). From this KB, we found that:

- Each patient and each disease are associated with phenotype;
- A patient and a disease can have phenotype in common.

Thus, the algorithm was adapted as follows: when a patient and a disease has a phenotype in common, they are related (See the E-NNC algorithm 1).

The generic algorithm can therefore be modified here by adding a parameter a. This parameter will allow us to check whether the number of common phenotypes calculated is equal to or greater than the expected value. In this task, the only value of $|a|$ used is 1.

Algorithm 3 Patient-disease interaction prediction using E-NNC algorithm

Require: $P, D, hpkg, |a|$
Ensure: $prediction$
  $prediction \leftarrow 0$
  $nbOfCommonNeighbor \leftarrow 1(P, D, hpkg)$
  if $nbOfCommonNeighbor \geq |a|$ **then**
    $prediction \leftarrow 1$
  end if

The following figure shows an example of application of the algorithm.

2.4. Gene-Disease Interaction Prediction

Task C consists of predicting if there is an interaction between a gene and a disease. To this end, we determined an identical number of elements linked to both the gene and the disease. Actually, the analysis of the HP KB allowed us to remark that: A gene is linked to a set of GO terms, a disease is linked to a set of phenotypes, each GO term and each phenotype are objects that can have common restrictions. The E-NNC algorithm was therefore adapted (see algorithm 4). It takes as parameter the variable $|a|$ which is the number of similarities.

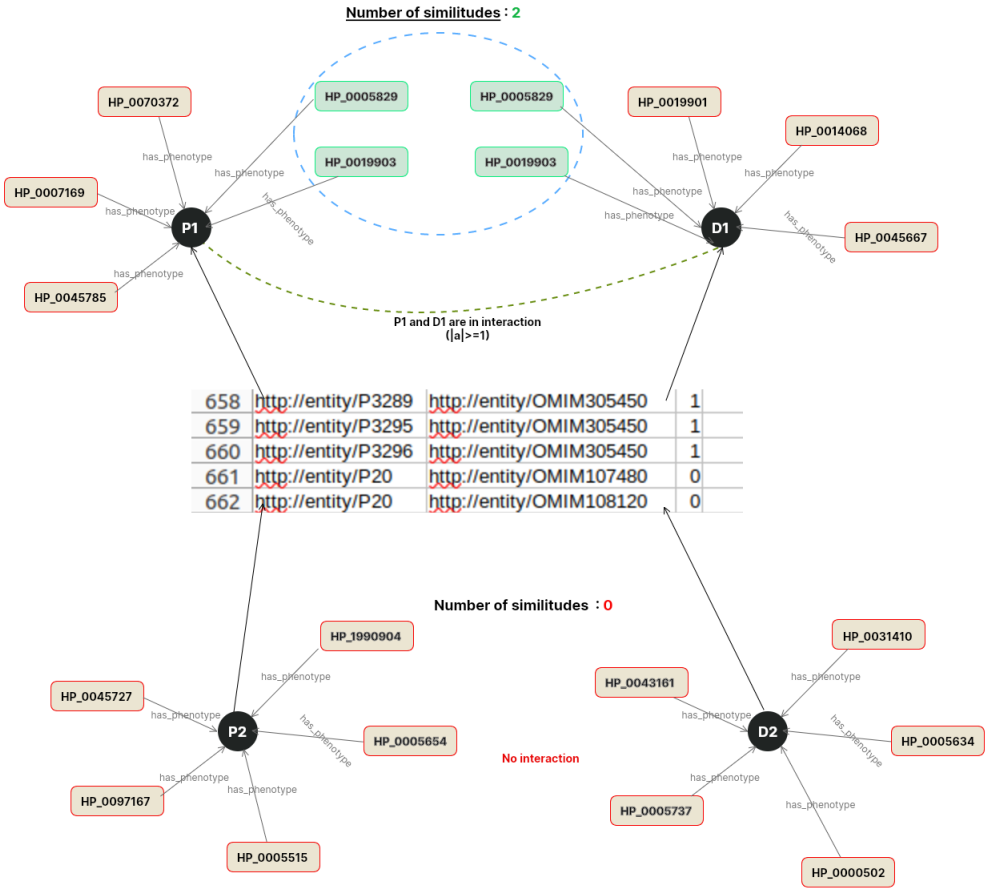

Figure 2: Interaction between patient and disease using only HP (KG)

Algorithm 4 Gene-disease interaction prediction using E-NNC algorithm

Require: $G, D, gohpkg|a|$
Ensure: $prediction$
  $prediction \leftarrow 0$
  $nbOfCommonNeighbor \leftarrow 1(G, D, gohpkg)$
  if $nbOfCommonNeighbor \geq |a|$ **then**
    $prediction \leftarrow 1$
  end if

The following picture represents an application of this algorithm.

3. Result

This Section presents the results during the experimentations. Section 3.1 presents the results for Task A, Section 3.2 the results for Task B and Section 3.3 the results for task C.

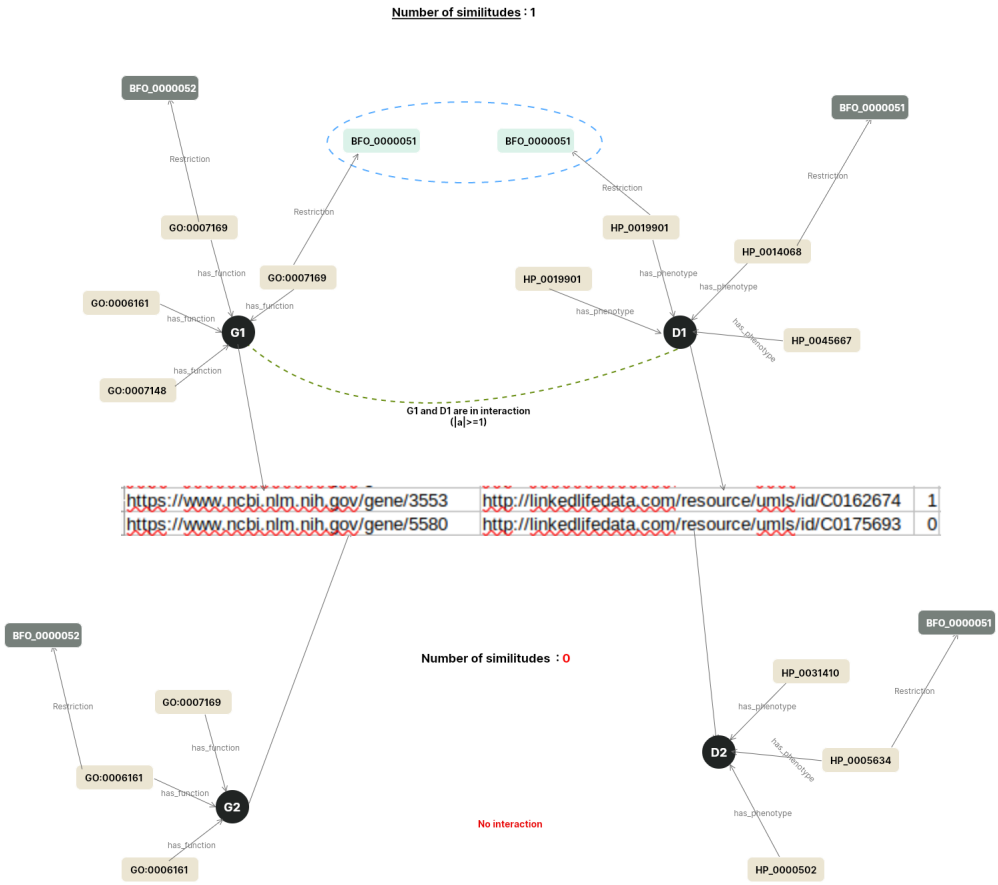

Figure 3: Interaction between patient and disease using only HP (KG)

3.1. Result for task A

The algorithm defined in Section 2 was applied on the dataset for different values of the parameter $|a|$ and $|s|$. The table 1 presents the results obtained for different values of the parameters $|a|$ and $|s|$. This table shows that for the best score is obtained when $|a| = 0$ and $|s| = 0.65$. Thus, when a relation exists between two entities and the string database predict with the minimum score that this relation exist, then it is more probable that this relation exists.

a \s	≥ 0.472	≥ 0.55	≥ 0.65	≥ 0.70
≥ 1	0.92	0.9204	0.9106	-
≥ 2	**0.947**	-	0.9086	-
≥ 7	-	-	-	0.835
≥ 8	-	-	-	0.801

Table 1
A descriptive summary of the result obtained in the task A (primary table)

Ablation To study the impact of the different datasets on the results, we decided to test on each of them. Then, for the ablation study, on the other hand, we will consider that $|a| = 0$ (the GO knowledge graph is not used) and on the other hand, $|s| = 0$ (the string knowledge graph is not used). The tables 2, 3 and 1 presents the different results obtained by considering the different source of knowledge. The table 3 presents the results when $|a| = 0$ and the table 2 presents the results when $|s| = 0$. These tables show that the highest values of the accuracy are obtained when the two knowledge base are combined.

$\|a\|$	≥ 7	≥ 8
Accuracy	0.79	0.76

Table 2
A descriptive summary of the result obtained in the task A Using only GO (KG)

STRING DB score (s)	≥ 0.472	≥ 0.55	≥ 0.65	≥ 0.7	≥ 0.8
Accuracy	0.9405	0.940	**0.943**	0.903	0.8609

Table 3
A descriptive summary of the result obtained in the task A Using only the score of STRING Database

The experimentations show that the best precision is obtained when one combine the GO Knowledge Graph with the STRING database. Thus, the results submitted for evaluation was the one obtained by the combination of these data sources.

3.2. Result for Task B

The model defined in Section 2.3 was applied to 10% of the train dataset for an unique value of the parameter $|a|$. The table 4 presents the results obtained for $|a| \geq 1$. This table shows that for the best score is obtained when $|a| = 1$. Thus, when a patient and a disease have at least one phenotype in common, it is very likely that the latter has already been diagnosed with this disease.

$\|a\|$	≥ 1
Accuracy	**0.901**

Table 4
A descriptive summary of the result obtained in the task B using HP (KG)

3.3. Result for Task C

The model defined in Section 2.4 was applied to the dataset for a unique value of the parameter $|a|$. The table 5 presents the results obtained for $|a| \geq 1$. This table shows that for the best score is obtained when $|a| = 1$. Thus, when a GO's term and a phenotype have at least one restriction in common, it is very likely that the gene and the disease are related.

$\|a\|$	≥ 1
Accuracy	**0.53**

Table 5
A descriptive summary of the result obtained in the task C using GO (KG) and HP (KG)

4. Conclusion

The knowledge graph predictions using negative statements (NEGKNOW) challenge aims to evaluate systems handling negative statements in knowledge graphs (KGs) during the relation prediction task.

In this paper, we present a new algorithm (entity-nearest neighbor count (E-NNC) algorithm consisting of counting the number of neighbors that two entities have in common and predict that these entities are related. We applied this algorithm on the three tasks of the NEGKNOW challenge and the results obtained were promising.

Future work consists of comparing this algorithm to other algorithms used for the same task such as TransE, TransH, TransR, DistMult, etc.

References

[1] A. Jiomekong, F. Asong, Designing, implementing and deploying an enterprise knowledge graph from a to z, in: Proceedings of the Federated Africa and Middle East Conference on Software Engineering, FAMECSE '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 87–88. doi:10.1145/3531056.3542761.

[2] P. Cimiano, H. Paulheim, Knowledge graph refinement: A survey of approaches and evaluation methods, Semant. Web 8 (2017) 489–508. doi:10.3233/SW-160218.

[3] T. Madushanka, R. Ichise, Negative sampling in knowledge graph representation learning: A review, 2024. URL: https://arxiv.org/abs/2402.19195. arXiv:2402.19195.

[4] R. T. Sousa, S. Silva, H. Paulheim, C. Pesquita, Biomedical knowledge graph embeddings with negative statements, in: T. R. Payne, V. Presutti, G. Qi, M. Poveda-Villalón, G. Stoilos, L. Hollink, Z. Kaoudi, G. Cheng, J. Li (Eds.), The Semantic Web – ISWC 2023, Springer Nature Switzerland, Cham, 2023, pp. 428–446.

[5] A. Jiomekong, B. Foko, U. M. Vadel Tsague, G. Camara, Towards an approach based on knowledge graph refinement for relation linking and entity linking, in: SMARTTask 2022, SMARTTask@ISWC, 2022.

5. Online Resources

The source code used in this work is available on github[2] under the Apache License, Version 2.0

[2] https://github.com/Teguimene/NEGKNOW_CHALLENGE