Embedding Knowledge Graphs Attentive to Positional and Centrality Qualities

Afshin Sadeghi[1,2(✉)], Diego Collarana[1,2], Damien Graux[3], Jens Lehmann[1,2]

[1] Smart Data Analytics Group, University of Bonn, Germany
[2] Fraunhofer IAIS, Sankt Augustin, Germany
[3] Inria, Université Côte dAzur, CNRS, I3S, France
{afshin.sadeghi,collaran,jens.lehmann}@.iais.fraunhofer.de
damien.graux@inria.fr

Abstract. Knowledge graphs embeddings (KGE) are lately at the center of many artificial intelligence studies due to their applicability for solving downstream tasks, including link prediction and node classification. However, most Knowledge Graph embedding models encode, into the vector space, only the local graph structure of an entity, i.e., information of the 1-hop neighborhood. Capturing not only local graph structure but global features of entities are crucial for prediction tasks on Knowledge Graphs. This work proposes a novel KGE method named Graph Feature Attentive Neural Network (GFA-NN) that computes graphical features of entities. As a consequence, the resulting embeddings are attentive to two types of global network features. First, nodes' relative centrality is based on the observation that some of the entities are more "prominent" than the others. Second, the relative position of entities in the graph. GFA-NN computes several centrality values per entity, generates a random set of reference nodes' entities, and computes a given entity's shortest path to each entity in the reference set. It then learns this information through optimization of objectives specified on each of these features. We investigate GFA-NN on several link prediction benchmarks in the inductive and transductive setting and show that GFA-NN achieves on-par or better results than state-of-the-art KGE solutions.

1 Introduction

Knowledge graphs (KGs) are capable of integrating heterogeneous data sources under the same graph data model. Thus KGs are at the center of many artificial intelligence studies. KG nodes represent concepts (entities), and labeled edges represent the relation between these entities[1]. KGs such as Wikidata, WordNet, Freebase, and Nell include millions of entities and relations representing the current knowledge about the world. KGs in combination with Machine Learning models are used for refining the Knowledge Graph itself and for downstream

[1] E.g. (Berlin, CapitalOf, Germany) is a fact stating Berlin is the capital of Germany.

tasks, like link prediction and node classification. However, to use KGs in Machine Learning methods, we need to transform graph representation into a vector space presentation, named Knowledge Graph embeddings (KGE).

KGE have many applications including analysis of social networks and biological pathways. Thus, many approaches have been proposed ranging from translation methods, e.g., Trans* family [3,14,30]; Rotation-based methods, e.g., RotatE [21]; Graph Convolutional methods, e.g., R-GCN [20], CompGCN [26], and TransGCN [4]; and Walk-based methods , e.g., RDF2Vec [17].

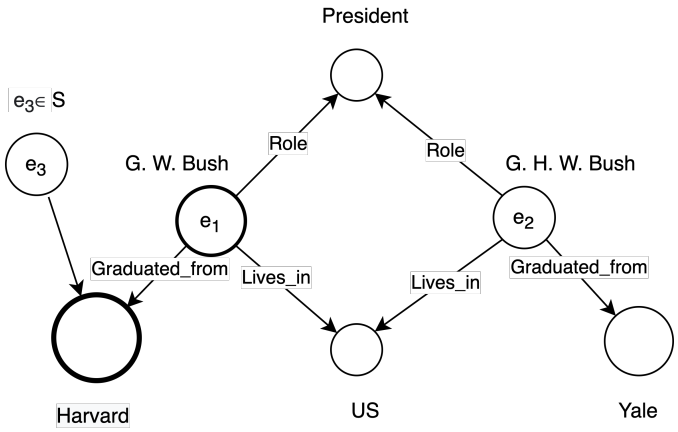

Fig. 1. Example Knowledge Graph in which nodes e_1 and e_2 are difficult to distinguish by a KGE model only using their neighborhood information.

Traditional graph embedding methods, however, rely exclusively on facts (triples) that are explicitly present in a Knowledge Graph. Therefore, their prediction ability is limited to a set of incomplete facts. A means of improvement is to incorporate complementary information in the embeddings. A class of methods applies external knowledge such as entity text descriptions [31] and text associations related to entities[27] into the KG modeling. In contrast, intrinsic methods extract complementary knowledge from the same KG. For example, the algorithms that derive logical rules from a KG and combine them with embeddings of the KG [6,29]. Analogously recent studies [36] consider graph structural features as an intrinsic aspect of KGs in the embedding.

We motivate our model by addressing a challenge of most KGE models; These methods independently learn the existence of relation from an entity to its hop-1 neighborhood. This learning strategy neglects the fact that entities located at a distance can still affect an entity's role in the graph. Besides that, the location of the entities in the network can be useful to distinguish nodes. Figure 1 illustrates such an example where the goal is to learn embeddings for e_1 and e_2 entities in the KG. Distinguishing between the two candidates, i.e., George W. Bush and

George H. W. Bush, is challenging for previous methods since e_1 and e_2 have almost the same neighbors, except George W. Bush graduated from Harvard University while George H. W. Bush did not. However, If we compare e_1 to e_2 using their eigenvector centrality, we can easily distinguish them. e_1 has a greater centrality than e_2 since e_1 is connected to Harvard which has a high eigenvector centrality. Analogously, if we consider the shortest path of e_1 and e_2 to e_3 which belongs to set of reference node S, their distance to e_3 is different. Intuitively, if a model could beforehand know the centrality and distance to e_3 as additional knowledge, it can more easily model e_1 and e_2 and rank them correctly.

With a new view to Knowledge Graph embeddings, we propose GFA-NN[2], an approach that learns both the local relations between the entities and their global properties in one model. In order to efficiently encode entity indicators in Knowledge Graph modeling, we focus on learning node centrality and positional indicators, (e.g., the degree, Katz, or eigenvalue centrality of entities in the graph) as well as the Knowledge Graph structure. For this purpose, we fuse the modeling of each entity indicator in the style of Multiple Distance Embedding (MDE) [18] where distinct views to Knowledge Graphs are modeled through independent embedding weights. GFA-NN extracts positional information and four centrality indicators of nodes from the KG and defines a learning function for each one. Then GFA-NN scores their aggregation with MDE.

Previously, different leanings were applied to embedding models using constraints in the loss function. Now that MDE has broken the limitation of using more than one objective function on independent embeddings, we directly add new extracted information about the entities as aggregated objective functions.

Centrality values and position of nodes in graphs are global measurements for nodes across the whole graph. If we use a local assignment, for example the number of paths between specific nodes, this measurement may have different wights based on what portion of the network is considered in the calculation.

Despite the exciting recent advancements, most of the previous works fail to learn the relation between entities regarding the whole graph. Therefore, we define relative position attentive and relative centrality attentive functions for embedding the relative importance of nodes and their position relative to the whole network. In the following Section, we discuss the relation between our work and the current state-of-the-art. Later in Section 3 introduce the preliminaries and notations required to explain our chosen method. We outline in Section 4 the idea of centrality and positional qualities learning and explain our approach. In Section 5, we mention the model's theoretical analysis; and we continue with experiments that evaluate our model in Section 6.

2 Related Work

A large and growing body of literature has investigated KGE models. A typical KGE model consists of three main elements: (1) entities and relations representation in a continuous vector space, (2) a scoring function to measure KG's facts

[2] Source code is available at: `https://github.com/afshinsadeghi/GFA-NN`

plausibility, and (3) a loss function that allows learning KGE in a supervised manner. Based on this formulation, we classify KGE models in: latent distance approaches, tensor factorization and multiplicative models, and neural networks.

Latent Distance Models, e.g., Trans* [3,14,30] family, measure a fact's plausibility by scoring the distance between the two entities, usually after a translation carried out by the relation. RotatE [21] combines translation and rotation. RotatE models relations as rotations from head to tail entities in the complex space and uses the Hadamard product in the score function to do these rotations.

Tensor factorization and multiplicative approaches define the score of triples via pairwise multiplication of embeddings. DistMult [34], for example, multiplies the embedding vectors of a triple element by element (h, r, t) as the objective function. However, DistMult fails to distinguish displacement of head relation and tail entities, and therefore, it cannot model anti-symmetric relations. ComplEx [24] solves DistMult's issue. Finally, LiteralE [11] propose a portable module for incorporating literals into these models, e.g., DistMult-LiteralE.

Unlike previous methods, the *neural network-based methods* learn KGE by connecting artificial neurons in different layers. Graph Neural Network (GNN) aggregate node formation using a message-passing architecture. Recently, hybrid neural networks such as CompGCN [25] and MDE$_{nn}$ [18] have raised. These methods benefit from neural network architectures to model relations with (anti)symmetry, inversion, and composition patterns.

Several studies have investigated the benefits of using graph features to bridge the graph structure gap and the numeric vector space. Muzzamil et al. [15] defined a Fuzzy Multilevel Graph Embedding (FMGE), an embedding of attributed graphs with many numeric values. P-GNN [36] incorporates positional information by sampling anchor nodes and calculating their distance to a given node (see Section 5.1 for an in-depth comparison with GFA-NN). Finally, it learns a non-linear distance weighted aggregation scheme over the anchor nodes.

This effort's main difference with previous approaches is in the message passing mechanism. Traditionally in GNNs, approaches learn just nodes' local features (similar to the modeling schema of KGEs) while focusing on neighbor nodes; here, our approach also learns nodes' features regarding the whole graph, known as global graph properties.

3 Background and Notation

A Knowledge graph KG, is comprised of a set of entities $e \in \mathcal{E}$ and a set of relations $r \in \mathcal{R}$. A fact in a Knowledge Graph is a triple of the form (h, r, t) in which h (head) and t (tail) are entities and r is a relation. A KG is a subset of all true facts KG $\subset \xi$. A KG can be conceived as a multi-relational graph. An entity in such formulation is equivalent to a node in graph theory, and an edge represents a relation. In this study, we use Node and Entity interchangeably. We use the term "Node" to emphasize its graphical properties. We use the term "Entity" to highlight the entity's concept.

Link prediction on Knowledge Graphs is made by a Siamese classifier that embeds KG's entities and relations into a low-dimensional space. Thus, a Knowledge Graph embedding model is a function $f : \mathcal{E}, \mathcal{R} \rightarrow \mathcal{Z}$, that maps entities $\mathcal{E}$ and relations $\mathcal{R}$ to d-dimensional vectors $\mathcal{Z} = \{z_1, \ldots, z_n\}$, $z_i \in \mathbb{R}$.

Centrality value of a node designates the importance of the node with regard to the whole graph. For instance, *degree* is a centrality attribute of a node that indicates the number of links incident upon it. When we consider degree as centrality value, the higher the degree of a node is, the greater is its importance in a graph. We provide a generalization of the position-aware embedding definition [36] that distinguishes our method from the previous works.

Structure-based Embedding: A KG embedding $z_i = f : \mathcal{E}, \mathcal{R} \rightarrow \mathcal{Z}$ is attentive to network structure if it is a function of entities and relations such that it models the existence of a neighborhood of an entity e_i using relations r_i and other entities $e_j \in \mathcal{E}$. Most Knowledge Graph embedding methods like QuatE and RotatE compute embeddings using the information describing connections between entities and, therefore, structure-based.

Property-Attentive Embedding: A KG embedding $z_i = f : \mathcal{E}, \mathcal{R} \rightarrow \mathcal{Z}$ is attentive to network properties of an entity if there exists a function $g_p(., ., ...)$ such that $d_p(v_i, v_j, ...) = g_p(z_i, z_j)$, where $d_p(,)$ is a graphical property in G. This definition includes both the property of a sole node such as its centrality and the properties that describe the inter-relation of two nodes such as their shortest path. Examples of Property-Attentive Embedding are P-GNNs and RDF2Vec which their objective function incorporates the shortest path between nodes into embedding computation.

We show that current KGE methods cannot recover global graph properties, such as path distances between entities and centrality of nodes, limiting the performance in tasks where such information is beneficial. Principally, structure-aware embeddings cannot be mapped to property-aware embeddings. Therefore, only using structure-aware embeddings as input is not sufficient when the learning task requires node property information. This work focuses on learning KGEs capturing both entities' local network structures conjointly with the global network properties. We validate our hypothesis that a trait between local and global network features is crucial for link prediction and node classification tasks. A KGE is attentive to node network properties if the embedding of two entities and their relation can be used to approximately estimate their network feature, e.g., their degree relative to other entities in the network.

You *et al.* [36] show for position attentive networks, there exists a mapping g that maps structure-based embeddings $f_{st}(v_i)$, $\forall\ v_i \in \mathrm{V}$ to position attentive embeddings $f_p(v_i)$, $\forall\ v_i \in V$, if and only if no pair of nodes have isomorphic local q-hop neighborhood graphs. This proposition justifies the good performance of KGE models in tasks requiring graphical properties and their under-performance in real-world graphs such as biological and omniscience KGs (e.g., Freebase, DBpedia), in which the structure of local neighborhoods are quite common. We demonstrate that this proposition does not hold for centrality attentive embed-

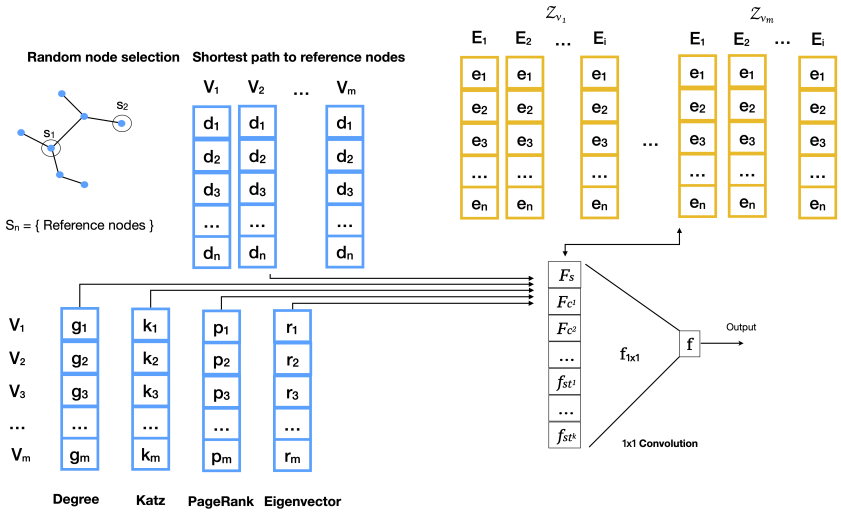

Fig. 2. The architecture of GFA-NN. GFA-NN first pre-computes the centrality property of nodes and distances to a set of to randomly selected reference nodes (**Left**). Then, node centrality and position embeddings attentive to position z_{v_m} are computed via scores $F_1, ..., F_k$ from the distance between a given node v_i and the reference-sets S_i which are shared across all the entities (**Top-middle**). To compute the embedding z_{v_1} for node v_1, an score of GFA-NN first computes via function F_i and then aggregates the F_i scores via 1×1 convolution and an activation function over obtains a vector of final scores. Inside 1×1 a vector w learned which is used to reduce scores into one centrality and position-aware score and produces embeddings z_{v_1} which is the output of the GFA-NN (**Right**).

dings, the proof is provided in the Appendix[3]. We show in Section 4 how we address this challenge for centrality learning.

4 Method

This Section details our proposed method for generating entity network properties attentive embeddings from Knowledge Graphs. We generalize the concept of Knowledge Graph embeddings with a primary insight that incorporating centrality and distance values enables KGE models to compute embeddings with respect to the graphical proprieties of entities relative to the whole network instead of only considering the direct local neighbors (Figure 2, left side).

When modeling the positional information, instead of letting each entity model the information independently and selecting a new reference set per iteration, we keep a set of reference entities through training iterations and across all the networks in order to create comparable embeddings. This design choice enables the model to learn the position of nodes with respect to the spectrum

[3] Uploaded as a complementary `.pdf` on the submission platform.

of different reference node positions and makes each embedding attentive to position (Figure 2, top left). GFA-NN models each graphical feature with a dedicated objective function, meaning that the information encrypted in centrality attentive embeddings does not interfere with the embedding vectors that keep the positional information (Figure 2, top right).

Centrality for nodes are individual values. While positional values are calculated relative to a set of nodes in a graph, only one centrality per entity is extracted. Still, learning this information is valuable because the centrality value of a node is meaningful despite the absence of a large portion of the network. This trait is particularly beneficial in inductive relation prediction tasks.

4.1 Model Formulation

The components of GFA-NN are as follows:

- Random set of reference nodes for distance calculations.
- Matrix M of distances to random entities, where each row i is a set of shortest distance of an entities to the selected set of random nodes.
- Structure-attentive objective functions $f_{st^1}(v_i), \ldots, f_{st^k}(v_i)$ that model the relatedness information of two entities with their local network, which is indicated by triples that consist of head and tail nodes (entities) connected by an edge (relation).
- Position-attentive objective function F_s that models the position of a node (entity) in the graph with respect to its distance to other nodes. This objective considers these distances as a factor of relatedness of entities.
- Centrality attentive objective functions F_c that model the relatedness information of two entities according to centrality properties of nodes (entities). In this setting, the global importances of nodes are learned relatively to the centrality of other nodes.
- Trainable aggregation function $f_{1\times1}$ is a 1×1 convolution [13] that fuses the modeling of the structure-based connectivity information of the entities and relations with their position aware and centrality attentive scoring.
- Trainable vectors r_d, h_d, t_d that project distance matrix M to a lower dimensional embedding space $z \in \mathcal{R}_k$.

Our approach consists of several centrality and position-attentive phases that each of which learns an indicator in a different metric of the status for entities relative to the network.

In the first phase, GFA-NN performs two types of computation to determine the position status and the centrality status of entities. The unit for centrality status computes the relative significance of entities as a vector of length one c_i^j, where j represents each of the centrality metrics. The unit for position status embedding samples n random reference-entities S_n, and computes an embedding for entities. Each dimension i of the embedding is obtained by a function F that computes the shortest path to the i-th reference entity relative to the maximum shortest path in the network.

Then objective functions $F_s, F_c^1, ..., F_c^4$ apply an entity interaction model to enforce the property features e_i^s into entity embeddings e_i, which in the next phase makes a 1×1 convolution [13] over the scores via weights $w \in \mathbb{R}^r$ and non-linear transformation *Tanhshrink*.

Specifically, each entity earns an embedding per attribute that includes values that reveal the relative status information from input entity network properties information. Calculation of the centrality for all nodes in the network leads to a vector representation of the graph for each measure, while the distances to the reference nodes S generate a dense matrix representation.

The network property attentive modeling functions are the same class of functions as used by existing translational KGEs plus a modeling function of embeddings that we extended to be performed in 3D using rotation matrix. In the following, we further elaborate on the design choices.

4.2 Centrality-Attentive embedding:

As shown in Section 3, the centrality values are not canonical. Therefore, the model learns their difference in a normal form, in which the equality of their norm does not mean they are equal. Degree centrality is defined as : $C_d(n) = \deg(n)$.

Katz centrality [8] extends degree centrality from counting neighbor nodes to nodes that can be connected through a path, where the contribution of distant nodes are reduced:

$$C_k(n) = \sum_{k=1}^{\infty} \sum_{j=1}^{N} \alpha^k A_{j,i}^k$$

where A is the adjacency matrix and α is attenuation factor in the range $(0, 1)$. Another included centrality measure is PageRank with the following formulation:

$$C_p(n) = \alpha \sum_j a_{j,i} \frac{C_p(j}{L(j)} + \frac{1-\alpha}{N}$$

where N is $|V|$, the number of nodes in the graph, and $L(j)$ is the degree of node j. Relative eingenvector centrality score of a node n is defined as:

$$C_e i(n) = \frac{1}{\lambda} \sum_{m \in KG} a_{m,n} x_m$$

where $A = (a_{v,t})$ is the adjacency matrix such that $a_{v,t} = 1$ if node n is linked to node m, and $a_{v,t} = 0$ otherwise. λ is a constant which fulfils the eingenvector formulation $Ax = \lambda x$.

Note that the method in first phase normalizes each of the centrality values. The normalization occurs with respect to minimum and the maximum value for nodes in the network and makes attributes relative to the whole network. For example, degree centrality is normalized as follows:

$$C_i^d = \frac{degree(i) - degree_{min}}{degree_{max} - degree_{min}}$$

The centrality-attentive modeling embeddings functions are the same class of dissimilarity functions used by existing KGEs plus a penalty we define on the difference of the entity embeddings as:

$$F_{c^d} = \|h_i - t_i\|_2 \quad - \quad \| \cos(log(C_h^d)) - \cos(log(C_t^d))\|_2 \tag{1}$$

where the function is normalized with the l_2 norm, h_i and t_i represent the vector representation of head and tail in a triple and lastly, C_h^d and C_t^d respectively denote the centrality values of the head and tail entities in that triple.

4.3 Position-Attentive embedding:

GFA-NN models the neighborhood structure using rotations in 3D space and a penalty that forces the method to encode the difference of distances of entities to the reference nodes. The formulation for the structure-attentive part of the score is:

$$F_{rot} = \| \ v_h - v_r \otimes v_t \ \|_2 \tag{2}$$

where $\otimes$ represents a rotation using a rotation matrix of Euler angles with the formulation of direction cosine matrix (DCM):

$$\begin{bmatrix} \cos\theta\cos\psi & -\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi & \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi \\ \cos\theta\sin\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\phi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \tag{3}$$

where ϕ, θ and ψ are Euler angles. The modeling of positional information is performed by a score function made from rotation matrices and a penalty:

$$F_p = F_{rot} - \| \cos(S_i^h) - \cos(S_i^t)) \ \|_2 \tag{4}$$

where S_C^i is the calculated distance from the head and tail nodes to the reference nodes. Hence, the score enforces to learn structure-attentive embeddings with a penalty which is the normalized scalar difference of distance to reference nodes. Here we use the l_2 norm to regularize the F_i score functions and apply negative adversarial sampling [21]. We utilise Adam [9] for optimization.

Reference-set selection relies on a Gaussian random number generator to select normally distributed random reference nodes from the network. GFA-NN keeps a fixed set of reference nodes during the training of different entities through different iterations to generate embeddings attentive to the position that are in the same space and, hence, comparable to each other.

Multiple Property aware scores can be naturally fused to achieve higher expressive power. This happens in $f_{1\times1}$.

Since canonical position-attentive embeddings do not exist, GFA-NN also computes structure-attentive embeddings h_v via the common distance-based modelings of MDE. These scores are aggregated with attribute attentive scores, and then the model using a linear combination of these scores forms a 1×1 convolution to produce only one value that contains both properties. The output of this layer is then fed into the nonlinear activation function.

It is notable that independent weights in MDE formulation allow restricting solution space without limiting the learnability power of the model. Note also that the method is still Semi-supervised learning, where the train and test data are disjoint, and the centrality and path information computation do not consider the portion of the unknown network to the model and only exist in the test data.

5 Theoretical analysis

5.1 Connection to Preceding KGE Methods

GFA-NN generalizes the existing Knowledge Graph embedding models. Taking the definition for the structure-aware and node properties attentive models into perspective, existing knowledge embedding models use the same information of connecting entities through different relations techniques, but use different neighborhood selection scoring function and sampling strategies, and they only output the structure-aware embeddings.

GFA-NN shares the score function aggregate training with MDE [18]. There, a linear combination of scores $f_{1\times1} = \sum w_i F_i$ is trained, where w_i weights are learnt together with the embeddings in the score functions F_i. GFA-NN also shares the concept of training independent embeddings with MDE. The direction cosine matrix used in modeling positional information is convertible into a four-element unit quaternion vector (q_0, q_1, q_2, q_3). The quaternions are the center of the structure-based model QuatE [37], where the relations are models as rotations in the quaternion space. Here, besides modeling rotation, we formulated the score to include a translation as well. RotatE [21] similarly, formulates the relations with a rotation and reduction in $\parallel v_h \circ v_r - v_t \parallel$, however RotatE models rotation in the complex space.

In the branch of Graph neural networks, the aggregate information of a node's neighborhood in one-hop [10,28,26] or nodes in the higher hops [33] is used in message passing mechanism.

P-GNN [36] explicitly learns the shortest path of random nodes for simple graphs. However, it takes a new set of reference nodes in each iteration, which makes the learning of shortest paths local and incremental. In addition, it makes it difficult to retain the structural information from positional embedding. GFA-NN generalizes positional learning by learning the distances to a fixed set of random nodes through the whole network, which makes the positional embedding vectors globally comparable. From the point of view of graph type, GFA-NN generalizes the positional learning to multi-relational graphs to support KGs.

GFA-NN not only learns a weight for each of the network features, but it also associates it with the existing relation types between the two entities that their features are being learned. By including the relation type into position-attentive embeddings, the position also is encoded into relation vectors that connect the entities. Note that relation type learning is sub-optimal for learning centrality values because the dimension of relation types is much more higher than dimension of the the node property values (one integer value) which makes the centrality value differentiation diminish when learnt together with the association

information belonging to relations. Another aspect that GFA-NN generalize the existing graph learning algorithms is that this method learns several centrality aspect and positional information at the same time.

5.2 Expressive Power

In this Section we explain how GFA-NN generalizes the expressive power of Knowledge Graph embedding methods in the perspective of a broader Inductive bias. Generally, inductive bias in a learning algorithm allows it to better prioritize one solution over another, independent of the observed data [2].

Assuming that a labeling function y, labels a triple (h, r, t) as $d_y^r(h, t)$, we predict y^r, similar to [36] from the prospective of representation learning, which is by learning an embedding function f, where $v_h = f(v, G)$ and f computes the entity embeddings for v_h, v_r and v_t. Thus, the objective becomes the task of maximizing the probability of the conditional distribution $p(y|v_h, v_r, v_t)$. This probability can be designated by a distance function $d_v(v_h, v_r, v_t)$ in the embedding space, which usually is an l_p norm of the objective function of the model.

A KGE model, with a goal to predict the existence of an unseen triple (h, r, t) learns embeddings weights v_h and v_t for the entities h and t and v_r for a relation r that lies between them.In this formulation, the embedding for an entity e is computed based on its connection through its one-hop neighborhood, which we express that by structural information S_e, and optimization over the objective function $f_\theta(e, S_e)$. Hereby, the neighborhood information of two entities S_{e_1} and S_{e_2} are computed independently. However, the network feature attentive objective function f_ϕ in GFA-NN poses a more general inductive bias that takes in the distance from a random shared set of reference-nodes which are common across all entities, and the centrality values, which are relative to all nodes. In this setting, any pair of entity embeddings are correlated through the reference-set and the spectrum of relative centrality and therefore are not independent anymore. We call this feature attentive information I.

Accordingly, we define a joint distribution $p(w_{e_1}, w_{e_2})$ over node embeddings, where $w_{e_1} = f_\phi(e_1, I)$ and $w_{e_2} = f_\phi(e_2, I)$. We formalize the problem of KG representation learning by minimizing the expected value of the likelihood of the objective function in margin-based ranking setting, in the following for a structure base KGE:

$$\begin{aligned} &\min_\theta \mathbb{E}_{e_1,e_2,e_3,S_{e_1},S_{e_2},S_{e_3}} \\ &\quad \mathcal{L}(d_v^+(f_\theta(e_1, S_{e_1}), f_\theta(e_2, S_{e_2})) - d_v^-(f_\theta(e_1, S_{e_1}), f_\theta(e_3, S_{e_3})) - m) \end{aligned} \tag{5}$$

and in GFA-NN:

$$\min_\theta \mathbb{E}_{e_1,e_2,e_3,I} \quad \mathcal{L}(d_v^+(f_\phi(e_1, I), f_\phi(e_2, I)) - d_v^-(f_\phi(e_1, I), f_\phi(e_3, I)) - m) \tag{6}$$

where d_v^+ is the similarity metric determined by the objective function for a positive triple, indicating existing a predicate between entities and by optimizing converges to the target label function $d_y(e_1, e_2) = 0$ for positive samples(existing

Table 1. Statistics of the data sets used in the Experiments.

Dataset	#entities	#relations	#train	#validation	#test
WN18RR	40943	11	86835	3034	3134
FB15k-237	14541	237	272115	17535	20466
ogbl-biokg	45085	51	4762678	162886	162870
WN18RR-v_3-ind	5084	11	6327	538	605
WN18RR-v_4-ind	7084	9	12334	1394	1429
NELL-995-v_1-ind	225	14	833	101	100
NELL-995-v_4-ind	2795	61	7073	716	731

triples) and $d_y(e_1, e_3) = m$ on negative samples. Here, m is the margin value in the margin ranking loss optimization setting. Note that the representations of entities are calculated using joint and marginal distributions, respectively.

Similar to the proof of expressive power in [36], considering the selection of entities $e_1, ..., e_i \in G$ as random variables to form any triples, the mutual information between the joint distribution of entity embeddings and any $Y = d_y(e_1, e_2)$ is greater than that between the marginal distributions. $Y : I(Y; X_{joint}) \geq I(Y; X_{marginal})$. Where,
$X_{joint} = (f_\phi(e_1, S_{e_1}), f_\phi(e_2, S_{e_2})) \sim p(f_\phi(e_1, S_{e_1}),\ f_\phi(e_2, S_{e_2}))$
$X_{marginal} = (f_\theta(e_1, I), f_\theta(e_2, I))$

Because the gap of mutual information is large when the targeted task is related to positional and centrality information of the network, we deduce that KGE embedding based on the joint distribution of distances to reference nodes and relative centrality values have more expressive power than the current structure-based KGE models.

5.3 Complexity Analysis

Next, we explain the complexity of the method and show it's complexity in comparison to the structure-based models. When the shortest paths are calculated on the fly, the learning complexity is added up by $O(b\ log(b))$ for finding the shortest paths on b entities in each batch, and similarly, the centrality computation aggregates to the complexity. We, therefore, pre-calculate this information in our implementation in the experiments to separate them from the learning complexity. The complexity of each of the objective functions on a batch with size b is $O(b)$, and suppose n property attentive features and m structure-aware scores be involved, the overall complexity becomes $O((n+m)\,b)$. It is observable that the larger number here is b and the complexity increases by b times each time one more graphical feature is involved in the learning.

6 Experiments

We evaluate the performance of our model with two link prediction experiments; First, the traditional transductive ranking evaluation, which is originally introduced in [3], and second, inductive relation prediction experiment. In the in-

Table 2. Results on `WN18RR` and `FB15k-237`. Best results are in bold.

	WN18RR			FB15k-237		
Model	MR	MRR	Hit@10	MR	MRR	Hit@10
ComplEx-N3	–	0.48	0.57	–	0.37	0.56
QuatE[2]	–	0.482	0.572	–	**0.366**	**0.556**
TuckER	–	0.470	0.526	–	0.358	0.544
CompGCN	3533	0.479	0.546	197	0.355	0.535
RotatE	3340	0.476	0.571	177	0.338	0.533
MDE	3219	0.458	0.536	203	0.344	0.531
GFA-NN	3390	**0.486**	**0.575**	186	0.338	0.522

ductive setting, the experiment evaluates a models ability to generalize the link prediction task to unseen entities. Table 1 shows the statistics of the datasets used during the experiments.

Metrics and Implementation: We evaluate the link prediction performance by ranking the score of each test triple against all possible derivable negative samples by once replacing its head with all entities and once by replacing its tail. We then calculate the hit at N (Hit@N), mean rank (MR), and mean reciprocal rank (MRR) of these rankings. We report the evaluations in the filtered setting. We determine the hyper-parameters by using grid search. We select the testing models which give the best results on the validation set. In general, we fix the learning rate on 0.0005 and search the embedding size amongst {200, 300, 400, 500}. We search the batch size from {250, 300, 500, 800, 1000}, and the number of negative samples amongst {10, 100, 200, 400, 600, 800, 1000}. We describe all GFA-NN hyper-parameters in the Appendix[4].

6.1 Transductive link prediction experiment

Datasets: We perform experiments on three benchmark datasets: WN18RR [5], FB15k-237 [23], and `ogbl-biokg` [7], which is comparably a sizeable Knowledge Graph assembled from a large number of biomedical repositories.

Baselines: We compare our model with several state-of-the-art structure-based embedding approaches. Our baselines include RotatE [21], TuckER [1], ComplEx-N3 [12], QuatE [37], MDE [18] and the recent graph neural network CompGCN [26]. We report results of each method on WN18RR and FB15k-237 from their respective papers, while the results of the other models in `ogbl-biokg` are from [7]. For RotatE, we report its best results with self-adversarial negative sampling, and for QuatE, we report the results with N3 regularization. For our model, we use the same self-adversarial negative sampling introduced in RotatE. This negative sampling schema is also applied to all the other models in the `ogbl-biokg` benchmark.

Results and Discussion: Table 2 and Table 3 summarize the performance of GFA-NN and other KGE models in the transductive link prediction task. We

[4] Uploaded as a complementary `.pdf` on the submission platform.

Table 3. MRR Results for `ogbl-biokg`. (Results of previous models are from [7].)

Method	Validation	Test
TransE	0.7456	0.7452
DistMult	0.8055	0.8043
ComplEx	0.8105	0.8095
RotatE	0.7997	0.7989
GFA-NN	**0.9011**	**0.9011**

observe that GFA-NN outperforms other state-of-the-art KGEs on WN18RR and is producing competitive results on FB15k-237.

Our analysis shows that the standard deviation of different positional and centrality measures through the network in WN18RR is ≈0.009, while in FB15k-237, it is ≈0.002, which is 4.5 times smaller. This comparison indicates that in WN18RR, these features are more diversified, but in FB15k237, they are close to each other. This analysis suggests the crucial impact of learning centrality and positional-attentive embeddings on the superiority of the GFA-NN on the WN18RR benchmark. While the result on the FB15k-237 is still very competitive to the state-of-the-art, as a lesson learned, we can declare it as a fixed procedure to perform the standard deviation analysis on a dataset before determining how much the network property attentive embedding learning method would be beneficial.

Table 3 shows the **MRR** evaluation results on the comparably large biological dataset named as `ogbl-biokg`. In this benchmark, the number of entity and training samples is much larger than the WN18rr and FB15k-237 datasets.

The capability of learning feature attentive embeddings is crucial in this transductive link prediction task. While the best KGEs can only achieve the MRR of 0.8105 on the validation and 0.8095 on the test dataset, GFA-NN reaches 0.901 on both datasets, improving state-of-the-art by 9 percent.

This wide gap between the results supports the assumption that property-attentive embeddings surpass prior methods in larger-scale real-world networks.

6.2 Inductive link prediction experiment

Datasets: For evaluations in the inductive setting, we select four variant datasets which Komal et al. [22] extracted from WN18RR and NELL-995 [32].

Baselines: Inductive baselines include GraIL [22], which uses sub-graph reasoning for inductive link prediction. RuleN [16] that applies a statistical rule mining method, and two differentiable methods of rule learning NeuralLP [35] and DRUM [19]. We report the results of these state-of-the-art models from Komal et al. [22].

Results: Table 4 summarizes the GFA-NN's Hit@10 ranking performance against methods specified on the inductive link prediction task. Although we did not explicitly design GFA-NN for this task, we observe GFA-NN performs very competitively in this setting and outperforms the best inductive learning

Table 4. Hit@10 results for inductive datasets. (Other models' results are from [22].)

Model	**WN18RR-v_3-ind**	**WN18RR-v_4-ind**	**NELL-995-v_1-ind**	**NELL-995-v_4-ind**
NeuralLP	0.4618	0.6713	0.4078	**0.8058**
DRUM	0.4618	0.6713	0.5950	**0.8058**
RuleN	0.5339	0.7159	0.5950	0.6135
GraiL	0.5843	0.7341	0.5950	0.7319
GFA-NN	**0.5893**	**0.7355**	**0.9500**	0.7722

models in most cases. This result supports our hypothesis that the Knowledge Graph embeddings attentive to positional and centrality qualities are beneficial for prediction tasks in challenging settings, i.e., inductive link prediction task.

7 Conclusion

In this article, with a new view to the relational learning algorithms, we propose to learn the structural information of the network conjointly with the learning of the centrality and positional properties of the Knowledge Graph entities in one model. We provide theoretical analyses and empirical evaluations to identify the improvements and constraints in the expressive power for this class of KGEs. In particular, we demonstrate, with proper formulation, that the learning of these global features is beneficial to the link prediction task, given that GFA-NN performs highly efficiently in a variety of benchmarks and often outperforms current state-of-the-art solutions in both inductive and transductive settings.

References

1. Balazevic, I., Allen, C., Hospedales, T.: Tucker: Tensor factorization for knowledge graph. In: EMNLP-IJCNLP. (2019) 5185–5194
2. Battaglia, P.W., Hamrick, J.B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al.: Relational inductive biases, deep learning, and graph networks. Preprint arXiv:1806.01261 (2018)
3. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in neural information processing systems. (2013) 2787–2795
4. Cai, L., Yan, B., Mai, G., Janowicz, K., Zhu, R.: Transgcn: Coupling transformation assumptions with graph convolutional networks for link prediction. In: Int. Conf. on Knowledge Capture, ACM (2019) 131–138
5. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2D knowledge graph embeddings. In: AAAI. (2018) 1811–1818
6. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Knowledge graph embedding with iterative guidance from soft rules. In: AAAI. (2018) 4816–4823
7. Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., Leskovec, J.: Open graph benchmark: Datasets for machine learning on graphs. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., eds.: NeurIPS. (2020)

8. Katz, L.: A new status index derived from sociometric analysis. Psychometrika **18**(1) (1953) 39–43
9. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In Bengio, Y., LeCun, Y., eds.: Int. Conf. on Learning Representations. (2015)
10. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: Int. Conf. on Learning Representations, OpenReview.net (2017)
11. Kristiadi, A., Khan, M.A., Lukovnikov, D., Lehmann, J., Fischer, A.: Incorporating literals into knowledge graph embeddings. In: ISWC. (2019) 347–363
12. Lacroix, T., Usunier, N., Obozinski, G.: Canonical tensor decomposition for knowledge base completion. In: Int. Conf. on Machine Learning, PMLR (2018) 2863–2872
13. Lin, M., Chen, Q., Yan, S.: Network in network. Preprint arXiv:1312.4400 (2013)
14. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI. (2015) 2181–2187
15. Luqman, M.M., Ramel, J.Y., Llads, J., Brouard, T.: Fuzzy multilevel graph embedding. Pattern Recognition **46**(2) (2013) 551 – 565
16. Meilicke, C., Fink, M., Wang, Y., Ruffinelli, D., Gemulla, R., Stuckenschmidt, H.: Fine-grained evaluation of rule- and embedding-based systems for knowledge graph completion. In: Int. Semantic Web Conference, Springer (2018) 3–20
17. Ristoski, P., Paulheim, H.: Rdf2vec: RDF graph embeddings for data mining. In: Int. Semantic Web Conference. (2016) 498–514
18. Sadeghi, A., Graux, D., Shariat Yazdi, H., Lehmann, J.: MDE: Multiple distance embeddings for link prediction in knowledge graphs. In: 24th European Conference on Artificial Intelligence, ECAI. (2020)
19. Sadeghian, A., Armandpour, M., Ding, P., Wang, D.Z.: DRUM: end-to-end differentiable rule mining on knowledge graphs. In: NeurIPS 2019. (2019) 15321–15331
20. Schlichtkrull, M.S., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: ESWC. Volume 10843 of Lecture Notes in Computer Science., Springer (2018) 593–607
21. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: RotatE: Knowledge graph embedding by relational rotation in complex space. In: ICLR. (2019)
22. Teru, K., Denis, E., Hamilton, W.: Inductive relation prediction by subgraph reasoning. In: Int. Conf. on Machine Learning. (2020) 9448–9457
23. Toutanova, K., Chen, D.: Observed versus latent features for knowledge base and text inference. In: Proceedings of the 3rd workshop on continuous vector space models and their compositionality. (2015) 57–66
24. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: ICML. (2016) 2071–2080
25. Vashishth, S., Sanyal, S., Nitin, V., Talukdar, P.: Composition-based multi-relational graph convolutional networks. In: ICLR. (2020)
26. Vashishth, S., Sanyal, S., Nitin, V., Talukdar, P.P.: Composition-based multi-relational graph convolutional networks. In: ICLR, OpenReview.net (2020)
27. Veira, N., Keng, B., Padmanabhan, K., Veneris, A.G.: Unsupervised embedding enhancements of knowledge graphs using textual associations. In: IJCAI. (2019) 5218–5225
28. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: ICLR. (2018)
29. Wang, W.Y., Cohen, W.W.: Learning first-order logic embeddings via matrix factorization. In: IJCAI. (2016) 2132–2138
30. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI. (2014)

31. Xie, R., Liu, Z., Jia, J., Luan, H., Sun, M.: Representation learning of knowledge graphs with entity descriptions. In: AAAI. (2016) 2659–2665
32. Xiong, W., Hoang, T., Wang, W.Y.: DeepPath: A reinforcement learning method for knowledge graph reasoning. In: EMNLP. (2017) 564–573
33. Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K., Jegelka, S.: Representation learning on graphs with jumping knowledge networks. In Dy, J.G., Krause, A., eds.: ICML. (2018) 5449–5458
34. Yang, B., Yih, W., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: ICLR. (2015)
35. Yang, F., Yang, Z., Cohen, W.W.: Differentiable learning of logical rules for knowledge base reasoning. In: NeurIPS. (2017) 2319–2328
36. You, J., Ying, R., Leskovec, J.: Position-aware graph neural networks. In: ICML. (2019) 7134–7143
37. Zhang, S., Tay, Y., Yao, L., Liu, Q.: Quaternion knowledge graph embeddings. In: NeurIPS. (2019) 2731–2741

Appendix

A Proof of the proposition from Section 3.1 about the non-existence of g

Reminder of the proposition:

For position attentive networks, we know (You *et al.* [36]) that there exists a mapping g that maps structure-based embeddings $f_sq(v_i)$, $\forall\ v_i \in$ V to position attentive embeddings $f_p(v_i)$, $\forall\ v_i \in V$, if and only if no pair of nodes have isomorphic local q-hop neighbourhood graphs.
This proposition does not hold for centrality attentive embeddings.

Proof. If no pair of nodes have isomorphic local q-hop neighborhood graphs, it is still possible for them to have the same centrally attentive embeddings. To show that, it is enough that two nodes have the same centrally value in the graph: for example, for degree centrality, when the two nodes have the same number of neighbors, that are consisting of different nodes, their neighborhoods are non-isometric; however, they have the same degree centrality.

Therefore $f_p(v_i)$ can not be a function of $f_{st}(v_i)$, and the two pairs of nodes would have different structure-aware node embeddings while their centrally attentive embeddings are equal. □

B Hyperparameters Settings

We list the best hyperparameters setting of GPA-NN on the benchmark datasets in Table 5. The learning rate in all the experiments is fixed to 0.0005, adversarial temperature for negative sampling is fixed to 2.5, and ψ, the dividend for the score aggregation in $f_{1\times1}$ is fixed to 14.

Table 5. Best hyperparameters setting of GPA-NN on the benchmark datasets.

Dataset	**Dimension**	**Batch size**	**#neg**	**#iterations**
WN18RR	400	300	800	100000
FB15K-237	1000	1000	200	200000
OGBL-BIOKG	400	600	850	700000
WN18RR-v_3-IND	300	1000	200	30000
WN18RR-v_4-IND	200	1400	10	30000
NELL-995-v_1-IND	200	300	600	20000
NELL-995-v_4-IND	200	1000	700	20000