Table of Contents

Neural Question Answering: The role of Knowledge Bases

Sabin Kafle

University of Oregon, Eugene, OR
skafle@cs.uoregon.edu

Abstract. Question Answering (QA) requires understanding of natural language queries along with information content to obtain an answer to the question. In this survey, we focus on the question answering methods specifically based on the neural network frameworks which are the state-of-art for many QA datasets. The crux of a neural network model lies in the representation of both question and answer along with auxiliary knowledge as a continuous real valued representation, called vectors or embeddings. Powerful QA models require processing of large information content accessible from Knowledge Bases (KBs). Many KBs are readily available and involve colossal quantities of information. KBs have been successfully incorporated to neural QA by embedding the relations and entities present in KBs and then using the learned embeddings. We survey several successful applications of KBs to neural question answering problem and study the role KBs play in neural QA.

1 Introduction

Neural network-based architectures [52], also called Deep Learning, have been successfully applied to many diverse and challenging problems of artificial intelligence. The novelty of such deep learning methods lie in the generality of their framework, with applications ranging from image generation [33] to machine translation [5]. Deep learning methods are capable of generalizing and integrating inputs from several different data sources by transforming input data into a continuous real-valued representation. This makes deep learning methods suitable for multi-modal problems. Question answering systems often require integration of different information source to answer a question successfully making deep learning methods highly applicable to solving question answering problems. Automatically answering questions asked in natural languages is a complex but integral problem in artificial intelligence [17]. Several Question Answering (QA) systems exist over different domains [117], question types [85], answer types [9] and resource access ([24], [41]). A general paradigm for QA is to map a resource to the question using either information retrieval (IR) [97] or semantic parsing based query formulations [112]. In this survey, we study application of deep learning methods to question answering.

A major problem for QA systems is the lack of structured information to answer a question. Knowledge Graphs (KGs) or Knowledge Bases (KBs) are the

primary source of such structured information. KBs can be used as either a graph network, fact triples or a querying database. Several KGs are readily available with huge amount of information and facts structured within. Some widely used KBs include Freebase [10], DBPedia [4], YAGO [89], Gene Ontology [3], Wordnet [66], ConceptNet [59] and Google Knowledge Graph [84]. Semantic parsing [8] approach to question answering parse a natural language question into structured query which is executed into KBs. Such systems suffer from limitations of both Knowledge Bases and the transformation system. A major limitation of a KB is its completeness - no KB exists with all the world's information content incorporated into it.

Question Answering is a difficult problem often requiring many components to effectively answer a question. In fact, the classical solution to QA is based upon different components which handle a specific sub task, working together in a cascading manner to extract answer to a question. Figure 1 gives an example architecture to question answering problem. The issue with such cascading system is propagation of errors. Moreover, as recent performance of deep learning has demonstrated, finding features suitable to a task manually is often an erroneous, time-consuming and in many cases infeasible. The broad spectrum of question answering methods could definitely benefit with models that can work on raw data without relying on human generated features. Figure 2 shows one such example which does not require any human engineering, making the system more reliant on learning capacity of the model as opposed to hand engineering features' ability to effectively discriminate the problem space.

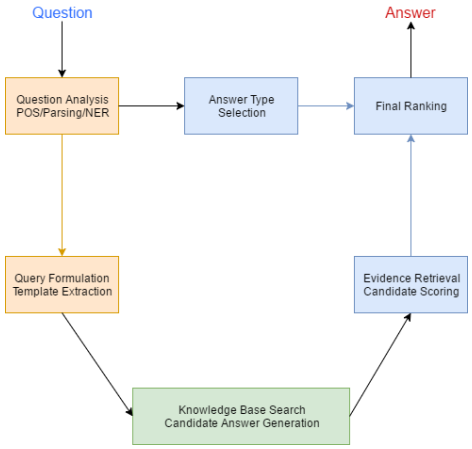

Fig. 1: A classical approach to problem of question answering [41].

Deep Learning methods are powerful framework with state-of-art results in natural language understanding [23] and information retrieval. KBs are an integral source of information for QA systems, especially factoid answers. It is

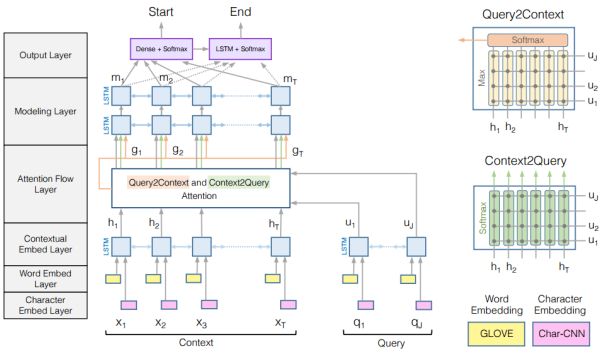

Fig. 2: A neural network based approach to question answering [81].

(almost) natural to integrate deep learning methodologies with information rich KBs in QA systems, primarily factoid QAs. However, there exists issue of representation of KBs before they are integrated to neural network architecture. KBs are represented as a triple of form $(head, relation, tail)$ with $head$ and $tail$ representing entities related via $relation$ with focus towards symbolic frameworks and languages. One example of such triple is $(car, has_part, engine)$. To solve this particular problem, researchers have devised methods to convert KB tuples into vector representation that are usable in neural networks [11]. Representation Learning of KB triples enables researcher to further enhance a KB with newer relations since embedding methods provide a functional estimate of a correct triple from an incorrect one (the problem is called triple prediction). We survey different methods to embed KBs as they form a crucial component for answering factoid questions using neural network architectures.

Deep Learning methods have gained unprecedented success in two subareas of question answering- visual question answering (VQA) and reading comprehension (RC). VQA [1] is answering questions posed in natural language about information contained within an image which is provided along with the question. RC [38] generates an answer to a question presented in natural language based on the sample of text provided with the question. Success of neural networks to these two problems could be attributed to representation learning and simplicity of multi-modal data integration by uniform representation. We study common theme in architectures for VQAs and RCs. We also focus on models which integrate further information such as KBs. KBs enable representation of world knowledge in machine understandable form and their integration to neural architecture enables the model to add additional constraints while generating or selecting an answer.

In this survey, we study neural question answering methods applied to wide range of question answering problems including factoid question answering, visual question answering and reading comprehension. We primarily explore the usability and contribution of KBs to neural question answering. While several

methods have been proposed to embed KBs, their usage is rather limited. We hypothesize that proper usage of KBs within a neural QA system should empower neural networks further. We also propose an architecture which aims toward that goal and state plausible future goals for their further usage with neural networks.

2 Deep Learning Methods

Deep Learning [52] methods are neural network architecture which enables learning of representation of data with multiple level of abstraction. These level of abstractions enable deep learning methods to generalize information while also being able to narrow down to a specific aspect of information. Deep Neural Networks, one of the most widely used deep learning model, have seen widespread application in natural language processing tasks such as parsing [76], language modeling [7], sentiment analysis ([46, 86]), factoid question answering [41], machine translation [92], visual question answering [1] among others. There are three widely popular neural network architecture for supervised deep learning - Convolutional Neural Networks (CNNs) [48], Recurrent Neural Networks (RNNs) [40] and a fully connected neural networks [80]. CNNs [51] are primarily designed towards image data with powerful property of shift invariance and their local receptors. RNNs, on the other hand, are neural networks with self connections for sequential dependencies that is very natural to textual data and time series. A fully connected neural networks are mostly useful for representation learning task [65] which requires shallow network for faster data processing. The common features of all deep learning methods is their gradient based optimization which is done by either backpropagation or backpropagation through time (BPTT) for recurrent neural networks.

2.1 Convolutional Neural Networks

A Convolutional Neural Network is a special form of neural network that is characterized by a convolution layer often followed by a sampling layer to lower parameters count. The structure of CNN is designed to take advantage of 2D structure of an input image which is achieved with local connections and tied weights followed by some form of pooling which results in translation invariant features. A general CNN architecture consists of convolutional and subsampling layers optionally followed by fully connected layers. A convolutional layers has filters, each of size smaller than original image, giving rise to locally connected structure which are each convolved with image to produce feature maps. This feature map is subsampled by pooling layer. A sequence of convolutional and pooling layers is usually followed by fully-connected layers for softmax computation, which is used for output prediction. CNNs are trained by gradient-based error propagation methods.

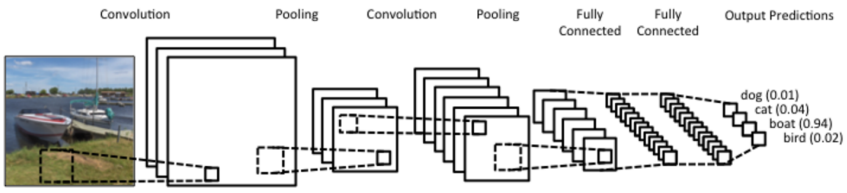

Fig. 3: A simple convolutional neural network (CNN) architecture for images. [18]

CNNs applied to text Convolutional Neural Network (CNNs) are developed primarily for image representation but their ability to represent image data has led to their usage in text data as well. The primary different between a CNN applied to text and image lies in the filter size. For text data, the width of a filter is always kept constant with the embedding size of incoming text data. This is generally followed by max-pooling over each feature map (output of convolution operation) to create a concatenated vector. Figure 4 depicts a CNN architecture applied to text data. While CNNs applied to text do not have a good structural justification evident upon image data, CNN architectures are faster to execute and produce competitive results in text data when compared to other deep learning architectures [46] . They are also extensively used in character-aware language models [47] which performs representation learning for characters and generates words and documents based on likelihood function of the model.

2.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are another special characterization of neural networks which makes use of sequential information present in input data as a form for recurring unit. They are called *recurrent* because they perform the same task for every element of a sequence, with the output being dependent on previous computations. A RNN needs unfolding in time of the computation as depicted in Figure 5. RNNs are very natural and intuitive to text data which are sequential by nature and thus provide a very powerful generative model. Simple RNNs though suffer from problem of vanishing [39] and exploding gradients [74] thus making their theoretical and modelling suitability to text useless. Different variants of RNN called Long short-term Memory (LSTM) [40] and Gated Recurrent Units (GRUs) [20] are selected as cell of RNNs for modeling sequential data.

RNN-based architecture are popular for language modeling problem [63]. The usability of RNN is enhanced by a variation of RNN-architecture called sequence-to-sequence models [92] where both inputs and outputs are sequential data e.g.; machine translation. Seq2Seq models (Figure 6), as they are widely called, are foundation of architecture for most of the more complex problems

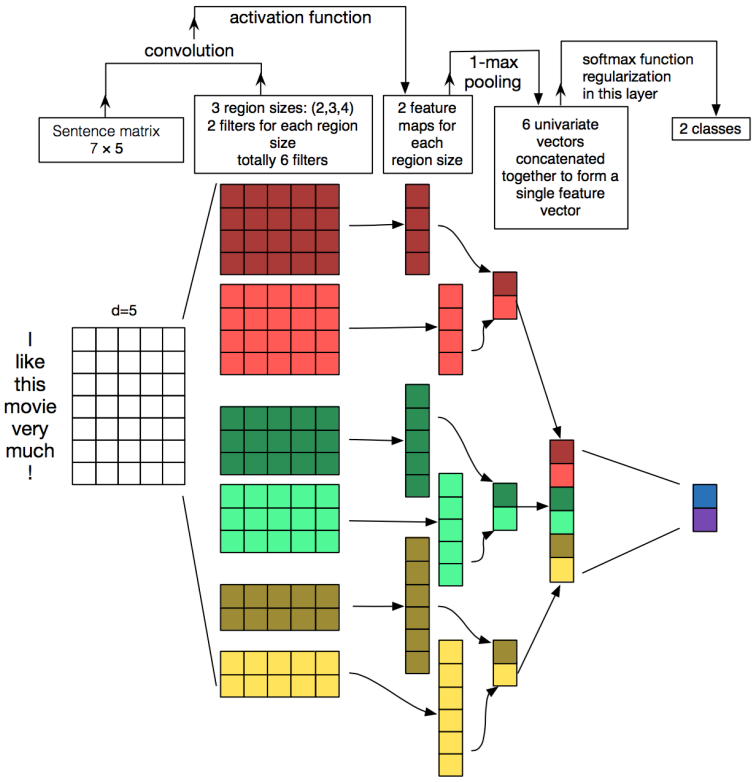

Fig. 4: A simple convolutional neural network (CNN) architecture applied to text data. The filter width is fixed for text and feature maps are of variable size. [18]

in machine learning which includes machine translations [62, 5], visual question answering [1], reading comprehension [38], and caption generation [105]. This formulation consists of an encoder - which reads input sequence and converts it into a fixed vector representation and a decoder - which uses the fixed vector representation to regenerate the output sequence.

2.3 Neural Networks for Embedding

A neural network used for learning embeddings (also called vector representations) of input data is a feedforward neural network with a single hidden layer [65] with non-linearities being absent. Such simple architectures are selected to process vast amount of resources (billions of words). The most popular architecture for learning embeddings is skip-gram model (Figure 7), whose training objective is to learn word vector representations that are good at predicting nearby words. Another model is continuous bag-of-words (CBOW) (Figure 8)

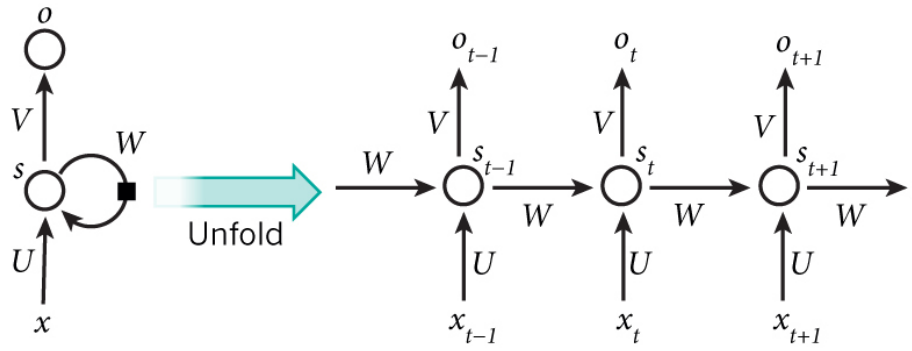

Fig. 5: A recurrent neural network unfolding in time of computation during forward propagation. [52]

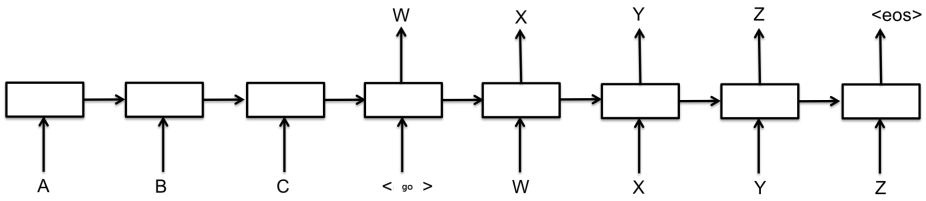

Fig. 6: A basic sequence to sequence architecture with an encoder and decoder component. Each square box represents a RNN cell, e.g.; LSTM [92]

which uses context word to predict target word [64]. Pennington et al. [75] generated word vectors using count based models.

The reason for learning embeddings for words or lower level structures (images when dealing with video processing) is to convert every entity into a vector space such that words which are used in common contexts are closer to each other in that space. Embeddings adds external context knowledge to machine learning models which cannot be obtained from other representations such as bag-of-words. Another nice property of word embeddings lies in the geometric properties such as $vector(king) - vector(male) + vector(female) \sim vector(queen)$ which is a highly desirable common-sense knowledge to the models as input.

2.4 Neural Networks with Attention

An extension of sequence-to-sequence neural networks is neural network with attention, in which the decoder attends to hidden state of encoder at each encoding step of incoming sequence. The decoder "attends" to different part of source sequence at each step of output generation. This process is learned by the model itself i.t. it learns which part to attend to when decoding. The current attention mechanism is a matrix $\alpha \in R^{T_T \times T_S}$ where T_S is source sequence

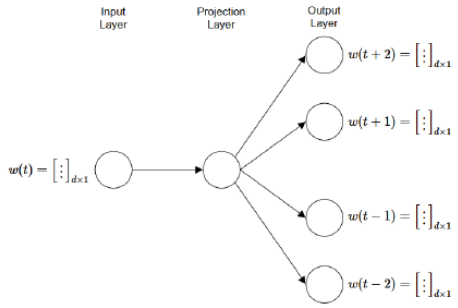

Fig. 7: A skip-gram model for learning word embeddings. [65]

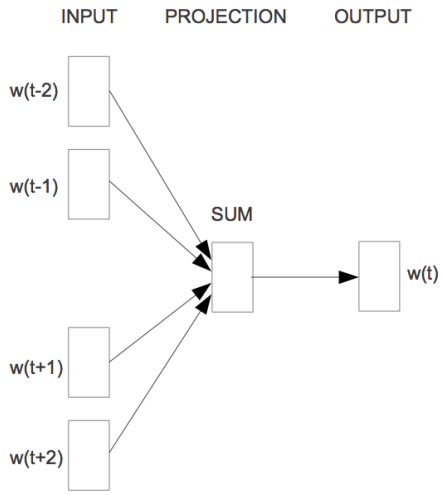

Fig. 8: A continuous bag-of-words model for learning word embeddings. [64]

length and T_T is target sequence length. It makes the decoded output y_t of a neural network dependent on weighted combination of all the input states based on attention weights, which is the normalized row vector $\alpha_{(t,:)}$. The approach for attention was popularized by its application to neural machine translation in [5]. A basic attention mechanism is illustrated in Figure 9.

Attention mechanism are also applied to question answering problem in which attention weights help the model focus on specific segment of text during many different tasks such as question embedding (some parts of question are more important than rest) [24] and text segments during reading comprehension [38].

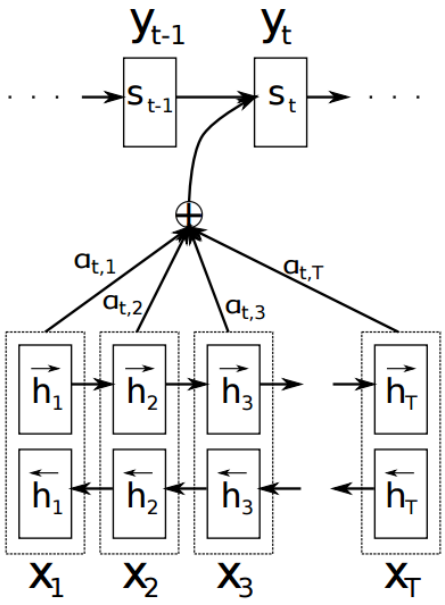

Fig. 9: An attention mechanism trying to generate t^{th} target word y_t given a source sentence $(x_1, x_2, ..., x_T)$. $a_{t,x}$ is the weight input encoded representation h_x has on generating y_t. A bidirectional RNN is in usage. [5]

2.5 Memory Networks

Memory Networks [102] and related NN architectures including Neural Turning Machine [34, 90] are neural networks with external memory. They are extremely useful paradigm for solving factoid question answering [16] and question answering involving reasoning [102]. Their novelty lies in their ability to manipulate external memory locations, such as a Knowledge Base (KB) or a Universal Schema [79].

An Input module is used to learn representations for the values to be stored in memory, which are then used to compute output representation and final predictions. The different mechanisms within a Memory Network could be different e.g. sentence representation can be either an averaging over word embedding or a position-encoded representation. The pluggable application and flexibility of the network makes Memory Networks state-of-art for factoid question answering [16]. Another advantage lies in different level of guidances applied i.e. additional information incorporation is easier than standard NN architecture [102, 114]. An end-to-end memory network (being end-to-end is not required of a memory network) is shown in Figure 10.

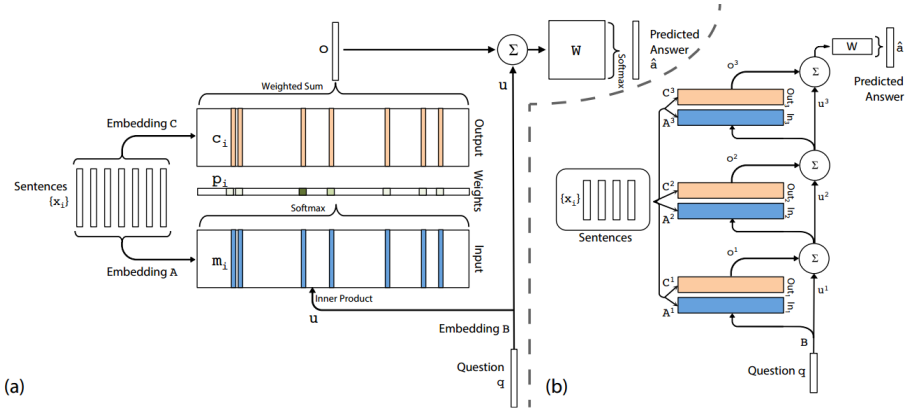

Fig. 10: An end-to-end memory network architecture in which sentences embedding are stored as memory values and multiple layers (one in left and three in right) are used to compute representation for final answer. [90]

2.6 Multi-modal Networks

An interesting application of neural networks is their easy integration to multi-modal problems, such as Visual Question Answering. Figure 11 gives an example architecture for multi-modal data problem. All data sources are converted into continuous vector representations which can then be integrated with each other to perform final joint inference. Fukui et al. [29] give an efficient algorithm for computing compact bilinear pooling over multi-modal data based on Fast Fourier Transform.

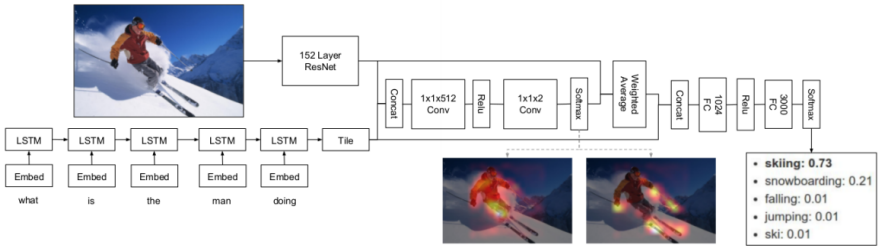

Fig. 11: A multi-modal neural network for visual question answering [45].

3 Knowledge Base

Knowledge Base (KB) is structured database with schema such as an ontology describing entities, relations and attributes which form the foundation of

structural information. Facts are then added to the KB in accordance with the structure, forming the entirety of a KB. A KB can also be represented as a graphs of facts, with entities representing the nodes of graph and relationship between entities being described by edges. For our survey, we treat Knowledge Base and Knowledge Graph as same entity. The reason for such treatment is potential transformation of KB into graphs. Also, the existing neural network literature does not draw any distinction between two terms and use them interchangeably.

Knowledge Graphs are typically stored as directed graphs of multi-relational data whose nodes correspond to entities and edges correspond to relations between them. KBs are represented as a triplet of form (h, l, t) or *(head, label, tail)* which indicates there exists a relationship of name *label* between the entities *head* and *tail*. There are numbers of Knowledge Base (KBs) in development and usage such as Freebase [10], DBPedia [4], YAGO [89], Gene Ontology [3], Wordnet [66], ConceptNet [59] and Google Knowledge Graph [84]. Such large KBs are constructed for usage as a querying system. Most Knowledge Graphs are written in formats (e.g. OWL [2]) which makes them accessible via query languages like SPARQL [77]. This itself is a significant research area and contributes to reasoner system such as HermiT [82] which can be used to generate an answer from such large knowledge graphs based on SPARQL query formulation.

The most widely used Knowledge Base is Freebase [10]. It is a structured KB in which entities are connected by predefined predicates or relations. All predicates are directional, connecting from subject to object. A triple (subject, predicate, object) denoted by (h, p, t) describes a fact; e.g, (US Route2, major cities, Kalispell) refers to the fact that US Route 2 runs through the city of Kalispell. Another KB, Reverb [26] is extracted automatically from text with minimal human intervention and is highly unstructured: entities are unique string and the lexicon for relationship is open. This leads to many more relationships, but entities with multiple references are not deduplicated, ambiguous referents are not resolved, and the reliability of the stored facts is much lower than in Freebase.

The usage of knowledge graphs is limited by two issues - completeness [87, 101] and compatibility. The issue of completeness arise from the fact that no KBs can be ever exhaustively completed. There are initiatives to keep on building KBs based on continuous stream of texts such as Never Ending Language Learner (NELL) [19] but even such learners are not expected to exhaustively cover all worlds knowledge. This inadequacy can lead to error in query based system which completely rely on KBs. Another challenge in usage of KB lies in its compatibility. Each KB has their own design decisions and thus even for same concepts and relations, different naming conventions are preferred which presents a challenge in applying more than one KBs to a problem. Application of more than one KB could potentially decrease the incompleteness of KBs [12]. A common solution is preferred to both problems - embedding of knowledge bases.

KB embedding is a challenging problem. The main issue is the computational complexity since most of the modern KBs tend to be huge with several millions entities which necessitates the usage of simpler model for learning embeddings while at the same time requiring nice embeddings with properties similar to

word2vec models. An initial step as part of KG embeddings is to flatten hugely hierarchical structure of such graphs into a triple format. This leads to loss of direct hierarchical relationship but limits the possible number of triples from being infeasible for learning embeddings. Such hierarchical representations are expected to be intrinsically represented within the embedded vectors. In this survey, we focus on embedding methods that are based on neural networks. There are several tensor factorization methods for relational learning that generate embeddings for KBs such as Nickel et al. [71], Nickel and Tresp [70], Nickel et al. [72]. Bayesian Clustering methods have also been successfully applied to embed a KB [91]. The objective function for embedding knowledge base is associated to link and triple predictions using a form of margin-loss function.

3.1 Embedding Models

Embedding models for KB completion associate entities and/or relations with dense feature vectors or matrices [11]. Such models obtain vectors which are able to produce state-of-art performance in tasks such as link prediction or triple predictions [12, 99, 35, 68, 69]. Such vectors can have interesting generalization properties similar to word embeddings where geometrical distance holds some semantic interpretation [49]. KB embedding models are primarily evaluated on two standard tasks - link prediction [12] and triple classification [87].

3.2 General Framework

For E entities and R relations where G denotes the knowledge graph consisting of a set of triples (h, r, t) such that $h, t \in E$ and $r \in R$. The embedding model defines a score function $f(h, r, t)$ for each triple, which is the score of its implausibility. The objective of embedding models is to choose f such that score of a plausible triple (h, r, t) is smaller than score of an implausible one (h', r', t'). The model parameters are learned by minimizing a margin-based objective function:

$$L = \sum_{(h,r,t)\in G,(h',r,t')\in G'} [\gamma + f(h, r, t) - f(h', r, t')]_+ \tag{1}$$

where $[x]_+ = max(0, x)$, γ is the margin hyper-parameter and

$$G' = \{(h', r, t)|h' \in E, (h', r, t) \notin G\} \cup \{(h, r, t')|t' \in E, (h, r, t') \notin G\} \tag{2}$$

is the set of incorrect triples generated by corrupting the correct triple $(h, r, t) \in G$. Table 1 shows scoring function for some successful neural-embedding models.

Table 1: **The scoring function of neural-embedding models for Knowledge Bases. The entities h and t are represented by vectors $v_h, v_t \in R^k$.**

Model	**Score Function $f(h,r,t)$**
Structured Embedding (SE) [11]	$\|W_{r,1}v_h - W_{r,2}v_t\|_{l_{1/2}}$ $W_{r,1}, W_{r,2} in R^{k\times k}$
SME [14]	$(W_{1,1}v_h + W_{1,2}v_r + b_1)^T(W_{2,1}v_t + W_{2,2}v_r + b_2)$ $b_1, b_2 \in R^n; W_{1,1}, W_{1,2}, W_{2,1}, W_{2,2} \in R^{n\times k}$
TransE [12]	$\|v_h + v_r - v_t\|_{l_{1/2}}; v_r \in R^k$
TransH [99]	$\|(I - r_p r_p^T)v_h + v_r - (I - r_p r_p^T)v_t\|_{l_{1/2}}$ $r_p, v_r \in R^k;$ $I: \quad Identity \quad Matrix \quad k \times k$
TransR [58]	$\|W_r v_h + v_r - W_r v_t\|_{l_{1/2}}$ $W_r \in R^{n\times k}; v_r \in R^n$
TransD [43]	$\|(I + r_p h_p^T)v_h + v_r - (I + r_p t_p^T)v_t\|_{l_{1/2}}$ $r_p, v_r \in R^n; h_p, t_p \in R^k;$ $I: \quad Identity \quad Matrix \quad n \times k$

Model	**Score Function** $f(h,r,t)$
lppTransD [116]	$\|(I + r_{p,1}h_p^T)v_h + v_r - (I + r_{p,2}t_p^T)v_t\|_{l_{1/2}}$ $r_{p,1}, r_{p,2}, v_r \in R^n; h_p, t_p \in R^k;$ $I: \quad Identity \quad Matrix \quad n \times k$
TranSparse [44]	$\|W_r^h \theta_r^h v_h + v_r - W_r^t \theta_r^t v_t\|_{l_{1/2}}$ $W_r^h, w_r^t \in R^{n \times k}; \theta_r^h, \theta_r^t \in R; v_r \in R^n$
STransE [69]	$\|W_{r,1} v_h + v_r - W_{r,2} v_t\|_{l_{1/2}}$ $W_{r,1}, W_{r,2} \in R^{k \times k}; v_r \in R^k$
DistMult [108]	$v_h^T W_r v_t; W_r \quad is \quad a \quad diagonal \quad matrix \quad \in R^{k \times k}$
NTN [87]	$v_r^T \tanh(v_h^T M_r v_t + W_{r,1} v_h + W_{r,2} v_t + b_r)$ $v_r, b_r \in R^n; M_r \in R^{k \times k \times n}; W_{r,1}, W_{r,2} \in R^{k \times k}$
HolE [73]	$\sigma(v_r^T (v_h \circ v_t))$ $v_r \in R^k, \circ \quad denotes \quad circular \quad correlation$

Model	**Score Function** $f(h,r,t)$
Bilinear-COMP [35]	$v_h^T W_{r1} W_{r2} ... W_{rm} v_t$ $W_{r1}, w_{r2}, .., W_{rm} \in R^{k \times k}$
TransE-COMP [35]	$\|v_h + v_{r1} + v_{r2} + ... + v_{rm} - v_t\|_{l_{1/2}}$ $v_{r1}, .., v_{rm} \in R^k$
Unstructured [14]	$\|v_h - v_l\|_{l_{1/2}}$

We describe some significant models in greater detail.

3.3 Translation-based Methods

Translation-based embedding methods learn embeddings for entities and relations based on the translation operation over head and tail entity using relation entity. The simplest of such model is Unstructured Model [14] which assumes head and tail vector are similar and does not take relationship into account. The Structured Embedding (SE) model [11] transforms head and tail entities into relation subspace using two distinct transformation matrices. The transformed head and tail entities are expected to be near one another in optimal subspace. The Semantic Matching Energy (SME) model learns triplet as a form a bigram function (using matrix multiplication) and as a trigram model (using tensor multiplication). The bilinear tensor based projection matrix obtains better performance on test task of link prediction confirming more complex model leads to better performance.

The pioneer work in translation-based embedding model is TransE [12]. It assumes all relations and entities can be represented by vectors of uniform size. The objective of TranE expects the head entity displaced by relation vector to be closer to the tail entity. The approach is noted for simplistic design, and flexible architecture leading to multiple improvements being made on top of TransE. One main problem with TransE model lies in its inability to differentiate between different relation mappings such as*one-to-one, many-to-one, one-to-many* which makes the model unsuitable for representing such relations.

TransH [99] treats each relation to be a different plane and modified TranE such that head and tail entities are projected to relation plane before being checked for displacement operation. This makes the method more flexible in handling various relation mappings since different relations would be projected into different planes, depending on its type. Figure 12 shows the geometrical contrast between TransH and TransE. Another translation method, TransD [43] considers diversity of both entity and relation. An entity is represented by two vectors with one vector being used to project the entity into relation subspace while the other calculates the translation with respect to other entity in triple. TransD primarily aims to cover different *type* associated with the entities. A similar idea of handling relations differently is followed in Lin et al. [57] where TransR treats a relation as a matrix which are then used to project entities to relation subspace for translation measurement. Another approach CTransR further considers multiple types of representation within a relation by clustering head and tail entities covered by the relation and learning a unique embedding for each cluster. The problem is constrained to ensure the learned representation does not stray too far from original relation matrix.

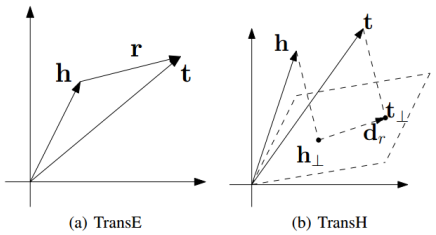

Fig. 12: The difference between TransE [12] and TransH [99] in geometrical space.

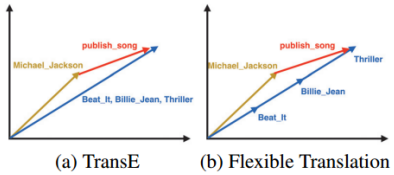

Fig. 13: The difference between TransE [12] and flexible Translation [28] which aims to ensure the sum of head and relation vector points in the same direction as tail vector.

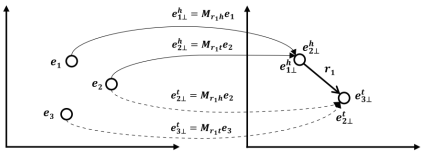

Fig. 14: A transitive property preservation transformation using role-based projection for TransE-lpp method [116]

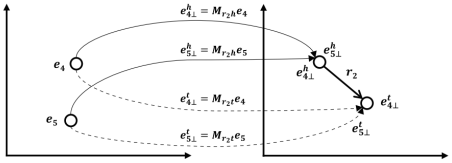

Fig. 15: A symmetric property preservation transformation using role-based projection for TransE-lpp method [116]

A flexible translation approach is proposed in Feng et al. [28] which uses the motivating example given in Figure 13 as a basis for reducing the constraint such that tail entity should only be in the same direction as the displacement of head and relation entity. This leads to tail entity being a plane as opposed to vector and hence a constraint is placed on possible plane for both head and tail entity during training. A translation-based method that preserves the transitivity and symmetricity of relations is given in Yoon et al. [116] where head and tails are mapped differently based on their position. This model is incremental in nature and can be integrated with any of the translation based approach as an additional constraint. The role based projection is illustrated in Figure 15 and Figure 14. STransE [69] follows the similar approach of projecting head and tail entity using separate projection matrices. The relation is represented using two matrices and a vector, making the model flexible to handle type differences within both relations and entities. Ji et al. [44] propose TransSparse with an assumption that some relations can be represented with fewer number of parameters as opposed to other relations. The embedding is thus a function of sparseness which is made adaptive and function of number of entities linked by the relation, while the rest of computation is similar to STransE giving greater flexibility for representing entities.

3.4 Tensor-based Methods

The neural tensor model [87] is one of the first work to use bilinear tensor operator to representation each relation as shown in Figure 16. Yang et al. [108] use a bilinear score function without any non-linearity to learn embeddings. Bilinear models are popular with matrix factorization based embedding methods [25]. Quadratic forms are used to model entities and relations in [37, 104, 95, 31,

93]. Such methods are similar due to the three-way interactions between relation, head and tail entities during the score computation. ProjE [83] use diagonal matrix and linear interaction to combine entity and relation similar to translation based methods, but sigmoid and tanh activation are used when projecting to a score metric. The objective is to learn by either pointwise learning or listwise learning based upon softmax regression function. The architecture by means of example for ProjE is at Figure 17. Nickel et al. [73] uses a circular correlation operation while learning embedding which can be interpreted as compression of tensor product.

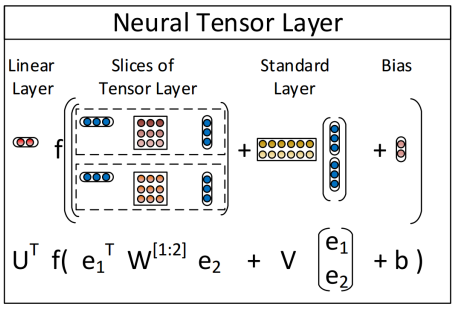

Fig. 16: A Neural Tensor Network Layer [87]

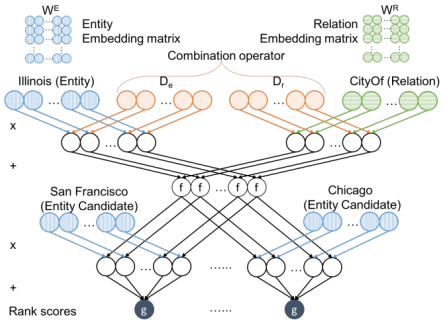

Fig. 17: An embedding projection for KG completion given an input example *(?, CityOf, Illinois)* [83]

3.5 Relation Path-based methods

More recent developments for embedding have shown that the relation path between entities in Knowledge Graph provide a richer context information which enables learning more structured embeddings [61, 30, 35, 54, 56, 94, 68]. A context dependent path based pre-training similar to Word2Vec [64] is proposed by Luo et al. [61]. The method first learns the embedding by obtaining context using connection between entities. The learned embeddings are then fine-tuned by using one of the translation based objective function. Guu et al. [35] use path queries to obtain a relational transformation which is then integrated into a translation model, such as TransE. The authors define some translation based methods as composable which is essentially decomposable into a head entity-relation pair without any dependency on tail entity. The paths are then obtained by randomly traversing through the knowledge graph using random walk. Lin et al. [56] extend the TransE method by additional objective of learning scoring from a different relation path representation, which is a summation over all relation paths that are termed reliable. The reliability is based upon path-constraint resource allocation algorithm which is recursive function weighted over the number of incoming edges to a node. The path representation is calculated by multiple operation such as addition, multiplication and RNNs as illustrated in Figure 18. Such path-based methods often suffer from problem of computing possible paths between entities exhaustively which can be computationally challenging. Toutanova et al. [94] propose a dynamic algorithm to enable efficient incorporation of relation paths of bounded length in compositional path models. Neelakantan et al. [67] propose a KB completion method using RNNs that are able to infer multi-hop relationships. Their approach is able to take vector embedding of relationships at each time step, the final output is the inference of the relationship between the first and last entity on the path. Such methods require large training corpus and the authors use 52M relational triples for training their model.

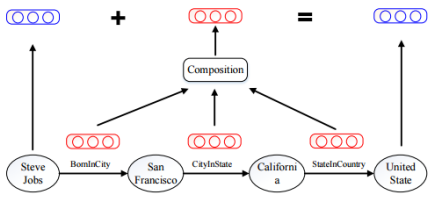

Fig. 18: A path representation computation for [56]

In addition to these classes of embedding methods for knowledge bases, some work aims at learning knowledge base embedding using external resources. Such methods enable the model to handle knowledge graph sparseness since the entity annotated corpus such as Wikipedia can be used for learning embedding

for knowledge base like Freebase. Wang and Li [100] use external text corpus to pointwise textual context and pairwise textual context, which describe the amount of co-occurrence between entities and words. Embeddings for knowledge base are learned using the margin based translation objective function augmented with averaged embedding representations obtained from co-occurrence. Such methods are able to leverage large about of additional resources. The evaluation criteria for Knowledge Graph embeddings are either link prediction or triple predictions with Xiao et al. [104] obtaining best result in Wordnet and Ji et al. [43] getting better performance in Freebase dataset for link prediction task.

4 Factoid Question Answering

Factoid Question Answering (FQA) refers to questions which can be answered effectively by a phrase or an entity of a Knowledge Graph. There are mainly two paradigms of FQAs - answering questions over a KG or obtaining answer from natural text using open factoid question answering. Few approaches exist which attempt to combine both resources or use multiple Knowledge Base. The problem of obtaining answers as a phrase from the text provided with question is a different challenge, called reading comprehension [38] or machine comprehension and follows different methodology to factoid question answering.

4.1 FQA from Knowledge Graphs

A Knowledge Graph-based factoid question answering involves mapping the question in natural language into triples of Knowledge Graphs. The distinction is made between FQA systems mapping to just one triples and mapping to multiple triples. The system which maps to a single triple is called Simple Question Answering (SimpleQA). Simple QA is a relatively easy problem compared to other factoid and non-factoid QAs. They are also teh most frequent type of questions asked [27]. A SimpleQA task involves answering a question such as *"What is the hometown of Obama?"* which asks for a direct topic of an entity *"Obama"* which is *"hometown"*. The challenges to SimpleQA systems lie in possibility to formulate a question in multiple of ways, making the mapping process hard to generalize.

A highly successful paradigm to factoid question answering involves converting the natural text into structured queries, which is then fed into the Knowledge Base systems to obtain answer, called semantic parsing [8, 112, 111]. The semantic parsers learn to understand natural language questions by converting them into logical forms. Semantic Parsing is highly successful in solving problems of factoid question answering such as those involving multiple relations and questions whose answers involve a list of ordering which is difficult to answer using look-up based FQA methods. There are few examples of neural network based semantic parsing methods [55] which use a data-based approach to convert natural text into formal query but they are still highly regulated. We skip such methods

for our study since they are not a neural network-based approach though it is useful to keep in mind that such methods exists and offer a powerful alternative to methods presented in this survey.

Simple Question Answering (SimpleQA) A common approach to solving SimpleQA problem is to extract a set of candidate answers from Knowledge Base using relation extraction [111, 112, 110, 6] or distributed representation [13, 24, 107]. Fader et al. [27] introduce WikiAnswers, a paraphrasing dataset which helps generalize for unseen words and question patterns. Another dataset SimpleQuestions is introduced by Bordes et al. [16]. SimpleQA involves embedding of knowledge base to find entity of knowledge base that is closest to the question's representation as the answer. The general framework for factoid question answering is: Given an input question sentence $S = \{w_1, w_2, .., w_Q\}$ and a sentence representation $s \in R^k$, we find the entity e in KB E such that $f(s, e) > f(s, e'), e' \cup e = E$.

A CNN based approach to answering Simple Questions is proposed by Yin et al. [115] with two-step pipeline - entity linking and fact selection. An active and passive entity linkers are used, with passive entity linking done via subsequence matching between question phrase and entities and knowledge bases and an active entity linking obtained through a BiLSTM-CRF [21] model to detect entity mentions. A sequentially labeled dataset is required for training an active entity linker. An active entity linking is followed by passive entity linking to obtain more candidates. A fact pool is obtained from facts containing candidate entities. For fact selection, two CNNs are proposed - a character CNN to match over KB entity and its mention in the question surface form; so that the generated representation is more robust even in presence of typos, spaces and other character violation. A word-level CNN with attentive max-pooling learns the match of the KB predicate with the pattern in question. Char-CNN and Word-CNN decompose each question fact into an entity mention surface-form match and a predicate-pattern semantic match.

Bordes et al. [16] perform simple question answering using Memory Networks. The memory network consists of a memory and a neural network that is trained to query it given some inputs. It consists of four components - Input map (I), Output map (O), Generalization (G) and Response (R). The workflow is to store Freebase into memory and then train MemNN to answer question. The object-triplet in Freebase are grouped and mediator nodes are removed by creating a single relationship. The grouping mechanism is to allow multiple objects to be linked to same subject and relationship, which is called as a KB-triplet. A KB triplet is represented in bag-of-words model with subject and relationship having value 1 and object entries set to $1/k$ where k is number of objects. The questions are mapped to bag-of-ngrams. The output module performs the memory lookups given the input to return the supporting facts destined to eventually provide the answer given a question. A candidate generation phase proceeds the output module operation. The scoring is obtained after performing cosine similarity on embedded representations where both the question and triple is summed

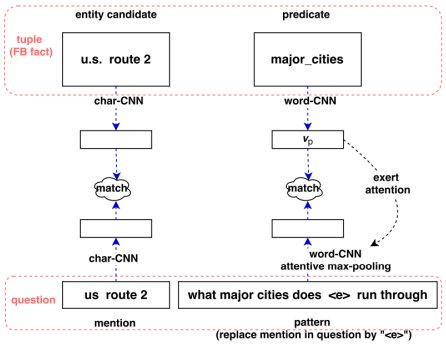

Fig. 19: A CNN-based SimpleQA system using char- and word-RNN for semantic and surface matching [115].

representation of underlying entities. The response module returns the set of selected supporting fact.

Lukovnikov et al. [60] follow neural network based approach to answer simple questions using KB (Figure 20). Question is encoded using word level GRUs and word is represented as concatenation of Glove vector with character level encoding. Subject are represented as concatenation of entity labels (char level) and type labels (word level). Char-level NNs are used to mitigate Out of vocabulary issues. Predicates are embedded using encoding GRUs initialized with word representations. The task is then to obtain the subject and predicate from question using scoring function. The caveat of model lies in candidate generation which is a simple substring match.

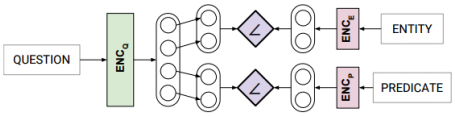

Fig. 20: Question Answering architecture for [60]

Golub and He [32] propose a character-level approach (Figure 21) based on the attention-enhanced encoder-decoder architecture [5]. The model of [32] consists of a character-level RNN-based question encoder and an attention-enhanced RNN decoder coupled with two separate character-level CNN-based entity label and predicate URI encoders. A pairwise semantic relevance function is used to measure the similarity between hidden states of the attention decoder and the embedding of an entity or predicate candidate. A cosine similarity measure is used to compute the likelihood of entity or relation being answer to the question.

The model is learned by maximizing the joint likelihood of entity and predicate candidate matching.

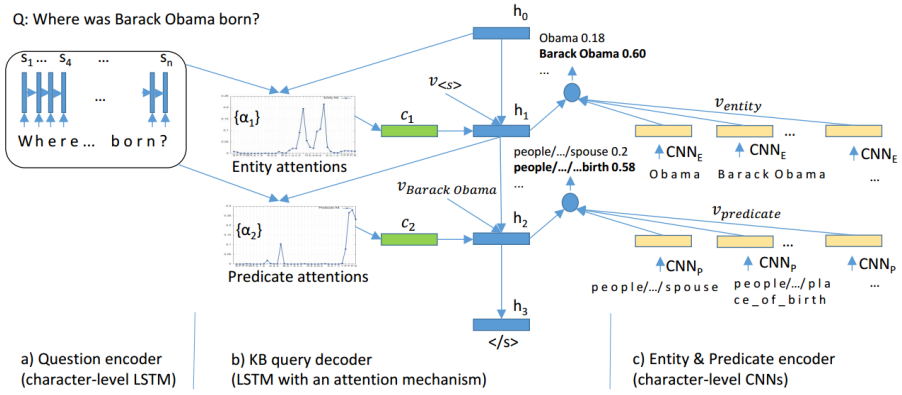

Fig. 21: A character-level QA approach for SimpleQA [32]

Dai et al. [21] propose a word-level RNN-based approach with emphasis on possible paraphrases of questions. The task of predicting subject and relation is factorized into two sub-tasks - prediction of relation first followed by entity given the relation and question (Figure 22). The model learns the question representation using GRU network , which is used to predict the likely relation candidate. The subject prediction is performed using join information of both relation and the question. Both [21] and [115] improve the performance of their approaches using a BiLSTM-CRF tagging model that is separately trained to label parts of the question as entity mention or context (relation pattern). Either log-likelihood or negative sampling method could be used for learning the parameters.

Multi-relation Question Answering The formulation of multi-relation question answering is driven by necessity to map questions in natural text to more than one triples in a knowledge base. WebQuestions [8] dataset is a popular benchmark for such problems with semantic parsing methods obtaining state of art performance in this task. A key necessity to complex factoid QA requires a large collection of text data with ClueWeb [111] a 5TB collection of text being used for KB matching.

For challenging questions such as " *What mountain is highest in North America?*"which requires learning a representation for mathematical function "highest", Xu et al. [107] use textual data to filter out wrong answers. The approach is a two-step process with first step for extracting entities from KB, which is then refined using Wikipedia text of topic entity as illustrated in Figure 23. A multi-relational question can be answered by decomposing original question into sub-questions using syntactic patterns with final answer obtained as intersection

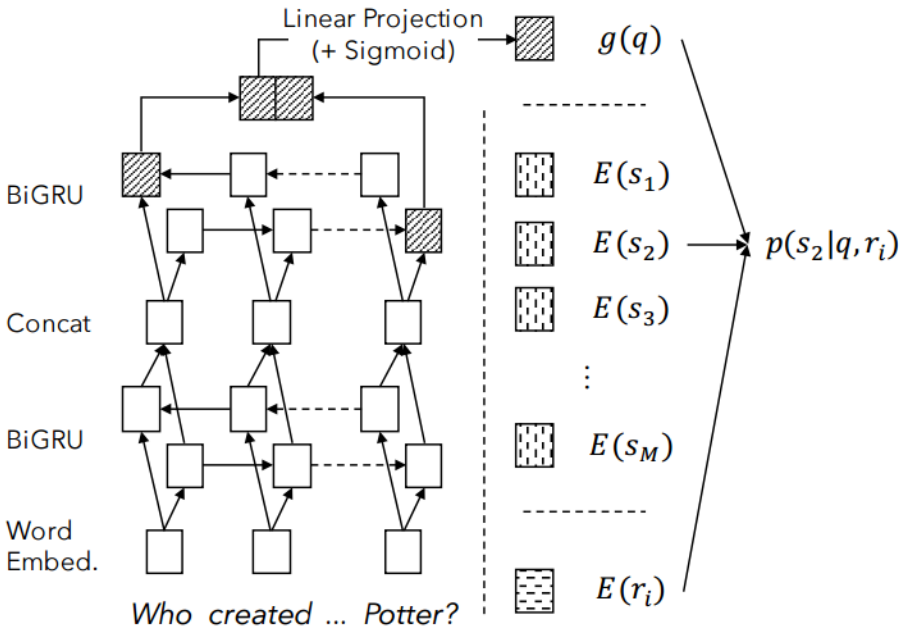

Fig. 22: Word-level RNN for SimpleQA [21]

of sub-question. The relation extraction module is based upon Multi-Channel CNN (Figure 24), with two channels being syntactic and sequential information. The syntactic feature is the shortest path between an entity mention and the question word in the dependency tree and the sequential information is words in the sentence excluding word and entity mention. The refinement model is more hand engineered with entity and relation clues to expand the answer search space. A gold standard of question-relation mapping is required for training the relation extraction. A dependency parser based query node expansion is devised in Yao and Van Durme [111] where ClueWeb text is used to learn correlation between KB relations and words using co-occurrence statistics with the alignment model.

Dong et al. [24] use multi-column CNNs to understand questions from three different aspect - answer path, answer context and answer type and learn their distributed representations. A low-dimensional embeddings of entities and relations in KB is jointly learned and QA pairs are used to train the model to rank candidate answers. This helps to analyze questions from multiple aspects. MC-CNNs help to improve question representation by considering word orders so that "who killed A?" and "who A killed?" are considered to be different as opposed to averaging over word-embeddings to obtain question representation [13]. A multiple-column CNN is used to learn multiple representation of same question which could be formulated to represent question from different stances. The

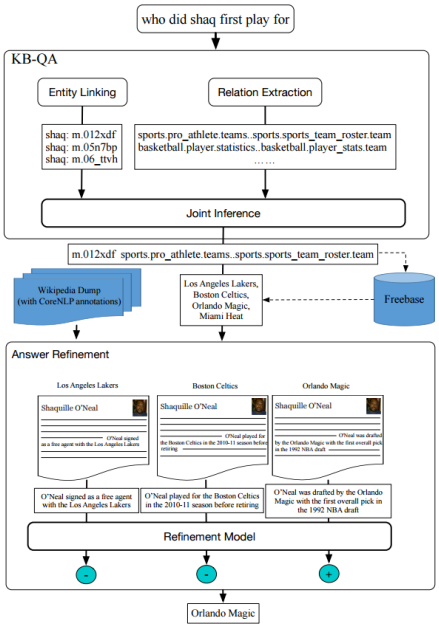

Fig. 23: An example of relation extraction and textual refinement [107].

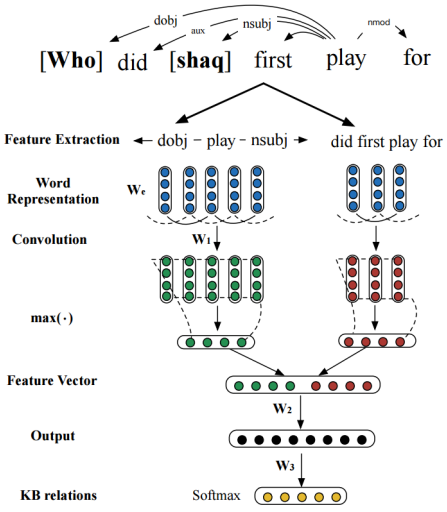

Fig. 24: MC-CNN for relation extraction [107].

vector representation of each candidate entity is also learned in multi-column stance i.e. multiple representations Then score for question stance and answer

candidate is computed based on sum over three different representation dot product. A simple Freebase search query is used for candidate generation, similar to Yao and Van Durme [111]. The candidate answers are embedded via answer path which averages all the relation in the path from question entity to answer candidate, answer context - which is the averaged embedding of all the entities and relations which appear in the context of the candidate answer and answer type - the emebdded representation of answer type. k random answers are sampled for each correct answer and a hinge loss is considered for pairs. The authors report that answer path and answer type are more direct clues for question. A paraphrase module is also learned for generalizing similar question word embeddings. The work achieves substantial improvements despite a naive candidate generation and pruning (random) mechanism. An example is given in Figure 25.

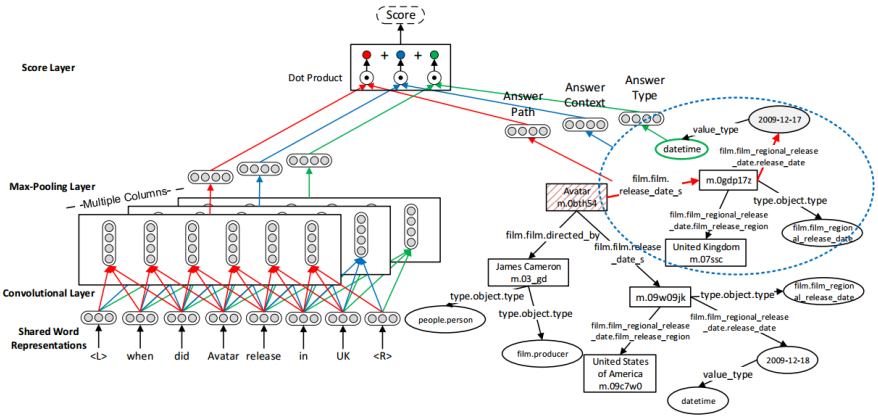

Fig. 25: An overview for question-answer pair using MC-CNN similarity approach. Left: network architecture for question understanding. Right: embedding candidate answer. [24]

Yang et al. [109] maps natural language to knowledge base by semi-automatically generating mappings between knowledge base triples and natural text using information extraction methods. A triple set is formed by merge between natural language and KB triples and they are jointly learned using translation-based embedding methods. Input questions are also associated with the joint embeddings following information extraction methods which is followed by summation based ranking measure. This model is particularly suited to SimpleQA problem, but is applicable to multi-relational QA task as well.

Bordes et al. [13] learns embedding for words and KB constituent which are then used to score natural language questions against candidate answers. WebQuestion [8] is used as evaluation benchmark. The dataset is augmented with ClueWeb extractions [8] and question paraphrasing [27]. This work learns the

embedding of each words, entites and relations. The candidate answers are represented either as one-hot vector, path representation from entity mentioned in question to answer entity, or subgraph representation which encodes all the entities surrounding the answer entity. The training is based on contrastive margin-loss objective (Figure 26). Embeddings are trained in multi-task manner taking paraphrase question dataset into considerations. A candidate answer is generated before inference and then score is computed.

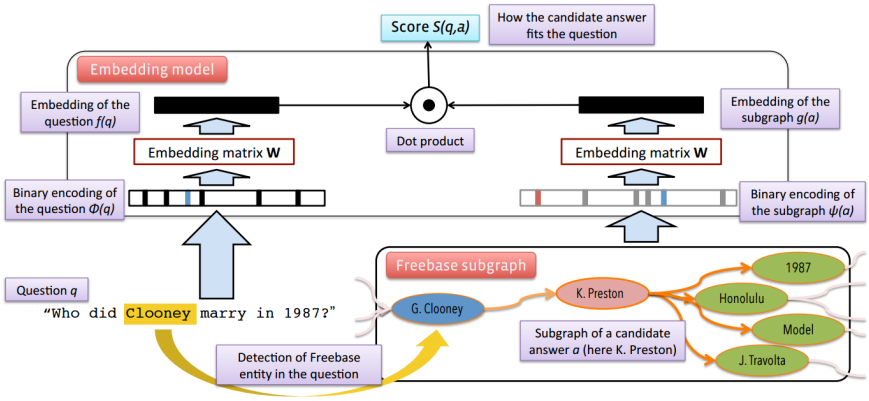

Fig. 26: Illustration of the subgraph embedding model scoring a candidate answer: (i) locate entity in the question; (ii) compute path from entity to answer; (iii) represent answer as path plus all connected entities to the answer (the subgraph); (iv) embed both the question and the answer subgraph separately using the learned embedding vectors, and score the match via their dot product. [13]

Yin et al. [113] propose an encoder-decoder framework model for factoid question answering with ability to enquire a KB. The key challenge addressed is switching between natural text and text from KB while generating answers to the question. They propose a model called GenQA (Figure 27) which consists of Interpreter, Enquirer, Answerer and an external KB. Interpreter transforms natural language question into en embedded representation and saves it to short term memory. Enquirer takes the question embedding and retrieves relevant facts from KB, which is summarized into an embedding. Enquirer first performs term matching followed by embedding similarity to obtain relevant triples. The similarity function could be either a bilinear model or an CNN-based matching model. The answerer (Figure 28) generates an answer using sequential model which is sampled from either the vocabulary from KB or from words.

Jain [42] preprocess Freebase to remove dummy entities and obtain more direct triples. A L-hop factual memory network is constructed for computational layers where each layer access candidate facts and question embedding. The similarity is computed based on translation sum of subject and relation embeddings

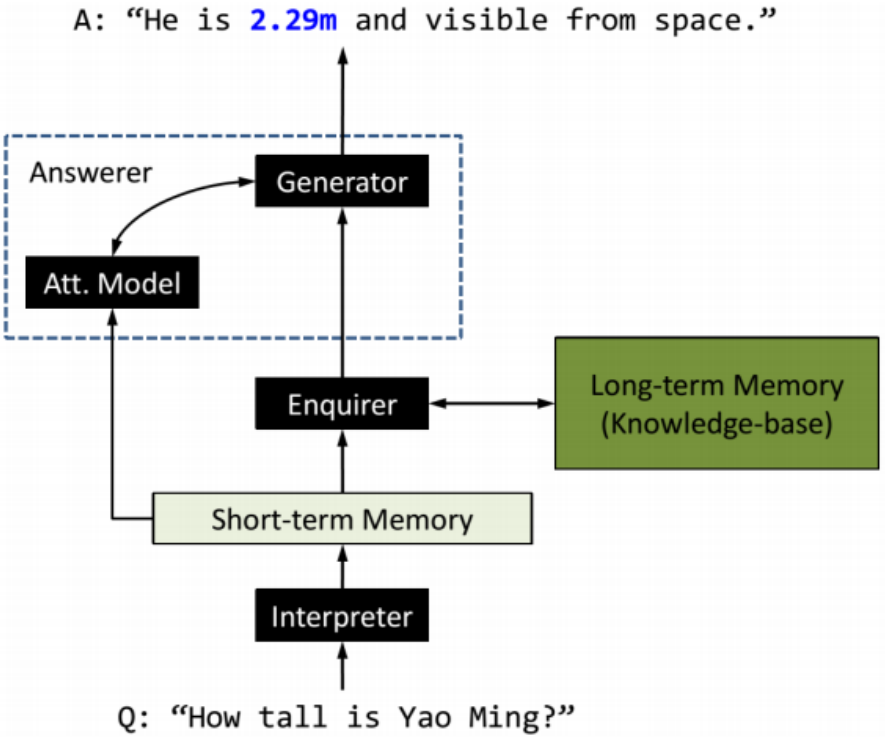

Fig. 27: System diagram of GenQA [113]

of fact and question embedding. For multiple object facts, averaging is done to compute the score. Each computation layer is followed by pruning of facts, recomputation of question vector based on relevant facts and addition of new facts. The model is trained using loss across each computation layer, weighted by its distance from question layer of distance between the true answer and facts considered in that computation layer. A paraphrasing loss is also included to project paraphrases into same sub regions. This method obtains the state-of-art performance in Web Questions dataset [8].

4.2 FQA over fixed answer set

Natural Language text is the most common source of answers in systems which do not depend upon KB to obtain an answer. A common paradigm is to have a closed FQA system with the possible answers already fixed, then the answer prediction step can be replaced by a simple softmax layer.

Iyyer et al. [41] use a Dependency-Tree Recursive Network (DT-RNN) [88] to learn the sentence embedding of each question. A dependency tree based RNN is suited for question answering task since the impact of negations significantly

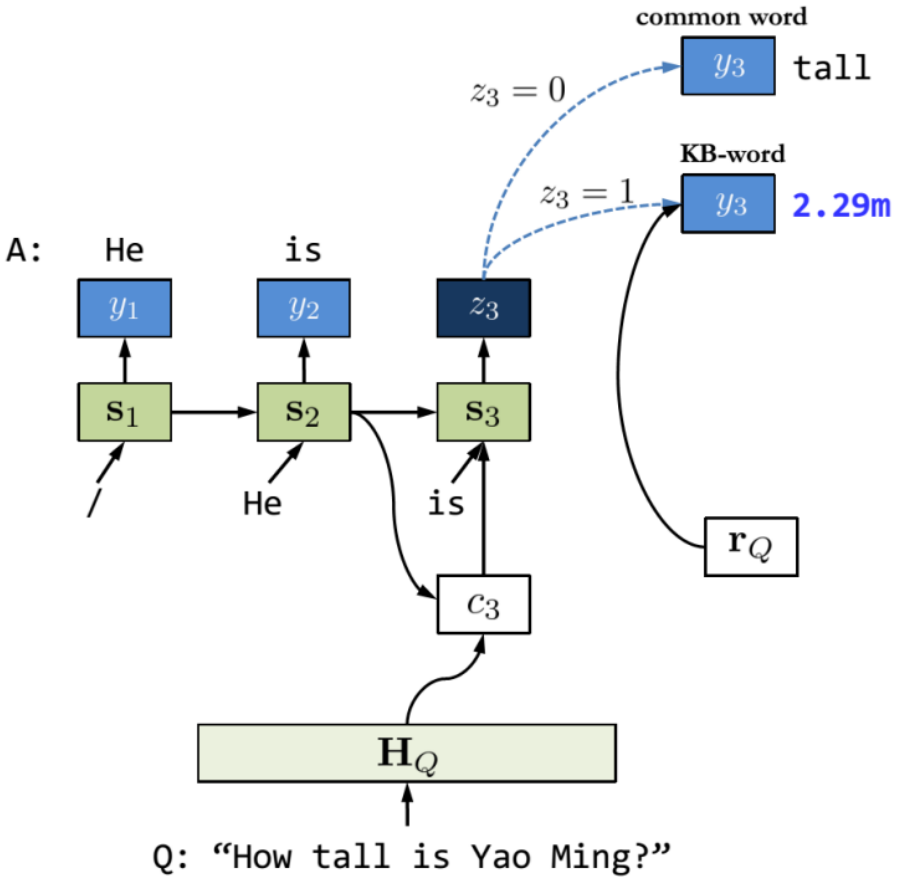

Fig. 28: Answerer module of GenQA [113]. Answerer is able to synthesize sequential answers using natural words with KB terms.

affect the answer and as such each word in question has different significance. Iyyer et al. [41] use a contrastive max-margin loss function that is applied to the dot product between the question sentence and the correct answer representation, multiplied by the rank difference between the incorrectly sampled answer to correct answer which is mapped to a simple linear functional. An answer representation learning framework helps with the generalization in both question and answer aspect. Since a fixed answer size is used for learning the question-answer mapping, it is a very domain specific approach.

4.3 FQA over multiple sources

A multiple-source based FQA requires question interpretation for different sources and alignment between multiple sources. Commonly used multiple sources include either multiple KBs or a single KB augmented with textual information to overcome sparseness of KBs.

Das et al. [22] apply question answering using memory networks over a universal schema which helps to jointly answer questions concerning either knowledge bases or texts. The universal schema matrix is attended over in memory networks to find the set of relations that are relevant to the question to finally obtain a softmax layer to answer question on. An attention mechanism between key of schema which is constructed by concatenating embedding of column and row values of schema and input question is used to learn the mapping from question to answer entity as shown in Figure 29. The model complexity makes this approach unsuitable for application to small set FQAs.

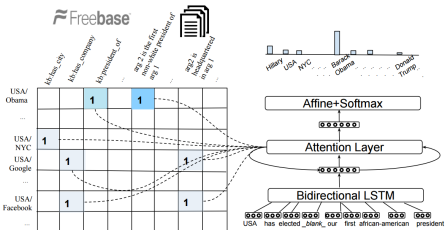

Fig. 29: Memory Networks attending the facts in universal schema [22].

Bordes et al. [15] use Reverb to generate question pattern from templates which are then used to learn an embedding which maps questions to answers into embedding representations. All words and entites and relations in KBs are mapped into one single vector representation using averaging. This work is one of the earlier work on embedding-based QA and suffers from limitation of training data when evaluated on WikiAnswers dataset [27].

The major constraint on factoid question answering models is data limitation. While there are multiple ways to phrase a single question, the dataset size suffers from sparseness and is unable to work with methods that require more training datasize. SimpleQA have made substantial progress recently due to the introduction of SimpleQuestions [16] dataset, making larger neural network models trainable till convergence without overfitting. While the focus on SimpleQA task is to generalize mapping of question to facts, non-simple QA tasks and multi-resource open domain QA task require learning mathematical and functional dependencies required to answer the question. This makes the problem considerably more complex, while at the same time, limited training data constrains the model to use lesser parameters. There are also very few methods that attempt to leverage multiple knowledge sources.

5 Attention-based Question Answering

Attention-based QA are extremely popular approaches for multi-modal data problem such as Visual Question Answering (VQA) and problems requiring deeper understanding of input data such as Reading Comprehension (RC) (also called Machine Comprehension). A common approach to VQA is illustrated in Figure 30. The baseline model concatenates visual and textual representations obtained from CNN and RNN respectively to perform joint inference. This approach can be improved upon by introduction attention maps for input image - with embedding for a certain section of image - which are then attended over using attention mechanism for learning a joint embedding, which then performs the final classification or sequence generation task. Multimodal bilinear compact pooling [29] propose an efficient but highly optimized bilinear pooling over two data sources enabling a robust embedding for visual question answering.

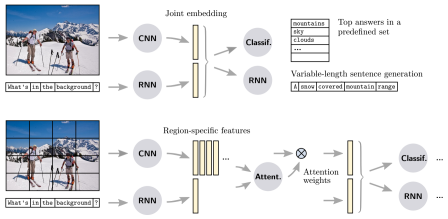

Fig. 30: Common approaches to visual question answering [103]

Kazemi and Elqursh [45] obtain state-of-art performance in Visual Question Answering. Their architecture (Figure 31) consists of embedding an image using ResNet [36]. The high level representation of image is a three-dimensional tensor. The questions are embedded using standard LSTM-network. A multiple attention distributions over spatial dimension of the image features are computed using the embeddings of image and question. The attention distribution gives image glimses over the original input image, which are concatenated with image and text embeddings and fed to a fully connected neural network for prediction.

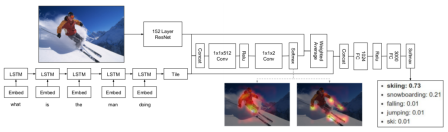

Fig. 31: An overview of Kazemi and Elqursh [45] approach to VQA.

Another problem that is modeled using neural networks with attention is RC task. In fact, RC and VQA architectures are closely related. Kumar et al. [50]

apply the same architecture to both problem (Figure 32) and obtain substantial improvements over baseline methods.

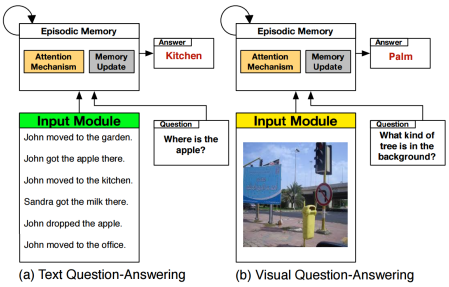

Fig. 32: Dynamic Memory Networks application to textual and visual question answering [50].

R-Net [98] (Figure 33) obtain state-of-art result on most complete RC dataset, SQuAD [78]. The difference between VQA and RC lies in decoding stage of inference where VQA decoding is done based upon preset vocabulary. RC datasets might require sampling of input text to generate answer phrases or sequences. This requires probabilistic decoding using combination of language decoding and pointer networks [96] to obtain answer effectively. R-Net uses GRU to learn embeddings for input question and sentence which is then passed to gated attention-based recurrent networks to determine importance of information in the passage regarding a question. Each passage representation incorporates aggregating matching information from the whole question. Another gate is added to determine importance of passage parts relevant to the question. Another attention to match over itself is used to incorporate context into question-aware embeddings. Pointer Network is used to predict the start and end position of the answer.

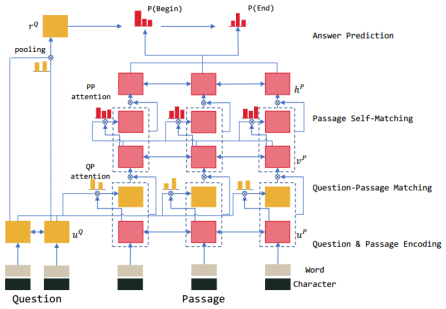

Fig. 33: R-Net architecture for reading comprehension [98]

While there are many different variants of visual question answering and reading comprehension methods in literature (see [1] for more details), the underlying mechanism entails learning the fixed vector representation for both question and input data (either image or text), then using the attention or bilinear pooling to learn a joint embeddings. The learned vectors are used for making predictions. We do not attempt to cover the entire attention-based question answering methods due to space and time constraints.

6 Future Directions

I propose two directions for future: 1) Adding structural constraints to knowledge base embedding methods to tighten dependencies over relation path and entity types and 2) applying embedded knowledge bases to neural models for multi KB question answering.

6.1 KB Embedding with structural constraints

The current approaches to embedding large entities such as KGs or a KB with ontology treat the data as a triple, essentially ignoring structural information and constraints present in the original representation. There is some progress in including structural information during embedding ([35], [69]) where a compositional relation is formulated between facts. The tail fact of a triple can be reached from head fact , in a large-scale KG, through multiple paths. Such path is analogous to structure information of underlying KBs, which acts as a regularizer to learn better embeddings. Such method can be compared to word embedding methods [53] which exploit context information to learn structure of natural texts. However, there is even more structure information that can be used from a Knowledge Graph (KG) or a Knowledge Base (KB). Such information can be easily obtained in form of constraints, especially for KBs. For example; the *type* information in Freebase is a constraint, as it restricts domain and range of a relation.

We propose a novel method for learning embeddings of large-scale Knowledge Base (KB) such as WordNet. We formulate the embedding learning process as a transformation where the *head* and *tail* entities are transformed non-linearly by a relation such that a true *head* and *tail* entity transformed by true relation are closest in the projected sub-space. We treat all elements of a triple as sequences, such that each element is provided with additional information. For ontology embedding, each sequence comprises of the entity type, relations and sub-classes information regarding each element. For KGs, a sequence could be a list of elements near the entity. We use a Recurrent Neural Network (RNN) architecture for learning representation of entities, which are then jointly attended over by relation embedding to obtain relation-transformed representations. Our architecture is illustrated in Figure 34.

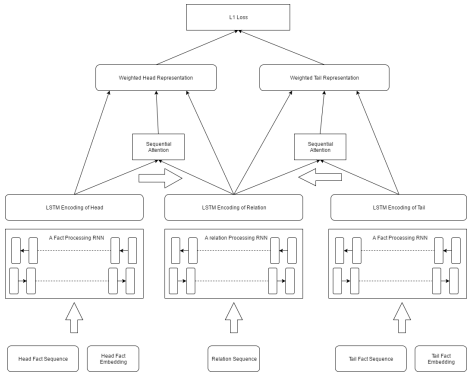

Fig. 34: Our proposed architecture for learning embedding for KBs.

Applying Learned Embeddings The current approaches exclusively use the learned embeddings from KB for similarity score computation between incoming question and triple. Such approaches while able to relate between texts and facts fail to exploit the representation of KB. One primary issue is due to the much larger size of entities and relations in a KB, making them exhaustively difficult to consider completely during training and inference. A constraint-based KB embeddings where each entity is identified by its constraint along with learned representation, makes it possible to use KB in inclusive manner during learning. An attention-based dynamic question embedding with optimal fact extraction methods can be devised efficiently with a type and attribute constrained knowledge base embeddings.

6.2 Applying multiple KBs to Neural Networks

Any question answering system can benefit from having more knowledge resources during answer generation. Similar to [106] where additional Wikipedia text source is used to verify an answer, an additional KB can ideally improve the performance of a QA system significantly. The lack of application of multiple KBs to QA problem is due to the current implementation being unable to handle one single KB effectively. A recurrent paradigm for learning embeddings can be applied to obtain a higher level representation of a sub-tree within a a knowledge graph. The neural question answering models can benefit from sub-tree embedding learning of KGs making it possible to use multiple KBs within the same problem framework.

Bibliography

[1] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2425–2433, 2015.

[2] Grigoris Antoniou and Frank Van Harmelen. Web ontology language: Owl. In *Handbook on ontologies*, pages 67–92. Springer, 2004.

[3] Michael Ashburner, Catherine A Ball, Judith A Blake, David Botstein, Heather Butler, J Michael Cherry, Allan P Davis, Kara Dolinski, Selina S Dwight, Janan T Eppig, et al. Gene ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–29, 2000.

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

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[6] Hannah Bast and Elmar Haussmann. More accurate question answering on freebase. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1431–1440. ACM, 2015.

[7] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[8] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, volume 2, page 6, 2013.

[9] Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. Finding the right facts in the crowd: factoid question answering over social media. In *Proceedings of the 17th international conference on World Wide Web*, pages 467–476. ACM, 2008.

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

[11] Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. Learning structured embeddings of knowledge bases. In *Conference on artificial intelligence*, number EPFL-CONF-192344, 2011.

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

[13] Antoine Bordes, Sumit Chopra, and Jason Weston. Question answering with subgraph embeddings. *arXiv preprint arXiv:1406.3676*, 2014.

[14] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259, 2014.

[15] Antoine Bordes, Jason Weston, and Nicolas Usunier. Open question answering with weakly supervised embedding models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 165–180. Springer, 2014.

[16] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*, 2015.

[17] Eric Brill, Jimmy J Lin, Michele Banko, Susan T Dumais, Andrew Y Ng, et al. Data-intensive question answering. In *TREC*, volume 56, page 90, 2001.

[18] Denny Britz. Understanding convolutional neural networks for nlp. `http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/`. Accessed: 2016-09-18.

[19] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, volume 5, page 3, 2010.

[20] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[21] Zihang Dai, Lei Li, and Wei Xu. Cfo: Conditional focused neural question answering with large-scale knowledge bases. *arXiv preprint arXiv:1606.01994*, 2016.

[22] Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. Question answering on knowledge bases and text using universal schema and memory networks. *arXiv preprint arXiv:1704.08384*, 2017.

[23] Li Deng, Dong Yu, et al. Deep learning: methods and applications. *Foundations and Trends® in Signal Processing*, 7(3–4):197–387, 2014.

[24] Li Dong, Furu Wei, Ming Zhou, and Ke Xu. Question answering over freebase with multi-column convolutional neural networks. In *ACL (1)*, pages 260–269, 2015.

[25] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610. ACM, 2014.

[26] Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545. Association for Computational Linguistics, 2011.

[27] Anthony Fader, Luke S Zettlemoyer, and Oren Etzioni. Paraphrase-driven learning for open question answering. In *ACL (1)*, pages 1608–1618. Citeseer, 2013.

[28] Jun Feng, Mantong Zhou, Yu Hao, Minlie Huang, and Xiaoyan Zhu. Knowlege graph embedding by flexible translation. *arXiv preprint arXiv:1505.05253*, 2015.

[29] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847*, 2016.

[30] Alberto Garcia-Duran, Antoine Bordes, and Nicolas Usunier. *Composing relationships with translations*. PhD thesis, CNRS, Heudiasyc, 2015.

[31] Alberto García-Durán, Antoine Bordes, Nicolas Usunier, and Yves Grandvalet. Combining two and three-way embedding models for link prediction in knowledge bases. *Journal of Artificial Intelligence Research*, 55:715–742, 2016.

[32] David Golub and Xiaodong He. Character-level question answering with attention. *arXiv preprint arXiv:1604.00727*, 2016.

[33] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[34] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[35] Kelvin Guu, John Miller, and Percy Liang. Traversing knowledge graphs in vector space. *arXiv preprint arXiv:1506.01094*, 2015.

[36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[37] Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. Learning to represent knowledge graphs with gaussian embedding. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 623–632. ACM, 2015.

[38] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701, 2015.

[39] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[40] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[41] Mohit Iyyer, Jordan L Boyd-Graber, Leonardo Max Batista Claudino, Richard Socher, and Hal Daumé III. A neural network for factoid question answering over paragraphs. In *EMNLP*, pages 633–644, 2014.

[42] Sarthak Jain. Question answering over knowledge base using factual memory networks. In *Proceedings of NAACL-HLT*, pages 109–115, 2016.

[43] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. Knowledge graph embedding via dynamic mapping matrix. In *ACL (1)*, pages 687–696, 2015.

[44] Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. Knowledge graph completion with adaptive sparse transfer matrix. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[45] Vahid Kazemi and Ali Elqursh. Show, ask, attend, and answer: A strong baseline for visual question answering. *arXiv preprint arXiv:1704.03162*, 2017.

[46] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[47] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. *arXiv preprint arXiv:1508.06615*, 2015.

[48] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[49] Denis Krompaß, Stephan Baier, and Volker Tresp. Type-constrained representation learning in knowledge graphs. In *International Semantic Web Conference*, pages 640–655. Springer, 2015.

[50] Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. *CoRR, abs/1506.07285*, 2015.

[51] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[52] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[53] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185, 2014.

[54] Chen Liang and Kenneth D Forbus. Learning plausible inferences from semantic web knowledge by combining analogical generalization with structured logistic regression. In *AAAI*, pages 551–557, 2015.

[55] Percy Liang. Learning executable semantic parsers for natural language understanding. *Communications of the ACM*, 59(9):68–76, 2016.

[56] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling relation paths for representation learning of knowledge bases. *arXiv preprint arXiv:1506.00379*, 2015.

[57] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187, 2015.

[58] Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W Black, and Isabel Trancoso. Finding function in form: Compositional character models for open vocabulary word representation. *arXiv preprint arXiv:1508.02096*, 2015.

[59] Hugo Liu and Push Singh. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.

[60] Denis Lukovnikov, Asja Fischer, Jens Lehmann, and Sören Auer. Neural network-based question answering over knowledge graphs on word and character level. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1211–1220. International World Wide Web Conferences Steering Committee, 2017.

[61] Yuanfei Luo, Quan Wang, Bin Wang, and Li Guo. Context-dependent knowledge graph embedding. In *EMNLP*, pages 1656–1661, 2015.

[62] Minh-Thang Luong and Christopher D Manning. Achieving open vocabulary neural machine translation with hybrid word-character models. *arXiv preprint arXiv:1604.00788*, 2016.

[63] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, page 3, 2010.

[64] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[65] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[66] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[67] Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. Compositional vector space models for knowledge base completion. *arXiv preprint arXiv:1504.06662*, 2015.

[68] Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. Neighborhood mixture model for knowledge base completion. *arXiv preprint arXiv:1606.06461*, 2016.

[69] Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. Stranse: a novel embedding model of entities and relationships in knowledge bases. *arXiv preprint arXiv:1606.08140*, 2016.

[70] Maximilian Nickel and Volker Tresp. Logistic tensor factorization for multi-relational data. *arXiv preprint arXiv:1306.2084*, 2013.

[71] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 809–816, 2011.

[72] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing yago: scalable machine learning for linked data. In *Proceedings of the 21st international conference on World Wide Web*, pages 271–280. ACM, 2012.

[73] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. Holographic embeddings of knowledge graphs. *arXiv preprint arXiv:1510.04935*, 2015.

[74] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. Understanding the exploding gradient problem. *CoRR, abs/1211.5063*, 2012.

[75] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[76] Slav Petrov. Announcing syntaxnet: The world's most accurate parser goes open source. *Google Research Blog, May*, 12:2016, 2016.

[77] Eric Prud, Andy Seaborne, et al. Sparql query language for rdf. 2006.

[78] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[79] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. Relation extraction with matrix factorization and universal schemas. 2013.

[80] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[81] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*, 2016.

[82] Rob Shearer, Boris Motik, and Ian Horrocks. Hermit: A highly-efficient owl reasoner. In *OWLED*, volume 432, page 91, 2008.

[83] Baoxu Shi and Tim Weninger. Proje: Embedding projection for knowledge graph completion. *arXiv preprint arXiv:1611.05425*, 2016.

[84] Amit Singhal. Introducing the knowledge graph: things, not strings. *Official google blog*, 2012.

[85] Eriks Sneiders. Automated question answering using question templates that cover the conceptual model of the database. In *International Conference on Application of Natural Language to Information Systems*, pages 235–239. Springer, 2002.

[86] Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the conference on empirical methods in natural language processing*, pages 151–161. Association for Computational Linguistics, 2011.

[87] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934, 2013.

[88] Richard Socher, Andrej Karpathy, Quoc V Le, Christopher D Manning, and Andrew Y Ng. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2:207–218, 2014.

[89] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM, 2007.

[90] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015.

[91] Ilya Sutskever, Joshua B Tenenbaum, and Ruslan R Salakhutdinov. Modelling relational data using bayesian clustered tensor factorization. In *Advances in neural information processing systems*, pages 1821–1828, 2009.

[92] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[93] Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Falk Brauer. Random semantic tensor ensemble for scalable knowledge graph link prediction. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 751–760. ACM, 2017.

[94] Kristina Toutanova, Xi Victoria Lin, Wen-tau Yih, Hoifung Poon, and Chris Quirk. Compositional learning of embeddings for relation paths in knowledge bases and text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 1434–1444, 2016.

[95] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. *arXiv preprint arXiv:1606.06357*, 2016.

[96] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700, 2015.

[97] Ellen M Voorhees, Donna K Harman, et al. *TREC: Experiment and evaluation in information retrieval*, volume 1. MIT press Cambridge, 2005.

[98] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[99] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119. Citeseer, 2014.

[100] Zhigang Wang and Juanzi Li. Text-enhanced representation learning for knowledge graph. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 1293–1299. AAAI Press, 2016.

[101] Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. Knowledge base completion via search-based question answering. In *Proceedings of the 23rd international conference on World wide web*, pages 515–526. ACM, 2014.

[102] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[103] Qi Wu, Damien Teney, Peng Wang, Chunhua Shen, Anthony Dick, and Anton van den Hengel. Visual question answering: A survey of methods and datasets. *Computer Vision and Image Understanding*, 2017.

[104] Han Xiao, Minlie Huang, Yu Hao, and Xiaoyan Zhu. Transg: A generative mixture model for knowledge graph embedding. *arXiv preprint arXiv:1509.05488*, 2015.

[105] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.

[106] Kun Xu, Yansong Feng, Siva Reddy, Songfang Huang, and Dongyan Zhao. Enhancing freebase question answering using textual evidence. *arXiv preprint arXiv:1603.00957*, 2016.

[107] Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. Question answering on freebase via relation extraction and textual evidence. *arXiv preprint arXiv:1603.00957*, 2016.

[108] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

[109] Min-Chul Yang, Nan Duan, Ming Zhou, and Hae-Chang Rim. Joint relational embeddings for knowledge-based question answering. In *EMNLP*, volume 14, pages 645–650, 2014.

[110] Xuchen Yao. Lean question answering over freebase from scratch. In *HLT-NAACL*, pages 66–70, 2015.

[111] Xuchen Yao and Benjamin Van Durme. Information extraction over structured data: Question answering with freebase. In *ACL (1)*, pages 956–966. Citeseer, 2014.

[112] Wen-tau Yih, Xiaodong He, and Christopher Meek. Semantic parsing for single-relation question answering. In *ACL (2)*, pages 643–648. Citeseer, 2014.

[113] Jun Yin, Xin Jiang, Zhengdong Lu, Lifeng Shang, Hang Li, and Xiaoming Li. Neural generative question answering. *arXiv preprint arXiv:1512.01337*, 2015.

[114] Pengcheng Yin, Zhengdong Lu, Hang Li, and Ben Kao. Neural enquirer: Learning to query tables. *arXiv preprint arXiv:1512.00965*, 2015.

[115] Wenpeng Yin, Mo Yu, Bing Xiang, Bowen Zhou, and Hinrich Schütze. Simple question answering by attentive convolutional neural network. *arXiv preprint arXiv:1606.03391*, 2016.

[116] Hee-Geun Yoon, Hyun-Je Song, Seong-Bae Park, and Se-Young Park. A translation-based knowledge graph embedding preserving logical property of relations. In *Proceedings of NAACL-HLT*, pages 907–916, 2016.

[117] Pierre Zweigenbaum. Question answering in biomedicine. In *Proceedings Workshop on Natural Language Processing for Question Answering, EACL*, volume 2005, pages 1–4. Citeseer, 2003.