Institute for Natural Language Processing

University of Stuttgart
Pfaffenwaldring 5B
D–70569 Stuttgart

Master Thesis

Question Answering on Knowledge Bases: A Comparative Study

Vishnudatha Kanjur

Course of Study:	Computer Science
Examiner:	Prof. Dr. Ngoc Thang Vu
Supervisor:	Maximilian Schmidt, M.Sc.
Commenced:	15 July, 2020
Completed:	15 January, 2021

Abstract

Question Answering intends to automatically extract accurate and relevant information as the answer to a particular question. A large amount of data from the Web is stored as Knowledge bases in a structured way. Question answering on Knowledge bases is a research field that involves multiple branches of computer science like natural language processing, information retrieval and artificial intelligence. Knowledge Base Question Answering (KBQA) research involves various challenges to be solved in multiple aspects. This thesis aimed to compare several state-of-the-art methods for single relation KBQA. The widely used standard single relation dataset, SimpleQuestions dataset was used in the study against Freebase Knowledge Base (KB). A comprehensive analysis of the underlying models and their architecture was performed. Furthermore, to identify the drawbacks and possible enhancements, several approaches for evaluating the models were explored. The results show how the models were performed and the suitability of considering them for solving real-world problems in question answering.

Contents

List of Figures

List of Tables

Acronyms

BiGRU Bidirectional Gated Recurrent Unit. 7

BiLSTM Bidirectional Long Short Term Memory. 7

CNN Convolutional Neural Network. 7

CRF Conditional Random Field. 7

GRU Gated Recurrent Unit. 7

KB Knowledge Base. 3

KBQA Knowledge Base Question Answering. 3

KBs Knowledge Base. 13

LSTM Long Short Term Memory. 20

MID Machine Identifier. 27

NLP Natural Language Processing. 13

QA Question Answering. 15

RNN Recurrent Neural Network. 7

1 Introduction

Question answering has undergone substantial research in the field of Natural Language Processing (NLP), machine learning and information retrieval. Internet or web is a huge archive of data. Many information extraction systems mine these data and store them as Knowledge Bases (KBs). Enabling this enormous information for answering questions involves solving challenges in the field of machine learning, information retrieval and natural language processing. Unstructured natural language questions need investigation against the structured knowledge bases. The first step towards solving these challenges is to inspect the single relation question answering over knowledge bases, which later on could be extended to multiple relations and multi constrained question answering. A comparative study on the current trends helps to explore various researches and gain a better understanding of the current models.

The objective of this thesis is to conduct a comparative analysis of current researches on the question answering on knowledge bases. There are multiple aspects based on which the comparison of KBQA can be done. This thesis focuses on single relation factoid questions answering over knowledge bases. The research is based on the question answering on Freebase knowledge base. Freebase knowledge base is used for most of the previous researches. This thesis base the analysis on the Simple Question data set, which is one of the popular benchmarks used for analysing single relation factoid question answering.

The main contributions of this thesis can be summarized as:

1. This work analyses how various systems work with single relation factoid question answering.
2. This work analyses how models perform with respect to accuracy and F1 score.
3. This work analyses how each system performs with reduced data.
4. This work provides a brief error analysis on various models.
5. This work provides an overview of the machine learning aspects explored in various models.

Outline

The remaining layout of this thesis is organized as follows:

- **Chapter 2** – Literature Review: provides an overview of the related work relevant for this thesis.
- **Chapter 3** – Theoretical Background: addresses the main theoretical topics related to this comparison work, such as the machine learning concepts and deep neural networks that are used in the compared models, accuracy, F1 score which are the aspects of comparison.
- **Chapter 4** – Resources: explains the resources used for the comparison such as single factoid relations, knowledge base used, data set used for comparison,
- **Chapter 5** – Model details: explains the relevant models that are compared.
- **Chapter 6** – Evaluation: various aspects of the models are compared and the results are presented.
- **Chapter 7** – Conclusions and Outlook: summarizes this work and provides the inferences of the study.

2 Literature Review

This chapter gives a glimpse of the researches in Question Answering (QA) and briefly summarizes the current researches in the field of KBQA.

2.1 Question Answering

QA is a computer science field intended to retrieve the relevant and accurate information, using natural language processing and artificial intelligence [HG01]. Hirschman and Gaizauskas [HG01] discusses the various aspects of questions and answers and its evaluation. Their research differentiates the questions, based on the answer type, as questions which answers factual answers, opinions and summarizing the content. Based on the question structure, their research divides questions as yes/no questions, 'wh'-questions, indirect questions and commands (Ex: Name, List, Provide etc.). Their research paper analyzes the difficulty level of various queries briefly. Also, classification of answers as long or short based on the answer length is explained.

Multimodal aspects of the QA systems are also explored by various authors [MM11]. Based on the multimodal aspects, QA systems can be mainly divided into three modalities: voice, text and image-based question answering. Voice-based question answering includes incorporating the speech into the question answering system [ZDK+17], whereas, visual aspects were compared in image-based question answering [SZD+18] [NLS18] [WWS+17]. Text-based question-answering in natural language processing involves research in multiple regards, like comprehension question answering, question answering using dialogue systems etc. Some of the major software product companies like IBM used their deep learning research in developing products like IBM Watson[FLB+13]. Amazon Alexa, Apple Siri and many other similar products have been popular in recent days which make uses of question answering using voice input. All these researches signify the relevance of automated question answering systems.

2.2 Question Answering on Knowledge Bases

For a particular natural language question, KBQA aims to automatically retrieve the best possible answer from the KB. KBQA systems can be widely grouped into two kinds: semantic parsing approaches and information retrieval approaches [CWZ19]. Semantic

parsing approaches construct an intermediate representation of the natural language question by extracting the semantics of the question, which is then executed on the KB to derive the answer. Information retrieval approaches, as the name signifies, are mainly concerned on retrieval of the answers from the question. Additionally, another aspect in which KBQA systems can be divided, is based on how they handle single relation factoid questions and multi relation questions answering. Single relation question answering comprises of questions that are answerable with one relation from the main entity of the question. It does not require the processing of multiple constraints in finding the answers. Multi relation question answering involves solving the answers by multi relation and constraints derived from the questions. Both of these domains are being widely researched in the development of efficient KBQA systems. Having said that, this thesis focuses on the single relation question answering.

Yih et al. [YCHG15] used semantic parsing methodology by introducing the query graph as an intermediate representation of the question. Their newly introduced method tried to solve the multiple relation question answering and produced good results against WebQuestions dataset on Freebase. Luo et al. [LLLZ18] also researched on similar query graph methodology by encoding the semantics of the complex query graph generated from the questions. In their research paper, ComplexQuestions, WebQuestions and SimpleQuestions datasets were tested on Freebase. Their model performs the best for ComplexQuestions, whereas, for WebQuestions and SimpleQuestions the scores were not the best in their comparison. Query graph methodology was later enhanced by Lan and Jiang [LJ20]. Their model was analysed as part of this comparative study. Their research provided state-of-the-art results on ComplexWebQuestions and WebQuestions dataset on Freebase [LJ20]. Bast and Haussmann [BH15] used query templates as an intermediate semantic representation of queries, which evaluated their performance on Free917 dataset and WebQuestions dataset on Freebase. Nevertheless, further researches provided much better results when compared to the query template method.

Sorokin and Gurevych [SG18] used a Gated Graph Neural Network topology in their research for question answering, wherein, the entities and relations were represented using directed graphs. Their research produced good results when tested on a subset of WebQuestions dataset on Wikidata KB. Considering the feasibility of adapting the model to Freebase KB, their model was not considered in this study.

Another research on the single relation question answering was conducted by Qu et al. [QLK+18], where they used a recurrent neural network and convolutional neural network for KBQA task. Their research was included in this comparison study as it provided good results on SimpleQuestions dataset and Freebase knowledge graph. Mohammed et al. [MSL18] tested their research model with several neural networks and verified results on single factoid questions. Their model performed state of the art accuracy on SimpleQuestions dataset on Freebase. Hence, it was also included as part of this study.

Zhou et al. [ZHZ18] researched on multi-hop question answering by iteratively analysing the question to derive the answer at each hop. Their research was not included in this comparative study as the repository consisting of the research was not publicly released

[CCC+19]. Petrochuk and Zettlemoyer [PZ18] model the probability distribution over the questions to identify the subject in the question and a recurrent network to classify the relation. Their research was on single relation question answering with good results against SimpleQuestions dataset on Freebase, and thus their model was included in this comparative study.

Chen et al. [CWZ19] explored the information retrieval methodology, which uses minimal handcrafted methods using bidirectional attention mechanism for question answering. Their research model was intended to work on multi-hop questions, which was also included in this comparative study, to analyze how their model performed for the single relation question answering.

Saxena et al. [STT20] is a recent research, which uses knowledge graph embedding in their model for KBQA task. Their research was not included in this study because of the following reasons. Firstly, the knowledge graph embedding for Freebase2M dataset is unavailable. Secondly, defining the hyperparameters and training the knowledge graph embedding for Freebase2M dataset may not be feasible within the given timeline.

3 Theoretical Background

This section provides the background information, which helps to understand KBQA better.

3.1 Knowledge base (KB)

Let E represent a set of entities, and let R represent a set of relations. A knowledge base K is a set of triples (subject, relation, object), mathematically denoted as $(s, r, o) \in E \times R \times E$. Subject and object are entities. Relation describes the link between the entities. It follows a graph structure data model to store the data. Semantic Web stores the extensive volumes of information in the form of KBs. Freebase, DBPedia, Wikidata, Google Knowledge graph are some of the important knowledge bases [DLSM18]. RDF is the format that stores the data in KB [DLSM18]. SPARQL is the query language used widely for querying the KB [DLSM18].

3.2 Machine learning/Deep learning models

This section gives a theoretical overview of the machine learning or deep learning techniques used in the compared models.

3.2.1 Convolutional Neural Network (CNN)

A CNN is a neural network architecture, comprising of an input layer, output layer and multiple of hidden layers[AMA17][ON15][YNDT18]. CNNs are used mainly for spatially independent data and it fetches the most relevant features from the input data. The hidden layers are composed of multiple convolution filters, which extracts the relevant features and followed by max-pooling layer for gathering the convolution results. Convolution filters acquire the knowledge of local features that are position-independent. The filter performs convolution operation; multiplication or dot product over the striding window over the data and reduces the data parameters by concentrating on the prominent features. Max pooling layer collects the convolution layer results thereby ignoring the irrelevant data. Figure 3.1 shows a CNN architecture which is used for classification of images[YNDT18].

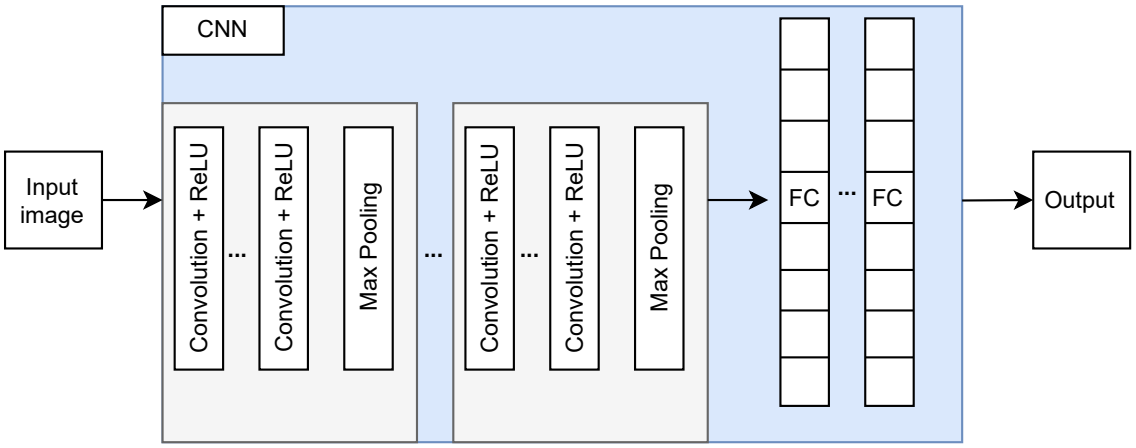

Figure 3.1: CNN model overview. This diagram is adapted from Yamashita et al. [YNDT18]

The sample architecture consists of multiple convolution layers followed by a max pooling layer and one or more fully connected layers. The image with its pixel values becomes the input layer. The convolution layer extracts the local features. Rectified linear unit is used with the convolution layer for activation. The number of samples is reduced by the max-pooling layer. Lastly, fully connected layers predict the output class from the local features collected from the previous layers.

3.2.2 Recurrent Neural Network (RNN)

RNNs are neural networks that are capable of handling sequential data with temporal aspects. In this type of network, the output at a particular time step depends on the results from the previous time step. Figure 3.2 below depicts the basic RNN and how at various time steps the network works. Basic RNN follows a successive layer organization in which each node is connected to its next layer[VAGS16][Bul13]. Each node has time dependant activation and weights W_i, W_o, W_h, where i, o and h denotes the input, output and hidden states. The output vector y_t is predicted based on the previous history at step $t-1$ and the current timestamp t. There are multiple variations of RNN like Elman network, Jordan network, bidirectional RNN, Long Short Term Memory (LSTM) etc.

3.2.3 Long Short Term Memory(LSTM)

Hochreiter and Schmidhuber [HS97] proposed the concept of Long short term memory RNN(LSTM) in 1997. LSTM is a neural network which is capable of storing information over time and also commands which information needs to be stored or forgotten. Figure 3.3 shows the recurrent module in LSTM model[Ola15]. It comprises of gates which can pass or retain information selectively. As a first step, the current input at time t, x_t is

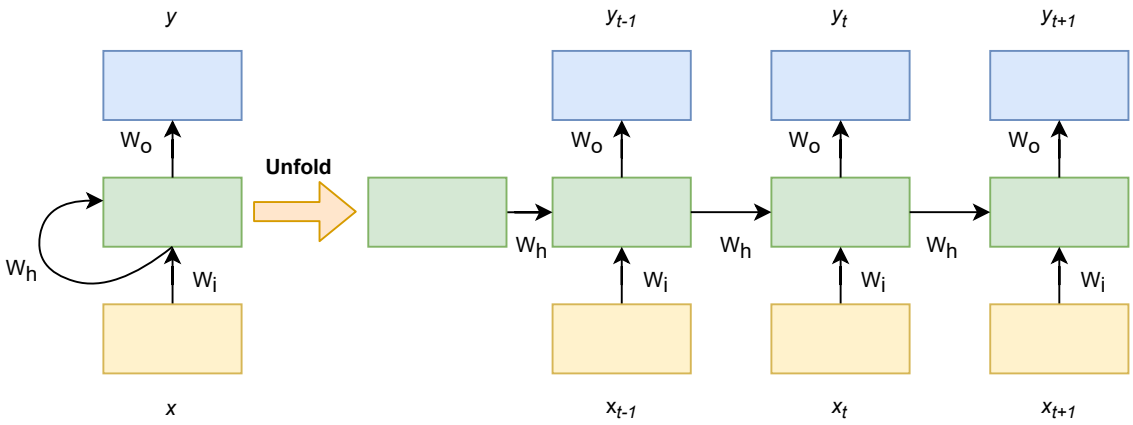

Figure 3.2: Basic RNN model unfold. This diagram is adapted from Olah [Ola15].

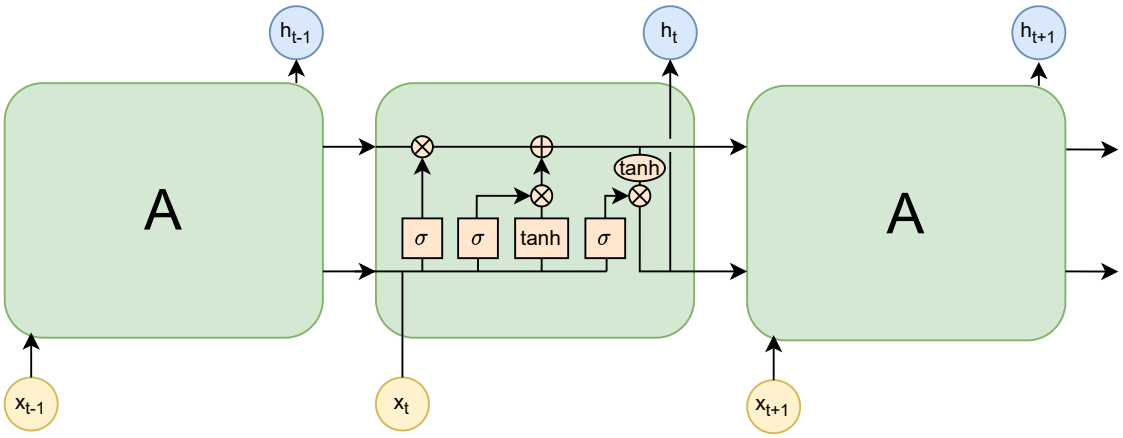

Figure 3.3: LSTM model. This diagram is adapted from Olah [Ola15].

combined with the previous hidden state output h_{t-1} and passed through the forget gate σ. Mathematically this could be represented below:

$$f_t = \sigma(W_f \cdot \left[h_{t-1}, x_t\right] + b_f) \tag{3.1}$$

σ function has the control to decide which information to retain or discard. Secondly, the network regulate the newly gained information. This is done by using two functions: $tanh$ and σ. σ decides the input to be retained and $tanh$ creates the new information which gets passed on to further steps. Mathematically, it can be shown as below:

$$i_t = \sigma(W_i \cdot \left[h_{t-1}, x_t\right] + b_i) \tag{3.2}$$

$$\tilde{C}_t = tanh(W_C \cdot \left[h_{t-1}, x_t\right] + b_C) \tag{3.3}$$

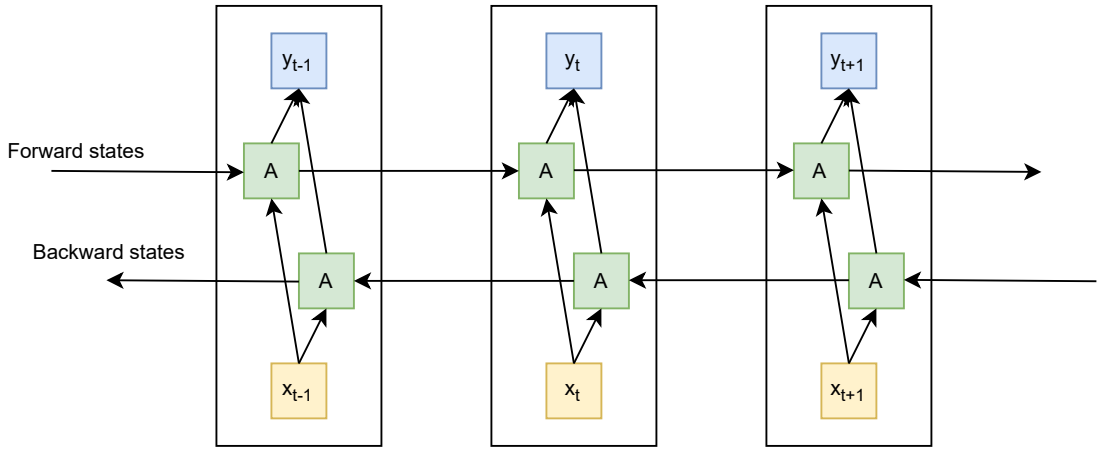

Figure 3.4: Bidirectional RNN model. This diagram is adapted from Schuster and Paliwal [SP97].

As a third step, the memory of the current LSTM cell updates using the output of first and second steps. Let C_{t-1} be the information from the previous LSTM cell and C_t the information of the current cell. Mathematically, it computes as below:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{3.4}$$

Fourth and the final step computes the output of the LSTM cell. This is performed using another σ function and $tanh$ function which finally decides, which information from the memory needs to be passed on further. Mathematically, it can be shown as below:

$$o_t = \sigma(W_o \cdot \left[h_{t-1}, x_t\right] + b_o) \tag{3.5}$$

$$h_t = o_t * tanh(C_t) \tag{3.6}$$

LSTM has been proven to solve various NLP problems efficiently, by retaining the right information from previous historical data for time dependant prediction.

3.2.4 Bidirectional Long Short Term Memory (BiLSTM)

Schuster and Paliwal [SP97] put forward the idea of Bidirectional Recurrent Neural Network(Bi-RNN). BiLSTM is Bi-RNN in which each cell is a LSTM as explained in previous Section 3.2.3. Each recurrent cell contains two nodes of LSTM; backward and forward states(A as in Figure 3.4), which increases the input size and there by increases the learning of information. A is a LSTM cell as explained in Figure 3.3.

In NLP BiLSTM is used, where the context of the sequential input is relevant for prediction. The prediction at each time step incorporates data from past, present and future. Hence it learns the context of the problem better.

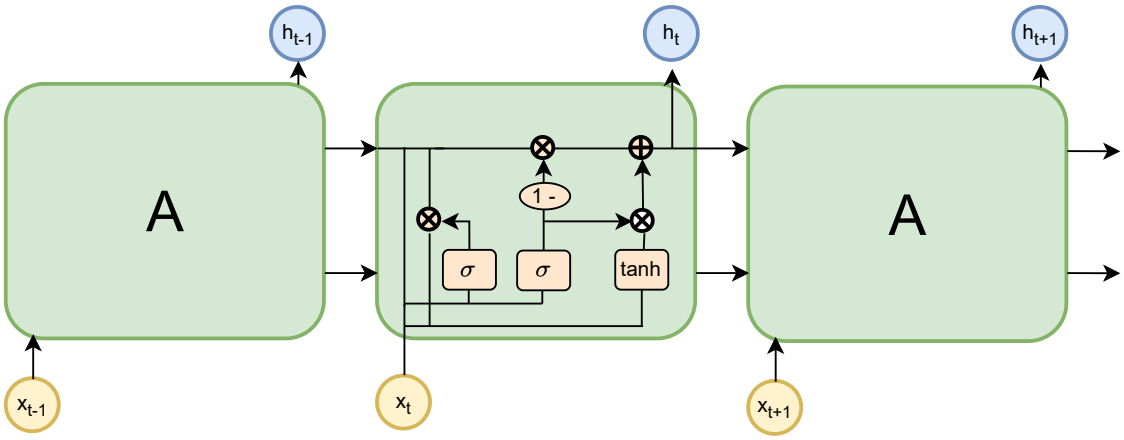

Figure 3.5: GRU model. This diagram is adapted from Olah [Ola15].

3.2.5 Gated Recurrent Unit (GRU)

Cho et al. [CMG+14] initiated the concept of gated mechanism in recurrent neural networks. Figure 3.5 shows the individual node of a GRU. It adaptively stores the required information and forgets based on the gate mechanism. The gate mechanism is realised using the reset gate and the update gate. The update gate is the combination of forget gate and input gate(explained in the LSTM). Update gate z_t is computed using σ function of the input x_t and h_{t-1}. Mathematically, it is computed as below:

$$z_t = \sigma(W_z \cdot \left[h_{t-1}, x_t\right]) \tag{3.7}$$

Reset gate r_t is computed using σ function of the input x_t and h_{t-1}. The value of r_t is used in the computation of $\tilde{h}t$. It can be mathematically shown as below:

$$r_t = \sigma(W_r \cdot \left[h_{t-1}, x_t\right]) \tag{3.8}$$

Intermediate hidden state $\tilde{h}t$ is later on computed using $tanh$ activation of the previous hidden state h_{t-1}, input x_t.

$$\tilde{h_t} = tanh(W \cdot \left[r_t * h_{t-1}, x_t\right]) \tag{3.9}$$

The intermediate $\tilde{h_t}$ processed with the update gate z_t and added with h_{t-1} multiplied with $(1 - z_t)$ to compute the final hidden state output. The following equation shows the final hidden state result:

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h_t} \tag{3.10}$$

GRU model has lesser complexity when compared to standard LSTM and also very much used in various NLP applications.

3.2.6 Bidirectional Gated Recurrent Unit (BiGRU)

Bidirectional Gated Recurrent Unit is a kind of bidirectional RNN as in Figure 3.4. Each unit A shown in the bidirectional RNN is a GRU component. GRU has a lower number of parameters in the network when compared to LSTM. So BiGRU automatically reduces the number of parameters and its complexity. Also additionally, BiGRU retains information from previous and future time steps.

3.2.7 Conditional Random Field (CRF)

Lafferty et al. [LMP01] proposed the probabilistic graphical model framework called Conditional Random Field for sequence data. The context of the data modelled using graphs encodes the dependencies between the prediction and data. Linear chain CRFs used extensively in NLP. One of the models compared uses linear chain CRF mechanism to tag the subject in the question.

Lafferty et al. [LMP01] specify CRFs using graph $G = (V, E)$ where V the set of vertices and E the set of edges, such that, $Y = (Y_{v \in V})$; so that Y is indexed by the vertices of G. Then (X, Y) is a condition random field if it follows Y_v conditioned on X, obeys Markov property: $P(Y_v|X, Y_w, w \neq v) = P(Y_v|X, Y_w, w \sim v)$, where $w \sim v$ means that w and v are neighbors in G.

3.3 Objective Performance Evaluation

This section explains the metrics that are used in evaluating the performance of models in KBQA task.

3.3.1 Accuracy

In machine learning, accuracy is the fraction of correctly predicted data samples out of the total data samples[Met78]. Mathematically, it is a ratio of the number of true positives and true negatives to the number of true positives, true negatives, false positives, and false negatives. It is a statistical measure, which depicts how well a model predicts its classes. It is computed as,

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{3.11}$$

where TP = True positive; FP = False positive; TN = True negative; FN = False negative.

3.3.2 F1 score

In machine learning, Precision and Recall are the metrics used to evaluate how well a model predicts its classes.

Precision is the ratio of the correctly identified positive samples to the total predicted positive samples.

(3.12) $Precision = TP/(TP + FP)$

where TP = True positive; FP = False positive.

The recall is the ratio of the correctly identified positive samples to the total actual positive samples.

(3.13) $Recall = TP/(TP + FN)$

where TP = True positive;FN = False negative.

Another metric for evaluating the classification algorithm is by using F1 score, which provides a better intuition on the incorrectly classified classes. It is computed as a harmonic mean of Precision and Recall[SF07].

(3.14) $F1 = 2 * (Precision * Recall)/(Precision + Recall)$

The harmonic mean in F1 score calculation penalizes the extreme values, so it is a good metric to evaluate a model when the class distribution is imbalanced.

4 Resources

4.1 Freebase

Freebase is a knowledge base developed by Metaweb in 2007 [BEP+08] to store general human knowledge efficiently. Freebase contained data that are created collaboratively and maintained in a structured manner. It provided an application programming interface(API) for its users to access the data. It was acquired by Google in 2010. Eventually, Freebase KB was officially shut down in May 2016 and API service was discontinued. However, Freebase data dumps are provided by Google[Cha18]. Later on, data from Freebase was partially migrated to Wikidata KB[PVS+16]. Freebase was based on four concepts: object, facts, type, properties[PVS+16]. Each entity in a Freebase is uniquely identified using a Machine Identifier (MID). For a particular entity, it can contain one or more types. Entity is linked using a specific property to a fact or another entity[PVS+16].

Freebase Knowledge base has been extensively studied for various Natural language processing problems including Question Answering. As Knowledge bases map human knowledge in a triple format, the size of the Knowledge base becomes considerably large. For research purposes, scientists use a subset of these knowledge bases. FB2M and FB5M are two major subsets of Freebase. Table 4.1 provides the details about the two versions of Freebase[BUCW15]. Most of the researches on question answering on knowledge bases are based on FB2M dataset, therefore this thesis report is based on the same.

	FB2M	**FB5M**
Entities	2,150,604	4,904,397
Relationships	6,701	7,523
Atomic facts	14,180,937	22,441,880
Facts (grouped)	10,843,106	12,010,500

Table 4.1: Details about FB2M and FB5M datasets. The table is referenced from Bordes et al. [BUCW15].

4.2 SimpleQuestions dataset

The SimpleQuestions dataset is the widely used dataset for single factoid question answering [BUCW15]. Bordes et al. [BUCW15] created this dataset for question answering on Freebase KB. It consists of a set of 100K human-annotated questions. Their research used a two-phase process that was used to collect the questions. Firstly, the facts from the Freebase were shortlisted for the questions. Secondly, human annotators were assigned to generate questions from the shortlisted facts. The facts were carefully sampled to ensure variability in the questions. Human annotators were asked to formulate questions from these samples which follow a format: the subject, relationship and answer being the object.

Knowledge bases encode the data as triples of the format (subject, relationship, object). SimpleQuestions dataset follows the same structure and contains the facts corresponding to Freebase Knowledge base. The 108,442 simple questions are split into training, validation and test sets. Training set comprises of 75910 questions, which is 70% of the total questions. Validation and test sets comprise of 20% and 10%, 10845 and 21,687 questions respectively. The dataset introduces two versions of Freebase Knowledge base FB2M and FB5M[BUCW15]. This thesis used the SimpleQuestion dataset, which is the benchmark used in researches.

5 Model details

This section describes the overall architecture of the models that are included in the comparison.

5.1 Model 1: Attentive RNN with Similarity Matrix based CNN(AR-SMCNN)

Qu et al. [QLK+18] introduces the concept of attentive recurrent neural network with similarity matrix based convolutional neural network model for QA. The model tries to represent the hierarchical information by making use of the benefits of both RNN and CNN. The model comprises of two components:

1. Entity detection for identifying the topic entity
2. Relation detection for capturing the correlation between question and candidate relation.

Figure 5.1 explains the architecture followed by this model in a nutshell.

For a particular query question, entity detection is carried out to generate entity mentions. The entity mentions are later searched in Freebase KB for finding the corresponding entity names or alias matches. The entity candidates are all the entities whose name or alias are found from the entity mention in the KB, and relations candidates are all of the relations associated with the entity candidates. The question is transformed into question pattern by generating a structure in which the entity mention is replaced with symbol <s>. AR-SMCNN model finds the relation between question pattern and the relation candidates for finding answers.

Figure 5.2 depicts the entity detection introduced in the model. The query Q is passed through a BiLSTM. The BiLSTM classifies whether each word in the question belongs to entity mention or not. All the words which are detected as entities form a set C. The adjacent words in set C are combined to form substring S. The longest substring S generated is searched in Freebase to match the name or alias field. The set of Freebase entities are labelled as entity candidates E. If there are no direct entity names or aliases based on the longest substring in S, the string is expanded or narrowed with upmost 2 words around the query and searched further. If there is still no match found, each word in S is separately

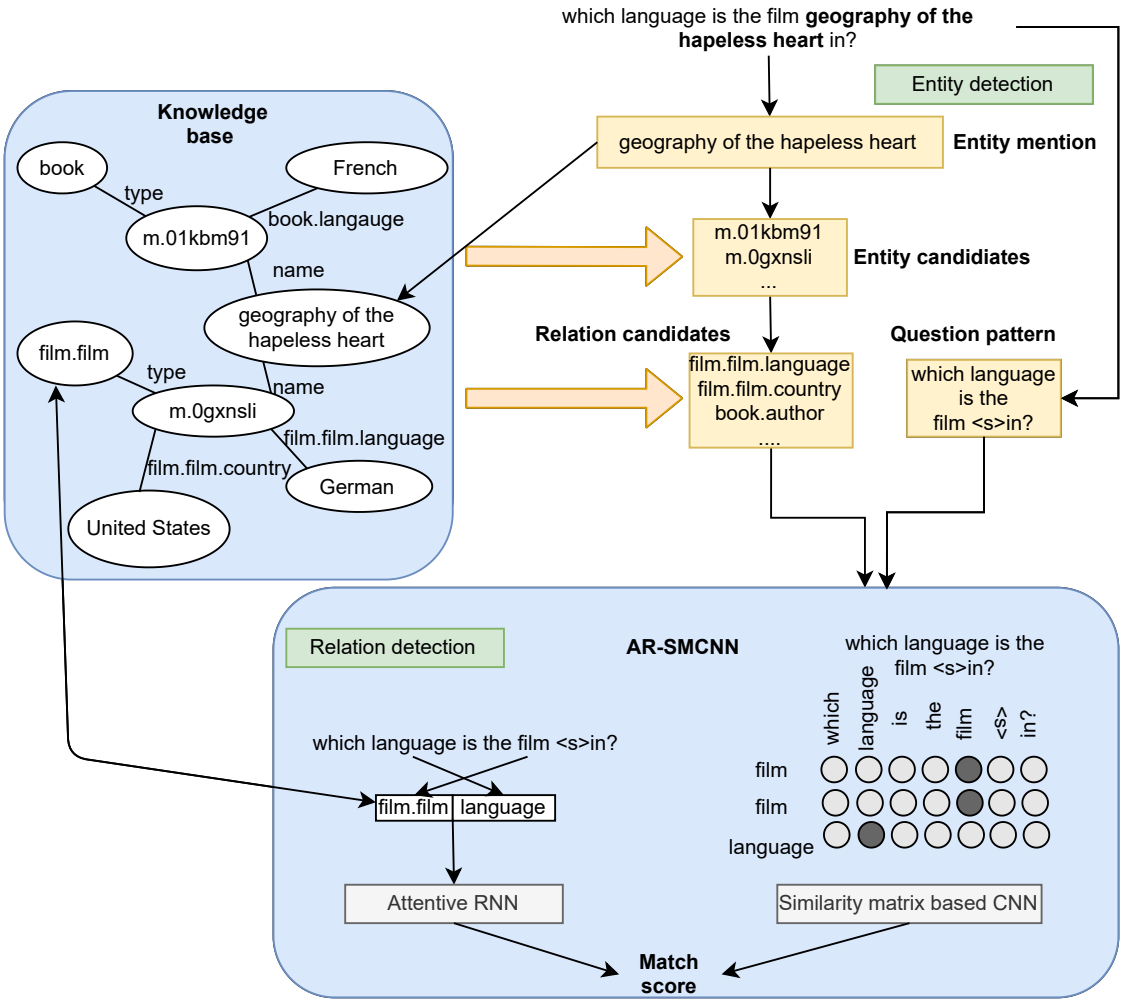

Figure 5.1: Attentive RNN with Similarity Matrix based CNN(AR-SMCNN). This is adapted from Qu et al. [QLK+18].

searched in Freebase. Set entity candidates E and set entity mentions X are then generated based on this search. The query is also transformed into a Query Pattern based on the entity mentions found.

Figure 5.3 shows the AR-SMCNN model in detail. For the generated question pattern P, each relation in the candidate relation is computed against a matching score function, which finds the correspondence between the two: relation candidates and question pattern. Semantic level and literal matching aspects were considered in the relation detection model.

As per the Figure 5.3, each relation in Relation candidates are encoded into two parts r_1 and r_2, that is, relation 'film.film.language' contains film.film, which is the first part: the type of the subject and 'language', which is the second part: the real relationship between

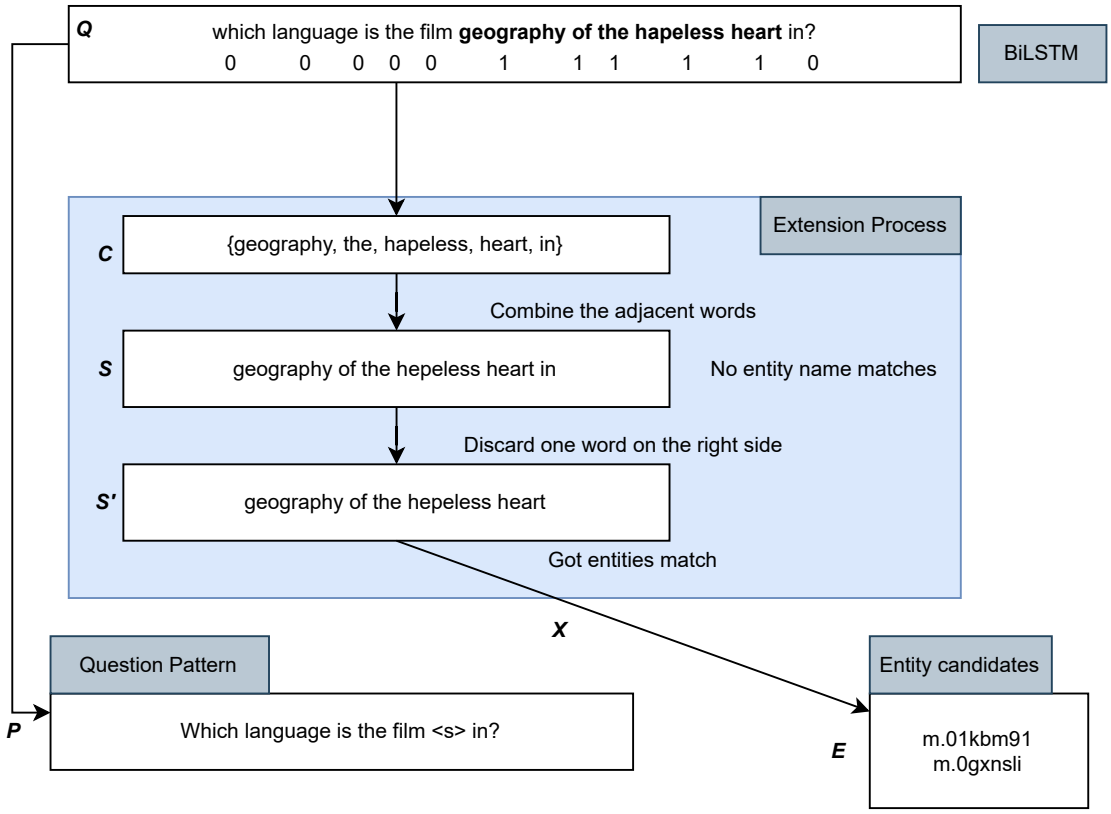

Figure 5.2: Entity detection used in AR-SMCNN model. This is adapted from Qu et al. [QLK+18].

the subject and answer. Word embeddings are then generated for each word in the question $\{q_1, ..q_L\}$. These embeddings are passed through a BiGRU, in order to generate the set of hidden states $H_{1:L}$. L represents the length of the question.

For each relation r_i, each question pattern representation p_i is as:

$$p_i = \sum_{j=1}^{L} \alpha_{ij} h_j \tag{5.1}$$

$$\alpha_{ij} = \frac{exp(w_{ij})}{\sum_{k=1}^{L} exp(w_{ik})} \tag{5.2}$$

$$w_{ij} = v^T tanh(W^T \left[h_j; r_i\right] + b) \tag{5.3}$$

where $i \in \{1, 2\}$, α_{ij} is the attention weight of j^{th} word in question in terms of r_i.

Parameters W and v need to be learned which is of the dimension $W \in R^{(c*(m+n))}$ and $v \in R^{(1*c)}$, where m is the dimension of r_i and n is the dimension of h_i.

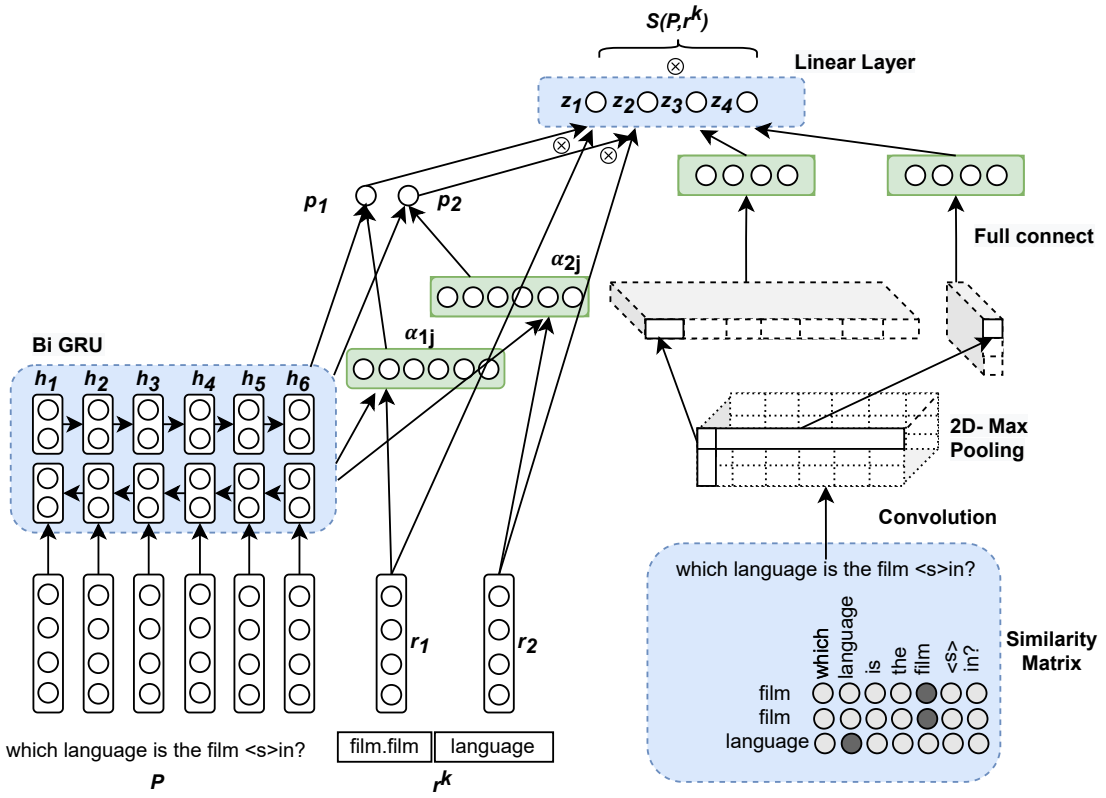

Figure 5.3: Relation detection used in AR-SMCNN model. This is adapted from Qu et al. [QLK+18].

For each r_i and p_i, similarity measure is calculated as the dot product $\otimes$ of the two vectors :

$$z_i = p_i \otimes r_i (i = 1, 2) \tag{5.4}$$

The literal level matching is retrieved by using the similarity matrix of the question pattern. Similarity matrix M is generated as the cosine similarity $\otimes$ of the i^{th} word embedding u_i in question and j^{th} word embedding v_j in relation:

$$M_{ij} = u_i \otimes v_j \tag{5.5}$$

A convolution layer is then employed to mine the word level and phrase level matching. It generates the feature map g^k from the similarty matrix M as output of the k^{th} kernel w^k. It can be mathematically denoted as follows:

$$g^k_{i,j} = \sigma(\sum_{s=0}^{r_k-1} \sum_{t=0}^{r_k-1} w^k_{s,t} \cdot M_{i+s,j+t} + b^k) \tag{5.6}$$

where r_k is the size of the k^{th} kernel.

The maximum matching feature is found out of the question and relation by a max pooling 2D layer. This is performed as follows:

$$y_i^{(1,k)} = \max_{0<=t<d_1} g_{i,t}^k \tag{5.7}$$

$$y_j^{(2,k)} = \max_{0<=t<d_2} g_{t,j}^k \tag{5.8}$$

Final feature is generated after a fully connected layer. Feautre z_3 and z_4 are computed with K the total number of kernels, $[y^{(i,0)}; y^{(i,K)}]$ as the concatenation of K pooling layer outputs, activation function as ReLU and weight of the i-th MLP layer, W_i as:

$$z_3 = W_2\sigma(W_1\left[y^{(1,0)}; y^{(1,K)}\right] + b_1) + b_2 \tag{5.9}$$

$$z_4 = W_2\sigma(W_1\left[y^{(2,0)}; y^{(2,K)}\right] + b_1) + b_2 \tag{5.10}$$

The two granularities are then combined in the final 4 feautures(z_1, z_2, z_3, z_4). The final linear layer is used to learn both the semantic features from z_1,z_2 and literal features from z_3,z_4:

$$S(P, r^k) = \text{Sigmoid}(W^T[z_1, z_2, z_3, z_4] + b) \tag{5.11}$$

For each question pattern P, the similarity matching score $S(P, r^k)$ is calculated for all possible relation candidates r^k and the prediction is the r^k with maximum matching score $S(P, r^k)$. This model is developed for single relation question answering and their original research was trained and tested on SimpleQuestions dataset against Freebase KB.

5.2 Model 2: Bidirectional Long Short Term Memory network (BiLSTM) and Gated Recurrent Unit (GRU)

Mohammed et al. [MSL18] explores the KBQA task over single relations using the basic LSTMs and GRUs. Through their paper, they compared various deep learning models for the task of question answering. The best accuracy is provided by the model which uses BiLSTM and GRU. Therefore, the model using BiLSTM and GRU is used as part of the thesis to compare with other models.

The model presents four steps for the question answering process task which includes :

1. Entity detection

2. Entity linking
3. Relation prediction
4. Evidence integration.

Figure 5.4 shows the overall architecture of the model proposed by Mohammed et al. [MSL18]. This model explores the semantic parsing of the QA, in which, the logical part of the question is mapped to a structured query format for solving the QA task. Firstly, the question is preprocessed by downcasing the question text and each word is tokenized. These tokenized words are transformed into 300 dimension vectors using pre-trained Glove 300 dimension word embedding. Entity detection in this model is designed as a neural network model which uses a recurrent neural network. For entity detection, RNN network used is a BiLSTM. The BiLSTM concatenates the hidden states in both directions, from forward and backward passes. This resultant is then passed to a linear layer and later on, batch normalization is performed. It is then passed through a ReLU activation function and dropout is performed. Lastly, the functions on the hidden states are then transformed into tag space using a linear layer. Each word token in the question is classified whether an entity or not, from the output vector in tag space.

The output vectors in the tag space after the entity detection is passed to the entity linking step. The entity linking step is not a neural network model. It contains a pre-built inverted index of n-grams on the Freebase KB entities. The candidate entity is then searched in the pre-built n-gram for all entity name matches. The inverted index returns the corresponding candidates MIDs from the Freebase. These matches are then scored based on the Levenshtein distance to the golden canonical label of MID. Levenshtein distance is a metric used to find the difference in string sequences.

The model proposes an independent relation prediction model. It is designed as an RNN network, similar to entity detection. BiGRU is the specific RNN used in the model. The relation classification decision is based on the hidden states of the final token. The output of the BiGRU is passed on to a linear layer, batch normalized, ReLU activation, dropout and a linear layer, similar to an entity detection model. The final linear layer maps the operations performed in the hidden space to the target relation space and predicts the corresponding relation.

As shown in Figure 5.4, entity prediction/entity linking and relation prediction is performed separately on the question text, so this need to be integrated to predict the correct entity and relation and eventually correct answer. Let m and r represents the top entities and top relations from the question text. The evidence integration module firstly computes the $m \times n$ tuples and their corresponding scores as the product of individual components. This is then pruned to remove the relations that are not existing for a particular entity. Further, the scores are analyzed to remove ranking ties in values. A heuristic method is used for the same, ie, the entities with more incoming edges are considered more important in scoring. Entities containing Wikipedia entry in the Freebase are also considered relevant than other counterparts. These heuristic methods are employed based on the knowledge base structure, the lexical structures of entities in KB are not considered.

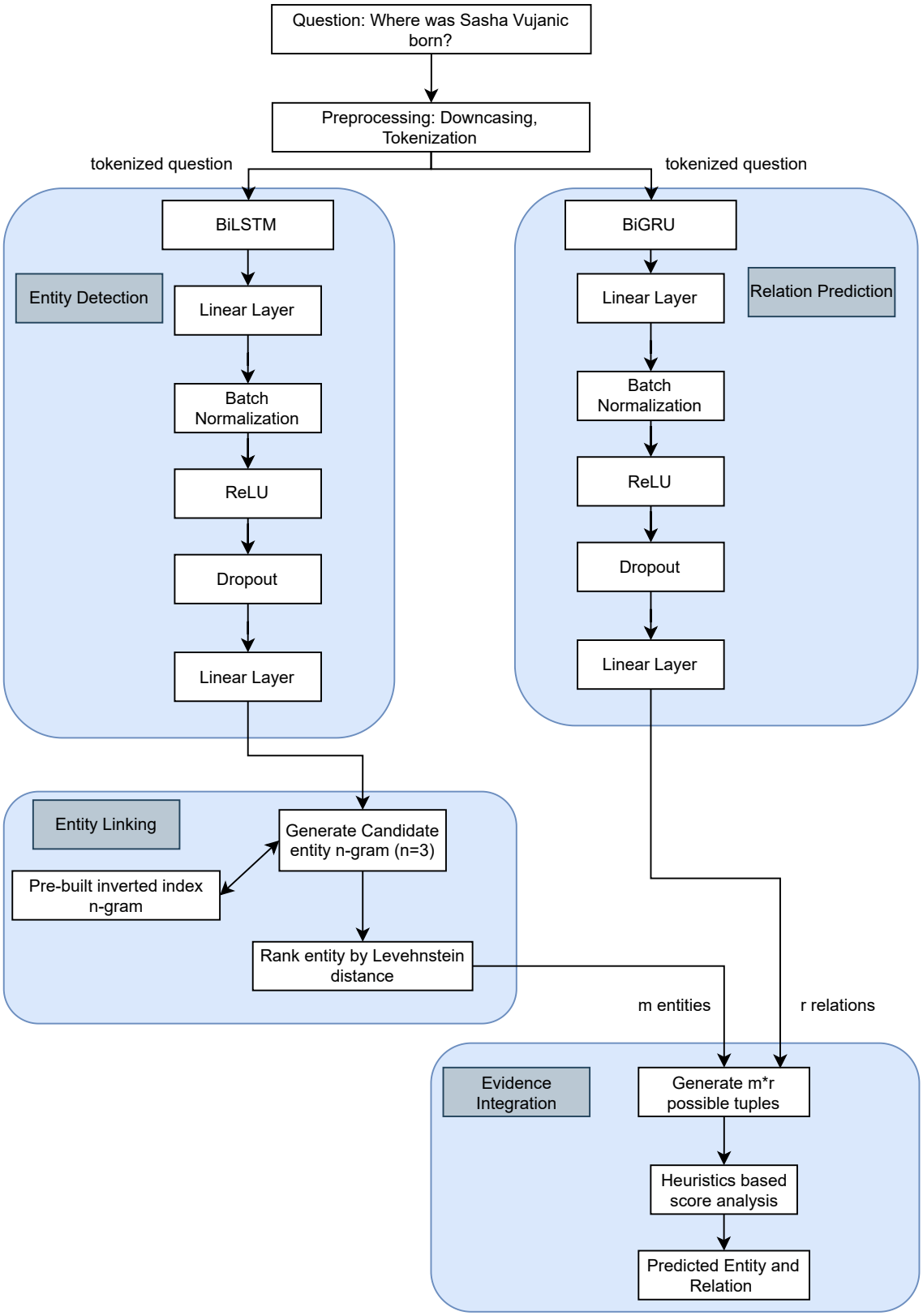

Figure 5.4: Model architecture using BiLSTM for entity prediction and BiGRU for relation prediction. This diagram is generated based on Mohammed et al. [MSL18]

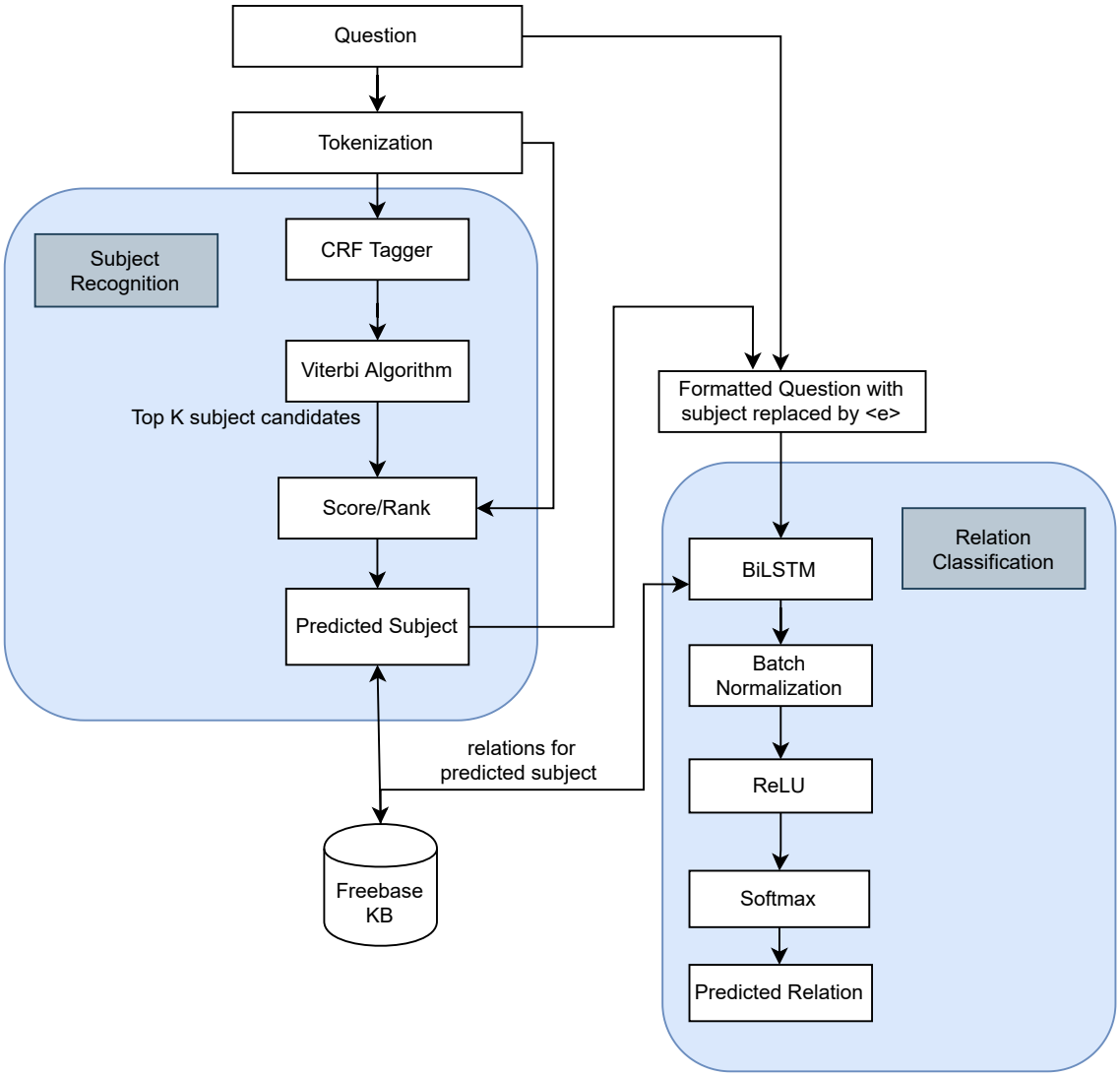

Figure 5.5: Model architecture using CRF tagger for subject prediction and BiLSTM for relation prediction. This diagram is generated based on Petrochuk and Zettlemoyer [PZ18]

5.3 Model 3: Conditional Random Field (CRF) and Bidirectional Long Short Term Memory network (BiLSTM)

Petrochuk and Zettlemoyer [PZ18] uses two concepts in their paper for the SimpleQuestion Answering task. The two of them are:

1. a CRF tagger for subject identification in the question
2. a BiLSTM classification for relation prediction

Figure 5.5 shows the overall architecture of the model. For a given question q, subject recognition model models it as a probability distribution, $P(a|q)$ which ranges over the correct text A over question q. A constitutes of set of the correct subjects in the question. $P(a|q)$ is realized using CRF. Secondly, the relation classification is also modelled as a probability distribution $P(r|q, a)$, which is realized using an LSTM.

The question is firstly tokenized and passed to the linear chain CRF tagger. Conditional log likely hood loss objective is used with the CRF tagger to generate the probability distribution of subject within the question text. These predicted tags are later on passed to k best Viterbi algorithm which finds the most likely top k tags. It is a dynamic programming algorithm that helps in predicting the k probable candidates. The top k candidates are checked against the question tokens to generate a score. The score is calculated based on the Levenshtein distance between the candidates and individual tokens. This score is used as the criteria for subject prediction.

The relations corresponding to the predicted subject are then retrieved from KB. This act as one of the inputs to the BiLSTM in relation prediction module. The question is then formatted with the predicted subject as abstract predicate string <e> and used further as an input to the BiLSTM. The output of the BiLSTM is batch normalized, then passed through ReLU activation function and a softmax to generate the relation. The predicted subject and relation can be further computed to generate the answer for a particular function by querying the KB.

5.4 Model 4: Bidirectional Attentive Memory Networks (BAMnet)

Chen et al. [CWZ19] put forward the concept of two-layered bidirectional attention network. It comprises of two networks, one primary and another secondary network. The primary attentive network captures the relevant aspects of the question with reference KB and relevant aspects of KB with reference to the question. The secondary network enriches the representations of KB and questions by using two-way attention.

Figure 5.6 explains the Bidirectional attentive network architecture. For a given question $Q = q_i{}_{i=0}^{|Q|}$, word embedding layer is used to generate a sequence of word embeddings (q_i). This is then passed to a BiLSTM to yield H^Q which is the concatenation of hidden states. Memory module takes set of candidate answers as the input $\{A_i\}_{i=0}^{|A|}$, which represents all the entities that are h hop near to the topic entity of the question. Every answer candidate A_i contains three information, namely, Answer type, Answer path and Answer context. Answer type encodes the type information of the entity, which is passed through a BiLSTM to derive d-dimensional vector, $H_i^{t_1}$. Answer path comprises of the candidate relations that are linked to the topic entity. It is depicted using two vectors, $H_i^{p_1}$ via a BiLSTM and $H_i^{p_2}$ as average of the relation embedding through the relation embedding layer. Answer context is the collection of nearby entities which are used for answering the questions with

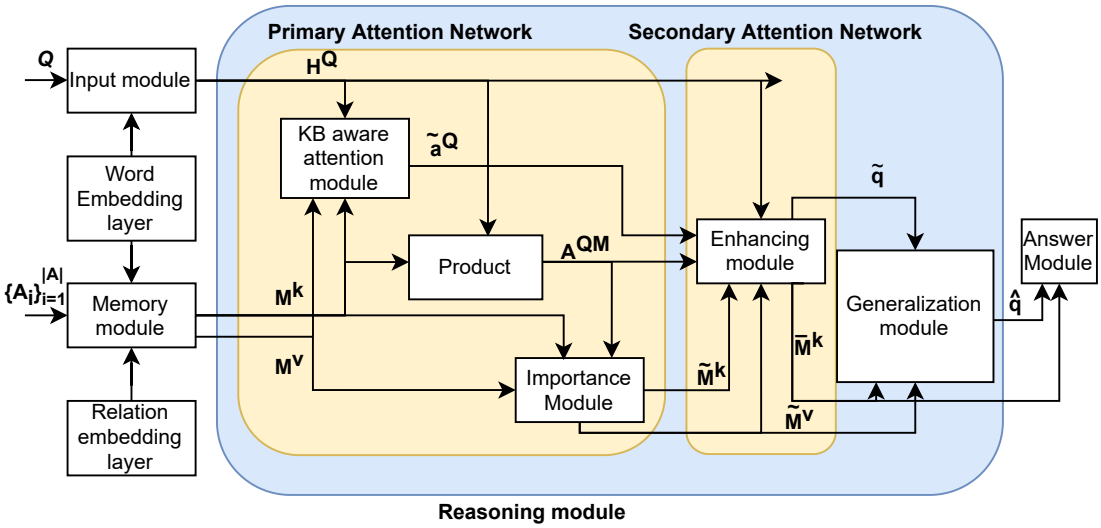

Figure 5.6: Bidirectional Attentive Memory Networks Model architecture. This diagram is adapted from Chen et al. [CWZ19]

constraints. Firstly, the longest common subsequence between the context entities and question is calculated. This is then passed to a BiLSTM if a non-stop word substring is contained in the subsequence. The average of all such entity representations is taken as the answer context H_i^c. The answer candidates are then stored via a key-value memory network[MFD+16]. Each component (path, type and context) is separately passed through the key-value network to generate M in the key-value memory as: $M_i = \{M_i^k, M_i^v\}$. Here $M_i^k = [M_i^{k_t}; M_i^{k_p}; M_i^{k_c}]$ and $M_i^v = [M_i^{v_t}; M_i^{v_p}; M_i^{v_c}]$ contains $[;]$ which is column-wise concatenation operation. $M_i^{k_x}$ are computed by linear transformations for each x, ie, text, path and context. Similarly, $M_i^{v_x}$ are computed by linear transformations for each x, ie, text, path and context of the answer.

The reasoning module as in Figure 5.6 contains the primary and secondary attention network. KB aware attention module is the first component in primary network. It tries to analyze the relevant parts of the question by self attention over H^Q. It is calculated as follows:

(5.12) $A^{QQ} = softmax((H^Q)^T H^Q)$

(5.13) $q = BiLSTM(\left[H^Q A^{QQT}, H^Q\right])$

The q is a d-dimensional vector which is the question summary obtained from KB aware module. An additive attention is then applied over the memory, which has text, path and context components. Additive attention over each component is separately calculated as follows:

$$a_x = Att_{add}(q, M^{k_x}) \tag{5.14}$$

$$m_x = \sum_{i=1}^{|A|} a_i^x \cdot M_i^{v_x} \tag{5.15}$$

where $x \in \{t, p, c\}$, t is the text, p is the path, c is the context and $Att_{add}(x, y) = softmax(tanh(\left[x^T, y\right]W_1)W_2)$ and W_1, W_2 are trainable weights. m will contain KB enhanced answer candidates with respect to question for each specific component. Question enhanced KB properties can be calculated $A^{Qm} = H^{Q^T} m$. A max pooling is applied on the A^{Qm} which selects the most important connection, $a_i^Q = \max_j A_{ij}^{Qm}$. A softmax is later applied to generated $\tilde{a}^Q$ which is the KB-aware question attention vector.

The second main component in the primary network is the importance module which brings in the relevant KB aspects by their importance to question. Attention tensor A^{QM} is created which tries to find the advantage of the connection between each pair of question words and candidate answers. A max function is applied on the resultant and later on normalized to $\tilde{A}^M$. Mathematically it can be shown as:

$$A^{QM} = (M^k H^Q)^T \tag{5.16}$$

$$A^M = \max_i \{A_i^{QM}\}_{i=1}^{|Q|} \tag{5.17}$$

$$\tilde{A}^M = softmax(A^{M^T})^T \tag{5.18}$$

The question aware memory representations $\tilde{M}^k$ and $\tilde{M}^v$ are then computed separately as follows:

$$\tilde{M}_i^k = \tilde{A}_i^M M_i^k \tag{5.19}$$

$$\tilde{M}^k = \{\tilde{M}_i^k\}_{i=1}^{|A|} \tag{5.20}$$

$$\tilde{M}_i^v = \sum_{j=1}^{3} M_{ij}^v \tag{5.21}$$

$$\tilde{M}^v = \{\tilde{M}_i^v\}_{i=1}^{|A|} \tag{5.22}$$

The secondary attention network consists of enhancing module and generalization module. Enhancing module computes KB enhanced question representation $\tilde{q}$ and question enhanced KB representation $\tilde{M}^k$. For $\tilde{q}$, max pooling is performed on the A^{QM} that is $A_M^Q = \max_k\{A_k^{QM}\}_{k=1}^3$. A_M^Q is normalized to determine the $\tilde{A}_M^Q$ to get question-to-KB attention matrix. The hidden state representation of $\tilde{H}^Q = H^Q + \tilde{a}^Q \odot (\tilde{A}_M^Q \tilde{M}^v)^T$ is then evaluated before computing the $\tilde{q}$, which is the d-dimensional KB-enhanced representation of the question as follows:

$$\tilde{q} = \tilde{H}^Q \tilde{a}^Q \tag{5.23}$$

Question enhanced KB representation is computed as follows:

$$\tilde{A}_Q^M = softmax(A_M^{Q\,T}) \tag{5.24}$$

$$\tilde{a}^M = (\tilde{A}_M^Q)^T \tilde{a}^Q \tag{5.25}$$

$$\bar{M}^k = \tilde{M}^k + \tilde{a}^M \odot (\tilde{A}_Q^M (\tilde{H}^Q)^T) \tag{5.26}$$

In the generalization module, one hop attention module is applied to the question representation $\tilde{q}$ and KB enhanced question representation $\bar{M}^k$.

$$a = Att_{add}^{GRU}(\tilde{q}, \bar{M}^k) \tag{5.27}$$

Appropriate data from value memory is also extracted:

$$\tilde{m} = \sum_{i=1}^{|A|} a_i \cdot \tilde{M}_i^v \tag{5.28}$$

A GRU layer and batch normalization is performed:

$$q' = GRU(\tilde{q}, \tilde{m}) \tag{5.29}$$

$$\hat{q} = BN(\tilde{q} + q') \tag{5.30}$$

The answering module, which predicts the answer, calculates the similarity score$S(\hat{q}, \bar{M}^k)$ between the $\hat{q}$ and the question enhanced KB representation $\bar{M}^k$. Similarity score is determined for each pair of answer candidate and the question using as $S(q, a) = q^T \cdot a$. The answer candidates are ranked based on the similarity score.

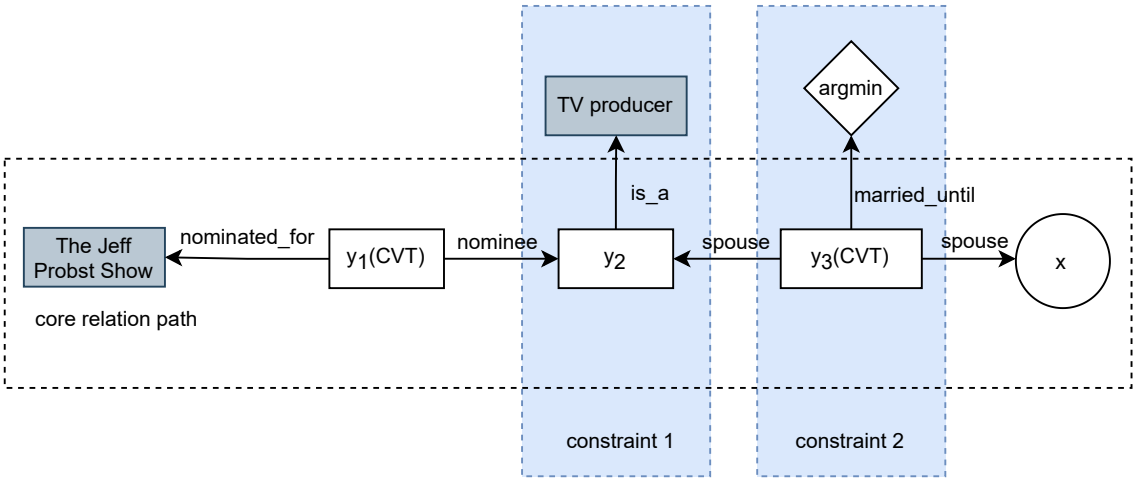

Figure 5.7: Query Graph. This diagram is adapted from Lan and Jiang [LJ20].

5.5 Model 5: Query Graph (Multi-hop Complex KBQA)

Lan and Jiang [LJ20] proposes a model based on staged query graph generation for multi-hop KBQA. It is based on the semantic parsing method, in which it generates a query graph that can be executed on KB. The method uses the query graph as described in Figure 5.7. A query graph comprises of four types of nodes: grounded entity, existential variable, lambda variable and an aggregate function. The entity from the KB is called grounded entity and it is shaded grey rectangle in the figure. Other rectangles which are non-shaded are not present in the KB and are non grounded entity called as the existential variable. Functions like $count$ and $argmin$ are functions that can be performed on the entities are called aggregate functions. The answer to that particular function is called the lambda variable. Therefore, the query graph should have exactly one lambda variable, at least one or more ground entity and existential variable may or may not be present.

Staged query graph generation consists of mainly 4 steps. Firstly, the core relation from grounded entity found from the question is extracted to find a relation to the lambda variable. Secondly, the constraints are identified from the question, which has a link with a grounded entity or aggregate function. As a third step, the identified candidate query graphs are ranked or scored with the help of a neural network model using CNN. Lastly, the top-ranked query graph is executed against the KB.

The model employs beam search iteratively to generate the query graphs. The model generates a set of K query graphs at the t-th iteration, which are denoted as G_t. In the next iteration, it tries to add a modification to the graphs in G_t using the aggregate functions: $\{extend, connect, aggregate\}$. This is repeated for all graphs in G_t. All the resulting graphs in the t + 1-th iteration are ranked and top-K are selected for G_{t+1}. This iteration is discontinued when there is no increase in the scores after each iteration. The $extend$ operation increase the graph by adding a core relation path. $connect$ function tries to link

the grounded entity with the lambda variable or to another existential variable, which is a CVT node. $Aggregarte$ function is added on to the graph when aggregation is detected to a lambda variable or existential variable, which is a CVT node, that is connected to the same lambda variable.

Graph ranking score is calculated by using a 7-dimensional vector derived for each graph and then passed to a fully connected layer and softmax. A BERT based semantic matching model is used to generate the 1st dimension of this 7-dimensional vector, in which the graph is converted to a sequence of tokens based on the actions performed on the graph during the construction. The textual content of the entities and relations involved at each step are then added to the sequence. The second dimension is the entity linking scores of each of the ground entities of the query graph. The third one is the total number of ground entities in the graph. The fourth dimension is the number of types of entities. The fifth one incorporates the expressions containing temporal constraints. The sixth dimension contains the superlative constraints in the query graph. The last one depicts the number of lambda nodes in the graph, ie, the number of answers. The model is trained using the reinforcement learning algorithm by learning the policy function. The policy function for a given question is a representation of query graph in terms of parameters, which is defined using the BERT parameters [DCLT19] and seven-dimensional vector for each query graph. BERT is Bidirectional Encoder Representations from Transformers. It is a language representation model that is used for NLP tasks. In this model, the pre-trained BERT model is used and then learned further as part of the model training. The F1 score is used as the reward in the reinforcement algorithm to iteratively learn the policy function.

6 Evaluation

This chapter evaluates various models and provides the results. Firstly, the five models were evaluated against the SimpleQuestions dataset on FB2M KB. For comparison of the models, accuracy and F1 score were considered as the main measure of comparison. Also, various machine learning aspects were analyzed to understand and to evaluate each model. The models were compared on their performance when trained with reduced data. The machine learning aspects were considered along with the error analysis, to understand the possible areas of improvement.

6.1 Accuracy

Accuracy is a measure that can help to understand how well a system can predict the correct answer. The technical details on how it is calculated can be found in Section 3.3.1. Table 6.1 provides the exact values on how each of the compared models performed on SimpleQuestions dataset against Freebase 2M KB. Model 1 on their research paper gave an overall accuracy of 77.9 [QLK+18], whereas, this research reported accuracy of 70.45% on FB2M dataset. Model 2 on their research paper compared various neural network models. On that, the best accuracy was reported with an architecture using BiLSTM and GRU [MSL18]. As part of this study the model reported 74.68% accuracy, whereas, their paper reported accuracy of 74.9%. Model 3 possessed the best accuracy, their research paper reported 78.1% [PZ18] and this research resulted in an accuracy of 78.2%.

Model 4 and Model 5 were not tested in their respective research papers against this SimpleQuestions dataset. Model 4 and Model 5 were developed for multi-relation question answering and hence did not perform well with regard to accuracy. These models had to be adapted to work with SimpleQuestions dataset. The lower accuracy could be explained by the constraints and relations introduced in Model 4 and Model 5. Model 4 incorporates answer context information in candidates which may not be required for answering simple questions. The answer context contains the nearby nodes which are not required for single relation question answering. Model 5 uses query graph generation, that includes the other nodes in the graph which predicts the answers by analyzing the constraints in the question. Although Model 4 and Model 5 performed poorly with respect to the accuracy, they performed relatively better in terms of F1 score. This will be further discussed in the next section.

Model	Accuracy
1. Attentive RNN with Similarity Matrix based CNN	70.44
2. Bi-LSTM and GRU	74.68
3. CRF and BiLSTM	78.20
4. Bidirectional Attentive Memory Networks	46.04
5. Query Graph	47.54

Table 6.1: Accuracy comparison for different models

Model	Precision	Recall	F1 score
1. Attentive RNN with Similarity Matrix based CNN	0.73	0.82	0.73
2. Bi-LSTM and GRU	0.42	0.87	0.48
3. CRF and BiLSTM	0.88	0.22	0.35
4. Bidirectional Attentive Memory Networks	0.57	0.90	0.62
5. Query Graph	0.57	0.70	0.60

Table 6.2: F1-score comparison for different models

6.2 F1 score

F1 score comparison is one of the methods for evaluating deep learning models. Section 3.3.2 explains how F1 score is computed. Table 6.2 provides the result of the comparison of various models with respect to their average F1 performance. Model 1 resulted in an F1 score of 0.73 on SimpleQuestions dataset. That model showed a precision of 0.73 and a recall of 0.82 which resulted in the best F1 score among the compared models. This indicates that true predictions are higher and false predictions are lower.

Model 2 reported a good recall of 0.87 but poor precision of 0.42. The overall F1 score was reported at 0.48. This indicates that false positives are higher. Model 3 had a good precision of 0.88, but the recall was reported 0.22, which resulted in a very low F1 score of 0.35. This indicates that false negatives are highly predicted by the model.

Model 4 and Model 5 reported good F1 scores of 0.62 and 0.60 respectively. Model 4 reported precision of 0.57 and a recall of 0.87. Model 5 reported precision of 0.57 and a recall of 0.70. Both models Model 4 and Model 5 performed well although the models were primarily focused on multi relation and multi constraint question answering.

Model	100%	10%
1. Attentive RNN with Similarity Matrix based CNN	70.44	65.70
2. Bi-LSTM and GRU	74.68	67.65
3. CRF and BiLSTM	78.20	73.73
4. Bidirectional Attentive Memory Networks	46.04	47.51
5. Query Graph	47.54	47.00

Table 6.3: Accuracy comparison for different models that are trained with 10% data

	100%			10%		
Model	**P**	**R**	**F1**	**P**	**R**	**F1**
1. AR-SMCNN	0.73	0.82	0.73	0.73	0.82	0.73
2. Bi-LSTM and GRU	0.42	0.87	0.48	0.44	0.79	0.45
3. CRF and BiLSTM	0.88	0.22	0.35	0.87	0.22	0.35
4. BAMnet	0.57	0.90	0.62	0.51	0.81	0.56
5. Query Graph	0.57	0.70	0.60	0.57	0.69	0.60

Table 6.4: F1 comparison for different models that are trained with 10% data. P, R and F1 represents Precision, Recall and F1 score respectively.

6.3 Training with less data

It was of interest to observe the performance of the models against a reduced data set. Such analysis helps to select the model that works with reduced data, as in some cases, training data may be limited. Accuracy and F1 scores were reported when trained with fewer data. 10% of the data was used for training and validation. This trained model was tested against full test data for comparison. Table 6.3 shows the accuracy of the compared models.

Model 1 and Model 3 reported the accuracies reduced by approximately 5% each, i.e., accuracies of 65.70% and 73.73% respectively. Model 2 reported an accuracy of 67.65% which was 7% lesser than the training accuracy on full data. Model 4 resulted in an accuracy of 47.51%, which was an increment than that of the full data set. Model 5 reported a slight decrease of 0.54%, i.e., 47%. Table 6.4 shows the average F1 scores of various models with fewer data.

For Model 1, Model 3 and Model 5, the F1 scores remained the same with reduced data however for Model 2 showed a slight reduction in F1 score. For Model 4, F1 score reduced from 0.616 to 0.56. Therefore, Model 1 proved to be a good model with consistent F1 scores and showing only a slight reduction in accuracy.

6.4 Error Analysis

Error analysis is useful in understanding which areas of the model introduce maximum errors. The result of the error analysis can be used to tune the model and reduce the overall error percentage.

Model 1, Model 2 and Model 3 perform the QA task in a two-step process, whereas, Model 4 and Model 5 were adapted from their original research to predict the answer for the questions, by using the topic entity mentioned in the SimpleQuestions dataset. Therefore, error analysis was performed separately for both kinds of models. The first three models were analyzed to understand the error distribution among the two steps. For Model 4 and Model 5, an eyesight sample of errors was analyzed to understand which all categories of errors were created by them.

Table 6.5 provides the details on where the errors get introduced in the system for Model 1, Model 2 and Model 3. The total number of errors in each model were classified into three categories based on where the errors were reported. For Model 1, the number of errors where relation detection was wrong were higher, and hence, it shows that relation prediction of that model should be made a focus while improving the model. On the contrary, Model 2 and Model 3 had more errors introduced in the entity detection module. The exact number of errors and the three subdivision is given in the next paragraph.

Model 1 had a total of 7158 wrong answers, of which, number of entity detection wrong and relation detection correct was 1776. In 2270 cases, entity detection were correct and relation detection were reported wrong. Both modules being wrong were 3112 cases. Therefore, this shows that more errors were introduced in relation detection. For Model 2, out of 5491 total wrong answers, the number of entity detection wrong and relation detection correct were 2059 cases. Both entity detection wrong and relation detection were wrong in 2279 cases. The number of cases, in which, entity detection were correct and relation detection were wrong, were 1153. This shows that more errors were introduced in the entity detection phase for Model 2. For Model 3, there were a total of 4770 wrong answers. Number of entity detection wrong and relation detection correct were reported for 2191 cases and both modules were wrong for 1511 cases. In 1068 cases, entity detection were correct and relation detection were reported wrong. Therefore, more errors were introduced in entity detection. The error percentages are reported in Table 6.5 for clarity.

Unlike the previous 3 models, the error analysis for Model 4 was performed differently. Out of the test data, an eyeball set of samples were taken out to analyze the errors. 100 sample test questions with individual F1 scores less than 1 were analyzed. Out of the 100 samples, in 30 cases the wrong relation was captured by the model for answering the question. For the question, 'What did Fred Schwarz write?' the model predicted 'author' as the answer, whereas, 'communism' is the correct answer. In 9 cases, the prediction was very generic. Out of these for a question, 'Where is MVM arts and science college at?' it predicted 'India', whereas, the golden answer was specific city 'Dindigul'. In 9 cases, the model predicted partial wrong answers. For question, 'What is the genre for the film Cow

Model 1: Attentive RNN with Similarity Matrix based CNN	Error distribution
Both Entity detection and relation prediction wrong	43.47%
Entity detection correct and relation prediction wrong	31.72%
Entity detection wrong and relation prediction correct	24.81%
Model 2: Bi-LSTM and GRU	**Error distribution**
Both Entity detection and relation prediction wrong	41.52%
Entity detection correct and relation prediction wrong	20.99%
Entity detection wrong and relation prediction correct	37.49%
Model 3: CRF and BiLSTM	**Error distribution**
Both Entity detection and relation prediction wrong	31.68%
Entity detection correct and relation prediction wrong	22.38%
Entity detection wrong and relation prediction correct	45.94%

Table 6.5: Error analysis for model 1, 2 and 3 in percentage

Belles?' the model predicted 'family' and 'television', whereas, 'family' was the correct answer. In the remaining 52 cases, the model predicted multiple correct answers. The model predicted for the question 'What kind of film is Días de combate?' as 'crime fiction', 'mystery', 'thriller' whereas the golden answer contains only 'thriller'.

Similarly, for Model 5, which uses query graph generation technique, an eyeball set of samples were analyzed. A sample of 100 questions was analyzed to understand the common errors predicted by the model. These sample questions selected were having individual F1 scores less than 1. Out of the 100 samples, 45 answers were wrongly predicted. For the question, 'What artist influenced Ellsworth Kelly?', the model predicted 'painting' and 'sculpture', whereas, the golden answer was 'Pablo Picasso'. In 9 cases partially wrong answers were predicted. The question 'What state is Riegelwood in?' was answered 'United States of America', 'North Carolina' and 'Columbus county', for the correct answer 'North Carolina'. And for 46 cases multiple correct answers were predicted, i.e., for the question, 'What genre is the movie Half empty?' the model predicted 'comedy-drama' and 'crime fiction', whereas, 'crime fiction' is the golden answer.

6.5 Machine learning aspects

Model 1 retained the same F1 score when trained with both 100% data and 10% data. The ablation study performed by Qu et al. [QLK+18] shows the relevance of attention mechanism in Model 1. It enhances the performance in single relation question answering.

Model	Total trainable parameters
1. Attentive RNN with Similarity Matrix based CNN	43,112,779
2. Bi-LSTM and GRU	8,065,105
3. CRF and BiLSTM	22,851,713
4. Bidirectional Attentive Memory Networks	128,861,856
5. Query Graph	66,956,553

Table 6.6: Total trainable parameters used in the various models.

Apart from attention, the model makes use of both RNN and CNN for solving the KBQA task. Table 6.6 provides the number of trainable parameters in each network. Model 1 had a third highest number of total trainable parameters among the compared models.

Model 2 is the model with the least number of trainable parameters. This model specifically separates the entity detection and relation detection. Afterwards, the results are combined from both the steps incorporating heuristics. As both steps are processed independently, there may be many relations that are not relevant to a particular entity. Heuristics includes, the relevance of predicted entity by incorporating data on Wikidata KB also proved to improve the prediction as explained in Section 5.2. It may not be a generalized model, even though, it performed second best with respect to accuracy. It performed poorly with respect to F1 score with a high false positive rate.

Model 3 implements hyperparameter optimization, in which, various hyperparameters used in the models are tested and the one with best results are used. Model 3 performed as a good accurate and precise system in the comparative study. However, because of high false negative cases predicted by the model, it did not perform well with respect to F1 score. It had the second least trainable parameters among the compared models and remained consistent with regard to F1 score when the model was trained with less data. Model 1, Model 2 and Model 3 were specifically developed in their researches for solving the single relation question answering. The real world problems comprises of more complicated cases in QA. So how well these models work on other datasets was not explored.

Of all the models compared, Model 4 contained the most complex networks. The network architecture of Model 4 is the most complex of all the compared models and it contains the most number of trainable parameters. Also, it is an information retrieval method with least or minimal handcrafted features. This model utilizes the attention mechanism for KBQA task. The ablation study performed on WebQuestions dataset in their orignal research shows the relevance of attention mechanism[CWZ19]. The performance with regards to accuracy and F1 score was slightly less for this model as the model incorporates extra information in the form of answer context.

The time complexity of the models could not be compared as various models uses different storage options for the Freebase2M KB. But of the compared models, Model 5 took the most time for training. This might be because of the reinforcement learning used during

the training as explained in Section 5.5. This model has the second-highest number of trainable parameters as it uses pre-trained BERT parameters and during training, these parameters are fine-tuned. Lan and Jiang [LJ20] performed ablation study in their research and proved that this model not only performs well by using BERT, but also with LSTM on ComplexWebQuestions dataset. This model included contraints in the query graph which is suitable for complex question answering task. This did not help in achieving a higher performance when tested on SimpleQuestions dataset, however, it performed consistently when trained with reduced data.

7 Conclusion and Outlook

Conclusion

The comparative study provides several insights on the single relation question answering over KB. Model 1, Model 2 and Model 3 were addressing the single relation question answering, whereas, Model 4 and Model 5 were adapted as part of this study, in order to make them suitable for the single relation question answering.

Model 1 performed the best with respect to F1 score and provided a relatively good accuracy, whereas, Model 3 performed the best with respect to Accuracy. Considering both these aspects together, Model 1 is the preferred model for answering single relation questions.

In their original researches Model 4 and Model 5 performed well with respect to multi relation question answering. For single relation question answering, they produced relatively good F1 scores. Hence for a broader question answering system, Model 4 and Model 5 could be considered. However, Model 4 uses the most complex neural network architecture and it requires the least handcrafted features of all the models. Hence Model 4 is favourable than Model 5.

Most of the real-world problems involve answering questions with constraints. Model 1, Model 2 and Model 3 lack support for the multi relation question answering. Hence the adaptability of these models must be explored further before positioning them in the real world scenarios. Heuristics applied in Model 2 is suitable for solving single relation question answering against Freebase KB, however, it may not be a generalized solution suitable for all situations. The time complexity of the models could not be analyzed as each of the models used different storage options for the KB. Some models used file-based storage options where others used database options.

SimpleQuestions dataset was considered for this comparative study since it is the benchmark dataset for the single relation question answering. The dataset contains only a single ground truth answer for each question. However, it was observed during error analysis that the SimpleQuestions dataset contains many questions which can have multiple correct answers. The erroneous data in the dataset may have a slight impact on the results of the models.

Outlook

This comparative study intended to provide a basic understanding of current researches, that are happening in the field of single relation question answering over knowledge bases. Most of the real-world problems in question answering require systems which could handle the aspects of multihop and multi-relation question answering. This comparative study could be further extended to cover these aspects. Datasets like WebQuestions dataset, QALD dataset etc. could be used for comparing the multi-relation question answering systems.

This study was limited to Freebase 2M knowledge base, but, Freebase has been shut down, and hence, the data in KB is not up to date. The API support is unavailable, which adds limitations in exploring the real-time performance of QA systems on Freebase KB. As an alternative, the Wikidata KB could be used to analyze the models. Such studies on different KBs may help to understand and analyze how differently the models behave.

A Implementation details

All models were implemented using Python version 3 and executed on GPUs using CUDA version 10.2. Specific packages and databases used are mentioned below:

A.0.1 Model 1

- PyTorch - used for training, testing and prototyping of the model.
- NLTK module - for NLP related operations in the model.
- Virtuoso - for storage of Freebase 2M data.

A.0.2 Model 2

- PyTorch - used for training, testing and prototyping of the model.
- NLTK module - for NLP related operations in the model.
- Torchtext package - for generic data processing.
- fuzzywuzzy package - for computing string differences using Levenshtein distance.

A.0.3 Model 3

- PyTorch - used for training, testing and prototyping of the model.
- NLTK module - for NLP related operations in the model.
- Pandas package - for data processing.
- Allennlp - NLP library for implementing CRF tagger.
- Jupyter notebook - for web based analysis of the model.
- PostgreSQL database - for Freebase data storage.

A.0.4 Model 4

- PyTorch - used for training, testing and prototyping of the model.
- NLTK module - for NLP related operations in the model.
- rapidfuzz package - for performing string matching.

A.0.5 Model 5

- PyTorch - used for training, testing and prototyping of the model.
- SPARQLWrapper module - for interfacing with the database.
- Virtuoso - for storage of Freebase 2M data..

Bibliography

[AMA17] S. Albawi, T. A. Mohammed, S. Al-Zawi. "Understanding of a convolutional neural network". In: *2017 International Conference on Engineering and Technology (ICET)*. IEEE. 2017, pp. 1–6 (cit. on p. 19).

[BEP+08] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor. "Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge". In: *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*. SIGMOD '08. Vancouver, Canada: Association for Computing Machinery, 2008, pp. 1247–1250. ISBN: 9781605581026. DOI: 10.1145/1376616.1376746. URL: https://doi.org/10.1145/1376616.1376746 (cit. on p. 27).

[BH15] H. Bast, E. Haussmann. "More accurate question answering on freebase". In: *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 2015, pp. 1431–1440 (cit. on p. 16).

[BUCW15] A. Bordes, N. Usunier, S. Chopra, J. Weston. "Large-scale simple question answering with memory networks". In: *arXiv preprint arXiv:1506.02075* (2015) (cit. on pp. 27, 28).

[Bul13] J. A. Bullinaria. "Recurrent neural networks". In: *Neural Computation: Lecture* 12 (2013) (cit. on p. 20).

[CCC+19] Z.-Y. Chen, C.-H. Chang, Y.-P. Chen, J. Nayak, L.-W. Ku. "UHop: An Unrestricted-Hop Relation Extraction Framework for Knowledge-Based Question Answering". In: *arXiv preprint arXiv:1904.01246* (2019) (cit. on p. 17).

[Cha18] N. Chah. "OK Google, What Is Your Ontology? Or: Exploring Freebase Classification to Understand Google's Knowledge Graph". In: *arXiv preprint arXiv:1805.03885* (2018) (cit. on p. 27).

[CMG+14] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. Ed. by A. Moschitti, B. Pang, W. Daelemans. ACL, 2014, pp. 1724–1734. DOI: 10.3115/v1/d14-1179. URL: https://doi.org/10.3115/v1/d14-1179 (cit. on p. 23).

[CWZ19] Y. Chen, L. Wu, M. J. Zaki. "Bidirectional Attentive Memory Networks for Question Answering over Knowledge Bases". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Ed. by J. Burstein, C. Doran, T. Solorio. Association for Computational Linguistics, 2019, pp. 2913–2923. DOI: 10.18653/v1/n19-1299. URL: https://doi.org/10.18653/v1/n19-1299 (cit. on pp. 15, 17, 37, 38, 48).

[DCLT19] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Ed. by J. Burstein, C. Doran, T. Solorio. Association for Computational Linguistics, 2019, pp. 4171–4186. DOI: 10.18653/v1/n19-1423. URL: https://doi.org/10.18653/v1/n19-1423 (cit. on p. 42).

[DLSM18] D. Diefenbach, V. Lopez, K. Singh, P. Maret. "Core techniques of question answering systems over knowledge bases: a survey". In: *Knowledge and Information systems* 55.3 (2018), pp. 529–569 (cit. on p. 19).

[FLB+13] D. Ferrucci, A. Levas, S. Bagchi, D. Gondek, E. T. Mueller. "Watson: Beyond Jeopardy!" In: *Artificial Intelligence* 199-200 (2013), pp. 93–105. ISSN: 0004-3702. DOI: https://doi.org/10.1016/j.artint.2012.06.009. URL: http://www.sciencedirect.com/science/article/pii/S0004370212000872 (cit. on p. 15).

[HG01] L. Hirschman, R. Gaizauskas. "Natural language question answering: the view from here". In: *natural language engineering* 7.4 (2001), p. 275 (cit. on p. 15).

[HS97] S. Hochreiter, J. Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780 (cit. on p. 20).

[LJ20] Y. Lan, J. Jiang. "Query Graph Generation for Answering Multi-hop Complex Questions from Knowledge Bases". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Ed. by D. Jurafsky, J. Chai, N. Schluter, J. R. Tetreault. Association for Computational Linguistics, 2020, pp. 969–974. URL: https://www.aclweb.org/anthology/2020.acl-main.91/ (cit. on pp. 16, 41, 49).

[LLLZ18] K. Luo, F. Lin, X. Luo, K. Zhu. "Knowledge base question answering via encoding of complex query graphs". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2018, pp. 2185–2194 (cit. on p. 16).

[LMP01] J. Lafferty, A. McCallum, F. C. Pereira. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data". In: (2001) (cit. on p. 24).

[Met78] C. E. Metz. "Basic principles of ROC analysis". In: *Seminars in nuclear medicine*. Vol. 8. 4. WB Saunders. 1978, pp. 283–298 (cit. on p. 24).

[MFD+16] A. Miller, A. Fisch, J. Dodge, A.-H. Karimi, A. Bordes, J. Weston. "Key-value memory networks for directly reading documents". In: *arXiv preprint arXiv:1606.03126* (2016) (cit. on p. 38).

[MM11] S. Mittal, A. Mittal. "Versatile question answering systems: seeing in synthesis". In: *International journal of intelligent information and database systems* 5.2 (2011), pp. 119–142 (cit. on p. 15).

[MSL18] S. Mohammed, P. Shi, J. Lin. "Strong Baselines for Simple Question Answering over Knowledge Graphs with and without Neural Networks". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*. Ed. by M. A. Walker, H. Ji, A. Stent. Association for Computational Linguistics, 2018, pp. 291–296. DOI: 10.18653/v1/n18-2047. URL: https://doi.org/10.18653/v1/n18-2047 (cit. on pp. 16, 33–35, 43).

[NLS18] M. Narasimhan, S. Lazebnik, A. Schwing. "Out of the box: Reasoning with graph convolution nets for factual visual question answering". In: *Advances in neural information processing systems*. 2018, pp. 2654–2665 (cit. on p. 15).

[Ola15] C. Olah. "Understanding lstm networks, 2015". In: *URL http://colah. github. io/posts/2015-08-Understanding-LSTMs* (2015) (cit. on pp. 20, 21, 23).

[ON15] K. O'Shea, R. Nash. "An introduction to convolutional neural networks". In: *arXiv preprint arXiv:1511.08458* (2015) (cit. on p. 19).

[PVS+16] T. Pellissier Tanon, D. Vrandečić, S. Schaffert, T. Steiner, L. Pintscher. "From freebase to wikidata: The great migration". In: *Proceedings of the 25th international conference on world wide web*. 2016, pp. 1419–1428 (cit. on p. 27).

[PZ18] M. Petrochuk, L. Zettlemoyer. "SimpleQuestions Nearly Solved: A New Upperbound and Baseline Approach". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Ed. by E. Riloff, D. Chiang, J. Hockenmaier, J. Tsujii. Association for Computational Linguistics, 2018, pp. 554–558. DOI: 10.18653/v1/d18-1051. URL: https://doi.org/10.18653/v1/d18-1051 (cit. on pp. 17, 36, 43).

[QLK+18] Y. Qu, J. Liu, L. Kang, Q. Shi, D. Ye. "Question answering over freebase via attentive RNN with similarity matrix based CNN". In: *arXiv preprint arXiv:1804.03317* 38 (2018) (cit. on pp. 16, 29–32, 43, 47).

[SF07] Y. Sasaki, R. Fellow. "The truth of the F-measure, Manchester: MIB-School of Computer Science". In: *University of Manchester* (2007) (cit. on p. 25).

[SG18] D. Sorokin, I. Gurevych. "Modeling Semantics with Gated Graph Neural Networks for Knowledge Base Question Answering". In: *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*. Ed. by E. M. Bender, L. Derczynski, P. Isabelle. Association for Computational Linguistics, 2018, pp. 3306–3317. URL: https://www.aclweb.org/anthology/C18-1280/ (cit. on p. 16).

[SP97] M. Schuster, K. K. Paliwal. "Bidirectional recurrent neural networks". In: *IEEE transactions on Signal Processing* 45.11 (1997), pp. 2673–2681 (cit. on p. 22).

[STT20] A. Saxena, A. Tripathi, P. P. Talukdar. "Improving Multi-hop Question Answering over Knowledge Graphs using Knowledge Base Embeddings". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Ed. by D. Jurafsky, J. Chai, N. Schluter, J. R. Tetreault. Association for Computational Linguistics, 2020, pp. 4498–4507. URL: https://www.aclweb.org/anthology/2020.acl-main.412/ (cit. on p. 17).

[SZD+18] Z. Su, C. Zhu, Y. Dong, D. Cai, Y. Chen, J. Li. "Learning visual knowledge memory networks for visual question answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7736–7745 (cit. on p. 15).

[VAGS16] N. T. Vu, H. Adel, P. Gupta, H. Schütze. "Combining Recurrent and Convolutional Neural Networks for Relation Classification". In: *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*. Ed. by K. Knight, A. Nenkova, O. Rambow. The Association for Computational Linguistics, 2016, pp. 534–539. DOI: 10.18653/v1/n16-1065. URL: https://doi.org/10.18653/v1/n16-1065 (cit. on p. 20).

[WWS+17] P. Wang, Q. Wu, C. Shen, A. Dick, A. Van Den Hengel. "Fvqa: Fact-based visual question answering". In: *IEEE transactions on pattern analysis and machine intelligence* 40.10 (2017), pp. 2413–2427 (cit. on p. 15).

[YCHG15] W.-t. Yih, M.-W. Chang, X. He, J. Gao. "Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*. The Association for Computer Linguistics, 2015, pp. 1321–1331. DOI: 10.3115/v1/p15-1128. URL: https://doi.org/10.3115/v1/p15-1128 (cit. on p. 16).

[YNDT18] R. Yamashita, M. Nishio, R. K. G. Do, K. Togashi. “Convolutional neural networks: an overview and application in radiology”. In: *Insights into imaging* 9.4 (2018), pp. 611–629 (cit. on pp. 19, 20).

[ZDK+17] Y. Zhang, H. Dai, Z. Kozareva, A. J. Smola, L. Song. “Variational reasoning for question answering with knowledge graph”. In: *arXiv preprint arXiv:1709.04071* (2017) (cit. on p. 15).

[ZHZ18] M. Zhou, M. Huang, X. Zhu. “An interpretable reasoning network for multi-relation question answering”. In: *arXiv preprint arXiv:1801.04726* (2018) (cit. on p. 16).

All links were last followed on Jan 1, 2021.

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

Stuttgart, 15.01.2021 Vishnudatha.K

place, date, signature