A review of Knowledge graph link prediction using graph neural networks

Mohamad Zamini (Mohamad.Zamini@und.edu)
University of North Dakota
Hassan Reza
University of North Dakota
Minou Rabiei
University of North Dakota

Research Article

Keywords: Knowledge graphs, information extraction, knowledge graph embeddings

Posted Date: June 22nd, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1762419/v1

Abstract

Information extraction methods proved to be effective at triple extraction from structured or unstructured data. The organization of such triples in the form of (head entity, relation, tail entity) is called the construction of Knowledge Graphs (KGs). In order to use KGs in downstream tasks it is desirable to predict missing links in KGs. In this review we will study the current approaches and the challenges exists..

1. Introduction

A graph can be directed if the order of nodes in the graph is important or undirected if the order of the nodes in the graph is not important. A knowledge graph is a heterogeneous multi-digraph which means it is directed and multiple edges can exist between two nodes. Knowledge graph (KG) which is also known as knowledge base is a structured representation of facts that describes a collection of interlinked descriptions of entities, relationships and semantic descriptions of entities. The other advantage of KG is better representation of heterogeneous objects using a unified space to connect them.

The idea of structured knowledge in a graph was first introduced by [1] in 1988 and in 2012 this concept gained great attention after its usage in Google's search engine. Entities can be real-world objects, events, and abstract concepts which corresponds to a node and directed edges are considered as a relationships. The knowledge graph stores objective information structured in RDF-style triples which consists of two entities and one relation in the form of (*head, relation, tail*) or (*subject, predicate, object*) [2]. In a knowledge graph, labels are types of relations that can connect the facts; edges (relations) are specific facts connecting two nodes (head and tail entities).

FreeBase, WordNet, DBPedia, and Yago are some of the most common KGs with billions of facts and millions of entities. However, current real world knowledge graphs are usually incomplete and need inference engine to predict links and complete the missing facts among entities available in the KG. The process of completing incomplete triples (i.e. (Einstein, ?, Germany)) is called knowledge graph completion (KGC). An example of it can be seen in fig. 1. Common studies are focused on disambiguate entities which is called entity resolution, extract relations which is relation classification or inferring from already available information in KG which is called link prediction. In this review we focused on link prediction methods.

A common approach in link prediction is via embedding into vector spaces to learn representations of entities and relations and an embedding vectors of entities and relations can then be updated by maximizing the global plausibility. knowledge graph embedding compared to traditional one-hot representation approaches can better address the semantic computing by using a distributed representation method. A scoring function is defined to measure the plausibility of triples given the embeddings to enable updating the representation on the training data. Using different scoring functions in knowledge graph embeddings will reflect different designing criteria which will be discussed in next section. In this paper, we investigate comparative analysis of current scoring functions in knowledge graph embeddings and how they integrate with graph neural networks for knowledge graph link prediction task. unlike previous surveys we will focus more on graph neural network based approaches. The rest of the article is as follows. First we define knowledge embeddings and their scoring functions in section 2. In section 3 graph neural networks will be discussed. The challenges and gaps at section 4 will be described and in 5 the conclusion is expressed.

2. Knowledge Graph Embeddings

With the unprecedented growth of data volume around the world, utilizing traditional graph structure to construct and manipulate KG is hard. The traditional formal logic reasoning is not tractable or robust in large scale KGs. To calculate semantic relations between entities in KG, several link prediction embedding models have been recently proposed which will be discussed in this section.

KRL or knowledge graph embedding (KGE) is trying to map entities and relationships into a continuous vector space to better capture the semantic relation between entities in low dimensional space. For example, each entity *h* in a KG can be represented

by a point *h* in vector space and each relation *r* can be modeled as an operation like projection, translation, etc. in the space. The embedding procedure in a given KG starts with randomly representation of entities and relations in a vector space and with the help of an evaluation function the plausibility of each triples will be evaluated at each iterations. Then, the embedding vectors of entities and relations updates through optimization algorithms to maximize the global plausibility of facts.

Although various KGEs have been widely studied but still most of them can only train an embedding model based on its observed triples. Hence, most of current studies focused on generalizing KGE models. KGEs for relation prediction can be classified into translational, decompositional, CNN based and graph neural network based models[3]. In the following we will discuss each models separately.

2.1. Translational Models

Translational models interpret relations as simple translations over hidden entity represenations. TransE[4] is one of the common translational models where both entities and relations are considered as vectors in the same space. This model aims to model the inversion and composition patterns. Despite simplicity of TransE, it cannot perform well in one-to-many, many-to-one, and many-to-many relations [5, 6]. Although some of the complex models handle this issue but still they are not efficient in the process. For example relation *Writer of*, might learn similar vector representations for *Harry Potter*, *Fantastic Beasts and Where to Find Them*, and *The Ickabog* which are all books of *J.k. Rowling*. However, these entities are totally different. To overcome this issue, extensions of TransE inclduing TransH [5], TransR [6], TransD [7], TransM [8], and TransW [9] have been recently proposed which have different relation embeddings and scoring functions.

TransH [5] models a relation as a translating operation on a hyperplane with almost the same complexity as TransE. In this model, each relation is represented by two vectors, the norm vector of the hyperplane and the translation vector on the hyperplane. They addressed the issue of N-to-1, 1-to-N and N-to-N relations by enabling each entity have distinct distributed representations. The experiments on link prediction, triplet classification and fact extraction on benchmark datasets like WordNet and Freebase shows improvements compared to TransE.

TransE and TransH both simply put both entities and relations within the same semantic vector space. However, an entity may have multiple aspects and relations. each relation might focus on special aspect of that entity which might be far away from others. Besides, entities and relations are completely different objects which can make them not suitable to be represented in the same vector space. TransR build entity and relations embeddings in separate vector space and then build translation in the corresponding relation space. The comparison between TransR and the two previously introduced models shows significant improvements including link prediction, triple classification and relational fact extraction. TransR uses a projection matrix which projects entities from entity space to relation space [6].

TransD uses two vectors to represent an entity or a vector. One of them represents the meaning of the entity or relation and the other one is used to construct mapping matrix dynamically. This way, it covers both diversity of relations and entities. TransD is proposed to simplify TransR by eliminating matrix vector multiplication operations and also has less parameters which results in more applicability to large scale. The evaluation of the model in link prediction and triplet classification outperforms the previously mentioned models. In TransD, each entity-relation pair has a unique mapping matrix. The elimination of matrix vector operations in this model improved the performance.

In TransM [8] they leveraged the structure of knowledge graph via pre calculating the distinct weight for each training triplet regarding its relational mapping property. In this model, optimal function deals with each triplet based on its own weight. In this model, the transition model for triplets will be hold the same as TransE but the optimal function they proposed uses pre-calculated weight corresponding to the relationship. The main difference between TransE and TransM is it is more flexible when dealing with heterogenous mapping properties of KGs by minimizing margin based hinge loss function. The proposed model outperformed in link prediction and triplet classification tasks.

Recently, TransW[9] proposed using word embeddings for knowledge graphs embeddings to better deal with unseen entities or relations. Unlike previous works which ignores the detail of the words within triples, TransW aims to enrich a KG by missing

entities and relations using word embeddings. linear combination of word embedding of entities and relations in this model leads to detect unknown facts. The word embedding for relation and entities is calculated separately using Hadamard product. The results was outperforming compared to previous translational approaches.

RotatE [10] is another translational based approaches for KG representation learning. This model is able to infer different relation patterns of symmetry and antisymmetry. RotatE model defines each relation as a rotation from source entity to target entity in the complex vector space. Some relations are symmetric like marriage and some are antisymmetric like filiation; some relations are inverse like hypernym and hyponym; and finally some are composed of others like my dad's wife is my mom. How to infer these characteristics in KGs are essential to predict missing links. Unlike the above mentioned models RotatE aims to model and infer these characteristics at the same time.

HAKE [11] is a translational distance model with some similarities to RotatE [10]. Despite RotatE, HAKE aims to model the semantic hierarchy rather than modeling relation patterns. Unlike RotatE which models relations as rotations leads two entities to the same modulus, HAKE explicitly models modulus information.

Table 1. Scoring functions of state of the art translational based knowledge graph embedding models

Model	Score Function	Memory Complexity
TransE	$\|\|h + r - t\|\|_{l_1/l_2}$	$O(N_e d + N_r d)$
TransH	$\|\|(h - w_r^T h w_r) + d_r - (t - w_r^T t w_r)\|\|_2^2$	$O(N_e d + N_r d)$
TransR	$\|\|M_r h + M_r t\|\|_2^2$	$O\left(N_e d + N_r (d^2 + d\ )\right)$
TransD	$\|\|(r_p h_p^T + I)h + r - \left(r_p r_p^T + I\right)t\|\|_2^2$	$O(2N_e d + 2N_r d)$
TransM	$w_r \|\|h + r - t\|\|_{l_1/l_2}$	$O(N_e d + N_r k)$
TransW	$\|\| \left(\sum h_i \otimes w_{hi} + b_h\right) + \sum r_i \otimes w_{ri} - (\sum r_i \otimes w_{ti} + b_t)\|\|_{1/2}^2$	
RotatE	$-\|\|h \odot r - t\|\|$	$O(2N_e d + 2N_r d)$
HAKE	$\|\|h_m \circ r_m - t_m\|\|_2 + \lambda \|\|\sin((h_p + r_p - t_p)/2)\|\|_1$	$O(2N_e d + 2N_r d)$

Translational models compared to other models which will be discussed in the following are arguably faster, easier to train and have fewer parameters to fine-tune. However, results lacks expressiveness. Expressivity of the models for graphs can be defined as the diversity of their graph representations.

2.2. Tensor dompositional Models

The other types of embedding models is tensor decompositional models which uses tensor products to capture rich interactions. Tensor is a multidimensional numeric field that generates alises scalars, vectors, and matrices[12]. Among compositional models RESCAL [13] and its extensions are more trending. RESCAL uses a vector to capture to represent the latent semantics of each entities and a matrix which models pairwise interactions among latent factors [14]. The issue of this approach is the number of parameters which is $o(d^2)$. Also, RESCAL has never been tested on data with large numbers of relation types[15]. RESCAL has large number of parameters and this makes it prone to overfitting.

To simplify RESCAL, DistMult [16] proposed using bilinear diagonal matrices which is a special case of bilinear objective used in NTN and reduces the number of parameters to $o(d)$ per relation. In RESCAL each relation is represented by a square matrix and DistMult simplify it by using a diagonal matrix. The issue of DistMult is it can just deal with symmetric relations because of using diagonal matrices. Other than that, is a model, characterised by three way interactions between embedding parameters to produce a single feature per parameter. The linear transformation on entity embedding vectors cannot model

asymmetric relations. Using such models learn shallow features with less expressive features but it is scalable to large knowledge graphs[17].

ComplEx [18] also aimed to generalize DistMult by proposing complex-valued embeddings to improve asymmetric relations modeling. ComplEx embedding method infers new relational triplets with asymmetrical Hermitian product. In this model, entity and relation embeddings are in a complex space rather than real space. This enables ComplEx model asymmetric relations.

[19] subsumes DistMult and ComplEx with more generalizability by exploiting hypercomplex space for learning KG embeddings. Unlike standard vector space with a single component *i*, each quaternion embedding is a vector in the hypercomplex space H with the imaginary components *i,j,k* with a new scoring function with relational quarternion embedding through Hamilton product. It has been proved that Hamilton operator compared to Hermitian and inner product in Euclidean space has better expressiveness. However, rotation based models cannot model hierarchical structure. This model also is not capable of modeling multiple relations between two entities at the same time. To solve above mentioned issues, a new method called dual quaternion KGE (DualE) [20] is proposed. Embeddings in dual quaternion space are vectors in hypercomplex space. This model integrates and unifies translation and rotation operations.

Tucker [21] employs a different decomposition model called a Tucker Decomposition to compute a smaller core tensor and a sequence of three matrices where each matrix represents entity embedding and relation embedding separately.

Table 2
Scoring functions of state of the art tensor decompositional based knowledge graph embedding models

Model	Score function	Memory Complexity
RESCAL	$h.W_r.t$	$O\left(N_e d + N_r d^2\right)$
DistMult	$\langle h, r, t\rangle$	$O\left(N_e d + N_r d\right)$
ComplEx	$Re\left(\langle h, r, \bar{t}\rangle\right)$	$O\left(2N_e d + 2N_r d\right)$
Quaternion	$h \otimes r^{\triangleleft}.t$	$O\left(N_e d + N_r d\right)$
DualE	$h \otimes r^{\triangleleft}.t$	$O\left(N_e d + N_r d\right)$
Tucker	$W \times_1 h^T \times_2 M_r \times_3 t$	$O\left(N_e d + N_r d + d_e d_r d_e\right)$

The issue of the above mentioned models are the learned embeddings should solely be compatible within each individual fact. In another word, the above mentioned models only consider structural information observed in triples rather than external information. This results in incompatibility of downstream tasks[22]. This motivates researchers to include other information like entity types [23], logical rules [22], and relation paths [24] to improve the learning of embeddings.

2.3. Neural Network Models

Semantic matching energy (SME) [15] is one of the early neural network models which first projects entities and relations to their vector embeddings in the input layer. The proposed model capture the inherent complexity in the data by defining similarities among entities and relations. The energy function is encoded using neural network to extract relevant components of each argument's embedding by using relation types. The computed results can be comparable in a space. It works based on

energy function to assign low energies to plausible triplets of multi relational graph. Head entity and relation are combined and make a function, tail entity and relation also make another function in hidden layer. Then they calculate the fact score by their dot product of two function.

Neural Tensor Network (NTN) [25] is an expressive neural tensor network which is capable of reasoning over relations between entities. In this work, unlike previous works they represent each entity as the average of its word vectors. They used bilinear tensor layer that directly related two entity vectors across multiple dimensions. The model computes the relation by an NTN-based function. The scoring function of this model is:

$$f_r(h,t) = u_r^T \tanh(h^T M_r t + M_{r,1} h + M_{r,2} t + b_r)$$

1

The issue of these models are they learn more shallow and less expressive features than multi-layer models which limits the performance of KGs. Link prediction should be manageable in number of parameters and computational costs to be useful for KGs. To do so, they sacrifice the accuracy over speed by using simple operations like inner products and matrix multiplications over an embedding space [17]. To increase the expressiveness, it is essential to increase the embedding size. However, increasing the embedding size is proportional to the number of entities and relations exists in the graph. An intuitive example will be a model like DistMult with the embedding size of 200 on Freebase dataset will require 33 GB memory regarding its parameters[17]. Using fully connected models in multi-layer KGEs can be prone to overfitting. To solve that, convolutional layers evolved which are highly optimized to GPU.

Table 3
Scoring functions of state of the art tensor decompositional based knowledge graph embedding models

Model	Score function	Memory Complexity
SME	$g_{left}(h,r)^T g_{right}(r,t)$	$O(N_e d + N_r d)$
NTN	$r^T \tanh(h^T \widehat{M} t + M_{r1} h + M_{r2} t + b_r)$	$O\left(N_e d + N_r d^2 d\right)$

2.4. Convolutional based models

Although previous models were fast models and can be scaled to large KGs, they learn less expressive features than multi-layer models. ConvE [17] yields the same performance as DistMult and R-GCN[26] with much fewer parameters and effective at modelling nodes with high indegree. It uses 2D convolution layers for link prediction which consists of a convolution layer, a projection layer deals with embedding dimension, and an inner product layer. ConvE generates a matrix by wrapping each vector over several rows and concatenate the matrices. Although this model shows outperforming results but this model compared to common convolutional models in computer vision and other area is still shallow and needs to study deeper models to improve the performance. HypER [27] also applies convolutions but it uses a fully connected layer to avoid such wrapping and generate relation specific convolutional filters.

The number of interactions that ConvE can capture between relation and entity embeddings is limited. To increase this number, InteractE [28] is proposed which is based on multiple permutations to better capture possible interactions, substituting simple feature reshaping used in ConvE with checked reshaping and circular convolution to capture more feature interactions in a depth-wise manner. This way the interaction between entity and relation embeddings for learning better representations and circular convolution will be enhanced.

ConvKB extends ConvE by omitting the reshaping operation in encoding of representations in the convolution operation [29]. ConvKB [30] uses CNN to capture global relationships and transitional characteristics between entities and relationships. Each triple in this model is represented by a 3 column matrix where each column represents a vector of each elements of a triple. The matrix is input of a convolution layer to map to different feature space and then concatenate them to create a single feature vector as a input triple representation. Its plausibility score is calculated via dot product of the feature vector and a weight vector. Despite ConvE, ConvKB covers global relationships between same dimensional entries of an embedding triple.

ConEx [31] is a Hadamard product composition of a 2D convolution followed by an affine transformation and a Hermitian inner product on complex valued embeddings. The proposed model uses the asymmetric properties of Hermitian products and parameter sharing property of a 2D convolution. Hermitian product has been previously used in ComplEx embedding model which shows good expressiveness, however results shows that Hamiltion product is more expressive.

Although both previous convolutional models are parameter efficient and they showed outperforming results, but they consider each triples independently without considering the possible relationships between triples. Another challenge in embedding approaches mentioned above is failure to capture multistep relationships and most of them solely work on the observed facts. They train representation of each node and edge based on the context of triples they are involving. To solve these challenges, instead of computing numerical representations for graphs, an alternative approach is using machine learning architectures for graphs. Graph neural networks have been studied widely as a solution of these challenges. In graph neural networks, edges serve as weighted connections and nodes serve as neurons.

Table 4
Scoring functions of state of the art convolutional based knowledge graph embedding models

Model	Score function	Memory Complexity
ConvE	$f\left(vec\left(f\left(\left[\bar{h};\bar{r}\right]\star\Omega\right)\right)W\right)t$	$O\left(N_e d + N_r d + T m_\Omega n_\Omega + T d(2d_m - m_\Omega + 1)\left(d_n - n_\Omega + 1\right)\right)$
ConvKB	$concat\left(f\left(\left[h,r,t\right)\star\Omega\right)\right)w$	$O\left(N_e d + N_r k + 4T\right)$
HypER	$f\left(vec\left(h\star vec^{-1}\left(w_r H\right)\right)W\right)t$	$O\left(N_e d + N_r d\right)$
InteractE	$f\left(f(\,perm\left(\left[h;r\right]\right)\circledast w)\,W + b\right)t$	$O\left(N_e d + N_r d + T m_\Omega n_\Omega + 2Tpd^2\right)$
ConEx	$Re\left(\left\langle conv\left(h,r\right),h,r,\bar{t}\right\rangle\right)$	

3. Graph Neural Networks

Deep learning approaches have been exploited for graph data modeling and representation. An essential step to perform tasks on graph structured data is to learn better representations. Conventional neural networks are limited to handle only Euclidean data. By leveraging representation learning, graph neural networks have generalized deep learning models to perform on graph structural data with good performance. In GNNs an iterative process, propagates the entity state until equilibrium. This idea was extended by [32] to use gated recurrent units in propagation step. Graph neural network (GNN) models have been proved to be a powerful family of networks that learns the representation of an entity by aggregation of the features of the entities and neighbors [33]. Many GNNs have achieved state of the art performance. GNNs generally update node representations by

aggregating and propagating node features in the graph. Unlike embeddings, GNNs are capable of end-to-end supervised learning which can be used to perform various classification tasks[34].

Several attempts have been exploited to apply neural networks to deal with structured graphs. Recursive neural networks were applied as early work to process data in acyclic graphs [35]. This idea has been extended to graph neural networks in [36] to generalize recursive graph neural networks for directed and undirected graphs. They generally learn the target node's representation by its neighbor's information iteratively until it reaches to equilibrium point. Graph neural networks with the key factor of high dimensional data growing have been widely studied and applied to learn representations from complex graph-structured data with remarkable performance in different domains.

Overall graph based models regarding their embedding dimensionality can be classified into low-dimensional and high-dimensional embedding. Low-dimensional embeddings of nodes in large graphs proved extremely useful in different prediction tasks. However, most of existing approaches require all nodes in the graph to be present at the time of training the embedding model. These approaches are transductive and are not capable of being generalized to unseen nodes. Another group of approaches has been recently studied which are inherently inductive which generate node embeddings from previously unseen data. Generating new node embeddings in inductive setting is more difficult because generalizing to unseen nodes require aligning newly observed subgraphs to the node embeddings[37]. BoxE [38] and GraphSAGE [37] are examples of inductive embedding. GraphSAGE leverages node feature information to generate node embedding for unseen data in undirected graphs. BoxE modeling is region based supervised embedding learning which embeds entities as points and relations as a set of hyper-rectangles to spatially characterize logical rules and easier calculations in similarity representations.

3.1. Convolutional neural network models

By evolving CNNs, convolution exploited on graph data in parallel. Convolutional graph neural networks (ConvGNNs) are mainly divided into two main approaches: spectral based which depends on graph structure by depending on the Laplacian eigenbasis and spatial based which works on sampling a fixed-size neighborhood of each node and aggregating over it. This approach has proved powerful in several large scale inductive benchmarks [39].

Graph convolutional networks have gained a lot of attention which work under an encoder-decoder framework to aggregate local information in the graph neighborhood for each node. Similar to convolutional neural networks which operate over local regions of input data, GCNs go over a node and its neighbours in the graph.

In GCNs, convolution operator uses locality information in graphs to leverage attributes associated with nodes[40]. However, general GCN is limited to undirected graphs and are not suitable for highly multi-relational data which are directed[17]. Relational Graph Convolutional Network (R-GCN) [26] is one of the well-known approaches which is developed for knowledge base completion tasks such as link prediction and entity classification. GCN represents as a graph encoder but to make it capable of specific tasks it needs to be developed into R-GCN which takes the neighborhood of each entity equally by hierarchical propagation rules to be suitable for directed graphs. The encoder maps each entity to a real-valued vector and the decoder (scoring function) reconstructs edges of the graph based on vertex representations. In this model, DistMult factorization is used as scoring function where every relation is related to a diagonal matrix. Optimizing cross-entropy loss pushes the model to pick the better triples than negative ones. However, R-GCN does not take relation or attribute similarity between entities into account. Also, using R-GCN's scoring function generates many negative triples for a positive triple[41]. RA-GCN [42] have been proposed to solve these issues by improving the propagation extension for entity updating and extract additional entity and relation information through aggregation.

Unlike R-GCN which entity embedding learning was done through convolution based encoder and relation embedding learning was in the decoder, TransGCN [43] trains relation and entity embeddings simultaneously during graph convolution operation with fewer parameters compared to R-GCN by using relation as transformation operator on between head and tail entity in a triple. They used transE and RotatE on both datasets and RotatE-GCN showed better results compared to TransE-GCN. However, in TransGCN the relation embedding during learning ignores entity representations. To solve this issue, KE-GCN [44]

leveraged the strength of GCN model and KGC methods for relation and entity embedding updates. KE-GCN is heterogeneous learning model with the focus on jointly propagating and updating of knowledge embedding of both nodes and edges. Similarly, COMPGCN [45] also uses jointly vector representation learning for both nodes and edges in multi-relational graphs by leveraging various entity-relation composition operations from KGE models.

Table 5
Graph Neural Network based update functions

Model	Relation update	Entity update
R-GCN	-	$h_i^{(l+1)} = \sigma(h_i^{(l)} W_0^{(l)} + \sum_{j \in N_i^r} \sum_{r \in R} h_j^{(l)} W_r^{(l)})$
RA-GCN	-	$h_i^{(l+1)} = \sigma(h_i^{(l)} W_0^{(l)} + \sum_{j \in N_i^r} \sum_{r \in R} h_j^{(l)} W_r^{(l)} + \sum_{a \in A} h_a^{(l)} W_a^{(l)})$
TransE-GCN	$r_k^{l+1} = \sigma\left(W_1^{(l)} r_k^{(l)}\right)$	$v_i^{l+1} = \sigma(m_i^{l+1} + v_i^{(l)})$
KE-GCN	$m_r^{l+1} = \sum_{(u,r) \in \mathcal{N}(r)} \frac{\partial f_r^l(h_u^l, h_r^l, h_v^l)}{\partial h_r^l}$	$m_v^{l+1} = \sum_{(u,r) \in \mathcal{N}_{in}(v)} W_r^l \frac{\partial f_{in}^l(h_u^l, h_r^l, h_v^l)}{\partial h_u^l} + \sum_{(u,r) \in \mathcal{N}_{out}(v)} W_r^l \frac{\partial f_{out}^l(h_v^l, h_r^l, h_u^l)}{\partial h_v^l}$
CompGCN	$h_r^{l+1} = W_{rel}^l h_r^l$	$m_v^{l+1} = \sum_{(u,r) \in \mathcal{N}_{in}(v)} W_r^l \varphi_{in}(h_u^l, h_r^l) + \sum_{(u,r) \in \mathcal{N}_{out}(v)} W_r^l \varphi_{out}(h_u^l, h_r^l)$

Although proposed GCNs can effectively improve the accuracy but scalability is a major challenge for them. PinSage [46] can be a promising solution which is capable of handling billions of nodes and edges however to the best of our knowledge it has not been tested for knowledge graphs. In this approach they select a fixed number of neighbors for all given nodes which might result in dropping out some neighboring nodes and information loss in multi-relational data.

3.2. Attention neural network models

Unlike application of CNNs in cases of images which have predictable number of neighbours, in graph data the neighbourhood of nodes in a graph are unpredictable. To address this challenge, attention mechanism is proposed to learn those nodes that have more important features to the current node. Attention mechanisms have been widely used in sequential tasks. Attention mechanisms are suitable for dealing with variable sized inputs and working on the most relevant parts of inputs. If the attention mechanism is executed to compute a single sequence representation, it will be called self-attention. Self-attention architecture is parallelizable across node-neighbor pairs and also can be applicable to tasks where the model needs generalize to completely unseen graphs[39].

Despite GCNs which all neighbors share fixed weights and contributing equally during information passing, graph attention networks assign different levels of importance to each neighborhood of a specific node. In graph attention networks, layers are stacked and nodes are able to attend over their neighbours. Each node will hold different weight in a neighborhood. The advantage of these networks is that they don't require any knowledge about the structure of the graph and any matrix operations [39]. Graph attention network (GAT) [39] uses multi-head attention to stabilize the learning process and boost performance by concatenating *n* attention heads. However, using multi-head attention can have large size of parameters. To address this issue, relation aware graph attention network (RAGAT) [47] is proposed which defines relation aware message passing functions parameterized by relation specific network parameters and employs averaging instead of concatenating *n*

attention head. To validate the results of this model, the decoder (scoring function) uses two different decoders: ConvE and InteractE.

The other issue of GAT is they ignore relation features [3]. To solve this issue, a novel embedding is required to incorporate relation and neighboring node features in the attention mechanism. KBGAT [3] is a multi-hop and semantically similar relation extraction in the n-hop neighbourhood of any given entity in the knowledge graph. Learning embeddings are performed through a linear transformation to get absolute attention value over concatenation of entity and relation feature vectors corresponding to a particular triple. Although the results are impressive but the disadvantage of this method is the computational costs and requirement of pre-trained KGE as input for the neural network.

A novel aggregation of neighborhood strategy with local structure for knowledge graph completion have been proposed by [48]. LSA-GAT uses local structures to derive a sophisticated representation which covers semantic and structural information. The combination of LSA-GAT, local structure representation module, feature fusion module and CNN based decoder in this work showed significant results. However, just aggregating neighborhood entities fails to effectively model the critical relationships and ignore the distinct aspects of entities and relations. DisenKGAT [49] tried to learn the disentangled representation of entities by using micro- and macro-disentanglement as property of the KG. The robustness of this model is it can work with different kinds of score functions.

HRAN [50] proposed an attention based model for heterogeneous graph networks such as knowledge graphs which have various types of entities and relations by aggregating features from different semantic aspects and dedicating weights to the relation path. They aggregate the neighbors features of an entity first and the importance of each relation-paths is learned through relation features. The extracted features are aggregated with learned weights and generate embedding representations. The node feature aggregation of this model is performed through graph convolution. Next, due to heterogeneity in KGs, an entity-level relation-path-based aggregation is used. In the relation-level aggregation step, a novel relation-based attention mechanism to obtain importance of different relation-paths is proposed.

DecentRL [33] is a KG representation learning approach which encodes each node from the embeddings of its neighbors. This approach unlike other GNNs which consider the representation of both an entity itself and its neighbors, it can be generalized to represent unseen entities by just learning representations from their context neighbors. The idea of decentRL is based on averaging its neighbor embeddings which makes decentralize the semantic information of entites over their neighbors. DecentRL works based on decentralized attention network (DAN). DAN and GAT have identical layers but DAN has decentralized structure. In this case the entity participates in the attention scores and the aggregation of neighbors. If the entity is an open entity then GAT generates the embedding completely random and will be almost meaningless. In contrast, DAN generates the embedding of the entity without the requirement of its embedding. This shows the robustness and more descriptive aspect of DAN rather conventional GAT. This models is considered as prototype of graph attention mechanism under open-world setting. They proposed an efficient knowledge distillation algorithm for generating unseen entities. The results shows outperforming in entity alignment and entity prediction tasks compared to current models under open-world settings like AlignE, GAT and AliNet.

3.3. Pre-trained Neural Network models in knowledge graphs

Knowledge graph construction is mainly supervised and requires humans to manually define all the facts like Wikidata or Freebase. Extracting the facts can also be done in a semi-supervised way which still needs human supervision. With the evolution of language models such as BERT, outperforming results in various natural language tasks have been achieved. However, pre-trained language models (PLMs) struggle to capture rich knowledge. Existing PLMs learn useful knowledge from unlabeled text and cannot capture the facts well because of its sparsity and complex forms in text. In contrast, knowledge embedding models can represent relational facts in structured data rather than an unstructured text corpora. Recently some works have been studied the applicability of pre-trained models in the context of KGs which proved to be a promising solution.

The unified knowledge embedding and pre-trained language representation (KEPLER) [51] is proposed to integrate factual knowledge into pre-trained language model in addition to producing effective text-enhanced knowledge embedding. The textual entity descriptions are encoded with a pre-trained language model as their embeddings to jointly optimize the KE and language modeling objectives. In this approach entities are encoded into vectors by using their corresponding text. They produced Wikidata5M from Wikipedia data dump 2019 on two different settings of transductive and inductive. In transductive setting entities are shared and triple sets are disjoint in train, test and validation data; while in inductive setting, the entities and triplets are mutually disjoint across train, validation and test.

[52] presented an in-depth analysis of to what extent pretrained language models can store factual and common sense relational knowledge. The results on relational knowledge bases shows without fine-tuning, BERT provide competitive results compared to traditional methods in case of relational knowledge. They also showed that BERT can outperform on open-domain question answering compared to supervised methods. Finally they proved some certain types of factual knowledge are readily learned. However, this work evaluates knowledge present and didn't investigate link prediction model in OKGs.

[53] proposed an unsupervised end-to-end model named Match and Map (MAMA). It constructs KGs with a single forward passing of pre-trained language models. In the Match stage a set of candidate facts from a corpora will be created. The stored knowledge in the pre-trained language model will be matched with the target corpora at this stage and each of the extracted triples will be passed to the Map stage. In Map stage an OKG will be constructed with the candidate facts. If the constructed facts can be framed in a fixed KG schema then it will be mapped according to Wikidata schema. If the candidate is in an open schema, then it will be partially mapped. The issue of this work will be in increase the size of unmapped facts in large scale and lead to performance problems.

[54] showed BERT may not predict the correct entity for OKGs, but still it can well predict type compatible entities. The experiment result for entity linking was also the same[55]. As mentioned before, OKGs do not have underlying ontology. Hence, providing type information is expensive and time-consuming. Which BERT perdictions can improve OKGs link prediction[54]. In this work, they used applied BERT for improving OKG link prediction with a novel scoring function. OKGIT aimed to use unsupervised implicit type information present in the pre-trained BERT model into OKG embeddings instead of explicit entity types present in ontology. Results outperformed ConvE and CaRE.

[56] proposed knowledge graph using BERT pre-trained model to improve the performance with rich language information by capturing rich semantic patterns from free text. BERT predicts whether two input sentences are consecutive or not. A sentence in original BERT can be an arbitrary span of contiguous text or word sequence[56]. The plausibility of a triple is calculated by considering the sentences of (h,r,t) as a single sequence. The model uses sentences of entities h and t to predict the relation r between them. this way the knowledge graph completion task is converted into a sequence classification problem. The results shows in link prediction it outperformed in comparison with TransE, TransR, TransH, TransD and DistMult.

The important issue of current approaches which use language models are all trained based on general datasets like Wikipedia. As a result, if OKG wants to be specific in an area there might be issues in accuracy and it requires to pre-train a language model related to that area.

Table 6
Link prediction with different embedding settings. Lower values in MR and Higher Hits are better

Model	WN18RR		FB15k-237	
	MR	Hits@10	MR	Hits@10
TransE [10]	3384	50.1	357	46.5
TransH [56]	2524	50.3	255	48.6
TransR[56]	3166	50.7	237	51.1
TransD[56]	276	50.7	246	48.4
DistMult[56]	3704	47.7	411	41.9
ComplEx[56]	3921	48.3	508	43.4
Tucker [21]	-	52.6	-	54.4
ConvE [17]	5277	48	246	49.1
InteractE [28]	5202	52.8	172	53.5
ConvKB [30]	3324	52.4	311	42.1
ConEx [29]	-	55	-	55.5
LSA-GAT [39]	1947	44	273	60
HARN [50]	2113	54.2	156	54.1
R-GCN [26]	-	-	-	41.7
RotatE-GCN [43]	-	55.5	-	57.8
TransE-GCN [43]	-	47.7	-	50.8
COMPGCN [45]	3533	54.6	197	53.5
RotatE [10]	3384	50.1	177	53.3
HAKE [11]	-	58.2	-	54.2
KG-BERT [56]	97	52.4	153	42.0
QuatE [10]	2314	58.2	87	55
DualE [20]	2270	44.4	91	55.9
DisenKGAT [49]	1504	57.8	179	55.3
RAGAT[47]	2390	56.22	199	54.7
KBGAT [3]	1921	55.4	270	33.1
Inverse Model [17]	13219	36	7148	1.2
decentRL + TransE [33]	-	-	159	52.1
decentRL + DistMult[33]	-	-	151	54.1
RGCN + TransE[33]	-	-	325	44.3
RGCN + DistMult [33]	-	-	230	49.9

4. Challenges In Knowledge Graphs

Since now many challenges remained unsolved. These challenges include scalability, entity disambiguation, knowledge extraction from heterogeneous and unstructured data, and managing evolving knowledge management. Most current works are with the assumption of static knowledge graph and not changing the facts over time. A promising direction will be studying dynamic graph algorithms which can handle addition of new edges during a continuous time. Most of current representation learning lacks from capability of using in multilingual KGs. Using multi-source knowledge bases and multi-modality in knowledge graphs are not still well studied.

Other than the above mentioned challenges, most current real world knowledge graphs have low quality and constructing a special domain knowledge graph is cumbersome.

Most currently available KG embeddings are applicable in ontological KGs rather than OKGs. While most ontological KGs are canonicalized, OKGs are not. As an example Donald Trump, President Trump, and Trump are the same. Recently some works has been proposed but they are not still in state of the art point.

With the growth of graphs at scale extraction of knowledge from multiple structured and unstructured sources is still challenging. Using supervised learning approaches needs human annotation which is time consuming. This leads the researchers to use unsupervised and semi-supervised approaches.

5. Conclusions

Knowledge graph embedding as a method for embedding entities and relations into low-dimensional continuous vector space should impressive results in representing information in a structural manner. This paper reviewed the main scoring functions which can result in different levels of expressiveness of facts. They classified into translational, decompositional, neural network based and convolutional based models. Unlike previous works which focused on the knowledge embedding models, we added how they can be integrated with graph neural networks to predict links. For this purpose we introduced different graph neural network based knowledge graph embedding. Overall, results shows that the early models such as TransE although have faster computations but they carry out less expressiveness in results. On the other hand, using more complex scoring functions can also result in higher computational cost but better expressiveness. Graph neural network models which mostly work on encoder decoder based are one promising solution for link prediction and knowledge graph completion. Although many researches have been done but still most of current approaches are based on general datasets which are publicly available and the results in special domain are not well studied or evaluated.

Declarations

Conflicts of Interest:

The authors declare no conflict of interest.

References

1. Stokman FN, Vries PHd (1988) *Structuring knowledge in a graph*, in *Human-computer interaction*. Springer, pp 186–206
2. Wang S et al (2019) Decentralized construction of knowledge graphs for deep recommender systems based on blockchain-powered smart contracts. IEEE Access 7:136951–136961
3. Nathani D et al (2019) *Learning attention-based embeddings for relation prediction in knowledge graphs.* arXiv preprint arXiv:1906.01195,
4. Bordes A et al (2013) *Translating embeddings for modeling multi-relational data.*Advances in neural information processing systems, **26**
5. Wang Z et al (2014) *Knowledge graph embedding by translating on hyperplanes*. in *Proceedings of the AAAI Conference on Artificial Intelligence*.

6. Lin Y et al (2015) *Learning entity and relation embeddings for knowledge graph completion*. in *Twenty-ninth AAAI conference on artificial intelligence*.
7. Ji G et al (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)*.
8. Fan M et al (2014) *Transition-based knowledge graph embedding with relational mapping properties*. in *Proceedings of the 28th Pacific Asia conference on language, information and computing*.
9. Ma L et al (2019) *Composing knowledge graph embeddings via word embeddings.* arXiv preprint arXiv:1909.03794,
10. Sun Z et al (2019) *Rotate: Knowledge graph embedding by relational rotation in complex space.* arXiv preprint arXiv:1902.10197,
11. Zhang Z et al (2020) *Learning hierarchy-aware knowledge graph embeddings for link prediction*. in *Proceedings of the AAAI Conference on Artificial Intelligence*.
12. Hogan A et al (2021) *Knowledge graphs.* Synthesis Lectures on Data. Semant Knowl 12(2):1–257
13. Nickel M, Tresp V, Kriegel H-P (2011) *A three-way model for collective learning on multi-relational data*. in *Icml*.
14. Wang Q et al (2017) Knowledge graph embedding: A survey of approaches and applications. IEEE Trans Knowl Data Eng 29(12):2724–2743
15. Bordes A et al (2014) A semantic matching energy function for learning with multi-relational data. Mach Learn 94(2):233–259
16. Yang B et al (2014) *Embedding entities and relations for learning and inference in knowledge bases.* arXiv preprint arXiv:1412.6575,
17. Dettmers T et al (2018) *Convolutional 2d knowledge graph embeddings*. in *Proceedings of the AAAI Conference on Artificial Intelligence*.
18. Trouillon T et al (2016) *Complex embeddings for simple link prediction*. in *International conference on machine learning*. PMLR
19. Zhang S et al (2019) *Quaternion knowledge graph embeddings*.Advances in neural information processing systems, **32**
20. Cao Z et al (2021) *Dual quaternion knowledge graph embeddings*. in *Proceedings of the AAAI Conference on Artificial Intelligence*.
21. Balažević I, Allen C, Hospedales TM (2019) *Tucker: Tensor factorization for knowledge graph completion.* arXiv preprint arXiv:1901.09590,
22. Wang Q, Wang B, Guo L (2015) *Knowledge base completion using embeddings and rules*. in *Twenty-fourth international joint conference on artificial intelligence*.
23. Guo S et al (2015) *Semantically smooth knowledge graph embedding*. 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)*.
24. Lin Y et al (2015) *Modeling relation paths for representation learning of knowledge bases.* arXiv preprint arXiv:1506.00379,
25. Socher R et al (2013) *Reasoning with neural tensor networks for knowledge base completion.*Advances in neural information processing systems, **26**
26. Schlichtkrull M et al (2018) *Modeling relational data with graph convolutional networks*. in *European semantic web conference*. Springer
27. Balažević I, Allen C, Hospedales TM (2019) *Hypernetwork knowledge graph embeddings*. in *International Conference on Artificial Neural Networks*. Springer
28. Vashishth S et al (2020) *Interacte: Improving convolution-based knowledge graph embeddings by increasing feature interactions*. in *Proceedings of the AAAI conference on artificial intelligence*.

29. Yu D et al (2020) *Jaket: Joint pre-training of knowledge graph and language understanding.* arXiv preprint arXiv:2010.00796,
30. Nguyen DQ et al (2017) *A novel embedding model for knowledge base completion based on convolutional neural network*.arXiv preprintarXiv:1712.02121,
31. Demir C, Ngomo A-CN (2021) *Convolutional complex knowledge graph embeddings*. in *European Semantic Web Conference*. Springer
32. Li Y et al (2015) *Gated graph sequence neural networks.* arXiv preprint arXiv:1511.05493,
33. Guo L et al (2020) *Decentralized Knowledge Graph Representation Learning.* arXiv preprint arXiv:2010.08114,
34. Hogan A et al (2021) *Knowledge Graphs*.ACM Comput. Surv., **54**(4): p. Article 71.
35. Sperduti A, Starita A (1997) Supervised neural networks for the classification of structures. IEEE Trans Neural Networks 8(3):714–735
36. Gori M, Monfardini G, Scarselli F (2005) *A new model for learning in graph domains*. in *Proceedings. IEEE international joint conference on neural networks*. 2005
37. Hamilton W, Ying Z, Leskovec J (2017) *Inductive representation learning on large graphs*.Advances in neural information processing systems, **30**
38. Abboud R et al (2020) Boxe: A box embedding model for knowledge base completion. Adv Neural Inf Process Syst 33:9649–9661
39. Veličković P et al (2017) *Graph attention networks.* arXiv preprint arXiv:1710.10903,
40. Shang C et al (2019) *End-to-end structure-aware convolutional networks for knowledge base completion*. in *Proceedings of the AAAI Conference on Artificial Intelligence*.
41. Kazemi SM, Poole D (2018) *Simple embedding for link prediction in knowledge graphs*.Advances in neural information processing systems, **31**
42. Tian A et al (2020) *RA-GCN: Relational aggregation graph convolutional network for knowledge graph completion*. in *Proceedings of the 12th International Conference on Machine Learning and Computing*. 2020
43. Cai L et al (2019) *TransGCN: Coupling transformation assumptions with graph convolutional networks for link prediction*. in *Proceedings of the 10th International Conference on Knowledge Capture*.
44. Yu D et al (2021) *Knowledge embedding based graph convolutional network*. in *Proceedings of the Web Conference 2021*.
45. Vashishth S et al (2019) *Composition-based multi-relational graph convolutional networks.* arXiv preprint arXiv:1911.03082,
46. Ying R et al (2018) *Graph convolutional neural networks for web-scale recommender systems*. in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*.
47. Liu X et al (2021) RAGAT: Relation aware graph attention network for knowledge graph completion. IEEE Access 9:20840–20849
48. Ji K, Hui B, Luo G (2020) Graph attention networks with local structure awareness for knowledge graph completion. IEEE Access 8:224860–224870
49. Wu J et al (2021) *DisenKGAT: Knowledge Graph Embedding with Disentangled Graph Attention Network*. in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*.
50. Li Z et al (2021) Learning knowledge graph embedding with heterogeneous relation attention networks. IEEE Transactions on Neural Networks and Learning Systems
51. Wang X et al (2021) KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation. Trans Association Comput Linguistics 9:176–194
52. Petroni F et al (2019) *Language models as knowledge bases?* arXiv preprint arXiv:1909.01066,
53. Wang C, Liu X, Song D (2020) *Language models are open knowledge graphs.* arXiv preprint arXiv:2010.11967,
54. Talukdar PP (2021) *OKGIT: Open Knowledge Graph Link Prediction with Implicit Types.* arXiv preprint arXiv:2106.12806,

55. Chen S et al (2020) *Improving entity linking by modeling latent entity type information*. in *Proceedings of the AAAI conference on artificial intelligence*.
56. Yao L, Mao C, Luo Y (2019) *KG-BERT: BERT for knowledge graph completion.* arXiv preprint arXiv:1909.03193,

Figures

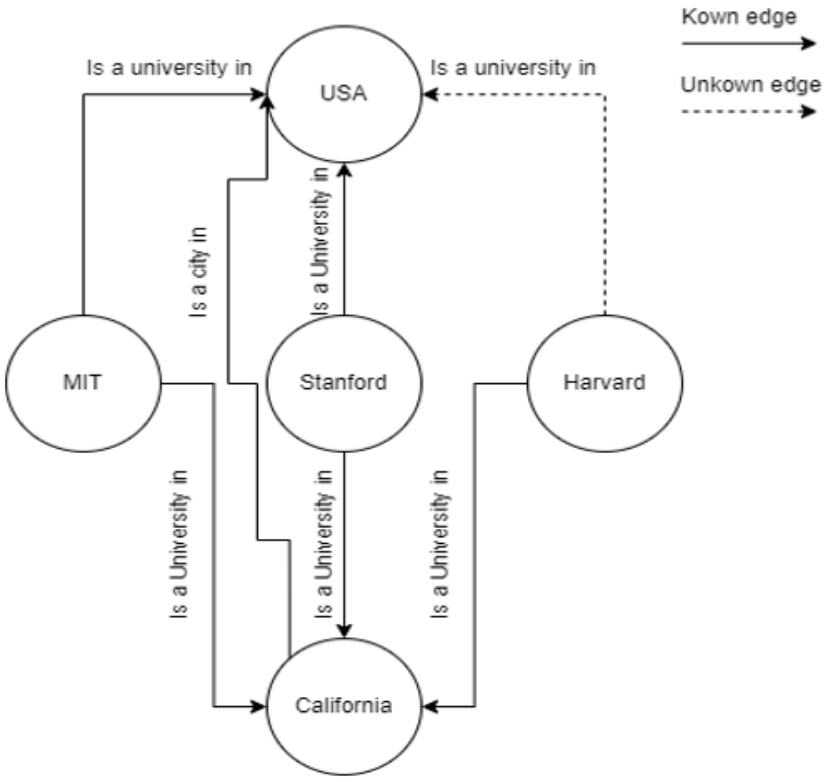

Figure 1

Sample KG where there exists a missing edge between the two nodes.

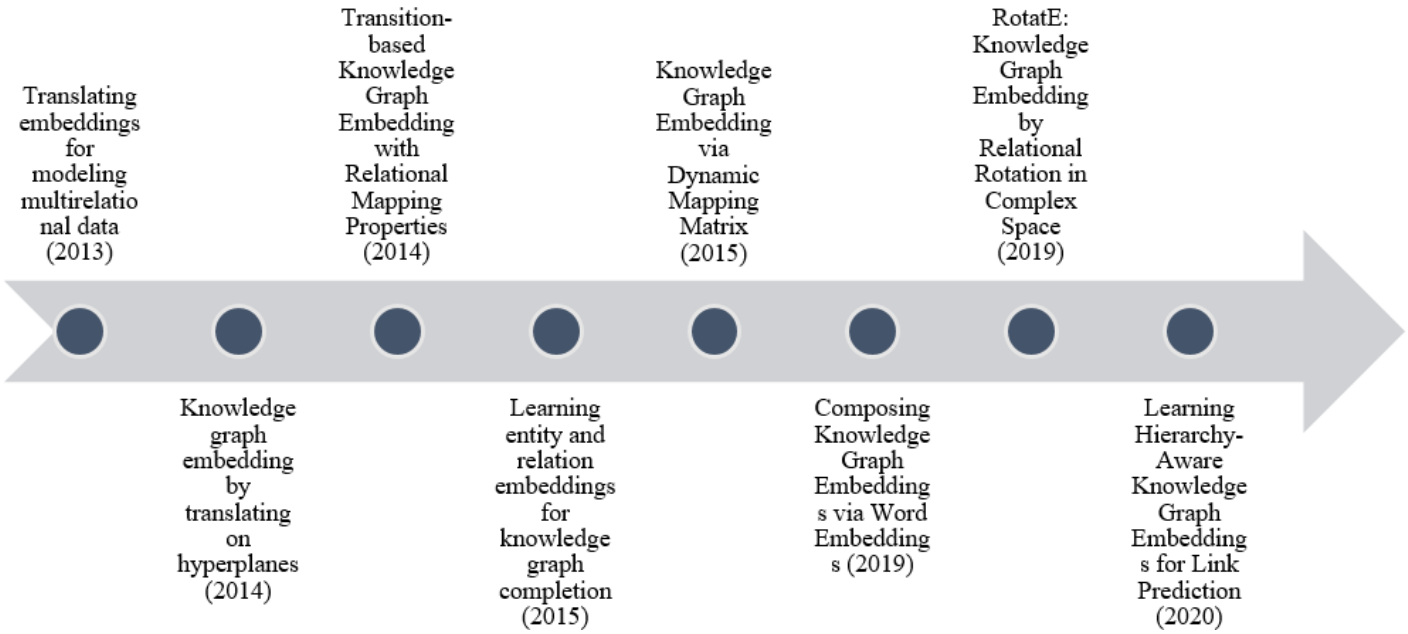

Figure 2

Translational model timeline

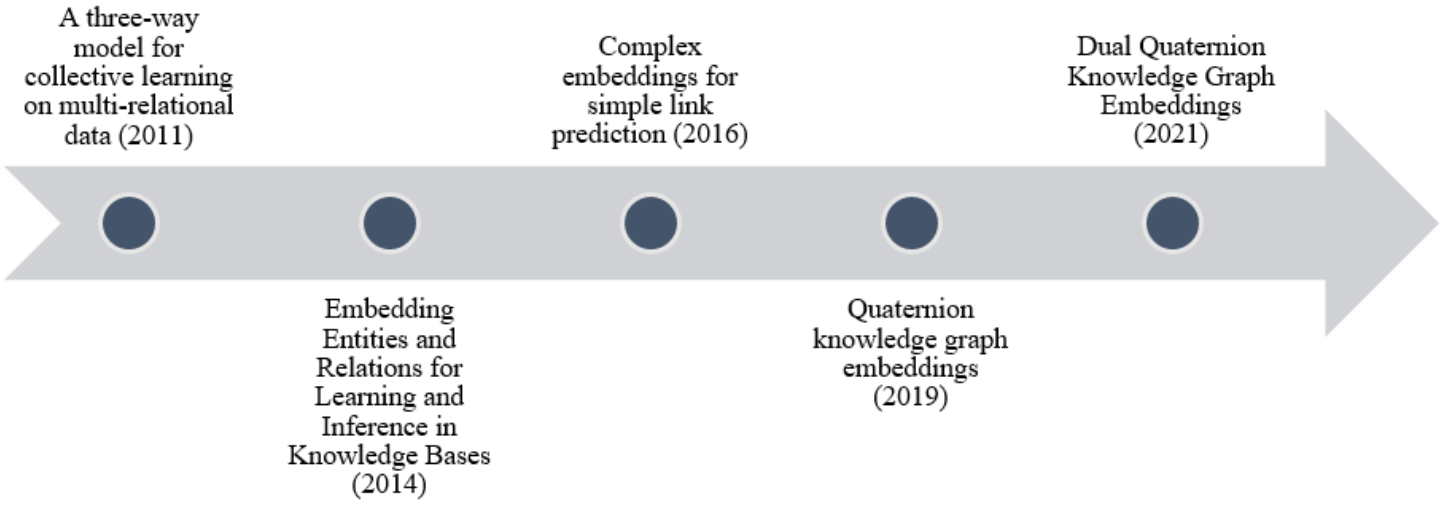

Figure 3

Tensor decompositional models timeline

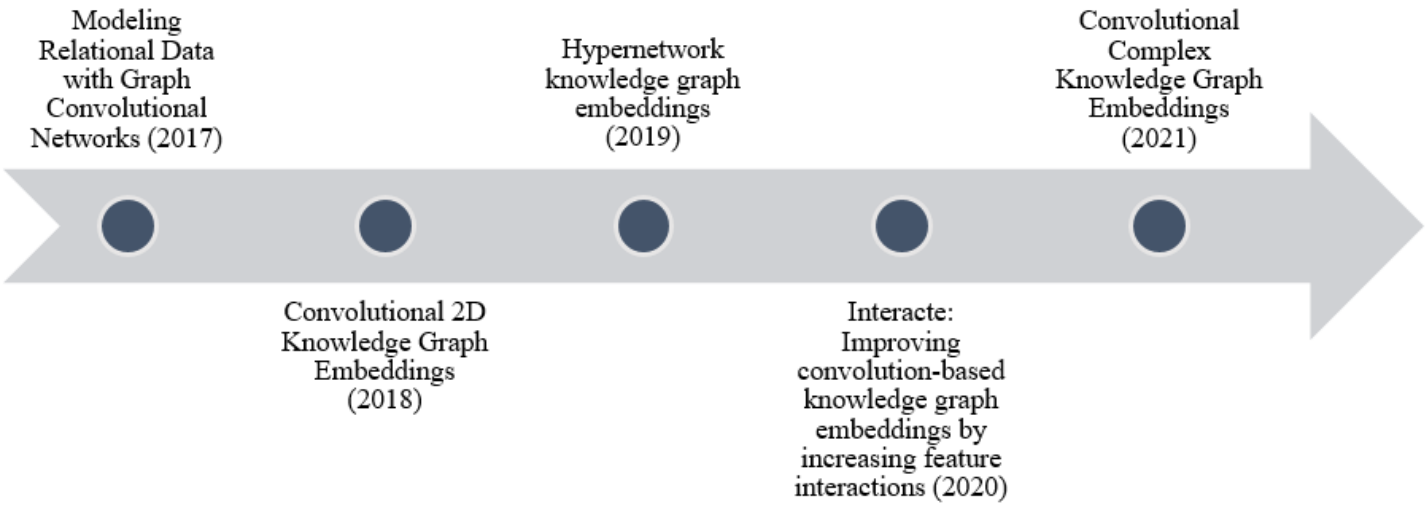

Figure 4

Convolutional based models timeline

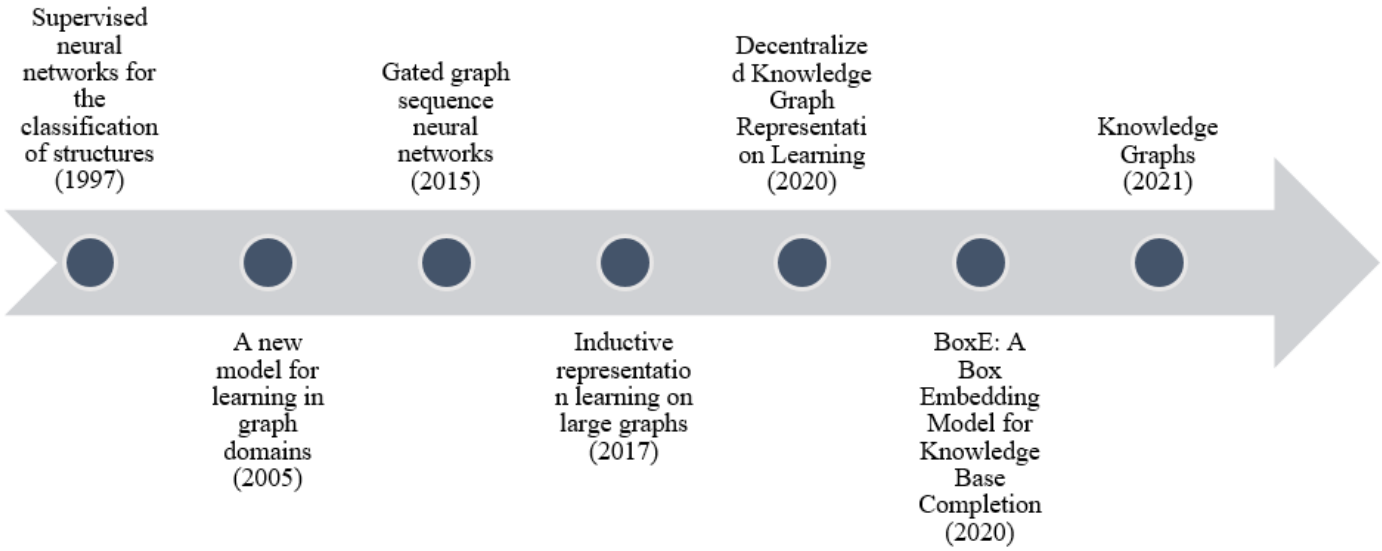

Figure 5

GNN based models timeline

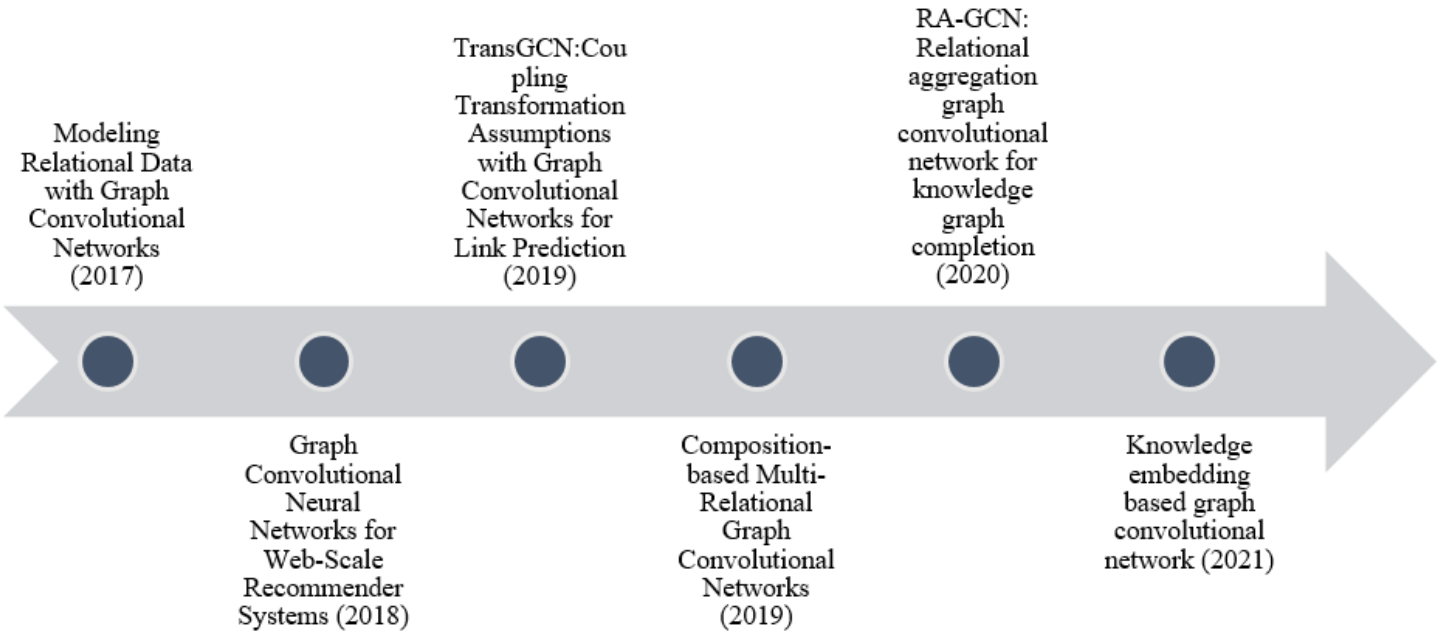

Figure 6

CNN based models timeline

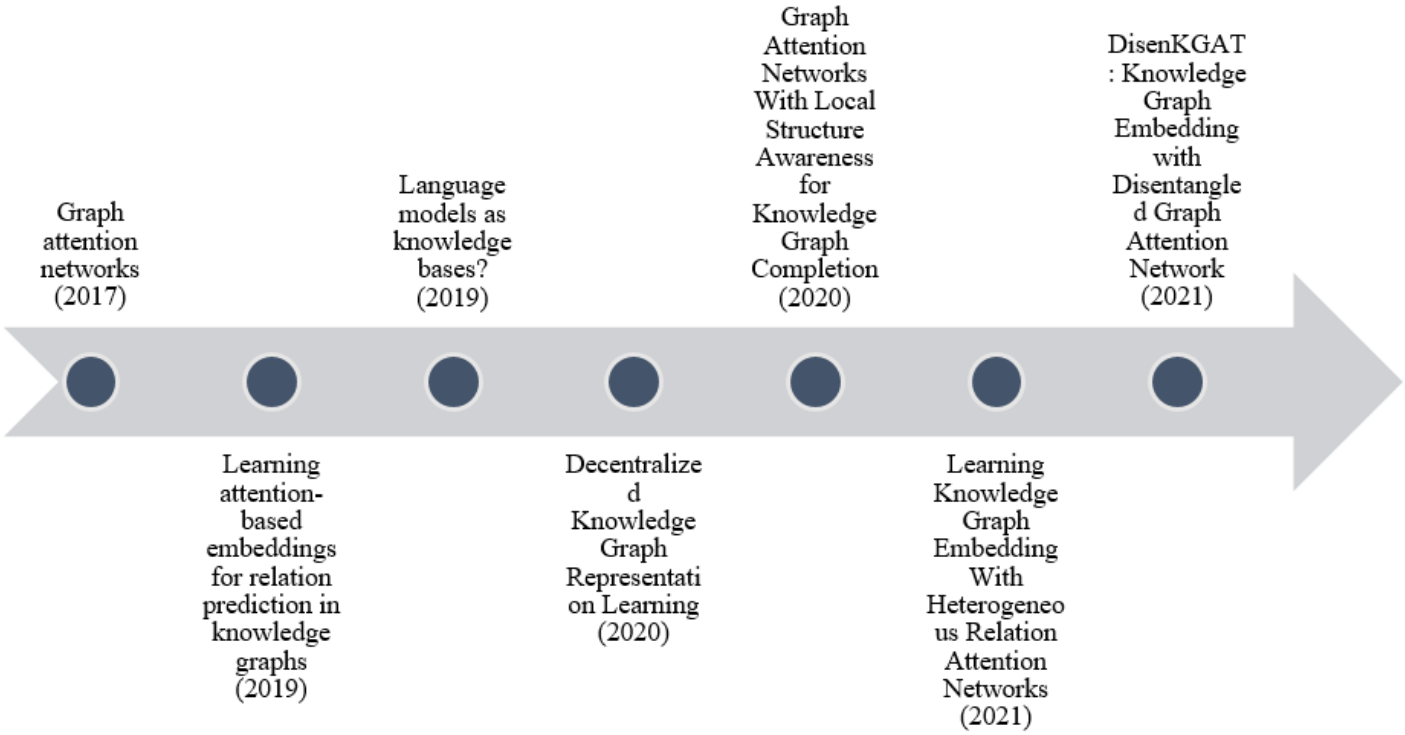

Figure 7

Attention neural network based models timeline