DeFormer: Decomposing Pre-trained Transformers for Faster Question Answering

Qingqing Cao, Harsh Trivedi, Aruna Balasubramanian, Niranjan Balasubramanian

ACL2020
Question Answering

- Question Answering (QA) has different forms: extractive QA, multiple-choice QA, open-domain QA, ...
- In this presentation, we focus on extractive QA.
- Extractive QA:
  > Given a question and a passage that contains the answer, the task is to predict the answer text span in the passage.

Context

The immune system is a system of many biological structures and processes within an organism that protects against disease. To function properly, an immune system must detect a wide variety of agents, known as pathogens, from viruses to parasitic worms, and distinguish them from the organism's own healthy tissue. In many species, the immune system can be classified into subsystems, such as the innate immune system versus the adaptive immune system, or humoral immunity versus cell-mediated immunity. In humans, the blood–brain barrier, blood–cerebrospinal fluid barrier, and similar fluid–brain barriers separate the peripheral immune system from the neuroimmune system which protects the brain.

Question

What is the immune system?

Answer

a system of many biological structures and processes within an organism that protects against disease

An example taken from SQUAD v1.1 dataset.
Question Answering

Recently, BERT and its variants (Devlin et al., 2018; Yang et al., 2019; Clark et al., 2020) have set the standard for solving the task. The concatenation of the question and the passage is believed to produce question-dependent representations for the passage.

> determine the answer span by identifying its start and end token.
> Introduce \textbf{START} vector \( S \in \mathbb{R}^H \) and \textbf{END} vector \( E \in \mathbb{R}^H \)

\begin{align*}
P(\text{start} = i) &= \frac{e^{S \cdot T_i}}{\sum e^{S \cdot T_i}} \\
P(\text{end} = j) &= \frac{e^{E \cdot T_j}}{\sum e^{E \cdot T_j}}
\end{align*}

> Training objective: maximize

\[ \log(P(\text{start} = i)) + \log(P(\text{end} = j)) \]
Pretrained Transformers have many layers

- Pretrained Transformers (e.g., BERT) usually have 12-24 layers, each involves input-wide self-attention.

Figure 1 (Devlin et. al, 2018)
Pretrained Transformers have many layers

- Pretrained Transformers (e.g., BERT) usually have 12-24 layers, each involves input-wide self-attention.

For extractive QA, the input is very long due as It is the concatenation of the question and the passage.
⇒ Expensive inference.
⇒ Do we really need the passage-token representations to depend on the question at every layer?

*Figure 1 (Devlin et. al, 2018)*
Do we need the passage to involve at every layer?

- The answer is No!
- This paper shows that in lower layers, the passage representations does not change as much as it does in the upper layers by changing the paired questions while keeping the same passage. In lower layers, we can compute the passage representations independently.

![Figure 2 in the paper](image-url)
Proposed Solution: Deformer

- Deformer: A decomposed Transformer.
  - Substitutes the full (input-wide) self-attention with question-wide and passage-wide self-attentions in the lower layers.
Deformer: Formal description

- Suppose that we perform a task whose input contains two segments of text $T_a$ and $T_b$.
- Token embeddings: $A = [a_1; a_2; \ldots; a_q]$ (1) $B = [b_1; b_2; \ldots; b_p]$ (2) $X = [A; B]$ (3)
- Input representations at layer $l+1$ of the Transformer: $X^{l+1} = L_i(X^l)$ (4)
- Compute representations for $A$ and $B$ independently in lower layers, and jointly compute them in the upper layers:
  $$[A^n; B^n] = L_{k+1:n}([L_{1:k}(A^0); L_{1:k}(B^0))]$$ (5)
- In QA task, $T_b$ is the passage and it is pre-computed.
- Runtime complexity of each lower layer is reduced from $O((p + q)^2)$ to $O(q^2 + c)$

where $c$ denotes cost of loading the cached representation.
Auxiliary Supervision for Deformer

- Decomposition in the lower layers might still loose some information.
- During training, they proposed to use auxiliary supervision to constrain the representations produced by Deformer in upper layers to be similar to those produced by the finetuned full Transformer.

**Knowledge distillation loss** (at token level):

\[
\mathcal{L}_{kd} = D_{KL}(P_{Deformer} \parallel P_{Full})
\]

**Layerwise Representation Similarity Loss**:

- \( v_i^j \) is the representation of token \( j \) at layer \( i \) of the Full Transformer.
- \( u_i^j \) is the representation of token \( j \) at layer \( i \) of the Deformer.

\[
\mathcal{L}_{lrs} = \sum_{i=k}^{n} \sum_{j=1}^{m} \| v_i^j - u_i^j \|^2
\]
Main Results: Question Answering

- Table 1: Performance of original fine-tuned vs fine-tuned models of DeFormer-BERT-base and DeFormerXLNet-base (#decomposed lower layers = 9).

<table>
<thead>
<tr>
<th>Model</th>
<th>Datasets</th>
<th>Avg. Input Tokens</th>
<th>Original base</th>
<th>DeFormer-base</th>
<th>Performance Drop (absolute</th>
<th>%age)</th>
<th>Inference Speedup (times)</th>
<th>Memory Reduction (%age)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>SQuAD</td>
<td>320</td>
<td>88.5</td>
<td>87.1</td>
<td>1.4</td>
<td>1.6</td>
<td>3.2x</td>
<td>70.3</td>
</tr>
<tr>
<td></td>
<td>RACE</td>
<td>2048</td>
<td>66.3</td>
<td>64.5</td>
<td>1.8</td>
<td>2.7</td>
<td>3.4x</td>
<td>72.9</td>
</tr>
<tr>
<td></td>
<td>BoolQ</td>
<td>320</td>
<td>77.8</td>
<td>76.8</td>
<td>1.0</td>
<td>1.3</td>
<td>3.5x</td>
<td>72.0</td>
</tr>
<tr>
<td>XLNet</td>
<td>SQuAD</td>
<td>320</td>
<td>91.6</td>
<td>90.4</td>
<td>1.2</td>
<td>1.3</td>
<td>2.7x</td>
<td>65.8</td>
</tr>
<tr>
<td></td>
<td>RACE</td>
<td>2048</td>
<td>70.3</td>
<td>68.7</td>
<td>1.6</td>
<td>2.2</td>
<td>2.8x</td>
<td>67.6</td>
</tr>
<tr>
<td></td>
<td>BoolQ</td>
<td>320</td>
<td>80.4</td>
<td>78.8</td>
<td>0.6</td>
<td>0.7</td>
<td>3.0x</td>
<td>68.3</td>
</tr>
</tbody>
</table>
Main Results: Pairwise input tasks

- Table 2: Performance of BERT-base vs DeFormer-BERT-base (#decomposed lower layers = 9).

<table>
<thead>
<tr>
<th>Avg. Input Tokens</th>
<th>BERT base</th>
<th>DeFormer-BERT base</th>
<th>Performance Drop (absolute</th>
<th>% age)</th>
<th>Inference Speedup (times)</th>
<th>Memory Reduction (% age)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNLI</td>
<td>120</td>
<td>84.4</td>
<td>82.6</td>
<td>1.8</td>
<td>2.1</td>
<td>2.2x</td>
</tr>
<tr>
<td>QQP</td>
<td>100</td>
<td>90.5</td>
<td>90.3</td>
<td>0.2</td>
<td>0.2</td>
<td>2.0x</td>
</tr>
</tbody>
</table>
Main Results: Large Decomposed is better than Base

- Table 3: A large Transformer can run faster than a smaller one which is half its size, while also being more accurate (SQUAD performance).

<table>
<thead>
<tr>
<th></th>
<th>Performance (Squad-F1)</th>
<th>Speed (GFLOPs)</th>
<th>Memory (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT-large</td>
<td>92.3</td>
<td>204.1</td>
<td>1549.6</td>
</tr>
<tr>
<td>BERT-base</td>
<td>88.5</td>
<td>58.4</td>
<td>584.2</td>
</tr>
<tr>
<td>DeFormer-BERT-large</td>
<td>90.8</td>
<td>47.7</td>
<td>359.7</td>
</tr>
</tbody>
</table>
Main Results: Speed comparison on different devices

- Table 4: Inference latency (in seconds) on SQuAD dataset for BERT-base vs DeFormer-BERT-base, as an average measured in batch mode.

<table>
<thead>
<tr>
<th>Device Type</th>
<th>BERT</th>
<th>DeFormer-BERT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tesla V100 GPU</td>
<td>0.22</td>
<td>0.07</td>
</tr>
<tr>
<td>Intel i9-7900X CPU</td>
<td>5.90</td>
<td>1.66</td>
</tr>
<tr>
<td>OnePlus 6 Phone</td>
<td>10.20*</td>
<td>3.28*</td>
</tr>
</tbody>
</table>

(* indicates a batch size of 1, the others use batch size of 32)
Ablation Study

- Table 5: Removing auxiliary losses leads to performance drop, showing its complementary role to the model.

<table>
<thead>
<tr>
<th></th>
<th>Base Model</th>
<th>Large Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>88.5</td>
<td>92.3</td>
</tr>
<tr>
<td>DeFormer-BERT</td>
<td>87.1</td>
<td>90.8</td>
</tr>
<tr>
<td>w/o LRS</td>
<td>86.2</td>
<td>88.9</td>
</tr>
<tr>
<td>w/o KD &amp; LRS</td>
<td>85.8</td>
<td>87.5</td>
</tr>
</tbody>
</table>
How many lower layers should we choose?

(a) F1 drop versus speedup on SQuAD for DeFormer-BERT-base without auxiliary supervision.

(b) F1 drop versus speedup on SQuAD for DeFormer-BERT-large without auxiliary supervision.
Thank you for listening!