DeFormer: Decomposing Pre-trained Transformers for Faster Question Answering

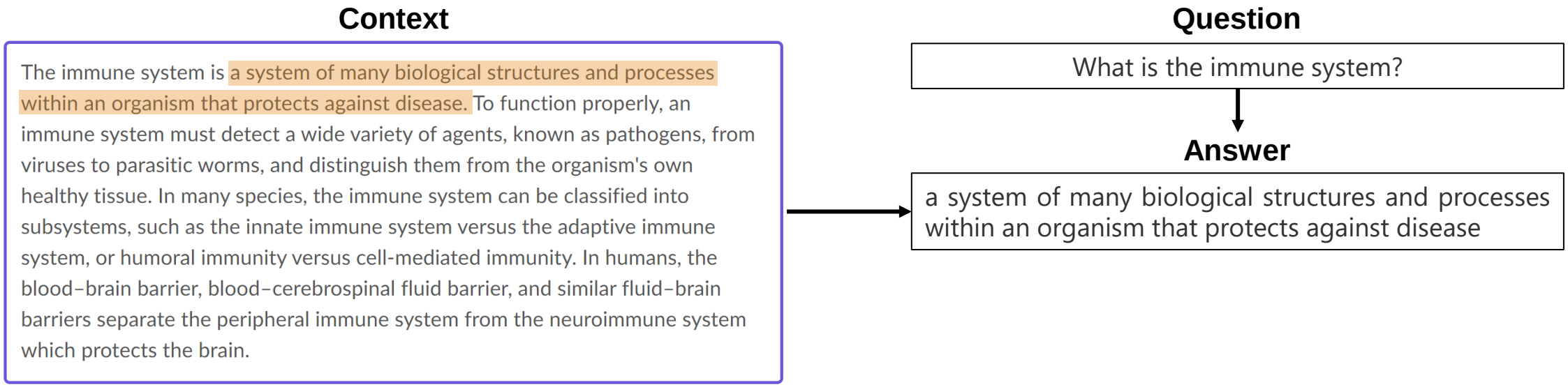
Qingqing Cao, Harsh Trivedi, Aruna Balasubramanian, Niranjan Balasubramanian

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.



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.

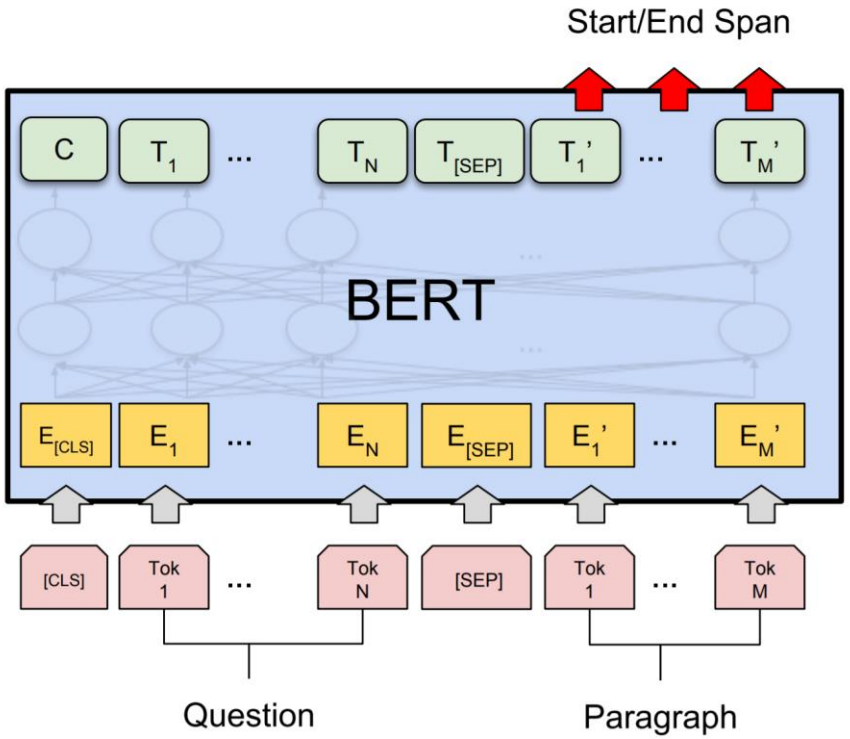


Figure 4c (Devlin et. al, 2018)

> determine the answer span by identifying its start and end token.
> Introduce **START** vector $S \in \mathbb{R}^H$ and **END** vector $E \in \mathbb{R}^H$
> Probability of the i-th token being the start of the answer:

$$P(\text{start} = i) = \frac{e^{S \cdot T_i}}{\sum e^{S \cdot T_{i'}}}$$

> Probability of the j-th token being the end of the answer:

$$P(\text{end} = j) = \frac{e^{E \cdot T_j}}{\sum e^{E \cdot T_{j'}}}$$

> 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.

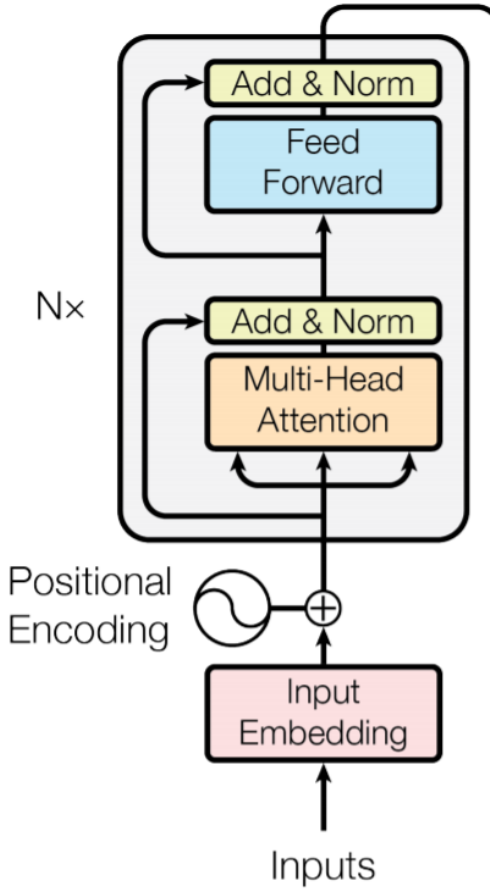


Figure 1 (Devlin et. al, 2018)

Pretrained Transformers have many layers

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

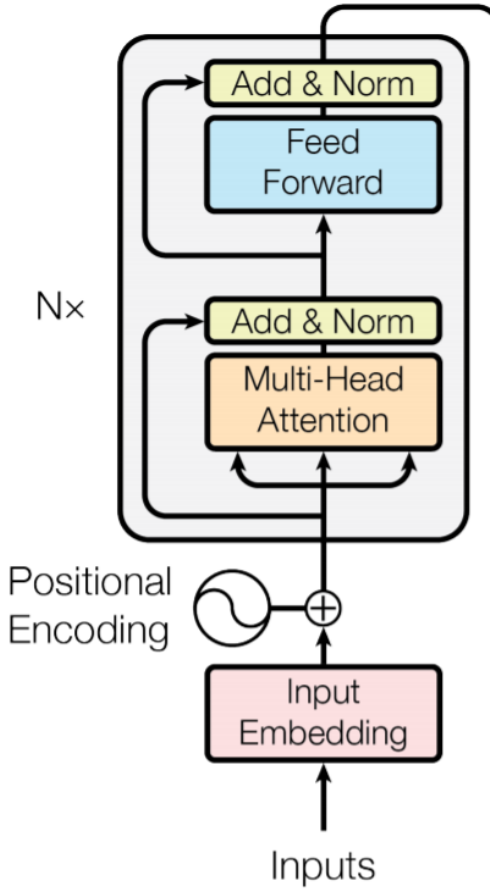


Figure 1 (Devlin et. al, 2018)

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?

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.

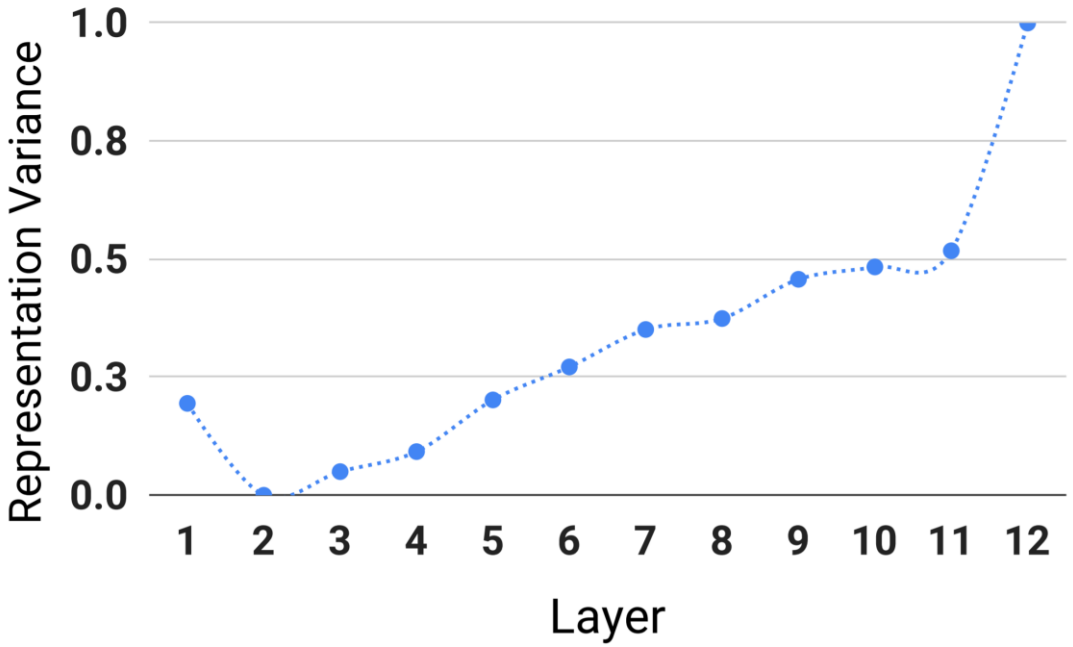
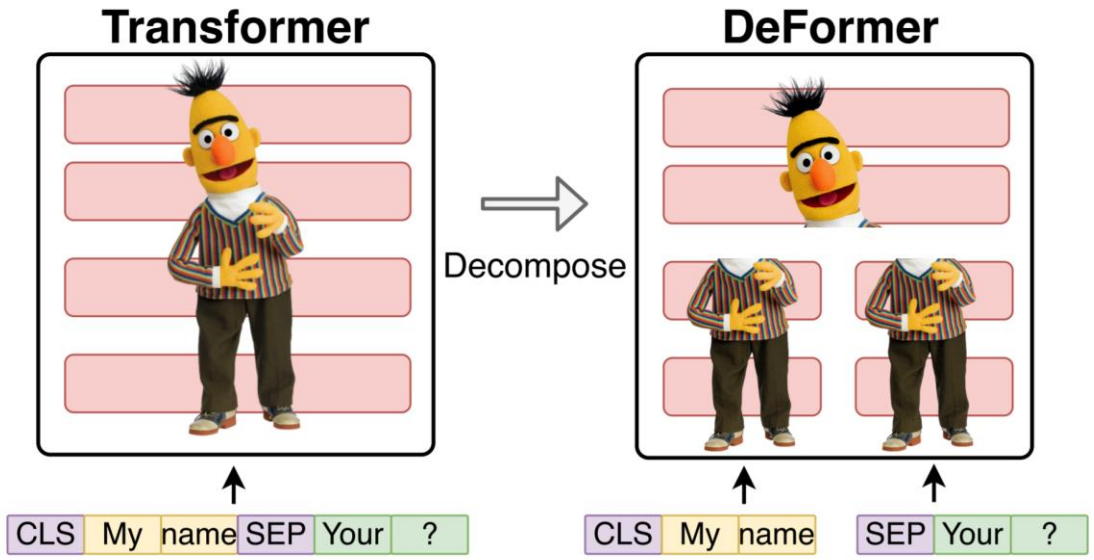


Figure 2 in the paper

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: $\mathbf{A} = [\mathbf{a_1}; \mathbf{a_2}; ...; \mathbf{a_q}]$ (1) $\mathbf{B} = [\mathbf{b_1}; \mathbf{b_2}; ...; \mathbf{b_p}]$ (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:

$$[\mathbf{A^n}; \mathbf{B^n}] = L_{k+1:n}([L_{1:k}(\mathbf{A^0}); L_{1:k}(\mathbf{B^0}))) \quad (5)$$

- 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.

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log\left(\frac{P(x)}{Q(x)}\right)$$

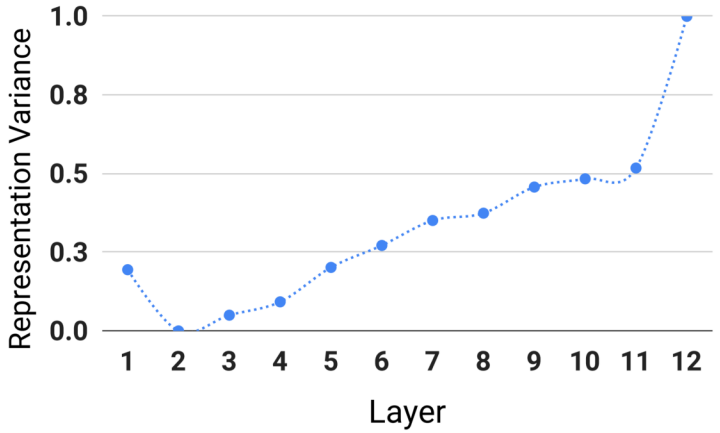
Auxiliary Supervision for Deformer

- Decomposition in the lower layers might still loose some information.
- During training, they preprosed 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):

$$\mathcal{L}_{kd} = D_{KL}(P_{Deformer} || P_{Full})$$

- **Layerwise Representation Similarity Loss**:
  - > $\mathbf{v}_i^j$ is the representation of token j at layer i of the Full Transformer.
  - > $\mathbf{u}_i^j$ is the representation of token j at layer i of the Deformer.

$$\mathcal{L}_{lrs} = \sum_{i=k}^{n} \sum_{j=1}^{m} \|\mathbf{v}_j^i - \mathbf{u}_j^i\|^2$$



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).

Model	Datasets	Avg. Input Tokens	Original base	DeFormer-base	Performance Drop (absolute | %age)	Inference Speedup (times)	Memory Reduction (%age)
BERT	SQuAD	320	88.5	87.1	1.4 | 1.6	3.2x	70.3
	RACE	2048	66.3	64.5	1.8 | 2.7	3.4x	72.9
	BoolQ	320	77.8	76.8	1.0 | 1.3	3.5x	72.0
XLNet	SQuAD	320	91.6	90.4	1.2 | 1.3	2.7x	65.8
	RACE	2048	70.3	68.7	1.6 | 2.2	2.8x	67.6
	BoolQ	320	80.4	78.8	0.6 | 0.7	3.0x	68.3

Main Results: Pairwise input tasks

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

	Avg. Input Tokens	BERT base	DeFormer-BERT base	Performance Drop (absolute | %age)	Inference Speedup (times)	Memory Reduction (%age)
MNLI	120	84.4	82.6	1.8 | 2.1	2.2x	56.4
QQP	100	90.5	90.3	0.2 | 0.2	2.0x	50.0

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).

	Performance (Squad-F1)	Speed (GFLOPs)	Memory (MB)
BERT-large	92.3	204.1	1549.6
BERT-base	88.5	58.4	584.2
DeFormer-BERT-large	90.8	47.7	359.7

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.

	BERT	DeFormer-BERT
Tesla V100 GPU	0.22	0.07
Intel i9-7900X CPU	5.90	1.66
OnePlus 6 Phone	10.20*	3.28*

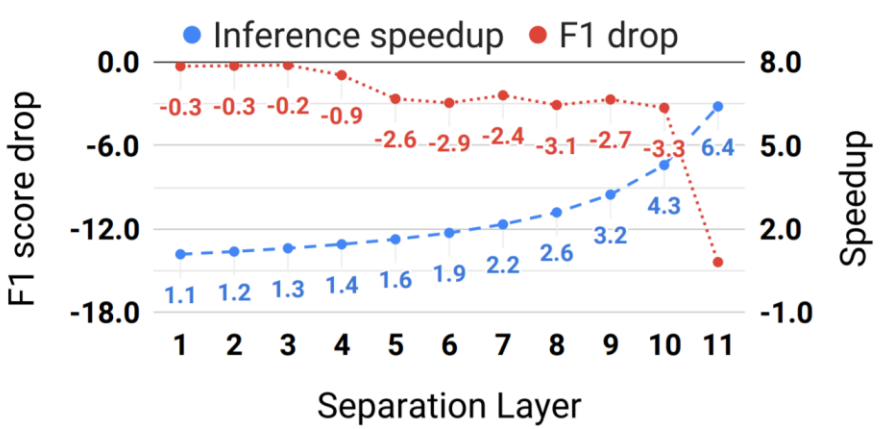
(* indicates a batch size of 1, the others use batch size of 32)

Ablation Study

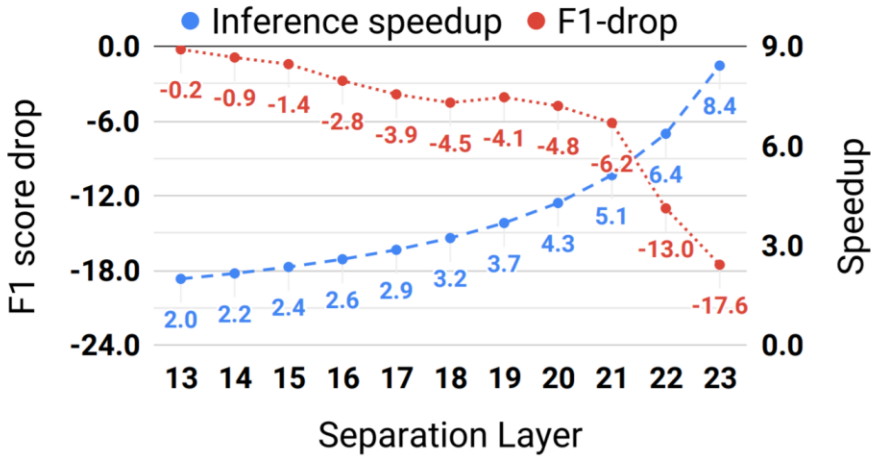
- Table 5: Removing auxiliary losses leads to performance drop, showing its complementary role to the model.

	Base Model	Large Model
BERT	88.5	92.3
DeFormer-BERT	87.1	90.8
w/o LRS	86.2	88.9
w/o KD & LRS	85.8	87.5

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!