Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Control Engineering

Master's thesis

Unstructured text comprehension and question answering

bc. Jakub Konrád

Supervisor: Ing. Jan Šedivý, CSc.

8th January 2018

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Konrád** Jméno: **Jakub** Osobní číslo: **406436**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra řídicí techniky**

Studijní program: **Kybernetika a robotika**

Studijní obor: **Systémy a řízení**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Porozumění nestrukturovanému textu a automatické odpovídání na dotazy

Název diplomové práce anglicky:

Unstructured text comprehension and question answering

Pokyny pro vypracování:

1. The thesis task is to find how is the performance of question answering models improving by narrowing the semantics of a training domain and by decreasing the training data size.
2. Start with training a large domain model. Use the Stanford Question Answering Dataset (SQuAD) reading comprehension dataset[3]. Focus on similar models as described in [1.2].
3. Create a small domain data sets selected and agreed on with the supervisor.
4. Create small domain models by retraining the SQuAD model.
5. Evaluate how is the performance improving with shrinking the domain and increasing the data.
6. The theses will result in a recommendation of the best meta-parameters for creating small domain models by retaining from a large general purpose model.

Seznam doporučené literatury:

[1] Zhang, Junbei, et al. "Exploring Question Understanding and Adaptation in Neural-Network-Based Question Answering." arXiv preprint arXiv:1703.04617 (2017).
[2] Liu, Rui, et al. "Structural Embedding of Syntactic Trees for Machine Comprehension." arXiv preprint arXiv:1703.00572 (2017).
[3] Rajpurkar, Pranav, et al. "Squad: 100,000+ questions for machine comprehension of text." arXiv preprint arXiv:1606.05250 (2016).

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Jan Šedivý, CSc., velká data a cloud computing CIIRC

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **17.07.2017** Termín odevzdání diplomové práce: **09.01.2018**

Platnost zadání diplomové práce: **31.01.2019**

Ing. Jan Šedivý, CSc. podpis vedoucí(ho) práce	prof. Ing. Michael Šebek, DrSc. podpis vedoucí(ho) ústavu/katedry	prof. Ing. Pavel Ripka, CSc. podpis děkana(ky)

Acknowledgements

First and foremost, I would like to thank my supervisor Ing. Jan Šedivý, CSc. for his patience, guidance and advice. Additionally, I would like to thank my family and my friends for their constant support during my study.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

In Prague on 8th January 2018 .....

Czech Technical University in Prague
Faculty of Electrical Engineering

Citation of this thesis

Konrád, Jakub. *Unstructured text comprehension and question answering.* Master's thesis. Czech Technical University in Prague, Faculty of Electrical Engineering, 2018.

Abstrakt

Tato práce se zabývá porozuměním nestruktorovanému textu a automatickým odpovídáním na otázky. Zaměřuje se na využití principů přenosu znalostí. Práce popisuje několik experimentů které ukazují zlepšení výkonu modelu natrénovaného na obecné bázi dat (SQuAD) přeneseného a přetrénovaného, na menší tematicky specifickou bázi dat.

Klíčová slova Závěrečná práce, Automatické odpovídání na dotazy, Přenos znalostí, SQuAD, DrQA.

Abstract

This thesis deals with comprehension of unstructured text and question answering. Specifically we focus the benefits of transfer learning. We show several experiments where we employ transfer learning to adjust model created on general large QA dataset (SQuAD) with data from smaller topic specific datasets. We explore influence of transfer learning on datasets focusing on specific topic not included in the main domain.

Keywords Thesis, Question Answering, Transfer Learning, SQuAD, DrQA.

Contents

List of Figures

List of Tables

Introduction

One of the more challenging but popular tasks in computer science and machine learning has always been reading comprehension and automatic question answering. We always wanted to teach computers to sort through large quantities of information and help us find the answers we are looking for. This can help us access information more easily. Moreover, it is an integral part of a broader goal in the field of NLP (natural language processing) and conversational AI, creating a general purpose artificial intelligence.

There is another reason why the question answering task has been very popular. The problem we often face with conversational AI is how to evaluate performance of a conversational agent in a general dialogue. This is challenging because we do not have an exact metric for how to rate the conversation. The task of question answering allows us to find evaluation criteria quite easily. We just compare the answer given by the agent to the expected answer. Another advantage is that the question answering tasks can be approached very broadly. Many tasks can be reformulated to fit the question answering task, therefore the development of efficient and accurate question answering systems is important for the field of NLP and conversational AI[1].

We can divide the question answering task into two subproblems - retrieval and inference [1]. Let's start with information retrieval. We need to find (retrieve) the requested knowledge and store it in some form. This can mean we store the article or an excerpt that contains said knowledge or we use for example an RDF database. As for inference, we must search through the stored knowledge and find the information that will help us answer the posed question.

In the specific case of reading comprehension, we want the computer to find an answer to our question, located in a specific excerpt of text that is available to us. We are not looking for the right answer reflected in the real world. Instead, our goal is to find the answer to the question in a way that is presented in specific text excerpt. We present the agent with a question and a text that is supposed to contain the answer. Then we require the agent

to sort through the information contained in the text and pick out the best possible answer to our question.

In this work, we will focus on question answering through reading comprehension. Specifically, we want to employ transfer learning to see how can a small amount of training data impact a performance of a model trained on a large dataset when presented with a test dataset of unfamiliar topics and types of questions. We have chosen SQuAD (The Stanford Question Answering Dataset)[2] for our research. In this work we employ one of the state of the art question answering systems, DrQA[3] developed at Facebook AI research.

We evaluate the performance of the system and then perform several experiments focusing on transfer learning. We have created three smaller focused datasets that we carved out from the SQuAD dataset. Two of them focus on questions regarding specific topical areas, that is people and places. The third one contains questions with specific syntactic pattern, that is "Who questions". We evaluate the performance of the large scale general model on these focused datasets. We try to increase the performance by incrementally training the large scale model on the training data from the focused datasets.

Motivation

Work on this thesis has been conducted as a part of the Alquist [4] project during the 2017 Amazon Alexa Prize. Alexa Prize is a university competition organized by the Amazon corporation[5]. The competition is focused on advancing the field of conversational artificial intelligence. During the competition teams were tasked with creating a social bot running on Amazon Alexa platform that was supposed to hold coherent and engaging conversation with the user for up to 20 minutes.

During the development of Alquist it became obvious that the social bot will be required to talk and answer questions about various topics and areas of interest. However, there was not always a suitable dataset or information source available. We have chosen the topic of this thesis in order to help alleviate this problem. We want to determine if, for the purposes of the Alquist socialbot, it is possible to use models trained on large scale datasets and employ transfer learning techniques to tweak them so that they will perform better in the areas we are specifically interested in.

Thesis structure

The first part of this thesis (chapters 1, 2, 3 and 4) focuses on theoretical background for the task and the experiments. In chapter 1 we talk about other related works and approaches to the question answering task. Chapter 2 focuses on topics and approaches important for this work, that is recurrent

neural networks and transfer learning. In chapter 3 we describe the SQUAD dataset and in chapter 4 we talk in detail about the DrQA system.

Chapters 5, 6, and 7 focus on the practical part of the thesis. In chapter 5 we conduct an error analysis of the DrQA system on the SQuAD dataset. Chapter 6 describes the proposed experiments in detail and chapter 7 evaluates the results of the performed experiments.

Chapter 1

Related work

Automatic question answering is a popular task in the field of artificial intelligence. Recently there has been many datasets and challenges published in order to move the state of the art froward.

Besides the SQuAD dataset that we focus on in this thesis there are others trying to achieve similar goals. The MCTest dataset[6] is a freely available dataset focusing on machine comprehension. It contains 660 stories with 4 questions per a story and 4 answer choices per question. Many of the QA pairs in the dataset require common sense reasoning [2]. Another popular dataset is WikiQA[7]. WikiQA is an open domain dataset of question and sentence pairs, collected and annotated for research on open-domain question answering. Finally, recently researches have constructed cloze datasets. Here the goal is to predict a missing word in a passage[2]. An advantage of these datasets is that they can be easily generated from naturally occurring data. Due to this they can be extremely large. The Children's Book Test[8], dealing with predicting a blanked out word in a sentence contains 688K entries.

If we look at the question answering task itself, one of the most important discoveries not only for question answering but for the entire field of computational linguistics were word embeddings. In this work we use the GLOVE embeddings[9]. Other possibilities include the original word2vec created by Tomas Mikolov and his team in 2013[10] or fastText published in 2016[11].

Automatic question answering and reading comprehension tasks can be approached from many directions. Rui Liu[12] uses *structural embedding of syntactic trees* to utilize syntactic information that is then encoded to the vector representation to improve accuracy of the model in machine comprehension task. Junbei Zhang[13] focuses on closely modelling questions in a neural network framework, encoding questions with their syntactic information and then modelling different types of questions as an adaptation task.

Recently, researchers have started exploring the possibilities of transfer learning in automatic question answering, which is also the topic of this thesis. We will now mention other works dealing with this area. Yu-An Chung explores the possibilities of supervised and unsupervised transfer learning in question answering in [14]. Specifically the author focuses on using simple transfer learning techniques to improve performance of the model in multi-choice question answering task. Sewon Min in [15] shows that the task of question answering on one dataset can benefit from models trained on different large, fine-grained QA dataset. The author shows that question answering with sentence-level supervision can greatly benefit from standard transfer learning of a question answering model trained on a large, span-level supervision[15].

Chapter 2

Theoretical Background

2.1 Recurrent neural networks

When working with classical neural networks we assume that all individual inputs are not mutually dependant. This is true for some tasks, but for many others it can lead to disappointing results[16]. The classical neural networks are limited to a fixed number of computational steps. They accept fixed size vector as input and produce a fixed size output with the inputs and outputs being independent on previous inputs and outputs[17]. Recurrent neural networks allow us to address this issue.

RNN takes the input vector presented to the network and uses fixed but learned function to combine it with their own state vector. This produces a new state vector for the RNN[17]. What this essentially means is that the new state vector of the RNN is dependant not only on the input but also on the previous state vector of the network and thus also on the previous inputs[18]. We can think about RNN as if they have "memory" of previous computations[16].

We can see a diagram of an unrolled RNN in the figure 2.1. What we did by unrolling is we displayed the entire sequence of inputs and outputs flowing through the network and their dependence. x_t is the input at step time t. s_t

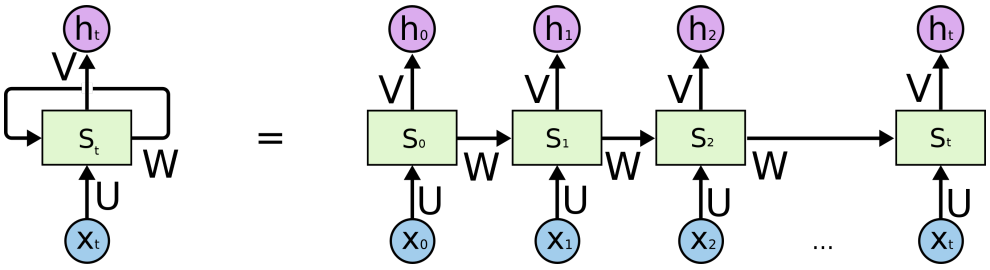


Figure 2.1: A schematic of an unrolled recurrent neural network[19]

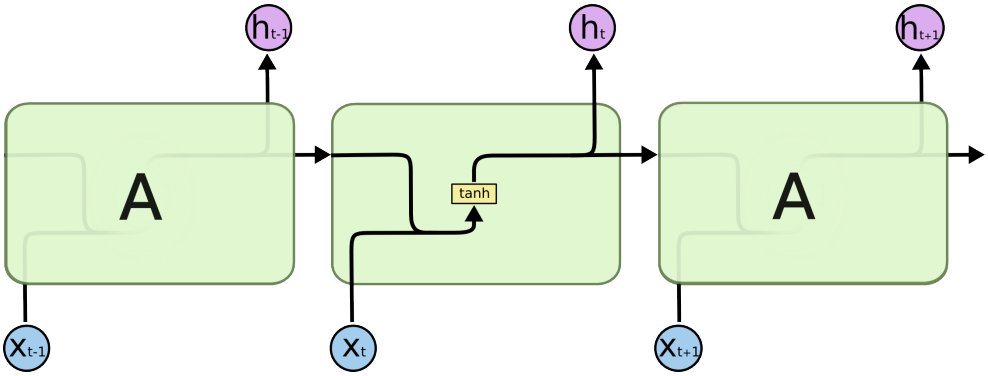


Figure 2.2: Recurrent neural network with a repeating module caontaining a single *tanh* layer[19].

represents the hidden state of the network in time step t. It is calculated from previous hidden state s_{t-1} and current input as

$$s_t = f(Ux_t + Ws_{t-1})$$

where U and W are weight matrices. The function f is usually a non-linearity such as $tanh$ or $ReLU$. h_t is the output at step t[16].

RNNs are currently employed in a variety of different machine learning tasks, such as language modelling [20] or text generation[21] and machine translation[22] or speech recognition[23] and of course question answering [13] and machine comprehension[12]

RNNs should be in theory able to make use of arbitrarily long sequences of information. However, in reality we observe that this effect is limited only on information presented in several most recent steps. This has been described in detail in [24].

When we train the RNN we essentially unfold it and create deep feed-forward network. Each time step is then represented by one layer of the network. After several time steps however, the contribution from the layers further from the current time step becomes vanishingly small when compared to contribution from the layers closest to the current time step. In applications where it is imperative that the network retains the long term memory, we need to be aware of this problem. One of the possible solutions are so called LSTM networks.

2.2 Long Short Term Memory Networks

Long Sort Term Memory networks were introduced in 1997 by Hochreiter & Schmidhuber in [25]. They are designed to be able to retain information over large number of steps[25].

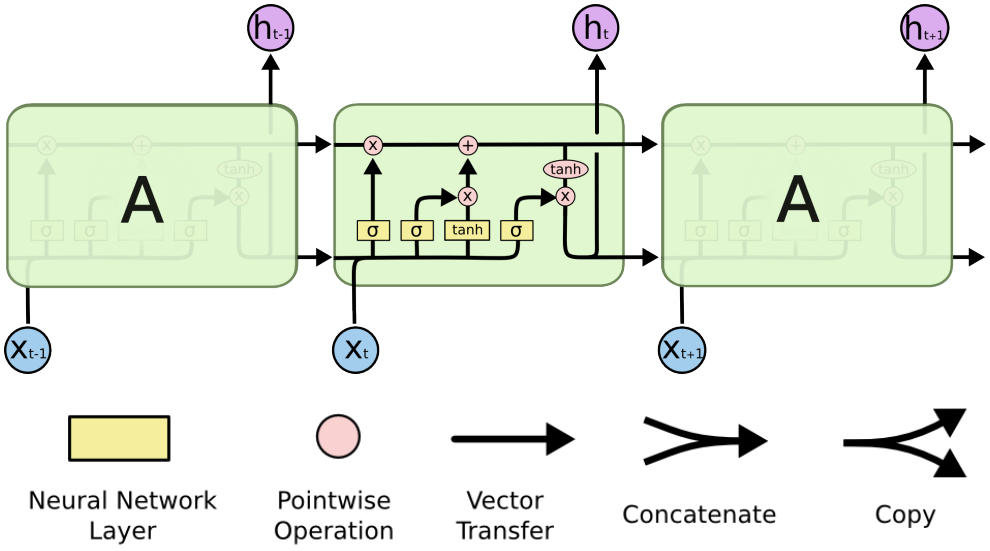
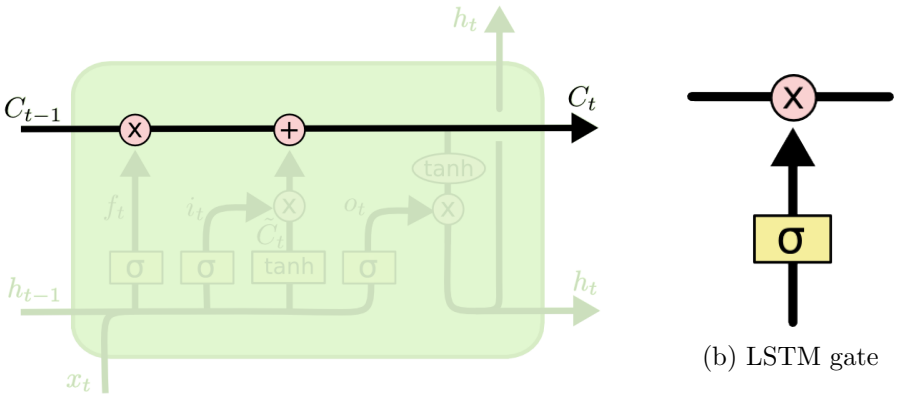


Figure 2.3: LSTM network whose repeating module contains four interconnected layers[19].



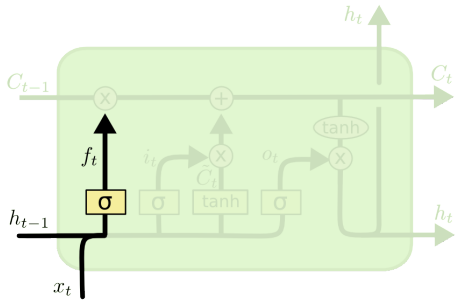
(a) LSTM cell state

(b) LSTM gate

Figure 2.4: Cell state and a gate of an LSTM network[19].

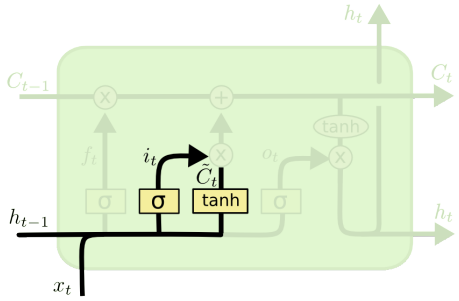
As we have previously shown, all RNNs have a repeating structure, but where classical RNNs have only one repeating layer (figure 2.2), LSTM networks are more complex, having four interconnected repeating layers (figure 2.3).

The main reason why are the LSTMs able to keep information through many steps is the cell state[19]. The cell state flows through the entire network (figure 2.4a). The LSTM can alter the information in the cell state through gates. Gates allow the network to let some information through. They compose of a sigmoid neural net layer and a point-wise multiplication



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

Figure 2.5: LSTM forget gate layer. It decides which information to retain and which information to forget from previous step[19].



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figure 2.6: LSTM forget gate layer. It decides which information to retain and which information to forget from previous step[19].

(figure2.4b). The output of the sigmoid is numbers between 0 and 1, which specify how much information from each component should be let through (value of zero means let nothing through)[19].

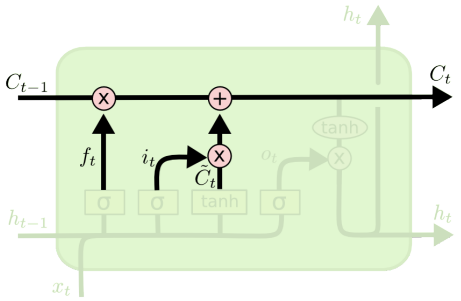
Since DrQA system is based on LSTMs, lets look at how LSTMs function in more detail.

2.2.1 LSTM function

We can divide the function of one LSTM cell into four steps.

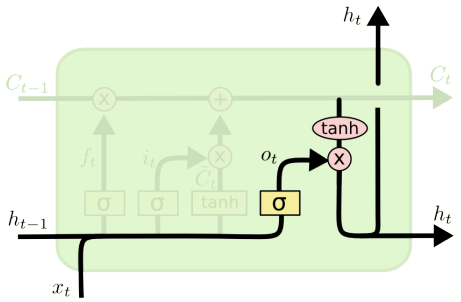
In the first step (figure 2.5) the LSTM network decides which information to retain and which to forget from previous step. The information passes through a sigmoid layer referred to as "forget gate layer"[26]. We look at the output from previous step h_{t-1} and the input from current step x_t. Depending on the value of the sigmoid for each number in cell state C_{t-1} the corresponding information is either retained or forgotten (again value of zero means completely forget, value of one means keep everything). We receive a new vector f_t that will be used in later steps to update the cell state. W_t is a wight matrix and b_t is a bias vector.

In the second step (figure 2.6), the LSTM cell decides which new inform-



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

Figure 2.7: LSTM forget gate layer. It decides which information to retain and which information to forget from previous step[19].



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 2.8: LSTM forget gate layer. It decides which information to retain and which information to forget from previous step[19].

ation should passed on to the cell state. This is done via an "input gate layer"[26]. First a sigmoid function determines what values will be updated in this step i_t. Then the cell creates a vector of new candidate values $\widetilde{C}_t$ via a *tanh* layer. These two are then processed in the next step[19].

In the third step (figure 2.7), the old cell state C_{t-1} is updated. The LSTM cell has already prepared the changes in steps one and two, now we just need to incorporate them. We first multiply the old cell state C_{t-1} by the vector f_t in order to forget information we decided to eliminate in the first step. Then we add $\widetilde{C}_t i_t$ that is the new candidate values from step two weighted by how much we decided to update each individual state value[19]. By doing this we obtain the new cell state C_t.

And finally, the network needs to create the output vector. First we take the input x_t in this step and the output h_{t-1} from previous step and we run them through a sigmoid layer to determine which parts of the cell state C_t the network should output. We run the cell state through a *tanh* layer, pushing the cell state values into -1 and 1 range and then multiply it by the output vector from the sigmoid o_t. The result of these operations if the final output h_t of the LSTM cell[19].

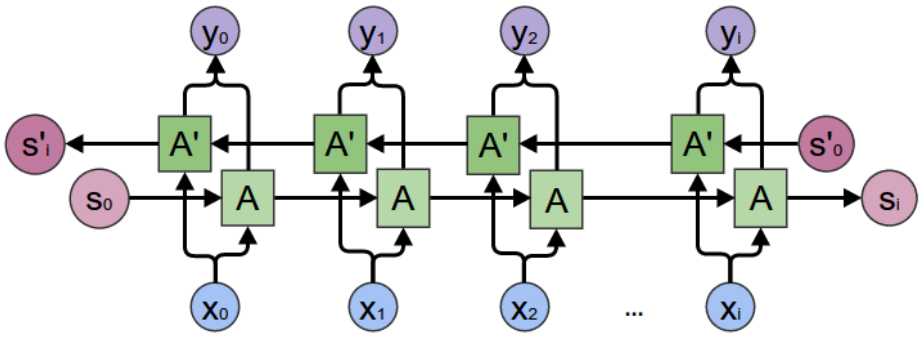


Figure 2.9: Bidirectional RNN architecture[29].

2.2.2 Bidirectional LSTM

Bidirectional recurrent neural networks were described in 1997 by by Schuster and Paliwal[27] as a way of increasing the input information available to the recurrent neural network. Standard recurrent neural network architecture allows for the network to retain information from previous time steps. Using bidirectional RNN architecture we are able to make use of both the past and the future contextual information for every time step of the input sequence in order to calculate the output sequences [28].

The architecture also contains two separate hidden layers, one containing forward states and another containing backward states, both computing contextual information. This allows us to further increase the performance on tasks where the current token in the sequence depends not only on its previous tokens but also on the tokens following after it. This is useful among others in reading comprehension as words in the sentence are often dependant not only words preceding them but also on words following after.

2.3 Transfer Learning

Transfer learning[30] is often also referred to as knowledge transfer. It is a vital machine learning technique. It aims to take a knowledge learned in one task and apply it to a different but related task in order to improve the performance or help reduce required training data etc.[15].

Transfer learning can be truly beneficial when when we have an abundance of labeled training data for one task, however the data is outdated. In order to create new, relevant training data we would be forced to expend a large amount of resources. With transfer learning we can transfer the knowledge we gained from the old, outdated training data to the new, relevant domain[30]. This is extremely important in cases where labeled training data is sparse, where the data is easily outdated or where we need to adapt the system to be able to handle new knowledge domains quickly.

All of the stated reasons are true for our application in the Alquist conversational system. The system needs to be able to talk about current events where there is no or only minimal labeled data available, the data is easily outdated and of course the system needs to be flexible and we need to be able to change its known domain based on the audience talking to the system.

2.4 Transfer learning definition and categorization

Let us now look into the definition of transfer learning and possible categorization of transfer learning tasks.

Transfer Learning. *Given a source domain D_S and learning task T_S , a target domain D_T and learning task T_T , transfer learning aims to help improve the learning of the target predictive function $f_T(\cdot)$ in D_T using the knowledge in D_S and T_S, where $D_S \neq D_T$, or $T_S \neq T_T$.*[30]

Transfer learning techniques can be divided into three main categories: *inductive transfer learning*, *transductive transfer learning*, and *unsupervised transfer learning*, based on different situations between the source and target domains and tasks.

Inductive transfer learning Here the source task and target tasks are different, no matter whether source and target domains are different or not. In this case we require some labeled data in the target domain to achieve good performance[30].

Transductive transfer learning The source and target tasks are the same, while the source and target domains are different[30]. No labeled data in the target domain is available while there is an abundance of labeled data in the source domain[30].

Unsupervised transfer learning The unsupervised transfer learning is similar to inductive transfer learning. The target task is different from but related to the source task. However, the unsupervised transfer learning focuses on solving unsupervised learning tasks in the target domain[30].

2.5 Transfer learning application

Transfer learning has been successfully applied in numerous domains. For example in computer vision, neural networks trained on large scale image classification datasets have shown to perform well in feature extraction for tasks such as image captioning[31] or visual question answering[32].

Transfer learning has also been successfully applied in NLP tasks. It showed promising results in named entity recognition[33], syntactic parsing[34] and others.

We employ transfer learning in the area of question answering. Transfer learning has been successfully applied to various domains in the computational linguistics and NLP fields, its applicability to question answering has yet to be well-explored[14]. We aim to conduct experiments that show the applications of knowledge transfer when using large generic pretrained models on a task based on smaller domain focusing on a specific topic or type of question.

Chapter 3

Stanford Question Answering Dataset (SQuAD)

SQuAD is a reading comprehension dataset created by Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev and Percy Liang from Computer Science Department at Stanford University. It consists of over 100,000 questions. The questions were posed by crowdworkers on 536 Wikipedia articles randomly selected from the top 10,000. For each article, individual paragraphs were extracted and converted to plain text form, while discarding paragraphs shorter than 500 characters. The final dataset contains 23,215 paragraphs from 536 articles. The articles were randomly divided into a training set containing 80% if the articles and a development set containing 10% of the articles and a test set also containing 10% of the articles[2].

As previously mentioned, the dataset contains over 100,000 questions. For each question, dataset contains at least three possible correct answers in the development and test sets. This ensures diversity in the possible ground truth answers and makes the automatic evaluation more robust.

The dataset contains articles on large variety of topics from historical figures to locations or current events. The dataset also shows a great diversity in answers and questions, and consists of large number of answers beyond proper noun entities[2]. The diversity in topics and questions, and the large volume of question answer pairs in the dataset allows us to extract smaller, focused datasets from SQuAD while still being able to create a generic model trained on the rest of the data, making SQuAD valuable tool for our research.

The dataset uses two evaluation metrics: exact string match (EM) and F1 score. The exact match metric measures the percentage of predictions matching any one of the ground truth answers exactly. The F1 score measures the average overlap between the prediction and the ground truth answer. The prediction and the ground truth are treated as bags of tokens and their F1 is computed. Maximum F1 is taken from all the ground truth answers for given question and then average over all the questions[2].

The human performance score on the test set of the dataset is 77.0% for the exact match metric and 86.8% for F1. The baseline logistic regression model created by the dataset authors reaches 40.4% on the EM metric and 51% for F1.

Chapter 4

DrQA system

DrQA system has been developed by Danqi Chen from Stanford University and Adam Fisch, Jason Weston Antoine Bordes from Facebook AI Research[3]. The system is built for the task of open domain question answering. The system uses Wikipedia as its knowledge source. One of the main advantages of using Wikipedia over other possible knowledge sources such as Freebase[35] or DBPedia[36] is that it is constantly evolving and contains detailed information about broad spectrum of topics[3].

The problem with using Wikipedia as a knowledge source is that unlike Freebase and DBPedia, the data contained in Wikipedia is not in a structured form (triplets) as in those databases. It is held as a plain text. That makes the extraction of information much more challenging for the QA system.

The DrQA system consists of two components, Document Retriever and Document Reader. Both components are able to work as separate systems independent on each other (figure 4.1).

The Document Retriever component focuses on reading and finding articles that are likely to be relevant to the question. It is a non-machine learning system based on inverted index look-up. We will not talk about Document Retriever in detail as it is not relevant for this thesis.

The second component is Document Reader. Document reader is inspired by successful RNN models, like AttentiveReader described in [37]. It uses LSTM RNN models to represent the text excerpt and the question and then finds the beginning and the end of the most probable answer span.

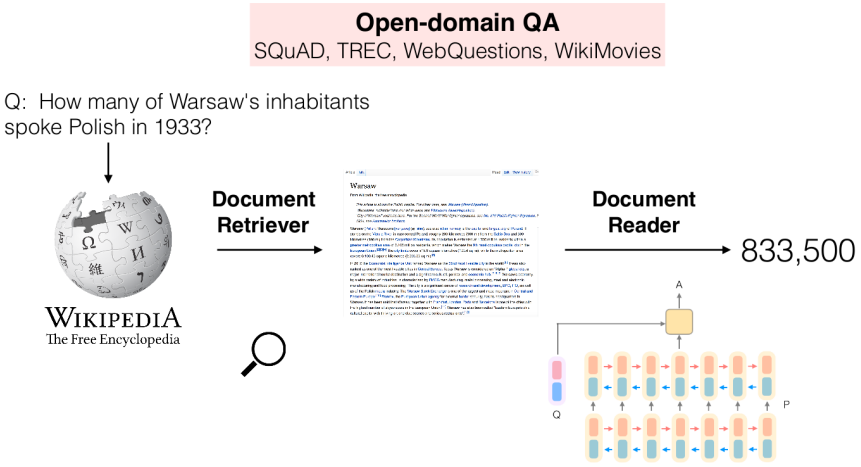


Figure 4.1: Overview of DrQA system [3].

4.1 DrQA document reader

In the reading comprehension task, we are given a question q and a paragraph of small length containing the answer p. The question q consists of l tokens

$$\{q_1, ..., q_l\}$$

and the paragraph p consists of m tokens

$$\{p_1, ..., p_m\}.$$

The DrQA system uses recurrent neural networks to encode the paragraph p and the question q and then uses a prediction algorithm to determine the span of words from the paragraph p corresponding to the most probable answer to the posed question q[3]. The system uses the Stanford CoreNLP toolkit[38] for tokenization of the questions and paragraphs.

4.1.1 Paragraph encoding

All tokens p_i in the paragraph p are initially represented as a sequence of feature vectors[3].

$$\tilde{\mathbf{p}} \in \mathbb{R}^d.$$

The feature vectors are then passed as an input to three layered bidirectional LSTM recurrent neural network. From there we obtain:

$$\{\mathbf{p}_1, ..., \mathbf{p}_m\} = \mathbf{RNN}(\{\tilde{\mathbf{p}}_1, ..., \tilde{\mathbf{p}}_m\}),$$

where $\mathbf{p}_i$ is the concatenation of each layer's hidden units in the end. This way $\mathbf{p}_i$ then contains useful information not only about the token but also about its context[3].

The feature vector consists of the following:

Word embeddings

The first part of the feature vector is the word embedding, a real number n-dimensional vector representing the word in vector space.

$$f_{emb}(p_i) = \mathbf{E}(p_i)$$

The system uses the 300-dimensional Glove word embeddings trained on 840B Web data crawl [9]. Most of the pretrained embeddings are fixed. Only 1000 most frequent words are being fine-tuned. This allows us to put emphasis on the representations of some key question words such as $what, how, where$ etc. as there can be crucial for question answering tasks[3].

Exact match

The second part of the vector consists of three binary features that indicate whether a token p_i can be exactly matched to to one of the question words in question q.

$$f_{exact_match}(p_i) = \mathbb{I}(p_i \in q)$$

The token can be matched either in its original form, lower-cased form or as a lemma[3]. The importance of these features is shown in the table 4.1.

Token features

The feature vector also contains features that reflect properties of token p_i in its context. These features are the token's part-of-speech (POS), named entity recognition (NER) tags and its (normalized) term frequency (TF)[3].

$$f_{token}(p_i) = (POS(p_i), NER(p_i), TF(p_i))$$

The part-of-speech and named entity tags are extracted using the Stanford CoreNLP toolkit[38].

Aligned question embeddings

Lastly, the feature vector contains so called aligned question embeddings.

$$f_{align}(p_i) = (\sum_j a_{i,j}\mathbf{E}(q_j))$$

The questions q and the paragraphs p often contain large lexical similarity or even overlap near the correct answer span. The aligned question embeddings are specifically designed to exploit these similarities[39].

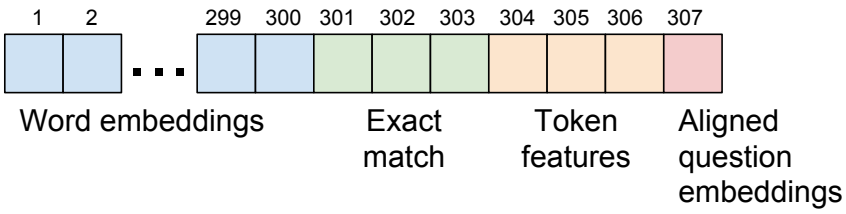


Figure 4.2: Feature vector for the paragraph encoder of the DrQA system. The vector consists of our parts: word embedding, exact match binary features, contextual token features and aligned question embedding.

These aligned question embeddings are computed via neural attention[40]. The attention score $a_{i,j}$ captures the similarity between the token p_i and each of the question words q_j. The attention score $a_{i,j}$ is computed by the dot product between nonlinear mappings of the word embeddings:

$$a_{i,j} = \frac{\exp(\alpha(\mathbf{E}(p_i)) \cdot \alpha(\mathbf{E}(q_j)))}{\sum_{j'} \exp(\alpha(\mathbf{E}(p_i)) \cdot \alpha(\mathbf{E}(q_{j'})))}$$

where $\alpha(\cdots)$ is a single dense layer with ReLU non-linearity[3].

These features fulfill a similar role to the *exactmatch* features. However, where the *exactmatch* features focused on the words being the same, these features add a soft alignment between similar but non-identical words such as *car* and *vehicle*[3].

We will talk more about the importance of these features in the section 4.1.4

4.1.2 Question encoding

For the question encoding DrQA uses recurrent neural network. However, the question encoding is much simpler. The feature vector consist only of word embeddings. On the vector a recurrent neural network is applied. Then the resulting hidden units are combined into one vector:

$$\{\mathbf{q}_1, ..., \mathbf{q}_l\} \rightarrow \mathbf{q}.$$

Then $\mathbf{q}$ is computed as

$$\mathbf{q} = \sum_j b_j \mathbf{q}_j$$

where b_j encodes the importance of each word contained in the question:

$$b_j = \frac{\exp(\mathbf{w} \cdot \mathbf{q}_j)}{\sum_{j'} \exp(\mathbf{w} \cdot \mathbf{q}_{j'})},$$

and $\mathbf{w}$ is a weight vector[3].

4.1.3 Answer prediction

The goal of the DrQA system is to process the posed question and the provided paragraph and then to predict a span of tokens that is most likely the correct answer to the question. In order to do this the DrQA system first takes the encoded paragraph vectors we described earlier

$$\{\mathbf{p}_1, ..., \mathbf{p}_m\}$$

and with them the question vector q as input. The system then trains two independent classifiers in order to predict the beginning and the end of the answer span[3]. To describe the classifiers in more detail, firstly the system uses a bilinear term to capture similarity between $\mathbf{p}_i$ and $\mathbf{q}$. The system computes probabilities of each token being start and end as:

$$P_{start}(i) \propto \exp(\mathbf{p}_i \mathbf{W}_s \mathbf{q})$$

for the start to the answer span and

$$P_{end}(i) \propto \exp(\mathbf{p}_i \mathbf{W}_e \mathbf{q})$$

for the end of the answer span[3]. System then tries to predict the the answer span in a following way. The system chooses the best span contained in the paragraph from token i to token i' such that following conditions are met.

Firstly, the chosen span must not be longer than fifteen words (tokens). That is

$$i \leq i' \leq i + 15.$$

And secondly the system selects the span so that

$$P_{start}(i) \times P_{end}(i')$$

is maximized[3].

Furthermore, if the system was provided multiple paragraphs, it uses unnormalized exponential and takes the argmax over all predicted paragraph spans in order to deliver the final prediction[3].

4.1.4 Importance of question related and contextual paragraph encoder feature vector features

In the table 4.1 we can see that all additional features added to the feature vector contribute to the final system performance. Additionally it is important to notice that while omitting only the exact match features or only the aligned question embedding features does not lead to a significant drop in performance of the system, omitting both of the question related features from the feature vector results in the system performance dropping significantly[3]. This further supports the claim that the using lexical and syntactical similarity of sentence in proximity to correct answer span leads to better results in question answering tasks.

Features	**F1**
Full	78.8
No f_{token}	78.0 (-0.8)
No f_{exact_match}	77.3 (-1.5)
No $f_{aligned}$	77.3 (-1.5)
No $f_{aligned}$ and f_{exact_match}	59.4 (-19.4)

Table 4.1: DrQA results on the SQUAD development set for different feature vectors. It is clear, that especially the question related (i. e. exact match and aligned question embeddings) features play a crucial role on the system performance.[3]

4.2 DrQA performance

Let us now look at the DrQA system performance on the SQuAD reading comprehension and question answering task. The DrQA system, specifically the document reader part of the system, was added to the SQuAD leaderboard in February 2017 with following results. It reached scores 69.5 for the exact match metric and 78.8 for F1 on the development set and 70.0 for the exact match and 79.0 for F1 on the test set[3]. Results are shown in the table 4.2. These results put DrQA among the top performing systems on the SQuAD leaderboard.

It is important to point out that SQuAD dataset and its question answering task is incredibly popular. There are many competing systems being released every month and the state of the art is moving forward at a fast pace. Because of this as of today, Nov 20 2017 there are systems outperforming DrQA on the squad leaderboard and it is likely that more systems with even better performance will be released in the near future.

Method	Dev		Test	
	EM	F1	EM	FI
Dynamic Coattention Networks[41]	65.4	75.6	66.2	75.9
Multi-Perspective Matching [42]	66.1	75.8	65.5	75.1
BiDAF [43]	67.7	77.3	68.0	77.3
R-net	n/a	n/a	71.3	79.7
DrQA Document Reader	69.5	78.8	70.0	79.0

Table 4.2: Comparison of DrQA results and results of other SQuAD leaderboard systems[3] as of Feb 6, 2017

DrQA system is not presently the top performing system on the SQuAD leaderboard. That does not in any way diminish the value of experiments performed on the system. Our goal is to show the benefits of transfer learning and incremental training in order to increase performance on smaller topically focused datasets. We are not trying to achieve the highest scores on the SQuAD question answering task.

Chapter 5

SQuAD task error analysis

As previously mentioned, in our work we are using the DrQA system, specifically its Document Reader portion, and we apply it on the SQuAD dataset. While the main goal of this work is to explore the benefits of transfer learning in automatic question answering, we have decided to first perform an error analysis on the results of the DrQA system on the vanilla SQuAD dataset.

The error analysis will allow us to better understand strengths and weaknesses of the DrQA system as well as the structure of the dataset.

Firstly, let us talk about DrQA implementation we are using. Currently there are two DrQA implementations available. Official standalone implementation of the entire DrQA system[3]. This is the implementation that is currently displayed on the SQuAD leaderboard with achieved scores 70.733 ExactMatch and 79.353 F1 on the SQuAD dataset[44]. Secondly, another implementation also by Facebook AI Research is available. This time as a part of ParlAI, a framework for dialog AI research, implemented in Python[45].

We have decided to use the ParlAI implementation as the framework allows us to adjust and experiment with the system more easily. It also contains prepared functions for viewing and exploring the dataset, which are useful both in the error analysis part of our experiment but in the transfer learning part of the experiment as well.

5.1 Model Training

As a first step of our error analysis we created a general model trained on the entire SQuAD training dataset. Similarly to the original system we used standard 840B glove embeddings with 300 dimensional vectors[9].

We set validation patience for the training to 5, where validation patience specifies how many times should the system continue the training even though the current model did not achieve better performance on the validation dataset then the previous model did. We set the time between individual validations to 7,200 seconds.

After approximately 18 hours of training and almost 130 000 turns (corresponds to 2,6 epochs) the training completed with following results: F1 - 76.4, ExactMatch - 65.8 on validation set. These results are in accordance with the results presented by the authors of the DrQA system in the ParlAI implementation[45].

With the model trained, we have performed an error analysis on the questions in the dataset that were incorrectly answered by our model.

5.2 Error Analysis

We have decided to analyze errors that our model made on the validation set of the SQuAD dataset. Due to the size of the dataset it was not possible to annotate all the incorrectly answered question answered pairs. Instead we have randomly selected 10 incorrectly answered question and answer pairs per article (or topic) in the validation set giving us a sample of 460 incorrectly answered question answer pairs. This corresponds to approximately 12% of all incorrectly answered question answer pairs.

In 33 percent of cases the system did not comprehend the question and highlighted obviously wrong and nonsensical answer. We were not able to categorize these question answer pairs further and we have not found any common characteristics among these pairs that would help us mitigate this kind of mistakes in the future. Example of this type of incorrectly selected answer can be seen here 5.1.

Excerpt:	The Panthers finished the regular season with a 15–1 record, and quarterback Cam Newton was named the NFL Most Valuable Player (MVP). They defeated the Arizona Cardinals 49–15 in the NFC Championship Game and advanced to their second Super Bowl appearance since the franchise was founded in 1995. The Broncos finished the regular season with a 12–4 record, and denied the New England Patriots a chance to defend their title from Super Bowl XLIX by defeating them 20–18 in the AFC Championship Game. They joined the Patriots, Dallas Cowboys, and Pittsburgh Steelers as one of four teams that have made eight appearances in the Super Bowl.
Question:	Who were the defending Super Bowl champions?
Correct Answer:	New England Patriots
DrQA agent:	20–18

Table 5.1: DrQA error type 1. The DrQA agent misunderstood the question and gave nonsensical answer.

In 30 percent of the reviewed incorrectly answered questions the system was able to correctly highlight the right entity type ie. when we asked about a score the system provided a score, when we asked about a place the system provided a place. However, the system failed to understand a finer context of the question and the paragraph leading to choosing the wrong entity. Example of this mistake can be seen in table 5.2. Another common mistake that we

Excerpt:	Super Bowl 50 was an American football game to determine the champion of the National Football League (NFL) for the 2015 season. The American Football Conference (AFC) champion Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers 24–10 to earn their third Super Bowl title.
Question:	What team was the NFC champion?
Correct Answer:	Carolina Panthers
DrQA agent:	Denver Broncos

Table 5.2: DrQA error type 2. System chooses right type of entity. However, it is contextually wrong.

encountered in 13% of annotated questions was that the system did properly identify the right answer. However, the system highlighted too long excerpt of the text leading to the evaluation script to mark these answers as incorrect. We provide example of this problem in table 5.3.

Excerpt:	The game was played on February 7, 2016, at Levi's Stadium in the San Francisco Bay Area at Santa Clara, California. As this was the 50th Super Bowl, the league emphasized the "golden anniversary" with various gold-themed initiatives, as well as temporarily suspending the tradition of naming each Super Bowl game with Roman numerals (under which the game would have been known as "Super Bowl L"), so that the logo could prominently feature the Arabic numerals 50.
Question:	Where was Super Bowl 50 held?
Correct Answer:	Levi's Stadium
DrQA agent:	February 7, 2016, at Levi's Stadium in the San Francisco Bay Area at Santa Clara, California

Table 5.3: DrQA error type 3. System chooses right entity but the chosen span is too long and is flagged as incorrect.

In 11 percent of the mistakes we found that there was a specific request in

Excerpt 1:	In early 2012, NFL Commissioner Roger Goodell stated that the league planned to make the 50th Super Bowl "spectacular" and that it would be "an important game for us as a league".
Question:	What one word did the NFL commissioner use to describe what Super Bowl 50 was intended to be?
Correct Answer:	spectacular
DrQA agent:	an important game for us as a league
Excerpt 2:	The league eventually narrowed the bids to three sites: New Orleans' Mercedes-Benz Superdome, Miami's Sun Life Stadium, and the San Francisco Bay Area's Levi's Stadium.
Question:	What three stadiums did the NFL decide between for the game?
Correct Answer:	Orleans' Mercedes-Benz Superdome, Miami's Sun Life Stadium, and the San Francisco Bay Area's Levi's Stadium
DrQA agent:	San Francisco Bay Area's Levi's Stadium

Table 5.4: DrQA error type 4. System is unable to fulfill a specific request in the question.

the question formulation that the system was not able to recognize. Therefore, the system answered correctly in the sense of highlighting the right part of the text but because it did not fulfill the specific request, it failed. Examples are provided in table 5.4

Similarly, to category 3, we found that in 8 percent of the mistakes, the system finds the right entity. However, the entity consists of multiple words and the DrQA agent only chooses part of the entire entity name as an answer. This answer is then obviously marked as incorrect. Example in table 5.5 And finally, we found that to answer 6 percent of the questions, the system

Excerpt:	The Broncos finished the regular season with a 12–4 record, and denied the New England Patriots a chance to defend their title from Super Bowl XLIX by defeating them 20–18 in the AFC Championship Game.
Correct Answer:	New England Patriots
DrQA agent:	New England

Table 5.5: DrQA error type 5. System selects incomplete entity whose span contains multiple words.

would need additional outside knowledge that is not contained within the

excerpt itself (table 5.6). These questions virtually impossible to answer for the system in the current form.

Excerpt:	The league eventually narrowed the bids to three sites: New Orleans' Mercedes-Benz Superdome, Miami's Sun Life Stadium, and the San Francisco Bay Area's Levi's Stadium.
Question:	Which Louisiana venue was one of three considered for Super Bowl 50?
Correct Answer:	Mercedes-Benz Superdome
DrQA agent:	San Francisco Bay Area's Levi's Stadium

Table 5.6: DrQA error type 6. System selects incomplete entity whose span contains multiple words.

5.3 Error analysis summary

We have analyzed the errors we discovered in our model's results. We have divided them in six categories detailed in previous section with percentages displayed in table 5.7.

Category	**Percentage**
Type 1 (Agent does not understand)	33%
Type 2 (Incorrect entity)	30%
Type 3 (Too long answer)	13%
Type 4 (Too specific request)	11%
Type 5 (Incomplete entity)	8%
Type 6 (Missing information)	6%

Table 5.7: Percentages for individual error categories

We will later explore changes in error rates for individual categories on the models we trained in our experiments.

Chapter 6

Experiment description

As a main part of this thesis we have performed several experiments. We have analyzed the contents of the SQuAD dataset in detail and we have created several subdatasets. Concretely, we have created two subdatasets focusing on specific topical ares that is geographical locations (places) and famous personalities (people). Additionally we have created a third dataset that contains a specific type of questions. In our case we have chosen "Who" questions.

We will train a model on a large-scale dataset that will be represented by a version of SQuAD without the specific topic or question type. We will use this model as a baseline for our experiments.

Our goal is to see if it is possible to increase the performance of the system on the smaller, focused datasets by employing transfer learning and incremental training. We will explore the performance of our models in order to see if we are able to achieve any significant improvement. Furthermore, we will study how the models perform based on the amount of available training data.

6.1 Focused dataset selection

We have performed an in depth analysis of the SQuAD dataset. We went through all the articles contained within the dataset and we have divided them into several categories based on their main topic and focus. The categories we found were:

- Organizations (e.g. University of Kansas, General Electric),
- Personalities (e.g. George IV, Beyoncé),
- Geographical locations (e.g. Boston, Israel),
- Abstract concepts (e.g. Anti-aircraft warfare, Separation of church and state in the United States),

- Art (e.g. To Kill a Mockingbird, Queen (band)),
- Sports (e.g. FC Barcelona, Premier League),
- and Other.

We have then found a proportional representation of each of these categories within the SQuAD dataset, both in training and in development set. The proportional representation of each of the selected categories is shown in the table 6.1.

Category	%	
	TRAIN	DEV
Organizations (e.g. University of Kansas, General Electric)	7.3	8.3
Personalities (e.g. George IV, Beyoncé)	6.8	6.3
Geographical locations (e.g. Boston, Israel)	19.8	14.6
Abstract concepts (e.g. Anti-aircraft warfare)	74.5	6.3
Art (e.g. To Kill a Mockingbird, Queen (band))	2.0	2.1
Sports (e.g. FC Barcelona, Premier League)	1.4	2.1
Other	58.2	60.4

Table 6.1: The list of topics contained in the SQuAD dataset and their proportional representation

From these categories we have selected two that have sufficient representation in both training and development set. That is, we have selected geographical locations (places) and personalities (people).

After selecting the categories we have extracted the paragraphs related to these categories as well as all the questions focused on these categories from both the training and development set of the dataset. This way we have obtained two separate datasets for each topic. We have a large, general dataset, that does not contain questions on articles about "places" and smaller focused dataset containing only questions on articles about "places". The same equivalently for the "people" topic. Tables 6.2 and 6.3 show the size of the datasets in terms of number of question paragraph pairs as well as their size compared to the original size of the entire SQuAD dataset.

As for the third dataset we use in our experiments, we have already talked about the large diversity in question answer types in the SQuAD dataset in the chapter 3. Since we have decided to exploit this in one of our experiments, let us look into the different question answer types in the dataset in detail. Table 6.4 shows percentages of answers sorted into different categories contained in the dataset.

Based on these categories, we have decided to select our third focused dataset. We have filtered all of the "Who" question answered pairs from the

Dataset	Train		Dev	
	QA pairs	SQuAD %	QA pairs	SQuAD %
PLACES dataset	20,009	22.8	1,278	12.1
SQuAD without PLACES	67,590	77.2	9,292	87.9
SQuAD	87,599	100	10,570	100

Table 6.2: Size of the PLACES dataset compared to SQuAD

Dataset	Train		Dev	
	QA pairs	SQuAD %	QA pairs	SQuAD %
PEOPLE dataset	8,862	10.1	1,235	11.7
SQuAD without PEOPLE	78,737	89.9	9,335	88.3
SQuAD	87,599	100	10,570	100

Table 6.3: Size of the PEOPLE dataset compared to SQuAD

SQuAD dataset. We believe that selecting only one specific type of questions might lead to more interesting results, as the WHO focused dataset will contain more information that the large-scale models that we expose to it previously haven't seen.

We have prepared the WHO focused and general dataset in the same way as datasets for PLACES and PEOPLE. Table 6.5 details the size of the WHO focused dataset.

Answer type	Percentage	Example
Date	8.9%	19 october 1512
Other Numeric	10.9%	12
Person	12.9%	Thomas Coke
Location	4.4%	Germany
Other entity	15.3%	ABC Sports
Common Noun Phrase	31.8%	property damage
Others	15.8%	quietly

Table 6.4: Different answer categories in the SQuAD dataset[2]

Dataset	Train		Dev	
	QA pairs	SQuAD %	QA pairs	SQuAD %
WHO dataset	9,430	10.8	1,312	12.4
SQuAD without WHO	78,169	89.2	9,258	87.6
SQuAD	87,599	100	10,570	100

Table 6.5: Size of the WHO dataset compared to SQuAD

6.2 Experiment Details

We will now describe the performed experiments in detail. Firstly, we take the DrQA system to train a model on the general part of our prepared datasets. That is we train three models, one on the dataset containing no "place" paragraphs, one on the dataset containing no "people" paragraphs and one on the dataset containing no "Who" questions. This way we simulate having a pretrained general model trained on a large scale question answering dataset that does not contain a topic of our dataset that we want to retrain the model for. We use the same parameters as in chapter 5. We use used standard 840B glove embeddings with 300 dimensional vectors[9], we fix all the embeddings except for 1000 most common, we set validation patience to 5 with validation step 3600 seconds. For the paragraph and question encoding we use three layer bidirectional LSTM with 128 hidden units. We use batch size 32 for the training examples and we apply dropout 0.3 to word embeddinbgs and all LSTM hidden units[3].

After the models have been trained, we test their performance on the development set of the focused datasets. This way we get a baseline for the trained models and we can see changes in their performance during the transfer learning phase.

After setting the baseline we setup the incremental training for the generic models. We take each of the pretrained models and we retrain it on the corresponding focused dataset. We use the same parameters as for the training of the models, but we set lower validation interval so that we have more detailed information about the training. For each of the training datasets we perform several experiments with different volumes of training data. We want to be able to tell how the performance is improving with increasing the amount of training data. This way we should be able to tell mow much training data is needed in order to significantly boost the performance. The different amounts of the training data for individual datasets are displayed in the table 6.6

Finally, we will evaluate results of the experiments.

Dataset	QA pairs	% of train data	% of general dataset
PLACES 1000	1,000	5 .00	1.48
PLACES 2000	2,000	10.00	2.96
PLACES 5000	5,000	24.99	7.40
PLACES 10000	10,000	49.98	14.80
PLACES FULL	20,009	100	29.60
PEOPLE 500	500	5.64	0.64
PEOPLE 1000	1,000	11.28	1.27
PEOPLE 3000	3,000	33.85	3.81
PEOPLE 5000	5,000	56.42	6.35
PEOPLE FULL	8,862	100	11.26
WHO 500	500	5.30	0.64
WHO 1000	1,000	10.60	1.28
WHO 3000	3,000	31.81	3.84
WHO 5000	5,000	53.02	6.40
WHO FULL	9,430	100	12.06

Table 6.6: Sizes of the different experimental training sets

Chapter 7

Experiment Results

In this chapter we will evaluate conducted experiments. First, we will focus on the experiments with the PLACES and PEOPLE datasets as they are in many ways similar. Secondly, we will evaluate the performance on the WHO dataset. We will compare the results of the individual experiments and comment on them.

7.1 PEOPLE and PLACES dataset experiments

Table 7.1 displays the results of the experiments for the PLACES dataset, whereas table 7.2 shows the results for the PEOPLE dataset. From the results we can see, that when training on small amounts of training data we have experienced a significant drop in the model performance. However, as the amount of training data increased, the performance of the model increased as well.

This trend is also shown in figures 7.1 resp. 7.2. In both figures we can see the rising performance of the model with the rising amount of training data. It is important to note, that we have achieved a 2.60 point (3.31%) for F1 for the PLACES dataset and only 1.55 (2.20%) increase for the PEOPLE model. This is most likely caused by the fact that the PLACES training set is approximately two times larger than the full PEOPLE training set.

We have documented the increase in performance on both PLACES and PEOPLE datasets. However, the performance does not increase until a large amount of training data is available for the system to learn from. While for the purpose of our experiments it was not an issue, it is possible that there is not going to be required amount of data available when trying to use make use of this technique in practice.

Dataset	Performance		Improvement	
	EM	F1	EM	F1
Generic model	67.68	78.43	-	-
PALCES 1000	59.39	71.21	-8.29	-7.22
PALCES 2000	61.74	73.63	-5.94	-4.80
PALCES 5000	66.43	77.29	-1.25	-1.14
PALCES 10000	69.48	80.16	1.80	1.73
PLACES FULL	**70.81**	**81.03**	**3.13**	**2.60**

Table 7.1: The table shows performance of the models trained with different amounts of training data on the PLACES dataset. From the results we can see that the performance improves with growing size of the training set. However, it is also clear that in order to achieve better performance then the generic model, we need large amount of training data.

Figure 7.1: The figure shows how the performance of the model trained on the PLACES dataset changes based on the different amount of training data.

Dataset	Performance		Improvement	
	EM	F1	EM	F1
Generic model	58.95	70.47	-	-
PEOPLE 500	54.74	65.16	-4.21	-5.31
PEOPLE 1000	54.25	64.52	-4.70	-5.95
PEOPLE 3000	55.22	66.58	-3.73	-3.89
PEOPLE 5000	57.81	68.99	-1.14	-1.48
PEOPLE FULL	**61.46**	**72.02**	**2.51**	**1.55**

Table 7.2: The table shows performance of the models trained with different amounts of training data on the PEOPLE dataset.

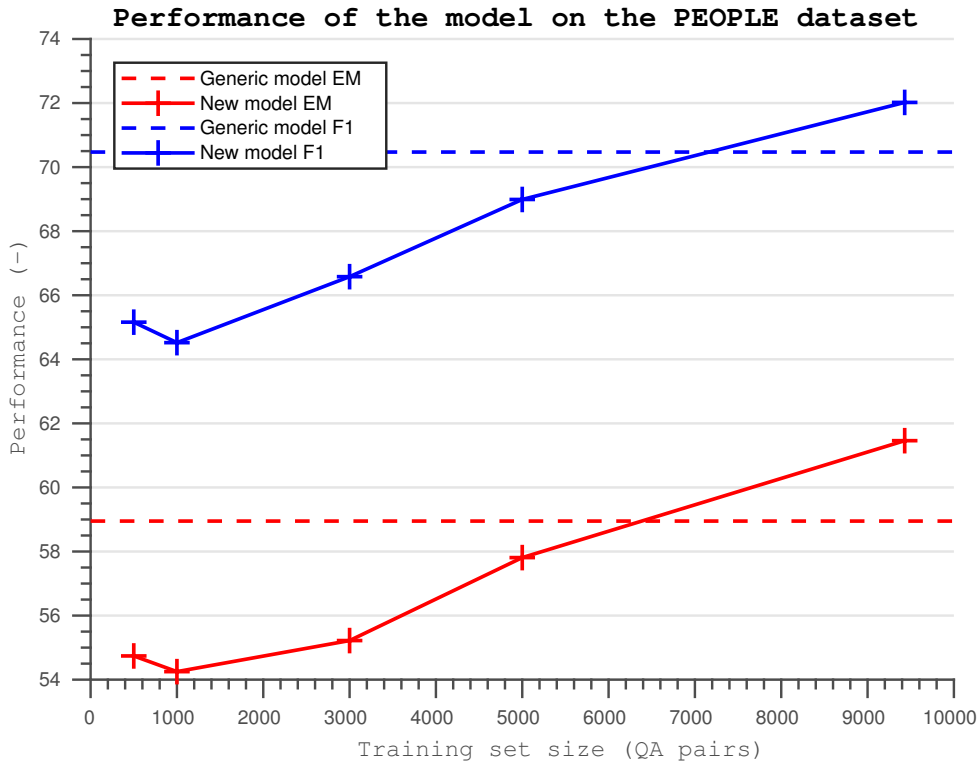


Figure 7.2: The figure shows how the performance of the model model trained on the PEOPLE dataset changes based on the different amount of training data.

7.2 WHO dataset experiments

We will now look into the results of the experiment for the WHO dataset. This dataset is different from the previous two. While the PLACES and PEOPLE focus on a single topical area, they contain broad range of different question and answer types. The WHO dataset focuses only on one type of questions, "Who" questions. Because of this we are able to say that the syntactical variety of the question answer pairs in the WHO dataset is significantly different from the remainder of the SQuAD datstes without the "who" questions.

We can see the results of the experiments on the WHO dataset in the table 7.3.

Dataset	Performance		Improvement	
	EM	F1	EM	F1
Generic model	66.31	73.85	-	-
WHO 500	63.43	69.19	-2.97	-4.66
WHO 1000	65.09	71.25	-1.22	-2.60
WHO 3000	69.36	75.35	3.05	1.05
WHO 5000	71.27	76.89	4.96	3.04
WHO FULL	**71.95**	**78.35**	**5.64**	**4.50**

Table 7.3: The table shows performance of the models trained with different amounts of training data on the WHO dataset. From the results we can see that the performance improves with the growing size of the training set.

The figure 7.3 depicts how did the performance of the model change based on the available amount of training data.

We can clearly see that with increasing amount of training data, the performance of the model increased as well. When we made available only 500 QA pairs for the training, the performance of the model did not improve at all. In fact for both 500 and 1000 QA pair training set the performance of the model decreased for up to 4.6 points in the F1 score. This is obviously caused by the insufficient amount of training data. However, when we look at the results for the 3000 QA pairs training sets, we can see the increase in the model performance. Even the performance increase on the 3000 QA training set is significant. We were able to achieve results 69.36 for EM which means increase for 3.05 points and 75.35 for F1, that is increase for 1.05 points.

As for the entire WHO questions training set. When we retrained the generic model on the full dataset, we were able to achieve results of 71.95 EM and 78.35 F1 on the validation set. That is increase of 5.64 points (8.5%) for EM and 4.50 (6.1%) for F1. This is a significant increase in the model performance for this task.

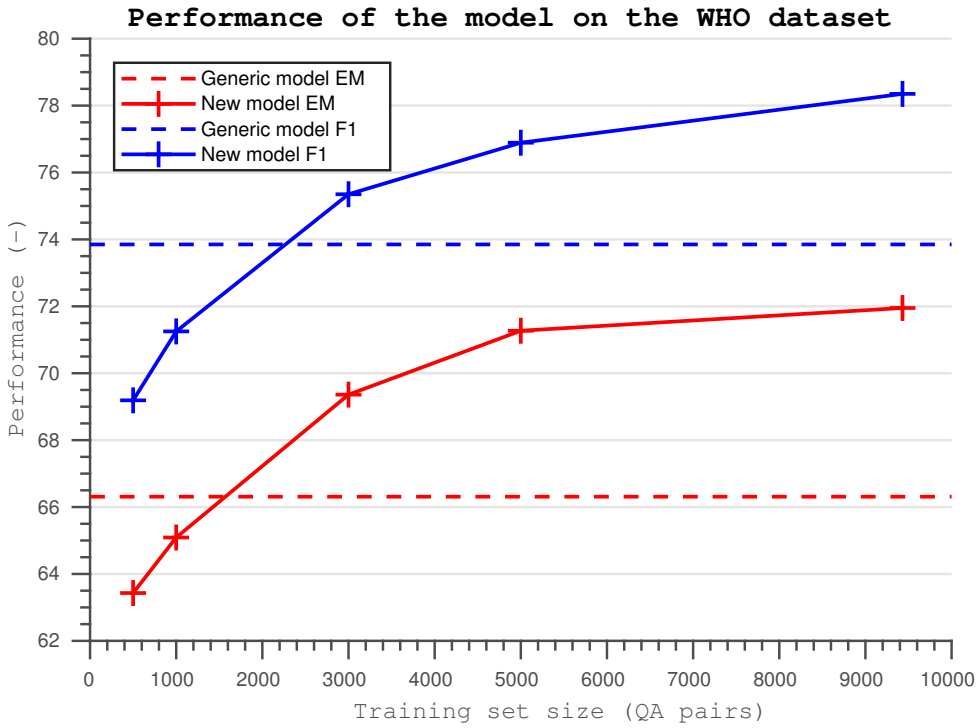


Figure 7.3: The figure shows how the performance of the model model trained on the WHO dataset changes based on the different amount of training data.

We can now see, that the increase in performance on the WHO dataset is more significant that that on the PEOPLE and PLACES datasets. We believe that this is caused by the fact that the WHO dataset focuses primarily on a very specific type of question that has not been represented in the large dataset used on training the generic model. The WHO dataset was then able to provide the pretrained generic model with new knowledge about the task and domain which led to the increased performance as can be seen in table 7.3.

We wanted to further explore this hypothesis, which is why we performed an error analysis on the WHO model performance similar to the one we performed on the performance of DrQA system on the SQuAD task. The table 7.4 shows the results of the performed analysis. We have annotated 125 (25%) of the incorrectly answered questions. The error categories remain the same.

Figure 7.4 shows a comparison between the DrQA error rates on standard SQuAD dataset and the error rates of the retrained model on the WHO dataset. We can see significant decrease in the first error category. This means that for the specific type of question (the datasets consists primarily of who questions) the system was able to recognize question patterns well enough to understand what entity type to expect. This means that much larger number

Category	Percentage
Type 1 (Agent does not understand)	24%
Type 2 (Incorrect entity)	35%
Type 3 (Too long answer)	22%
Type 4 (Too specific request)	9%
Type 5 (Incomplete entity)	6%
Type 6 (Missing information)	4%

Table 7.4: Percentages for individual error categories for the WHO dataset

of errors then falls into categories 2 and 3. The system is usually able to tell the type of entity it needs in order to answer posed question, but it either interchanges is for a different entity of the same type or it predict too long answer span.

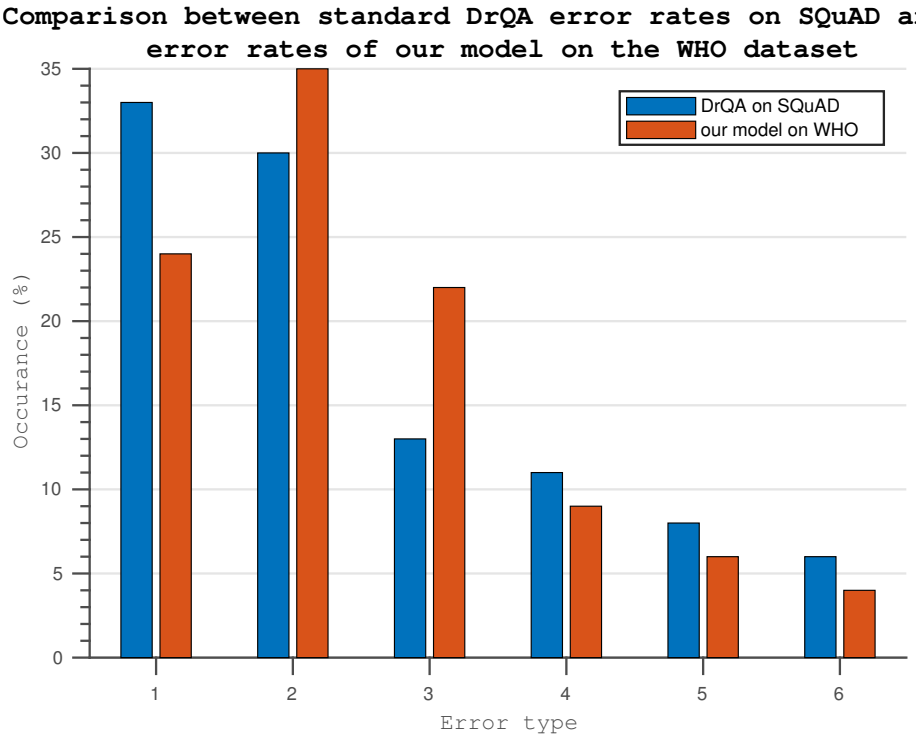


Figure 7.4: Comparison between the DrQA error rates on standard SQuAD dataset and the error rates of the retrained model on the WHO dataset. We can see significant decrease in the first error category, that is the category where system completely misunderstood the question.

Conclusion

The goal of this thesis was to explore the possibilities and applications of transfer learning in automatic question answering. Specifically the automatic question answering with application suitable for the conversational AI and the Alquist bot in particular.

We have summarized the sate of the art techniques in question answering and reading comprehension. We focused on various QA tasks and datsets as well as on new and promising QA approaches, including possible uses of transfer learning.

We have described in detail recurrent neural networks, specifically long-short term memory networks. We have described how they work, their strong suits and their shortcomings. We have also focused on their uses in different areas of machine learning. We have also described basic principles of transfer learning and stated the reasons why have we chosen to use it for this task (Chapter 2).

We have given a detailed description of the DrQA Document reader system, that we employ in our experiments. We have described both its function and implementation but also its performance on SQuAD dataset and various other tasks (Chapter 4).

In the practical part of the thesis we described three transfer learning experiments on three dataset that we created by extracting them from the SQuAD dataset. We have taken the generic large SQuAD dataset without its small subset and using DrQA system trained a generic model. Then we tested the performance of the generic model on a small focused dataset. After we got a baseline this way, we used transfer learning to increase the performance of the generic model on the given dataset.

We have reached following results. We documented an increase in performance for all three models when using the full training set. Specifically we achieved a 2.60 point (3.31%) increase for F1 and 3.13 point (4.62%) increase for EM for the PLACES dataset, 1.55 point (2.20%) increase for F1 and 2.51 point (4.25%) increase for EM for the PEOPLE dataset, and finally increase

of 5.64 points (8.5%) for EM and 4.50 (6.1%) for F1 for the WHO dataset.

We have performed an error analysis of the results of DrQA system on the SQuAD dataset as well as an error analysis on the errors of our system on the WHO dataset. We were able to observe significant decrease in the errors where system completely misunderstood the question. This means that for the specific type of question (the datasets consists primarily of who questions) the system was able to recognize question patterns well enough to understand what entity type to expect.

We have shown that transfer learning in the QA task can be a valuable tool. The results that we achieved show that the increase in the performance of the tested systems is significant.

As for where to continue with this work. We would like to focus on integration of transfer learning into the Alquist conversational system. It would be a great advantage if we were able to quickly boost Alquist's performance in certain areas of conversation by utilizing previously learned knowledge. Another area of focus is the selection of the suitable dataset for transfer learning. We can see, that the results on the PLACES dataset are better than the results on the PEOPLE dataset despite the fact that both of the datasets were chosen by the same method. This could to a certain degree be caused by the smaller size if the PEOPLE dataset (the PLACES dataset is approximately 2x larger). However, it is also possible that the QA pairs in the people dataset are much more diverse, which could cause the worse results. We would like to analyze all three datasets we ran experiments on in the future. Then we should be able to determine why are the results on the examined datasets different.

To conclude, we have shown that transfer learning in the QA task can be a valuable tool. The results that we achieved show that the increase in the performance of the tested systems is significant. We have also shown that the performance of the system increases with the increasing amount of training data. Small amounts of data actually led to decrease in performance. However, when we got to the 3000 QA pairs for the WHO dataset and 10000 QA pairs for the PLACES dataset we have seen a significant increase in performance.

We have shown that it is possible to create small domain models by retraining large general purpose models and that it can lead to significant performance increase. However, it is important to select a compact domain in order for the retraining to lead to positive results. We have seen much better results with smaller amount of data on the WHO dataset then on either of the other two tested datasets.

Bibliography

[1] Kapashi, D.; Shah, P. Answering Reading Comprehension Using Memory Networks. 2015. Available from: https://cs224d.stanford.edu/reports/KapashiDarshan.pdf

[2] Rajpurkar, P.; Zhang, J.; Lopyrev, K.; et al. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. *CoRR*, volume abs/1606.05250, 2016, 1606.05250. Available from: http://arxiv.org/abs/1606.05250

[3] Chen, D.; Fisch, A.; Weston, J.; et al. Reading Wikipedia to Answer Open-Domain Questions. *CoRR*, volume abs/1704.00051, 2017, 1704.00051. Available from: http://arxiv.org/abs/1704.00051

[4] Pichl, J.; Matulík, M.; Marek, P.; et al. Alquist: The Alexa Prize Socialbot. 2017. Available from: https://s3.amazonaws.com/alexaprize/2017/technical-article/alquist.pdf

[5] Ram, A.; Prasad, R.; Khatri, C.; et al. Conversational AI: The Science Behind the Alexa Prize. 2017. Available from: https://s3.amazonaws.com/alexaprize/2017/technical-article/alexaprize.pdf

[6] *MCTest: A Challenge Dataset for the Open-Domain Machine Comprehension of Text*, October 2013. Available from: https://www.microsoft.com/en-us/research/publication/mctest-challenge-dataset-open-domain-machine-comprehension-text/

[7] Yang, Y.; Yih, S. W.-t.; Meek, C. WikiQA: A Challenge Dataset for Open-Domain Question Answering. ACL – Association for Computational Linguistics, September 2015. Available from: https://www.microsoft.com/en-us/research/publication/wikiqa-a-challenge-dataset-for-open-domain-question-answering/

[8] Hill, F.; Bordes, A.; Chopra, S.; et al. The Goldilocks Principle: Reading Children's Books with Explicit Memory Representations. *CoRR*, volume abs/1511.02301, 2015, 1511.02301. Available from: http://arxiv.org/abs/1511.02301

[9] Pennington, J.; Socher, R.; Manning, C. D. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. Available from: http://www.aclweb.org/anthology/D14-1162

[10] Mikolov, T.; Chen, K.; Corrado, G.; et al. Efficient Estimation of Word Representations in Vector Space. *CoRR*, volume abs/1301.3781, 2013, 1301.3781. Available from: http://arxiv.org/abs/1301.3781

[11] Joulin, A.; Grave, E.; Bojanowski, P.; et al. Bag of Tricks for Efficient Text Classification. *CoRR*, volume abs/1607.01759, 2016, 1607.01759. Available from: http://arxiv.org/abs/1607.01759

[12] Liu, R.; Hu, J.; Wei, W.; et al. Structural Embedding of Syntactic Trees for Machine Comprehension. *CoRR*, volume abs/1703.00572, 2017, 1703.00572. Available from: http://arxiv.org/abs/1703.00572

[13] Zhang, J.; Zhu, X.; Chen, Q.; et al. Exploring Question Understanding and Adaptation in Neural-Network-Based Question Answering. *CoRR*, volume abs/1703.04617, 2017, 1703.04617. Available from: http://arxiv.org/abs/1703.04617

[14] Chung, Y.; Lee, H.; Glass, J. R. Supervised and Unsupervised Transfer Learning for Question Answering. *CoRR*, volume abs/1711.05345, 2017, 1711.05345. Available from: http://arxiv.org/abs/1711.05345

[15] Domain Adaptation in Question Answering. *CoRR*, volume abs/1702.02171, 2017, withdrawn., 1702.02171. Available from: http://arxiv.org/abs/1702.02171

[16] Britz, D. Introduction to RNNs. 2015. Available from: http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

[17] Karpathy, A. The Unreasonable Effectiveness of Recurrent Neural Networks. 2015. Available from: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[18] Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*. MIT Press, 2016.

[19] Olah, C. Understanding LSTM Networks. 2015. Available from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[20] Mikolov, T.; Karafiát, M.; Burget, L.; et al. Recurrent neural network based language model. In *INTERSPEECH*, edited by T. Kobayashi; K. Hirose; S. Nakamura, ISCA, 2010, pp. 1045–1048. Available from: http://dblp.uni-trier.de/db/conf/interspeech/interspeech2010.html#MikolovKBCK10

[21] Sutskever, I.; Martens, J.; Hinton, G. E. Generating Text with Recurrent Neural Networks. In *ICML*, edited by L. Getoor; T. Scheffer, Omnipress, 2011, pp. 1017–1024. Available from: http://dblp.uni-trier.de/db/conf/icml/icml2011.html#SutskeverMH11

[22] Liu, S.; Yang, N.; Li, M.; et al. A Recursive Recurrent Neural Network for Statistical Machine Translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, Maryland: Association for Computational Linguistics, June 2014, pp. 1491–1500. Available from: http://www.aclweb.org/anthology/P14-1140

[23] Graves, A.; Jaitly, N. Towards End-To-End Speech Recognition with Recurrent Neural Networks. In *Proceedings of the 31st International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, volume 32, edited by E. P. Xing; T. Jebara, Bejing, China: PMLR, 22–24 Jun 2014, pp. 1764–1772. Available from: http://proceedings.mlr.press/v32/graves14.html

[24] Bengio, Y.; Simard, P.; Frasconi, P. Learning Long-term Dependencies with Gradient Descent is Difficult. *Trans. Neur. Netw.*, volume 5, no. 2, Mar. 1994: pp. 157–166, ISSN 1045-9227, doi:10.1109/72.279181. Available from: http://dx.doi.org/10.1109/72.279181

[25] Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.*, volume 9, no. 8, Nov. 1997: pp. 1735–1780, ISSN 0899-7667, doi:10.1162/neco.1997.9.8.1735. Available from: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[26] Gers, F. A.; Schmidhuber, J. A.; Cummins, F. A. Learning to Forget: Continual Prediction with LSTM. *Neural Comput.*, volume 12, no. 10, Oct. 2000: pp. 2451–2471, ISSN 0899-7667, doi:10.1162/089976600300015015. Available from: http://dx.doi.org/10.1162/089976600300015015

[27] Schuster, M.; Paliwal, K. Bidirectional Recurrent Neural Networks. *Trans. Sig. Proc.*, volume 45, no. 11, Nov. 1997: pp. 2673–2681, ISSN 1053-587X, doi:10.1109/78.650093. Available from: http://dx.doi.org/10.1109/78.650093

[28] Singh, S. Deep Bidirectional LSTM based RNN for Casual Speech to Clear Speech conversion. 2017. Available from: https://www.evl.uic.edu/documents/shiwangisinghresearchprojectfinal-2.pdf

[29] Olah, C. Neural Networks, Types, and Functional Programming. 2015. Available from: http://colah.github.io/posts/2015-09-NN-Types-FP/

[30] Pan, S. J.; Yang, Q. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, volume 22, no. 10, Oct 2010: pp. 1345–1359, ISSN 1041-4347, doi:10.1109/TKDE.2009.191.

[31] Karpathy, A.; Fei-Fei, L. Deep Visual-Semantic Alignments for Generating Image Descriptions. *IEEE Trans. Pattern Anal. Mach. Intell.*, volume 39, no. 4, Apr. 2017: pp. 664–676, ISSN 0162-8828, doi:10.1109/TPAMI.2016.2598339. Available from: https://doi.org/10.1109/TPAMI.2016.2598339

[32] Xu, H.; Saenko, K. Ask, Attend and Answer: Exploring Question-Guided Spatial Attention for Visual Question Answering. *CoRR*, volume abs/1511.05234, 2015, 1511.05234. Available from: http://arxiv.org/abs/1511.05234

[33] Chiticariu, L.; Krishnamurthy, R.; Li, Y.; et al. Domain Adaptation of Rule-based Annotators for Named-entity Recognition Tasks. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 1002–1012. Available from: http://dl.acm.org/citation.cfm?id=1870658.1870756

[34] McClosky, D.; Charniak, E.; Johnson, M. Automatic Domain Adaptation for Parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, ISBN 1-932432-65-5, pp. 28–36. Available from: http://dl.acm.org/citation.cfm?id=1857999.1858003

[35] Bollacker, K.; Evans, C.; Paritosh, P.; et al. 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, New York, NY, USA: ACM, 2008, ISBN 978-1-60558-102-6, pp. 1247–1250, doi:10.1145/1376616.1376746. Available from: http://doi.acm.org/10.1145/1376616.1376746

[36] Auer, S.; Bizer, C.; Kobilarov, G.; et al. DBpedia: A Nucleus for a Web of Open Data. In *Proceedings of the 6th International The Semantic*

Web and 2Nd Asian Conference on Asian Semantic Web Conference, ISWC'07/ASWC'07, Berlin, Heidelberg: Springer-Verlag, 2007, ISBN 3-540-76297-3, 978-3-540-76297-3, pp. 722–735. Available from: http://dl.acm.org/citation.cfm?id=1785162.1785216

[37] Hermann, K. M.; Kociský, T.; Grefenstette, E.; et al. Teaching Machines to Read and Comprehend. *CoRR*, volume abs/1506.03340, 2015, 1506.03340. Available from: http://arxiv.org/abs/1506.03340

[38] Manning, C. D.; Surdeanu, M.; Bauer, J.; et al. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, 2014, pp. 55–60. Available from: http://www.aclweb.org/anthology/P/P14/P14-5010

[39] Lee, K.; Kwiatkowski, T.; Parikh, A. P.; et al. Learning Recurrent Span Representations for Extractive Question Answering. *CoRR*, volume abs/1611.01436, 2016, 1611.01436. Available from: http://arxiv.org/abs/1611.01436

[40] Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, volume abs/1409.0473, 2014, 1409.0473. Available from: http://arxiv.org/abs/1409.0473

[41] Xiong, C.; Zhong, V.; Socher, R. Dynamic Coattention Networks For Question Answering. *CoRR*, volume abs/1611.01604, 2016, 1611.01604. Available from: http://arxiv.org/abs/1611.01604

[42] Wang, Z.; Mi, H.; Hamza, W.; et al. Multi-Perspective Context Matching for Machine Comprehension. *CoRR*, volume abs/1612.04211, 2016, 1612.04211. Available from: http://arxiv.org/abs/1612.04211

[43] Seo, M. J.; Kembhavi, A.; Farhadi, A.; et al. Bidirectional Attention Flow for Machine Comprehension. *CoRR*, volume abs/1611.01603, 2016, 1611.01603. Available from: http://arxiv.org/abs/1611.01603

[44] Rajpurkar, P. The Stanford Question Answering Dataset. 2016. Available from: https://rajpurkar.github.io/SQuAD-explorer/

[45] Miller, A. H.; Feng, W.; Fisch, A.; et al. ParlAI: A Dialog Research Software Platform. *arXiv preprint arXiv:1705.06476*, 2017.

Appendix A

Acronyms

SQuAD The Stanford Question Answering Dataset

RNN Recurrent Neural Network

ReLU Rectified Linear Unit

LSTM Long Short Term Memory

EM Exact Match

QA Question Answering

AI Artificial Intelligence

NLP Natural Language Processing

Appendix B

ParlAI framework

A snapshot of the ParlAI framework used in this thesis is attached on the CD

Appendix C

Topically Focused Datasets

The topically focused datasets created over the course of this thesis are all attached on the CD