Natural Language Processing for Scientific Research

AUGUST 2018

AUTHOR:
Amina
Manafli

CERN IT-DI-OPL

SUPERVISORS(S):
Taghi Aliyev

CERN openlab

Abstract

The goal of this Openlab project is to create a Smart Data Analytics Platform for Science that will host analytical tools, publish data, share resources, interact with bots, collaborate and build communities of researchers with various backgrounds in a single ecosystem. With this platform we will address solutions of such research problems as finding reliable sources, lack or excess of data, interacting with researchers of different backgrounds and finding partners for research work, creating a set of common definitions and environments. Thus, the underlying goal is the promotion of especially innovative, interdisciplinary research projects, as well as dealing with high entrance barriers of the research field.
One of the crucial stages of building such platform is creating an intelligent chat bot, able to analyze and answer questions, as accessing and effectively using large amounts of data can be the most challenging and time consuming part of the research work. To do that we need to direct attention to Question Answering models, methods of Natural Language Processing and find a reliable dataset, for their further development and implementation.

Contents

List of Figures

1. Introduction

In an era where interdisciplinary research is considered critical to solving most scientific problems, getting started with the research career remains a challenging stage for any scientist. As found by International Association of Scientific, Technical and Medical Publishers, publication activity is growing with high rates in all fields, with about 1.8 million new scientific papers published every year and number of publishing scientists increasing by 4-5% annually while scientific research centres all around the world are producing ever growing amounts of data that needs to be processed [3]. Therefore the number of potentially related or useful and accessible research papers is increasing, also creating enlarging amounts of publications that contain irrelevant or repetitive results, turning the initial step of finding the right sources into a tedious task.

The Smart Data Analytic Platform for Scientists is aiming to help in overcoming the above mentioned barrier, therefore creating an environment facilitating scientists in storing, analyzing and sharing data, communicating with other users and speeding up source searching process with the use of smart chat bots. We will further focus on the last component of the platform, as it is one of the main ways of simplifying the task of finding the right sources for future work. Smart bot, as part of the platform, is required to have access to large data base of papers and articles, which, together with user friendly interface and implemented machine learning algorithms, turns the bot into a smart search engine.

Smart chatbots use a dialog system to process an interaction with the users. To process human information chatbot goes through following main steps. The first step is converting human input into an understandable context through input recognizers and decoders, which can analyze text. The next step is applying Natural Language Processing to analyze the plain text and search for semantics. All the while the input is managed and processed by a dialog manager to ensure a correct flow of information from and to the user. The dialog manager also makes sure that questions or issues are assigned to the right task manager and solved. After the tasks are solved the output manager translates the solution into human like output. This is done through a natural language generator to mimic human speech [6].

To implement more comprehensive search, at the second step we are addressing such issues of Natural Language Processing as Information Retrieval and Question answering.

Machine reading comprehension (MRC) is a challenging task in natural language processing, which requires that machine can read, understand, and answer questions about a text. Information Retrieval (IR) can be defined broadly as the study of how to determine and retrieve from a corpus of stored information the portions which are relevant to particular information needs. The information may be stored in a highly structured form or in an unstructured form, depending upon its application. A user seeks certain information which, therefore he has to express as a request for information in one form or another. Thus IR is concerned with the determining and retrieving of information that is relevant to his infor-

mation need as expressed by his request and translated into a query which conforms to a specific information retrieval system(IRS) used.

After the user has obtained the portion of information which deemed relevant by the system, the user might want to investigate further the content of the information by asking more specific questions and get specific answers together with sentences that support the answers. This kind of request for specific information brings us into question and answering (QA) systems. Question answering composes of reading comprehension tasks that demonstrates the understanding of the system about the document and means to show and to build up the meaning representation based on syntactic and semantic knowledge as well as the world knowledge on certain domain area under investigation [8].

In this report the Question Answering part of the project is covered, where we are setting base for the future work. The structure of the report corresponds to the order of steps taken. Therefore, Section 2 describes three question answering datasets that can be used for testing and training the chat bot, each of them demonstrates new approach to reading comprehension and differs by the source and volume of contained data. Some of the possible and best performing architectures are described in Section 3 with demonstration of advantages and disadvantages of used methods and some initial results obtained with them are mentioned in Section 4. Section 5 concludes the research and motivates the discussion on this topic and necessary future steps.

2. Dataset

Bot has to be capable of proceeding search on required information in local and global scales, analyzing data on answer suitability for given query. Thus, question answering problem of Natural Language Processing (NLP) became our main focus. The base of a machine learning project is training data to be used to teach the machine. The quality of the training depends on the quality of the data input. Finding or creating the perfect data set is not always a simple matter. It was decided to concentrate on the model rather than data set generation. Therefore several datasets were studied for potential of further training our model.

2.1 Shaping Answers with Rules through Conversation (ShARC)

The dataset is built up from 948 distinct snippets of rule text. Each has an input question and a ”dialog tree”. At each step in the dialog, there is a followup question posed and the tree branches depending on the answer to the followup question (yes/no). The ShARC dataset is comprised of all individual utterances from every tree, i.e. every possible point/node in any dialog tree. In addition, there are 6637 scenarios that provide more information, allowing some questions in the dialog tree to be skipped as the answers can be inferred from the scenario. When combined with scenarios and negative sampled scenarios, the total number of distinct utterances became 37087. As a final step, utterances were removed where the scenario referred to a portion of the dialog tree that was unreachable for that utterance, leaving a final dataset size of 32436 utterances.

It is divided into train, development and test sets such that each dataset contains approximately the same proportion of sources from each domain, targeting a 70%/10%/20% split. In terms of deciding whether the answer is a Yes, No or some follow-up question, the three annotators reach an answer agreement of 72.3% [4].

The size of the ShARC dataset may not be sufficient enough for training neural models. However the approach shown in this paper is an example of more comprehensive question answering, based on generating questions for getting more information from the user to give correct answer. This may be disadvantageous for general purpose NLP systems, however it is appropriate for an implementation in a QA system, where chat bot - user interaction takes place.

2.2 Microsoft MAchine Reading Comprehension (MS MARCO) [5]

The dataset comprises of 1,010,916 anonymized questions sampled from Bings search query logs each with a human generated answer and 182,669 completely human rewritten

generated answers. In addition, the dataset contains 8,841,823 passages-extracted from 3,563,535 web documents retrieved by Bing that provide the information necessary for curating the natural language answers. Therefore it may be more representative of a natural distribution of information need that users may want to satisfy using, say, an intelligent assistant. A question in the MS MARCO dataset may have multiple answers or no answers at all.

Corresponding to each question, a set of extracted passages from documents retrieved by Bing is provided in response to the question. The passages and the documents may or may not actually contain the necessary information to answer the question. Unanswerable questions are included in the dataset because the ability to recognize insufficient (or conflicting) information that makes a question unanswerable is important to develop for an MRC model.

As it was mentioned, dataset is based on the Bing search query, therefore the question formulations, are often complex, ambiguous, and may even contain typographical and other speech errors. This makes MS Marco dataset a suitable choice for QA system, as human factor is taken into consideration. However the results demostrated in the MS Marco Challenge Leaderboard do not reach Human Performance, therefore are less favorable in analysing and further use.

2.3 The Stanford Question Answering Dataset (SQuAD) [7]

Previously publicly available QA datasets were human annotated and relatively small in size, which made them unsuitable for high capacity models such as deep neural networks. The Stanford Question Answering Dataset (SQuAD), introduced by Rajpurkar et. al. (2016), is a crowdsourced reading comprehension dataset consisting of questions concerning context paragraphs of text and their associated answers as subspans of the associated context paragraph. This dataset consists of 107,785 question answer pairs and 53,775 unanswerable questions on 536 articles, which is significantly larger than all the previous human annotated datasets. The size of the SQuAD dataset has enabled the development of much more expressive models for the QA task. A unique feature of SQuAD is that all answers are entailed by the corresponding contexts.

Additionally, compared with other question answer datasets, where the answers are single words or entities, SQuAD answers can be much longer phrases and often include non-entities, or, in the latest version, SQuAD includes unanswerable questions and questions with plausible answers. The QA task with the SQuAD dataset can be formulated as identifying the span of words in a document (context) that answers a given question.

The SQuAD dataset is a popular metric for evaluating modern reading comprehension models. To complete that, SQuAD 2.0 is more challenging dataset to work with. A state-of-the-art model achieves only 66.3% F1 score when trained and tested on SQuAD 2.0, whereas human accuracy is 89.5% F1, a full 23.2 points higher. The same model architecture trained on SQuAD 1.1 gets 85.8% F1, only 5.4 points worse than humans.

It was decided to use SQuAD in the further work as it contains large scales of data, unanswerable questions and is well studied, therefore providing a number of existing and tested models and methods, available for public use.

3. Models

3.1 QANet [1]

Most end-to-end machine reading and question answering models are based on recurrent neural networks (RNNs) with attention. These models are often slow for both training and inference due to the sequential nature of RNNs, especially for long texts. A new Q&A architecture called QANet was published in 2018 by Adams Wei Yu (link) and it does not require recurrent networks: Its encoder consists exclusively of convolution and self-attention, where convolution is used to capture local structure of the text, while self-attention learns the global interaction between each pair of words. Better speed results of the model are caused by feed-forward nature of its architecture. QANet is an open source model which made it convenient for analyzing and testing. It was created and tested on the earlier version of SQuAD dataset with 84.454 EM and 90.490 F1. This numbers exceeded results of other end-to-end models at that moment. All these reasons led us to try the model on the most recent version of SQuAD (released May 2018) as it is more challenging. Along with our expectations results were not promising, lower than any submitted models for SQuAD 2.0 dataset at the moment. Further on we will talk more detailed on the models components, testing results and possible modifications for their improvement.

Problem formulation

The reading comprehension task considered is defined as follows. Given a content paragraph with n words $C = \{c_1, c_2, \ldots, c_n\}$ and the query sentence with m words $Q = \{q_1, q_2, \ldots, q_m\}$ output a span $S = \{c_i, c_{i+1}, \ldots, c_{i+j}\}$ from the original paragraph C, or return an empty string if there is no answer. In the following x will be used to denote both the original word and its embedded vector, for any $x \in C, Q$.

Model overview

The given model contains five main components: an embedding layer, an embedding encoder layer, a context-query attention layer, a model encoder layer and an output layer. These components are the same for most reading comprehension models, however here only convolutional and self-attention mechanisms are used in embedding and modeling encoders, despite the usual use of RNNs. As the result, model process input tokens in parallel. In brief the five layers of QANet model are following:

Input Embedding Layer

The standard techniques are used to obtain the embedding of each word w by concatenating its word embedding and character embedding. The word embedding is fixed during training and initialized from the $p_1 = 300$ dimensional pretrained GloVe word vectors, which are fixed during training. All the out-of-vocabulary words are mapped to an $< UNK >$ token, whose embedding is trainable with random initialization. The character embedding is obtained as follows: Each character is represented as a trainable vector of dimension p2= 200, meaning each word can be viewed as the concatenation of the embedding vectors for each of its characters. The length of each word is either truncated or padded to 16. We take maximum value of each row of this matrix to get a fixed-size vector representation of each word. Finally, the output of a given word x from this layer is the concatenation $[x_w; x_c] \in R_{p_1+p_2}$, where x_w and x_c are the word embedding and the convolution output of character embedding of x respectively. For simplicity, we also use x to denote the output of this layer.

Embedding Encoder Layer

The encoder layer is a stack of the following basic building block: [convolution-layer $\times\#$ + self-attention-layer + feed-forward-layer], as illustrated in the upper right of Figure 1. Depth-wise separable convolutions are used rather than traditional ones, as we observe that it is memory efficient and has better generalization. The kernel size is 7, the number of filters is $d = 128$ and the number of convolution layers within a block is 4. For the self-attention-layer,the multi-head attention mechanism is adopted defined in (Vaswani et al., 2017a) which, for each position in the input, called the query, computes a weighted sum of all positions, or keys, in the input based on the similarity between the query and key as measured by the dot product. The number of heads is 8 throughout all the layers. Each of these basic operations (conv/self-attention/ffn) is placed inside a residual block, shown lower-right in Figure 1. For an input x and a given operation f, the output is $f(layernorm(x)) + x$, meaning there is a full identity path from the input to output of each block, where $layernorm$ indicates layer-normalization proposed in (Ba et al., 2016). The total number of encoder blocks is 1. Note that the input of this layer is a vector of dimension $p_1 + p_2 = 500$ for each individual word, which is immediately mapped to $d = 128$ by a one-dimensional convolution. The output of this layer is a also of dimension $d = 128$.

Context-Query Attention Layer

This module is standard in almost every previous reading comprehension models such as Weissenborn et al. (2017) and Chen et al. (2017). We use C and Q to denote the encoded context and query. The context-to-query attention is constructed as follows: We first computer the similarities between each pair of context and query words, rendering a similarity matrix $S \in R^{n \times m}$. We then normalize each row of S by applying the softmax function, getting a matrix $\overline{S}$. Then the context-to-query attention is computed as $A = \overline{S} \times Q^T \in R^{n \times d}$. The similarity function used here is the trilinear function:

$$f(q, c) = W_0[q, c, q \odot c],$$

where $\odot$ is the element-wise multiplicaion and W_0 is a trainable variable.

Model Encoder Layer

Similar to Seo et al. (2016), the input of this layer at each position is $[c, a, c \odot a, c \odot b]$, where a and b are respectively a row of attention matrix A and B. The layer parameters are the same as the Embedding Encoder Layer except that convolution layer number is 2 within a block and the total number of blocks are 7. We share weights between each of the 3 repetitions of the model encoder.

Output Layer

This layer is task-specific. Each example in SQuAD is labeled with a span in the context containing the answer. We adopt the strategy of Seo et al. (2016) to predict the probability of each position in the context being the start or end of an answer span. More specifically, the probabilities of the starting and ending position are modeled as

$$p^1 = softmax(W_1[M_0; M_1]), p^2 = softmax(W_2[M_0; M_2]),$$

where and W_2 are two trainable variables and M_0,M_1,M_2 are respectively the outputs of the three model encoders, from bottom to top. The score of a span is the product of its start position and end position probabilities. Finally, the objective function is defined as the negative sum of the log probabilities of the predicted distributions indexed by true start and end indices, averaged over all the training examples:

$$L(\theta) = -\frac{1}{N} \sum_{i}^{N} \Big[log(p^1_{y^1_i}) + log(p^2_{y^2_i}) \Big],$$

where y^1_i and y^2_i are respectively the groundtruth starting and ending position of example i, and θ contains all the trainable variables. The proposed model can be customized to other comprehension tasks, e.g. selecting from the candidate answers, by changing the output layers accordingly.

Inference

At inference time, the predicted span (s, e) is chosen such that $p^1_s p^2_e$ is maximized and $s \leq e$. Standard dynamic programming can obtain the result with linear time.

3.2 No-Answer Approach

Existing reading comprehension models focus on answering questions where a correct answer is guaranteed to exist. However, they are not able to identify unanswerable questions but tend to return an unreliable text span. Given a context passage and a question, the machine needs to not only find answers to answerable questions but also detect unanswerable cases. Therefore, several modifications and their combination can be proposed for the given model to improve its results.

There are several kinds of approaches to model the answerability of a question. One approach is to directly extend previous MRC models by introducing a no-answer score to the score vector of the answer span (Levy et al. 2017;Clark and Gardner 2017) or adding final probability threshold for determination of chosen span being the answer. But this kind of approaches are relatively simple and cannot effectively model the answerability of a question. Another approach introduces an answer verifier to determine whether the question is unanswerable (Huet al. 2018; Tan et al. 2018). The answer pointer and answer verifier have their respective models, which are trained separately. No-answer pointer can also be implemented to avoid selecting any spans when question is defined as unanswerable. One more approach suggests verifying the answer after reading and finding the supposedly correct answer span. Those methods are either implemented by themselves or combined and one of the most successful models that include some of them together is recently published U-Net by Fu Sun, Linyang Li, Xipeng Qiu, Yang Liu from Shanghai Key Laboratory of Intelligent Information Processing, Fudan University [2].

3.3 U-Net [2]

Different from the state-of-art pipeline models, U-Net can be learned in an end-to-end fashion. The unanswerability problem is decomposed into three sub-tasks and combined into a unified model, which uses the shared encoding and interaction layers. A universal node is introduced making it possible to encode common information on context and query simultaneously. Referring to the original publication main components of the U-Net model are described next.

U-Net consists of four major blocks: Unified Encoding, Multi-Level Attention, Final Fusion, and Prediction. As shown in Figure (?), first the embedded representation of the question and passage is combined with a universal node u and they are passed through a BiLSTM to encode the whole text. The encoded representation is then used to do the information interaction. Then the encoded and interacted representation is used to fuse the full representation and fed into the final prediction layers to do the multitask training. Skipping first three layers, as they are mostly structured similarly to other models, last Prediction layer becomes the novelty of it. Here the three approaches are combined successfully: (1) answer pointer, (2) no-answer pointer and (3) answer verifier. (structure as in paper) We use this answer pointer to detect the answer boundaries from the passage when the question is answerable (i.e., the answer is a span in the passage).

1. Answer Pointer

This pointer is used to detect the answer boundaries from the passage when the question is answerable. This later is a classic pointer net structure. We use two trainable matrices W_s

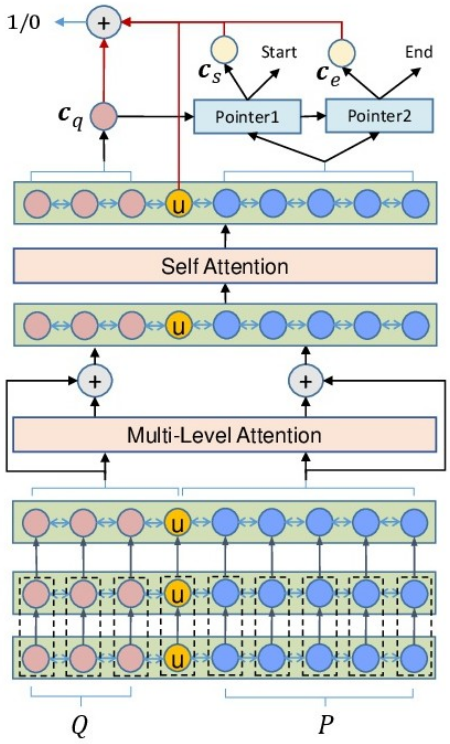

Figure 3.1: U-Net architecture

and W_e to estimate the probability of the answer start and end bounaries of the i_{th} word in the passage, α_i and β_i.

$$\alpha_i \propto exp(c_q W_s o_i^P),$$

$$\beta_i \propto exp(c_q W_e o_i^P),$$

Note here when the question is answerable, we do not consider the universal node in answer boundary detection, so we have $i > 0$ ($i = 0$ is the universal node in the passage representation). The loss function for the answerable question is:

$$\mathcal{L}_A = -(log(\alpha_a) + log(\beta_b)),$$

where a and b are the ground-truth of the start and end boundary of the answer.

2. No-Answer Pointer

Then the same pointer is used for questions that are not answerable. Here the loss $\mathcal{L}_N A$ is:

$$\mathcal{L}_{NA} = -(log(\alpha_0) + log(\beta_0)),$$

where α_0 and β_0 correspond to the position of the universal node, which is at the front of the passage representation O_p. For this scenario, the loss is calculated for the universal node. Additionally, since there exits a plausible answer for each unanswerable question in SQuAD

2.0, we introduce an auxiliary plausible answer pointer to predict the boundaries of the plausible answers. The plausible answer pointer has the same structure as the answer pointer, but with different parameters. Thus, the total loss function is:

$$\mathcal{L}_{NA} = -(log(\alpha_0) + log(\beta_0)) - (log(\alpha'_{a*}) + log(\beta'_{b*})),$$

where α' and β' are the output of the plausible answer pointer; $a*$ and $b*$ are the start and end boundary of the unanswerable answer.

3. Answer Verifier

The answer verifier is used to distinguish whether the question is answerable.
Answer verifier applies a weighted summary layer to summarize the passage information into a fixed-dim representation c_q.
The weight matrix obtained from the answer pointer is used to get two representations of the passage.

$$c_s = \sum_i \alpha_i \cdot o_i^P$$

$$c_e = \sum_i \beta_i \cdot o_i^P$$

Then the universal node o_{m+1} is concatenated with the summary of question and passage to make a fixed vector

$$F = [c_q; o_{m+1}; c_s; c_e].$$

This fixed F includes the representation c_q representing the question information, and c_s and c_e representing the passage information. Since these representations are highly summarized specially for classification, we believe that this passage-question pair contains information to distinguish whether this question is answerable. In addition, we include the universal node as a supplement. Since the universal node is pointed at when the question is unanswerable and this node itself already contains information collected from both the passage and question during encoding and information interaction, we believe that this node is important in distinguishing whether the question is answerable.
Finally, the fixed vector F is passed through a linear layer to obtain the prediction whether the question is answerable.

$$p^c = \sigma(W_f^T F)$$

where σ is a sigmoid function, W_f is a learnable weight matrix.
Here the cross-entropy loss is used in training.

$$\mathcal{L}_A V = -\big(\delta \cdot log(p^c) + (1-\delta) \cdot (log(1-p^c))\big),$$

where $\delta \in \{0, 1\}$ indicates whether the question has an answer in the passage.

Compared with other relatively complex structures developed for this MRC task, our U-Net model passes the original question and passage pair through embedding and encoding, which then interacts with each other, yielding fused information merged from all the levels. The entire architecture is very easy to construct. After we have the fused representation of the question and passage, we pass them through the pointer layer and a fused information classification layer in a multi-task setup.

4. Testing results on modified QANet model.

Minor changes were made in QANet model on dataset preprocessing stage to demonstrate results of described model on SQuAD 2.0 version. Model as it originally was didn't work with the new dataset and train on the No-Answer cases. Therefore, we have separated this special case and its correct proceeding. As a result the new model was able to proceed the dataset and show certain training results with learning rate on no-answer questions (returning empty span). It can be seen on Figure 4.1 that model reaches 50% EM on training dataset and the loss is decreasing in stable manner. On dev set results (Figure 4.2) are noticeably lower than on any base model created for the unanswerable question datasets. These results prove the new dataset's uniqueness and importance and the need for the spacial approach once more. Therefore, some of the approaches and their implementation will be described in the following section.

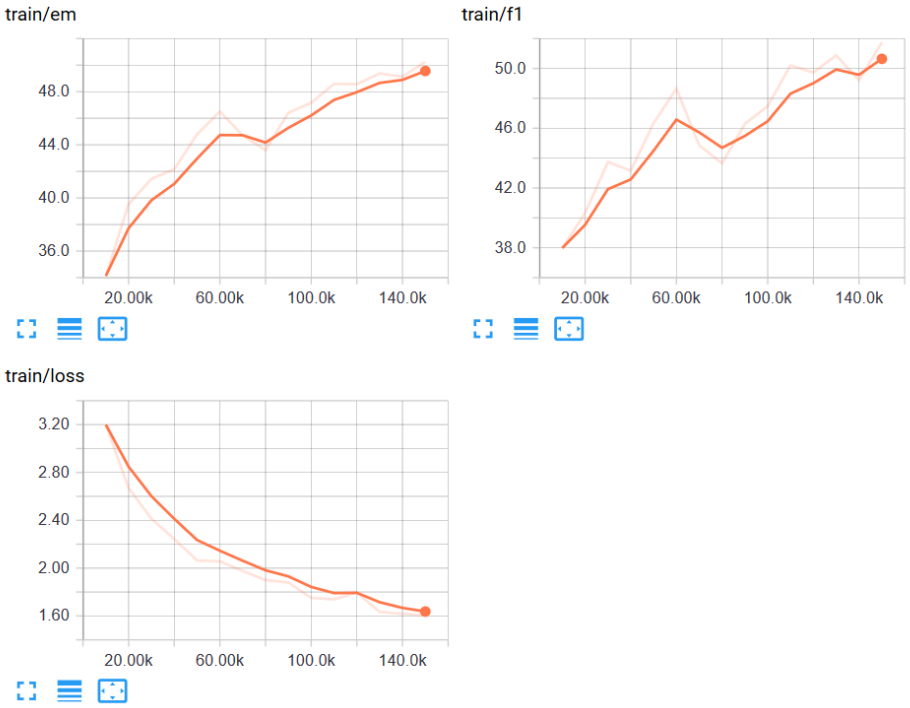

Figure 4.1: Modified QANet testing results on training dataset.

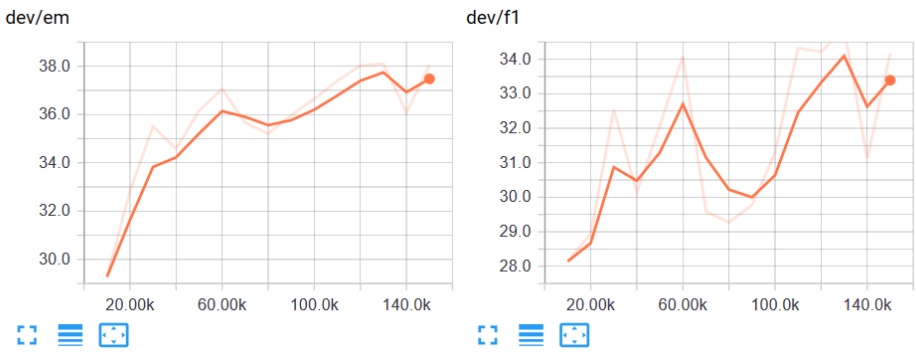

Figure 4.2: Modified QANet testing results on dev dataset.

5. Conclusions and Future Work

As a result of studying existing models and testing done it became clear that despite older models' ability to learn on unanswerable cases they require major model changes and special approach to perform well. Methods and approaches applicable to the problem were also studied and as support to choose them a fully implemented model (U-Net) is described. As part of our future plans we will work on optimal implementation of these methods considering purposes of the project as well as improvement of accuracy and speed up. In addition we're also looking into building our own dataset focused around the Scientific Research papers.

Bibliography

[1] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, Quoc V. Le. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. April 2018. iii, 5

[2] Fu Sun, Linyang Li, Xipeng Qiu, Yang Liu. U-Net: Machine Reading Comprehension with Unanswerable Questions. October 2018. iii, 8

[3] Mark Ware, Michael Mabe. The STM report: An overview of scientific and scholarly journal publishing. November 2012. 1

[4] Marzieh Saeidi, Max Bartolo, Patrick Lewis, Sameer Singh, Tim Rocktschel, Mike Sheldon, Guillaume Bouchard, Sebastian Riedel. Interpretation of natural language rules in conversational machine reading. August 2018. 3

[5] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao,Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen,Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, Tong Wang. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. December 2016. iii, 3

[6] Henk Pelk. Chatbots: A bright future in IoT? September 2016. 1

[7] Pranav Rajpurkar, Robin Jia, Percy Liang. Know What You Don't Know: Unanswerable Questions for SQuAD . June 2018. iii, 4

[8] Sembok, Tengku and Badioze Zaman, Halimah and Abdul Kadir, Rabiah. IRQAS: information retrieval and question answering system based on a unified logical-linguistic model. January 2008. 2