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 nal thesis.

In Prague on 8th January 2018 ....

Czech Technical University in Prague
Faculty of Electrical Engineering

Citation of this thesis

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

Abstrakt

Tato prace se zabyva porozumenm nestruktorovanemu textu a automatickym odpovdanm na otazky. Zameruje se na vyuzit principu prenosu znalost. Prace popisuje nekolik experimentu ktere ukazuj zlepsen vykonu modelu natrenovaneho na obecne bazi dat (SQuAD) preneseneho a pretrenovaneho, na mens tematicky speci ckou bazi dat.

Klcova slova Zaverecna prace, Automaticke odpovdan na dotazy, Prenos znalost, SQuAD, DrQA.

Abstract

This thesis deals with comprehension of unstructured text and question answering. Speci cally we focus the bene ts 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 speci c datasets. We explore in uence of transfer learning on datasets focusing on speci c 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 eld of arti cial 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 eld 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] usesstructural 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 di erent 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 xed number of computational steps. They accept xed size vector as input and produce a xed 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 xed 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 gure 2.1. What we did by unrolling is we displayed the entire sequence of inputs and outputs owing 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].

(b) LSTM gate

(a) LSTM cell state

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 (gure 2.2), LSTM networks are more complex, having four interconnected repeating layers (gure 2.3).

The main reason why are the LSTMs able to keep information through many steps is the cell state[19]. The cell state ows through the entire network (gure 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

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

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-

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

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 $\tilde{C}_t$ via a tanh layer. These two are then processed in the next step[19].

In the third step (gure 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 rst multiply the old cell state C_{t-1} by the vector f_t in order to forget information we decided to eliminate in the rst step. Then we add $\tilde{C}_t i_t$ 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 nally, 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 nal output h_t of the LSTM cell[19].