# GrCluster: A score function to model hierarchy in knowledge graph embeddings

Anonymous Author(s)

## ABSTRACT

Low-dimensional embeddings for knowledge graph entities and relations help preserve their latent semantics while enabling computation efficiency. These embeddings are often used to perform tasks such as machine translation, sentiment analysis, knowledge graph completion, and information extraction. Knowledge graph embedding methods aid in the representation of entities and relationships of a knowledge graph in continuous vector spaces. However, most existing techniques ignore the inherent hierarchical structure of entities of the knowledge graph, defined by ontological relationships between entity types. This paper introduces a novel score function called GrCluster that helps fill that gap. GrCluster is a simple, intuitive and efficient scoring function that considers the hierarchy of entities of a knowledge graph. The effectiveness of GrCluster is demonstrated by integrating it into several well known embedding models. The experimental results show consistent improvements across metrics and embedding models for the tasks of entity prediction and triplet classification.

## CCS CONCEPTS

• **Computing methodologies** → **Ontology engineering**; *Continuous space search*; Statistical relational learning;

## KEYWORDS

knowledge representation, knowledge graph embeddings, representation learning, relational learning, hierarchy, wordnet

**ACM Reference Format:**

Anonymous Author(s). 2020. GrCluster: A score function to model hierarchy in knowledge graph embeddings. In *Proceedings of ACM SAC Conference (SAC'20).* ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/xx.xxx/xxx_x

## 1 INTRODUCTION

Recent years have witnessed a phenomenon known as Information Explosion [2], which is the rapid increase in the amount of published data on the internet. Consequently, the size of publicly available Knowledge Graphs (KGs) such as WordNet [20], Freebase [7] and DBpedia [4] is exponentially increasing. These KGs store information in the form of triplets, which can represent facts describing entities (e.g., the date of birth of a person) or relationships between entities (e.g., the city where a person was born). Applications need large and accurate KGs, e.g., for tasks like question answering, but are incomplete [22] because it is hard to build them by hand and the best Information Extraction (IE) algorithms are still far from accurate enough. Reasoning with a closed-world assumption will result in substandard performances. Machine Learning algorithms for KG reasoning and completion rely on developments in Representation Learning. Representation Learning or Feature Learning [5] is a set of techniques that allows the machine to obtain representations that effectively preserve the semantics of objects while overcoming related issues of storage and computation.

Knowledge graph embedding [30] is a new direction of research that focuses on finding efficient embeddings for all entities and relationships in a knowledge graph. The key concept involves embedding components of a knowledge graph into continuous vector spaces to reduce computation during manipulation while preserving the latent structure of the knowledge graph. These embeddings can be further used in tasks such as knowledge graph completion [27], relation extraction [14], entity classification and entity resolution [33]. Although results obtained in previous works in the predictive tasks are quite satisfactory, most techniques ignore the entity hierarchical correlation in the knowledge graph. Encoding relationships while including information about the hierarchical nature of entities could produce more robust embeddings that can improve performance in predictive tasks such as entity prediction and triplet classification. This has led to the proposal of a novel score function that can be applied to the objective function of any existing knowledge graph embedding technique to improve obtained embeddings for entities and relations.

This paper proposes GrCluster, a novel score function that takes into consideration the hierarchy of entities in a knowledge graph. GrCluster is a simple and computationally inexpensive score function that employs only one additional parameter. The proposed score function may be incorporated in the objective function of any knowledge graph embedding method to obtain more robust embeddings. These embeddings incorporate the inherent entity hierarchy present in the knowledge graph.

The outline of this paper is as follows. Section 2 summarizes a few related knowledge graph embedding techniques introduced in previous papers. In section 3, the motivation and intuition behind GrCluster is explained. The mathematical formula for the GrCluster score function is derived and presented. Results from various experiments are presented in section 4. Section 4.1 describes the implementation details. Section 4.2 demonstrates the ability of the GrCluster score function to model hierarchy. Section 4.2 further demonstrates the improvement in performance on including GrCluster in the objective function. In section 4.3 and section 4.4, the proposed score function is included to modify the objective function of several embedding methods. Experimental results on



*SAC'20, March 30-April 3, 2020, Brno, Czech Republic*



ACM ISBN 978-1-4503-6866-7/20/03…$15.00
https://doi.org/xx.xxx/xxx_x

the tasks of entity prediction and triplet classification show improvements when the GrCluster score function is included in the objective function of the embedding methods. Section 5 provides the concluding remarks and the future scope of the proposed score function.

## 2 RELATED WORK

In this section, a few related knowledge graph embedding models have been summarized.

Nickel et al. [24] introduced RESCAL, a collective matrix factorization model which associates each entity with a vector to capture its latent inherent semantics. Relations are represented by matrices which represent the pairwise interactions between each entity vector. A score of a triplet (h, r, t) is given by $h^T M_r t$. Embeddings are obtained by maximizing the score for positive triplets and minimizing the score for negative triplets. Due to the computationally intensive nature of the model, this method is quite slow and does not scale well to large datasets [8].

Yang et al. [35] introduced DistMult, which is an extension of RESCAL that constraints the relation matrix to a diagonal matrix. The score function used by DistMult is given by $h^T diag(M_r)t$. This takes into account the pairwise interactions between the components of $h$ and $t$ which are along the same dimension. Although the constraint helps in reducing the number of trainable parameters, this simplified version of RESCAL cannot handle asymmetric relations.

Trouillon et al. [29] extends DistMult to develop ComplEx by introducing complex valued embeddings to improve asymmetric relation embeddings. Entity and relation embeddings lie in the complex space to model inverse relations. By enabling representation of real and imaginary numbers, the performance on antisymmetric relations such 1-N and N-1 type relations improved significantly from DistMult.

Nickel et al. [23] introduced the Holographic Embeddings that combine the power of RESCAL with the efficiency of DistMult by representing both entities and relations as vectors. They make use of the circular correlation operation to compress the pairwise interactions. Performance of HolE on asymmetric relations is comparable to RESCAL.

Bordes et al. [8] introduced TransE, which is the first and the most popular translational distance embedding model. TransE represents both entities and relations in the same space. Given a fact represented by the tuple $(h, r, t)$ where $h$ is the head entity embedded vector, $t$ is the tail entity embedded vector and $r$ is the relation embedded vector respectively, TransE views the relation as a vector which can be used to connect the head and tail entity embeddings using translation. In other words, it tries to optimize the embeddings such that $h + r = t$. It solves this optimization problem by minimizing $h + r - t$, while searching for optimal embeddings. Although this method worked better than previous semantic matching models, it failed to perform well over 1-N, N-1 and N-N relations. TransE established a foundation for most of the translation distance models.

To overcome the problems faced by TransE, Wang et al. [31] introduced TransH and the idea of relation specific entity embeddings. TransH proposed the idea of relationship specific hyperplanes. It maps the head entity embedded vector and the tail entity embedded vector to a relationship specific hyperplane and then minimizes $h_\perp + r - t_\perp$ where $h_\perp$ is the projected head entity embedded vector, and $t_\perp$ is the projected tail entity embedded vector. TransH enabled an entity to take on different roles when involved with different relationships, which resulted in an improved performance in 1-N, N-1, and N-N relationship types.

Lin et al. [19] introduced TransR, which extended the idea of relation specific entity embeddings. TransR differs from TransH by projecting entities to a relation specific space instead of hyperplanes. In TransR, entities and relations are associated with two different spaces and are modeled as translator vectors in their respective spaces. The more expressive nature of TransR promoted performance improvements in the task of entity prediction. The increase in the number of trainable parameters and matrix-vector multiplication operations caused a significant increase in training time.

TransD was introduced by Ji et al. [15] to overcome the problem of computational inefficiency associated with TransR. TransD proposed a simplification of TransR where the projection matrix is decomposed into a product of two vectors. In TransD, two vectors are defined for each entity and relation. The first vector represents the meaning of the entity or the relation. The second vector, also called the projection vector, represents how to project an entity embedding into the relation vector space and it will be used to construct the mapping matrices. Every entity-relation pair will have a unique mapping matrix. Therefore, this method utilizes less parameters and has no matrix-vector multiplication operations which can be applied to large scale graphs.

Xie et al. [34] introduced the Type-embodied Knowledge Representation Learning (TKRL) which takes into account the fact that a single entity can have different types associated with it. Entities in this model thus have a different representation based on their type. Type-specific projection matrices are constructed from the hierarchical type encoders for each type. After setting up projection matrices for each type of entity, the model is trained to minimize $M_{rh}h + r - M_{rt}t$ where $M_{rh}$ is the projected embedded vector for the head entity, $M_{rt}$ is the projected embedded vector for the tail entity, and $r$ is the relation embedded vector. Due to matrix-vector multiplication and the inclusion of hierarchical type-encoders, this method is computationally inefficient to scale for large scale knowledge graphs. This paper does not compare with this method since TKRL is a knowledge graph embedding model and the GrCluster is a score function that can be utilized with any knowledge graph embedding model.

Zhang et al. [36] introduced the concept of leveraging the rich information present in the relationship structure. They divide a relation into a three layered structure, which captures the semantic meaning of the relation to a very large extent. They proposed a novel score function which encodes the hierarchical relation structural data into the relation embedding space. The proposed approach follows a similar intuition, but aims to model the hierarchical structure of entities.

Poincaré embeddings, introduced by Nickel and Kiela [21], place the connected nodes of a graph close to each other and unconnected nodes far from each other. It models hierarchy by placing nodes high in the hierarchy close to the origin and nodes lower in the

hierarchy farther from the origin. This is achieved by defining a loss function that penalizes low distances between unconnected nodes and high distances between connected nodes, and then training over the list of connected nodes, along with randomly sampled negative examples, using gradient descent [26]. These embeddings are searched for in the hyperbolic space. This is because hierarchies are captured better in hyperbolic space that in Euclidean space. Moreover, capturing the exact distance between nodes in Euclidean space requires higher dimensions. Ganea et al. [11] introduces the hyperbolic entailment cones to overcome the problem of encoding asymmetric relations faced in the Poincaré Embedding model. They present an efficient algorithm for learning hierarchical embeddings of a directed acyclic graph. These methods learn higher quality embeddings and improves experimental results. Although both methods achieve satisfactory results in the task of entity prediction, they do not support modeling of relationships other than the parent-child relationship.

To the best of our knowledge, the first use of a hierarchical loss function was originally introduced in the context of document categorization with support vector machines[9]. However, that work assumed that weights to control the hierarchical loss would be solicited from domain experts which is not applicable to our setting.

# 3 GRCLUSTER: SCORE FUNCTION TO MODEL HIERARCHY

## 3.1 Motivation

The main motivation behind the proposed score function is two fold. The first motivation arises from the common presence of hierarchical relationships in knowledge graphs [25]. Due to the incomplete nature of datasets, this hierarchy may not be modeled properly. This leads to a compromised performance in predictive tasks [10]. The second motivation is derived from the concept of inheritance from the object-oriented programming paradigm [32]. The concept of inheritance allows an instance of a child class to substitute any instance of its ancestor class. Similarly, in a knowledge graph, a child entity is a specialization of a parent entity. For example, an entity 'red_car' can replace any entity 'car' without any change in the original semantics. Upon realization, the parent entity acts as a representative for all its descendants. This results in the formation of clusters containing entities, where each cluster can be represented by a single entity. Consequently, a child entity may act as a representative of a cluster comprising of its descendent entities. This forms a recursion of clusters. Modeling the hierarchy in such a manner would also model the correlation between entities leading to more robust representations in lower dimensional embedding spaces.

The inheritance concept also states that an instance of a parent class cannot be cast as an instance of its child class. This is because the child class may have additional functionality and specializations that are not modeled by any of its ancestor classes. Similarly, in a knowledge graph, both the parent entity and the child entity do not refer to the same real world object. For example, the entity 'ball' cannot substitute 'golf_ball' without any change in semantics. This dissimilarity can be illustrated through the concept of distance. The distance between two entities in a cluster can be measured by the number of edges involved while traversing over a hierarchical tree from one entity to the other entity. To model the entity hierarchy in a knowledge graph as a set of recursive clusters, the GrCluster score function has been proposed.

## 3.2 Deriving GrCluster

To describe the similarity between any two entities connected by a single edge in a hierarchical tree, the term discount factor, denoted by $\gamma$, has been introduced. This similarity measure is constrained to be between 0 and 1. To capture the similarity between any two connected entities in a hierarchical tree, $\gamma^d$ is computed where $d$ is the distance in terms of number of edges connecting the head entity and the tail entity.

To arrive at the similarity between any two entities in the hierarchy tree, a function $\Phi$ is defined. $\Phi$ is a function that takes as input the discount factor $\gamma$ and the number of edges between entities $d$. The similarity between any two entities $e_1$ and $e_2$ can be decomposed into the product of two terms. The first term is the similarity between entity $e_1$ and a neighbouring entity $e_n$, closest to $e_2$. The second term is the similarity function $\Phi$ between $e_n$ and $e_2$.

$$\Phi(\gamma, d) = \Phi(\gamma, 1).\Phi(\gamma, d-1) \tag{1}$$

From the definition of discount factor, $\Phi(\gamma, 1) = \gamma$.

$$\Phi(\gamma, d) = \gamma.\Phi(\gamma, d-1) \tag{2}$$

From equation 1,

$$\Phi(\gamma, d) = \gamma.\gamma.\Phi(\gamma, d-2) \tag{3}$$

$$...$$

$$\Phi(\gamma, d) = \gamma.\gamma.\gamma...(d-1\ times).\Phi(\gamma, 1) \tag{4}$$

$$\Phi(\gamma, d) = \gamma^d \tag{5}$$

If the head entity and the tail entity are the same, there are no edges to consider. Hence, $d$ will be 0.

$$\Phi(\gamma, 0) = \gamma^0 = 1 \tag{6}$$

This similarity value of 1 implies that the two entities are the same. For immediate neighbours, $d$ will equal 1, resulting in the original definition of the discount factor.

$$\Phi(\gamma, 1) = \gamma^1 = \gamma \tag{7}$$

For two disconnected entities, that is, there does not exist a sequence of edges which starts with $e_1$ and ends with $e_2$, the value for $d$ will tend to $\infty$.

$$\Phi(\gamma, \infty) = \lim_{d \to \infty} \gamma^d = 0 \tag{8}$$

$\Phi(\gamma, \infty)$ gives 0 which implies that there is no hierarchical correlation [12] between the two entities.

The range of the function $\Phi$ is constrained by the domain of $\gamma$, that is, $\Phi$ takes a value between 0 and 1. Since $\Phi$ gives a similarity measure between two entities, a distance measure $\Phi'$ can be given by

$$\Phi'(\gamma, d) = 1 - \Phi(\gamma, d) \tag{9}$$

The hierarchical information involving two entities must be encoded over all dimensions of the relation vector. A vector matching optimization problem can be formulated such that the distance between the two entities in all dimensions is equal to the hierarchical distance given by equation 9. Since $\Phi'$ outputs a scalar value, the

relation vector has been reduced to its mean over all dimensions. In addition to converting the vector into a scalar, this operation allows embedding of hierarchical information over all dimensions.

The GrCluster score, denoted by $L_{GrCluster}$, is given by

$$L_{GrCluster}(r, \gamma, d) = [(\frac{1}{N}\Sigma_{i=1}^{N}||r||_2) - \Phi'(\gamma, d)]_+ \quad (10)$$

where $||r||_2$ is the l2-normalized relation embedding vector, $N$ is the embedding vector size, $\gamma$ is the discount factor, $d$ is the hierarchical distance between the two entities and $[x]_+$ denotes the positive part of $x$. Since a child may substitute for any ancestor, the length from child to ancestor becomes zero, complying to the second motivation.

The discount factor is the only additional parameter required to model the hierarchical correlation between two entities. Two variants of the GrCluster score function are possible:

(1) Hyperparameter: The discount factor can be specified as a constant between 0 and 1.
(2) Parameter: The discount factor can be a trainable variable activated by the sigmoid activation function [17] given by

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (11)$$

GrCluster score function can be incorporated in the objective function of any knowledge graph embedding model using the addition operation as shown in equation 12.

$$L = L_{Original} + L_{GrCluster} \quad (12)$$

where $L$ is modified objective function, $L_{Original}$ is the objective function followed by the method originally, and $L_{GrCluster}$ is the proposed scoring function. It is important to note that the goal is to improve existing methods instead of proposing a new one.

# 4 EXPERIMENTS

## 4.1 Experimental Details

The GrCluster score function requires a distance measure between the two entities in the hierarchical tree. Given the set of triplets available in a knowledge graph, a forest of hierarchy trees was generated. After generating this forest, the distance between any two entities could be derived. To perform this task, two different algorithms were employed. Details about the algorithms can obtained from the supplementary material. The performance of the trained models were evaluated on the tasks of entity prediction [8] and triplet classification [28].

The Tensorflow [1] package was used to implement the knowledge graph embeddings methods. To ensure fair evaluation of all models, the training hyperparameters chosen for every model was kept constant. For experiments that involve the GrCluster score function in the objective function, selected values for the discount factor $\gamma \in \{0.1, 0.25, 0.5, 0.75, 0.9\}$. The parameterized variant of GrCluster was also used, represented by $\theta$, where the value of the discount factor is automatically searched using gradient descent [26]. Embedding models trained with the GrCluster score function have a 'Gr' prefixed to the model name. Hyperparameters that remained constant throughout the experiments include learning rate $\lambda$ = 0.001, margin $\epsilon$ = 1, embedding size for entities and relations $N$ = 100, batch size $B$ = 1024. All models were trained using the Adam Optimizer [18] as suggested by Kadlec et al. [16]. Experiments were performed using both the Uniform and the Bernoulli sampling techniques to create negative samples for training. The raw and the filtered test settings were used. The metrics used for evaluation are Mean Reciprocal Rank (MRR), Hits@3 (H3) and Hits@10 (H10).

Codes that were used for all experiments along with detailed experimental results and generated embeddings are available at http://www.tinyurl.com/grcluster-supplement.

## 4.2 Modeling the transitive closure of the Wordnet Noun Hierarchy with TransE

This experiment studies the ability of the GrCluster score function in modeling the latent hierarchical structure in a knowledge graph. The Wordnet Noun Hierarchy dataset (WNNH) was created using the script provided by Nickel and Kiela [21]. Since the dataset did not explicitly contain a train, validation and test split, 25,000 samples were randomly picked for the validation and test split, allowing the model to train with the remaining samples. Due to the presence of only a single relationship, the Uniform sampling method was employed to corrupt the golden triplets while using the TransE embedding model. The baseline for this experiment is TransE trained with its original objective function. GrTransE embedding models are trained by including the GrCluster score function in the original objective function in the manner described by equation 12.

On thorough exploration of the WNNH dataset, it was found that the tuples were of the form (*child_entity*, *is_a*, *parent_entity*). Since a parent entity may have several child entities, models were made to predict the parent entity given a child entity. Due to the presence of a single parent entity for a given child entity, the raw test setting was employed.

Performance of models was tested based on the entity prediction task in the raw setting using two metrics, Mean Rank and Hits@10. The triplet classification task was also used to compare the performance between TransE and variants of GrTransE. From table 2, an improvement can be observed in these predictive tasks. GrTransE with $\gamma = 0.75$ outperforms TransE in both entity prediction and triplet classification tasks. Other GrTransE models also show improvements, which is indicative of the improvement in performance due to the inclusion of the GrCluster term in the objective function.

*4.2.1 Detailed Results.* Table 3 demonstrates the working of the GrCluster score function. On observing the tail entities predicted by TransE, for those samples on which GrTransE($\gamma$=0.75) performed perfectly, one can realize that TransE is unable to accurately predict the required entity. TransE predicts those entities that are closely related to the required entity. For example, for the head entity 'american_lobster.n.01', TransE predicts in the following sequence: 'physical_entity.n.01', 'entity.n.01', 'matter.n.03', 'solid.n.01', 'food.n.02'. Although the first 4 entities may seem related to 'american_lobster.n.01', it is not as accurate as 'food.n.02'. Therefore, GrCluster score function allows for the embedding of the entity hierarchical correlation information within obtained knowledge graph embeddings.

## 4.3 Improved Robustness in Entity Prediction

In this section, an experiment was performed to observe the increase in the robustness of embeddings obtained when hierarchical data

| Dataset | Number of Entities | Number of Relations | Samples in Train Split | Samples in Validation Split | Samples in Test Split |
|---|---|---|---|---|---|
| **WNNH** | 82,115 | 1 | 693,086 | 25,000 | 25,000 |
| **WN18** | 40,943 | 18 | 141,442 | 5,000 | 5,000 |

**Table 1: Statistics of datasets used in various experiments.**

is included while modeling other relations. For this experiment, the WN18 dataset was utilized. As described in section 2, TransE is a simple and efficient knowledge graph embedding model. In its simplicity lies its drawback in modeling various types of relations. It does not perform well on 1-N, N-1 and N-N relation types. To further prove the working of the GrCluster score function, it was included in the objective function of the TransE embedding model and detailed results in the performance involving various relation types have been presented.

The baseline for this experiment was the TransE embedding model trained using its original objective function on the WN18 dataset. Following the previous section, GrTransE embedding models were trained by including the GrCluster score function in the original objective function in the manner described by equation 12. The same hyperparameters were used as discussed in section 4.1. Since the WN18 dataset contains 18 relation types, models were trained using the Uniform and Bernoulli sampling methods [31] for the corruption of golden triplets.

Table 4 shows significant improvements in performances with different relationship types which further validates the working of the GrCluster score function. GrTransE performs better in predicting entities in the 'N' side of a relation, overcoming the drawback mentioned in [8]. GrTransE with $\gamma = 0.50$ outperforms TransE in the entity prediction task involving various types of relationships.

## 4.4 Experiments on Translational Distance Models

In this experiment, several other translational distance knowledge graph embedding models were trained by including the GrCluster score function in the objective function. The implementation for various models was provided by TransX [3]. Translational distance models chosen for this experiment include TransE [8], TransH [31], TransR [19], and TransD [15]. The baselines for this experiment were the translational distance models trained on their original objective function. Both the Uniform and the Bernoulli negative sampling methods were used to train the models. Hyperparameter settings from the previous experiments were followed in this experiment.

| Task | Entity Prediction | | | Triplet |
|---|---|---|---|---|
| Method | MRR | H3 | H10 | Class. |
| TransE | 0.0113 | 0.0816 | 0.4913 | 0.8703 |
| GrTransE ($\gamma$=0.10) | 0.0105 | 0.0786 | 0.4879 | 0.8683 |
| GrTransE ($\gamma$=0.25) | **0.0140** | **0.0818** | 0.4448 | 0.8666 |
| GrTransE ($\gamma$=0.50) | **0.0122** | 0.0792 | 0.4787 | **0.8708** |
| GrTransE ($\gamma$=0.75) | **0.0115** | 0.0811 | **0.5036** | **0.8710** |
| GrTransE ($\gamma$=0.90) | **0.0130** | **0.0816** | 0.4705 | 0.8678 |
| GrTransE ($\gamma$=$\theta$=0.34) | 0.0109 | 0.0745 | 0.4701 | **0.8730** |

**Table 2: Results of Entity Prediction and Triplet Classification of GrTransE against TransE on the WNNH dataset in the raw test setting. GrTransE with $\gamma = 0.75$ outperforms TransE in both tasks.**

| Head entity | Predicted tail entity |
|---|---|
| american_lobster.n.01 | physical_entity.n.01, entity.n.01, matter.n.03, solid.n.01, **food.n.02** |
| club.n.06 | substance.n.01, physical_entity.n.01, part.n.01, material.n.01, entity.n.01, matter.n.03, **relation.n.01** |
| huddle.n.01 | abstraction.n.06, entity.n.01, conversation.n.01, discussion.n.02, auditory_communication.n.01, **communication.n.02** |
| convergence.n.04 | act.n.02, psychological_feature.n.01, event.n.01, change.n.03, degradation.n.01, abstraction.n.06, change_of_integrity.n.01, transportation.n.02, **action.n.01** |
| glossodia.n.01 | group.n.01, taxonomic_group.n.01, genus.n.02, anthemis.n.01, hexagrammos.n.01, **biological_group.n.01** |

**Table 3: A few examples of test samples where TransE fails to predict as well as GrTransE($\gamma$=0.75). The entities in bold are the correct entities for the given head entity. GrTransE($\gamma$=0.75) perfectly predicts the required entity at rank 1.**

| Task | Predicting head | | | | Predicting tail | | | |
|---|---|---|---|---|---|---|---|---|
| Relationship Type | 1-1 | 1-N | N-1 | N-N | 1-1 | 1-N | N-1 | N-N |
| TransE | 0.5714 | 0.7401 | 0.5194 | 0.5558 | 0.3571 | 0.5154 | 0.7627 | 0.5496 |
| GrTransE ($\gamma$=0.10) | 0.3810 | **0.7694** | **0.5381** | **0.5805** | 0.4762 | **0.5214** | **0.7996** | **0.5894** |
| GrTransE ($\gamma$=0.25) | 0.5238 | **0.7526** | **0.5623** | 0.5434 | **0.5000** | **0.5295** | **0.7638** | 0.5451 |
| GrTransE ($\gamma$=0.50) | **0.6190** | **0.7986** | **0.5416** | **0.5770** | **0.4524** | **0.5539** | **0.7956** | **0.5991** |
| GrTransE ($\gamma$=0.75) | 0.4762 | **0.7526** | **0.5533** | 0.5549 | 0.3333 | **0.5604** | **0.7653** | 0.5478 |
| GrTransE ($\gamma$=0.90) | **0.5952** | **0.7585** | 0.2590 | 0.4053 | **0.4762** | 0.2610 | **0.7769** | 0.4257 |
| GrTransE ($\gamma$=$\theta$=0.85) | **0.5714** | **0.7580** | **0.5336** | 0.5469 | **0.4524** | **0.5398** | **0.7638** | **0.5496** |

**Table 4: Detailed results for the entity prediction task by category of relationship. Values signify Hits@3 in the filtered setting. It is observed that the GrTransE variants outperform TransE in the entity prediction task across various relation types. GrTransE with $\gamma = 0.50$ outperforms TransE in all relationship types.**

| Task | Entity Prediction | | | Triplet |
|---|---|---|---|---|
| Method | MRR | H3 | H10 | Class. |
| TransE(U) | 0.0994 | 0.6147 | 0.8430 | 0.9304 |
| GrTransE($\gamma$=0.50)(U) | **0.1094** | **0.6521** | **0.8626** | 0.9288 |
| GrTransE($\theta$=0.24)(U) | **0.1015** | **0.6249** | **0.8516** | **0.9332** |
| TransE(B) | **0.1064** | 0.6099 | 0.8203 | **0.9193** |
| GrTransE($\gamma$=0.75)(B) | 0.1051 | **0.643** | **0.8516** | 0.9192 |
| GrTransE($\theta$=0.85)(B) | 0.1059 | **0.6346** | **0.8493** | 0.9189 |
| TransH(U) | 0.0620 | **0.6149** | **0.9000** | 0.9761 |
| GrTransH($\gamma$=0.75)(U) | **0.0668** | 0.6064 | 0.8966 | **0.9775** |
| GrTransH($\theta$=0.43)(U) | 0.0583 | 0.5868 | 0.8769 | **0.9766** |
| TransH(B) | 0.0583 | 0.6355 | 0.8947 | **0.9776** |
| GrTransH($\gamma$=0.90)(B) | **0.0637** | 0.6275 | 0.8893 | **0.9776** |
| GrTransH($\theta$=0.18)(B) | 0.0681 | 0.6411 | 0.9015 | 0.9766 |
| TransR(U) | 0.1358 | 0.7198 | 0.8797 | 0.8911 |
| GrTransR($\gamma$=0.75)(U) | **0.1431** | **0.7279** | **0.8859** | **0.8950** |
| GrTransR($\theta$=0.61)(U) | **0.1424** | **0.7338** | **0.8924** | **0.8911** |
| TransR(B) | 0.1891 | 0.7071 | 0.8572 | 0.7417 |
| GrTransR($\gamma$=0.25)(B) | **0.2288** | **0.7205** | **0.8707** | 0.7286 |
| GrTransR($\theta$=0.73)(B) | 0.1759 | **0.7093** | **0.8579** | **0.7491** |
| TransD(U) | 0.0601 | 0.6117 | 0.8863 | 0.9655 |
| GrTransD($\gamma$=0.25)(U) | **0.0630** | **0.6404** | **0.8960** | **0.9661** |
| GrTransD($\theta$=0.23)(U) | 0.0570 | **0.6218** | **0.8873** | 0.9648 |
| TransD(B) | 0.0554 | 0.6361 | 0.8744 | 0.9320 |
| GrTransD($\gamma$=0.75)(B) | **0.0571** | **0.6521** | **0.8935** | 0.9318 |
| GrTransD($\theta$=0.70)(B) | **0.0559** | **0.6611** | **0.8942** | **0.9326** |

**Table 5: Comparison of performance of translational distance models with the inclusion of GrCluster score in the objective function. 'U' signifies model has been trained with the Uniform sampling method. 'B' signifies model has been trained with the Bernoulli sampling technique. Results from the filtered test setting have been produced. The table shows the best results obtained when the hyperparameterized version of GrCluster was used. The table also shows the result of using the parameterized version of GrCluster.**

Results from table 5 show that models that have been trained with the GrCluster score included in their objective function show an increase in performance against their original counterparts. Similar results, as specified in section 4.3, can been noticed in terms of improvements in prediction of entities which take part in different types of relationships. A significant increase in performance can be observed while predicting the entity on the 'N' side of a relationship. Thus, a conclusion can be made that including hierarchy of entities while modeling relationships helps in obtaining robust embeddings for both entities and relationships.

## 4.5 Experiments on Semantic Matching Models

To further observe the advantage of the proposed score function, GrCluster was included in the objective functions of several Semantic Matching embedding methods. Implementation for the models DistMult [35], ComplEx [29] and HolE [23] were provided by the OpenKE framework [13]. Similar to the previous section, the baselines in this experiment were the semantic matching models trained with their original objective function. GrCluster score function was included in the objective function in the manner described in equation 12. Both the Uniform and the Bernoulli negative sampling methods were used to train the models. Hyperparameter settings from the previous experiments were followed in this experiment.

Since the ComplEx method searched for two sets of embeddings for each entity and relation, two GrCluster score functions were added to the original objective function which shared the same parameter for the discount factor $\gamma$.

Results displayed in table 6 demonstrate the advantages of including the proposed score function in the objective function of any knowledge graph embedding model. Significant improvements in performance have been observed when the GrCluster score function is included in the objective function while training the embedding models in both Uniform and Bernoulli negative sampling techniques.

## 5 CONCLUSION AND FUTURE SCOPE

In this paper, GrCluster, a novel score function has been proposed that considers the entity hierarchical correlation in a knowledge graph. Two variants of the proposed score function have been introduced. This score function can be included in the objective function of any knowledge graph embedding model to increase the

| Task | Entity Prediction | | | Triplet |
|---|---|---|---|---|
| Method | MRR | H3 | H10 | Class. |
| DistMult(U) | 0.7321 | 0.9095 | 0.9401 | 0.9736 |
| GrDistMult($\gamma$=0.25)(U) | **0.7369** | **0.9135** | **0.9429** | **0.9750** |
| GrDistMult($\theta$=0.58)(U) | 0.7306 | **0.9105** | **0.9428** | **0.9745** |
| DistMult(B) | 0.7304 | **0.9166** | 0.9427 | 0.9748 |
| GrDistMult($\gamma$=0.75)(B) | **0.7310** | 0.9159 | **0.9443** | **0.9760** |
| GrDistMult($\theta$=0.82)(B) | 0.4000 | 0.7903 | 0.9163 | **0.9748** |
| ComplEx(U) | 0.9202 | 0.9376 | 0.9434 | 0.9727 |
| GrComplEx($\gamma$=0.10)(U) | **0.9289** | **0.9418** | **0.9458** | **0.9744** |
| GrComplEx($\theta$=0.32) (U) | 0.9160 | **0.9391** | **0.9448** | **0.9732** |
| ComplEx(B) | 0.9223 | 0.9400 | 0.9447 | 0.9727 |
| GrComplEx($\gamma$=0.90)(B) | **0.9303** | **0.9448** | **0.9482** | **0.9746** |
| GrComplEx($\theta$=0.51)(B) | **0.9279** | **0.9425** | **0.9482** | **0.9732** |
| HolE(U) | 0.5665 | 0.7503 | 0.8573 | **0.9661** |
| GrHolE($\gamma$=0.90)(U) | **0.6394** | **0.7913** | **0.8706** | 0.9640 |
| GrHolE($\theta$=0.32)(U) | **0.6240** | **0.7744** | **0.8621** | **0.9661** |
| HolE(B) | 0.1765 | 0.3101 | 0.5094 | **0.9638** |
| GrHolE($\gamma$=0.50)(B) | **0.6210** | **0.7533** | **0.8316** | 0.9545 |
| GrHolE($\theta$=0.23)(B) | **0.4164** | **0.5972** | **0.7671** | 0.9563 |

**Table 6: Comparison of performance of Semantic Matching models with the inclusion of GrCluster score in the method's original objective function. 'U' signifies model has been trained with the Uniform sampling method. 'B' signifies model has been trained with the Bernoulli sampling technique. Filtered test results have been produced. The table shows the best results obtained when the hyperparameterized version of GrCluster was used. The table also shows the result of using the parameterized version of GrCluster.**

robustness of the obtained embeddings with a negligible increase in number of parameters. Experiments have been performed with several translation distance models and semantic matching models to offer evidence that GrCluster helps increase performance in the tasks of entity prediction and triplet classification.

Tables 5 and 6 compares the performance of embedding methods with the inclusion of the GrCluster score in the objective function. The results show consistent improvements across metrics and embedding models for the tasks of entity prediction and triplet classification.

Future work could include extending several other embedding models to incorporate the GrCluster score function to arrive at more robust embeddings. Random search approaches for hyperparameter optimization [6] may be used to select an optimal value for the discount factor $\gamma$ for a particular dataset and an embedding model. Furthermore, improvements can be made to the parameterized version of the proposed score function to search for optimal values for the discount factor $\gamma$ which can enable end-to-end training using gradient descent [26]. Embedding methods developed in the future could inherently include the GrCluster score function to produce improved results.

## REFERENCES


[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*. 265–283.

[2] Christine Alvarado, Jaime Teevan, Mark S Ackerman, and David Karger. 2003. Surviving the information explosion: How people find their electronic information. (2003).

[3] Natural Language Processing Lab at Tsinghua University. 2015. Tensorflow-TransX. https://github.com/thunlp/TensorFlow-TransX.

[4] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*. Springer, 722–735.

[5] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.

[6] James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research* 13, Feb (2012), 281–305.

[7] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. ACM, 1247–1250.

[8] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*. 2787–2795.

[9] Lijuan Cai and Thomas Hofmann. 2004. Hierarchical document categorization with support vector machines. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*. ACM, 78–87.

[10] Christopher De Sa, Albert Gu, Christopher Ré, and Frederic Sala. 2018. Representation tradeoffs for hyperbolic embeddings. *arXiv preprint arXiv:1804.03329* (2018).

[11] Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic entailment cones for learning hierarchical embeddings. *arXiv preprint arXiv:1804.01882* (2018).

[12] Yi Gu and Chaoli Wang. 2010. A study of hierarchical correlation clustering for scientific volume data. In *International Symposium on Visual Computing*. Springer, 437–446.

[13] Xu Han, Shulin Cao, Xin Lv, Yankai Lin, Zhiyuan Liu, Maosong Sun, and Juanzi Li. 2018. OpenKE: An Open Toolkit for Knowledge Embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 139–144.

[14] Xu Han, Zhiyuan Liu, and Maosong Sun. 2016. Joint representation learning of text and knowledge for knowledge graph completion. *arXiv preprint arXiv:1611.04125* (2016).

[15] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 687–696.

[16] Rudolf Kadlec, Ondrej Bajgar, and Jan Kleindienst. 2017. Knowledge base completion: Baselines strike back. *arXiv preprint arXiv:1705.10744* (2017).

[17] Bekir Karlik and A Vehbi Olgac. 2011. Performance analysis of various activation functions in generalized MLP architectures of neural networks. *International Journal of Artificial Intelligence and Expert Systems* 1, 4 (2011), 111–122.

[18] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[19] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*.

[20] George A Miller. 1995. WordNet: A lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[21] Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*. 6338–6347.

[22] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2016. A review of relational machine learning for knowledge graphs. *Proc. IEEE* 104, 1 (2016), 11–33.

[23] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic embeddings of knowledge graphs. In *Thirtieth AAAI conference on artificial intelligence*.

[24] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data.. In *ICML*, Vol. 11. 809–816.

[25] Heiko Paulheim. 2017. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web* 8, 3 (2017), 489–508.

[26] Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747* (2016).
[27] Baoxu Shi and Tim Weninger. 2018. Open-world knowledge graph completion. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
[28] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.
[29] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*. 2071–2080.
[30] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.
[31] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Twenty-Eighth AAAI conference on artificial intelligence*.
[32] Peter Wegner. 1990. Concepts and paradigms of object-oriented programming. *ACM Sigplan Oops Messenger* 1, 1 (1990), 7–87.
[33] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation learning of knowledge graphs with entity descriptions. In *Thirtieth AAAI Conference on Artificial Intelligence*.
[34] Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2016. Representation Learning of Knowledge Graphs with Hierarchical Types.. In *IJCAI*. 2965–2971.
[35] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575* (2014).
[36] Zhao Zhang, Fuzhen Zhuang, Meng Qu, Fen Lin, and Qing He. 2018. Knowledge graph embedding with hierarchical relation structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 3198–3207.