ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

DISI

Master in Artificial Intelligence

Andrea Pinto - andrea.pinto2@studio.unibo.it

Giorgio Buzzanca - giorgio.buzzanca@studio.unibo.it

Implementing QANet For Question Answering

NLP Project Report

Academic Year 2021 - 2022

Contents

Chapter 1

Executive Summary

We started by analyzing the SQuAD dataset[1], seeking a way to formalize the task of question answering, to get an idea of its properties and the preprocessing steps needed, as well as possible features to feed a model with.

After a shallow literature review on the main methods to tackle this task, we decided to go for an encoder-decoder-like architecture, as LSTMs with attention[2], transformers[3], or BERT[4]-like architectures.

In particular, given the huge training cost and the need to use more datasets associated with transformers, and the lack of parallelizability of recurrent layers, we decided to opt for an advanced architecture easier to train, which adopts a completely different approach.

Indeed, QANet[5] consists of only convolutional layers and relies on context-to-query attention and self-attention[3] After a thorough examination of the architecture, we were able to implement it from scratch, making personal choices where details were lacking in the paper.

We performed a full training/evaluation cycle by splitting the dataset we were provided with, and analyzed the predictions made by the model.

Chapter 2

Background

20202040

Andrea (alla fine)

Chapter 3

System Description

The architecture of QANet can be decomposed into 5 main building blocks: an input embedding layer, an encoding embedding layer, a context-query attention layer, a model encoder layer, and an output layer.

3.1 Input Embedding Layer

In this layer, we extract the word and character embeddings using their indices, with the first two elements always reserved to padding and OOV words respectively. While in the paper the OOV embedding was made trainable, we decided to keep it fixed in our implementation.

The character embeddings of every word are passed through convolutional layers, in order to aggregate the information between characters and obtain a new representation of the word.

At the end of this layer we'll have the concatenation of word embeddings and character embeddings. This new representation is passed through a Highway network[6], which has the purpose of facilitating the flow of information, through the usage of gating units, that helps with the training of deep neural networks.

3.2 Embedding Encoder Layer

This layer is composed by stacking different encoders, which are made of: convolutional layer, self-attention layer, and feed forward layer.

Each of this layers is preceded by a normalization layer, as proposed in [7], and wrapped inside a residual block.

The input dimension of the word embeddings is mapped into a lower dimension by a one-dimensional convolution.

We implemented the convolutions in the encoder block as depth-wise separable, as in the paper.

The multi-head attention mechanism that we have used is the one defined in [3], which computes attention as: $MultiHeadAttention(Q, K, V) = Concat(head_1, ..., head_n)W^O$ where $head_i = softmax(\frac{Q_i W_i^Q (K_i W_i^K)^T}{\sqrt{(d_k)}})V_i W_i^V$.

In our case, given that we are applying a self-attention, Q, K, and V are the same matrices.

3.3 Context-Query Attention Layer

Here the information between the context and the query is aggregated using attention.

This is done by computing the similarity matrix $S \in \mathbb{R}^{n \times m}$, filled with the similarities between each pair of context and query words, that is then normalized with a softmax.

The context-to-query attention is computed as $A = \bar{S} \cdot Q^T \in \mathbb{R}^{n \times d}$. The similarity funcion used is the trilinear function.

Furthermore, we used also a query-to-context attention. To do so, we computed the column normalized matrix $\bar{\bar{S}}$ of S with a softmax, then the query-to-context is computed as $\dot{\dot{B}}$. The output of this layer is, for each word, given by $[c, a, c \odot a, c \odot b]$

where a and b are rows of A and B.

3.4 Model Encoder Layer

In this layer we use the same encoder structure as in the embedding encoder layer.

The output of this layer is given by the concatenation of the outputs of the different blocks. In particular, we concatenated the output of the first and second block, and the output of the first and the third block.

Because of the concatenation, the dimension of the embedding in input is larger, so we decided to bring it down with a one dimensional convolution.

3.5 Output Layer

The two vectors obtained in the previous layer are passed into two separate linear layers with a softmax at the end.

This allows us to compute the start and the end of an answer's span. The loss function used is the sum of the cross-entropy losses computed both on the start position and the end position probabilities.

At inference time, the span is chosen such that the product of the probabilities of the indices is maximized.

In order to deal with the padding of the sequences, we compute a boolean mask that is passed throguh the layers, and used whenever necessary to compute a softmax function (i.e. in the attention(s) computation and in the output layer).

Chapter 4

Experimental Setup And Results

Chapter 5

Analysis Of The Results

Giorgio

Discussion

Andrea e Giorgio

Bibliography

[1] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016.

[2] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension, 2018.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[5] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. Qanet: Combining local convolution with global self-attention for reading comprehension, 2018.

[6] Rupesh Kumar Srivastava, Klaus Greff, and JÃ¼rgen Schmidhuber. Highway networks, 2015.

[7] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.