

DeBERTa: Decoding-enhanced BERT with Disentangled Attention

Pengcheng He, Xiaodong Liu, Jianfeng Gao and Weizhu Chen
Microsoft

DeBERTa – Overview

- Three techniques are proposed to improve the generalization performance of transformer language models
 - Disentangled attention
 - Enhanced mask decoder
 - Scale Invariant Fine-tuning

DeBERTa – The improvements on pre-training

- DA (Disentangled Attention)
- EMD (Enhanced Mask Decoder)

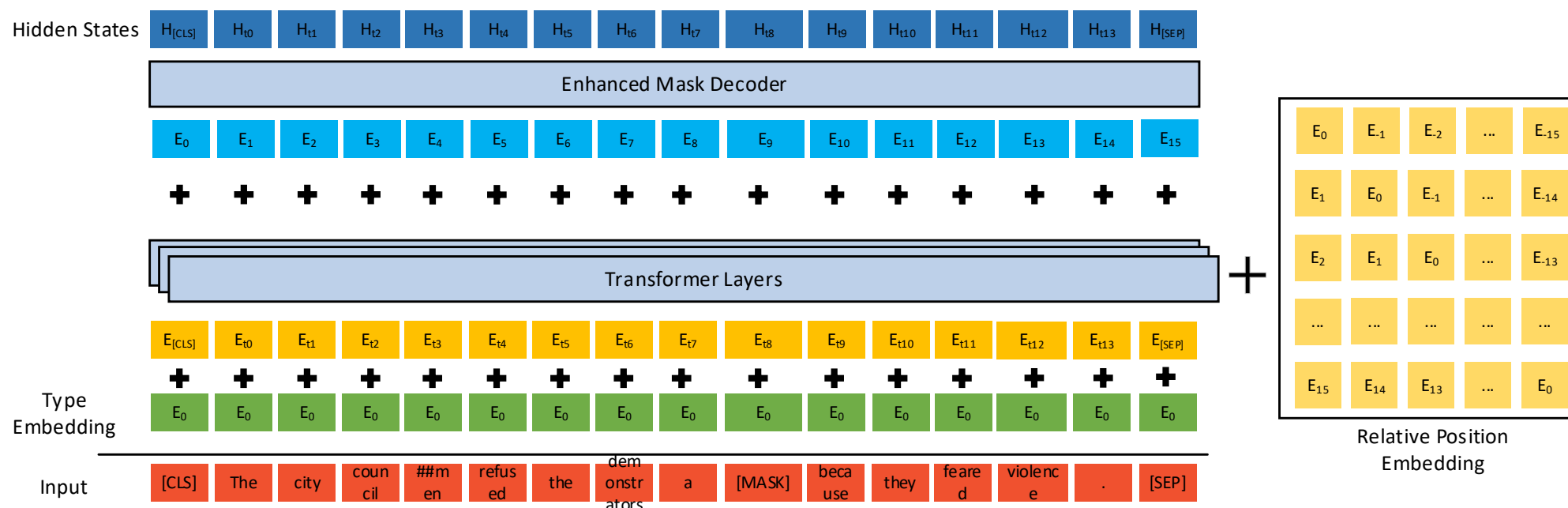


Fig 1. The model architecture of DeBERTa

DeBERTa – Disentangled Attention(DA)

- In **BERT**, each word in the input layer is represented using a vector that sums its word (content) embedding and position embedding. Then this vector is passed to self-attention layers to calculate the dependencies among words.
- Problem with the position encoding in BERT
 - The input vectors will be different at different positions for the same words.
 - A. *Deep learning* is an important area of artificial intelligence.
 - B. An important area of artificial intelligence is *deep learning*.

DeBERTa – Disentangled Attention

- In DA, the attention weights are composed of three matrices

$$Q_c = HW_{q,c}, K_c = HW_{k,c}, V_c = HW_{v,c}, Q_r = PW_{q,r}, K_r = PW_{k,r}$$

$$\tilde{A}_{i,j} = \underbrace{Q_i^c K_j^{c\top}}_{\text{(a) content-to-content}} + \underbrace{Q_i^c K_{\delta(i,j)}^r}_{\text{(b) content-to-position}} + \underbrace{K_j^c Q_{\delta(j,i)}^r}_{\text{(c) position-to-content}}$$

Content to content

❖ Deep Learning

	Learning
Deep	?

Content to position

❖ Deep ____

	+1
Deep	?

Position to content

❖ ____ learning

	Learning
-1	?

- Ablation study

Model	MNLI-m/mm Acc	SQuAD v1.1 F1/EM	SQuAD v2.0 F1/EM	RACE Acc
BERT _{base} Devlin et al. (2019)	84.3/84.7	88.5/81.0	76.3/73.7	65.0
RoBERTa _{base} Liu et al. (2019c)	84.7/-	90.6/-	79.7/-	65.6
DeBERTa _{base}	86.3/86.2	92.1/86.1	82.5/79.3	71.7
-DA	85.2/85.0	91.3/85.0	80.3/76.5	67.3

DeBERTa – Enhanced Mask Decoder

- Relative position encoding will have ambiguity issue with MLM in some cases.
 - A **new store** opened beside the **new mall**
- In mask decoding layer, EMD add absolute position to the query of attention layer to incorporate that information back without directly affect content encoding.
- Ablation study

Model	MNLI-m/mm Acc	SQuAD v1.1 F1/EM	SQuAD v2.0 F1/EM	RACE Acc
BERT _{base} Devlin et al. (2019)	84.3/84.7	88.5/81.0	76.3/73.7	65.0
RoBERTa _{base} Liu et al. (2019c)	84.7/-	90.6/-	79.7/-	65.6
DeBERTa _{base}	86.3/86.2	92.1/86.1	82.5/79.3	71.7
-EMD	86.1/86.1	91.8/85.8	81.3/78.0	70.3

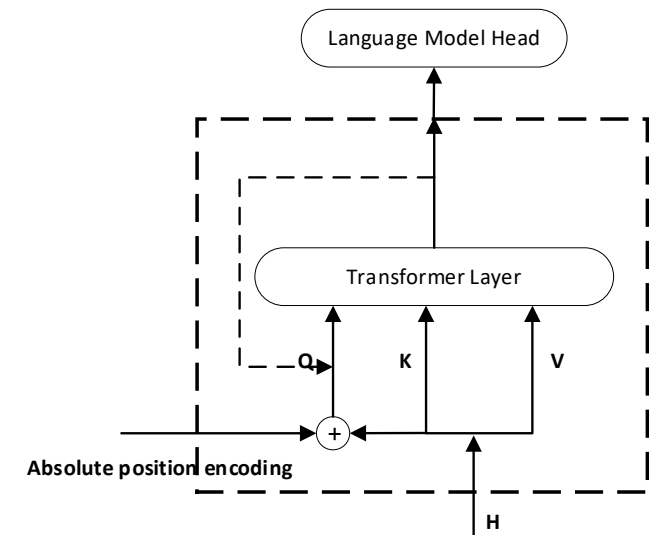


Fig 2. Enhanced Mask Decoder

SiFT - Scale Invariant Fine Tuning

- Virtual Adversarial Training
 - A regularization method for improving models' generalization
 - Suffers from instability issue when applies to large models as the variance of embeddings varies among words and models.
- SiFT
 - Add a normalization layer to the embeddings, and apply perturbation to normalized embeddings.

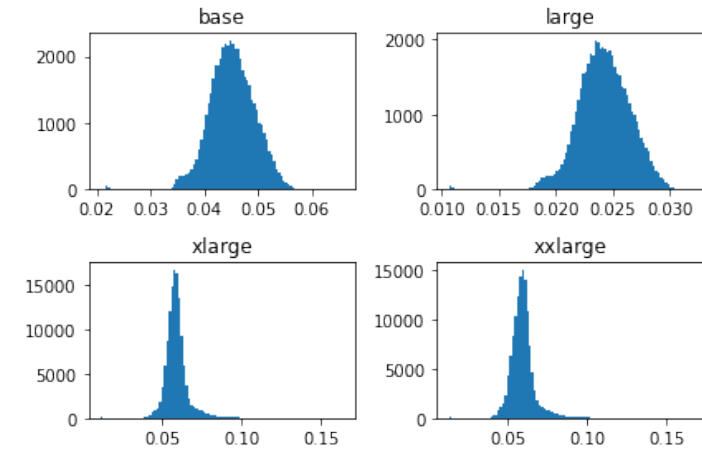


Fig 3. The distribution of the variance of word embeddings of different models

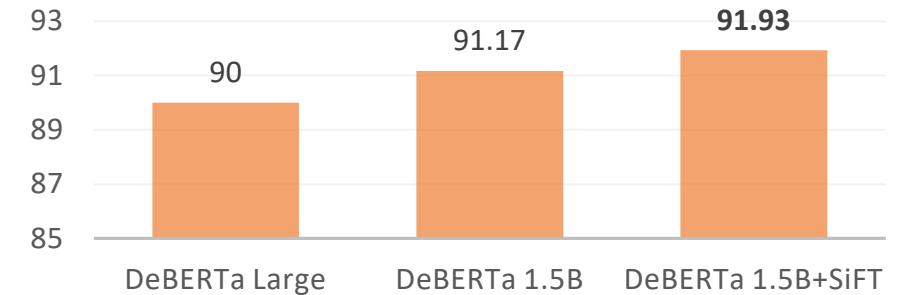


Fig 4. Performance on GLUE Dev

$$\delta = \operatorname{argmax}_{\|\delta\|_p < \epsilon} L(f(\tilde{x}, \theta), f(\tilde{x} + \delta, \theta)) \quad \tilde{x} = \frac{x - \mu_x}{\sigma_x}$$

Experiments – Large model

- Settings
 - Large model 24 layers, 16 heads, 1024 hidden size.
 - Training 1M steps with 2k batch size using 78GB data.
- GLUE dev average score with large models

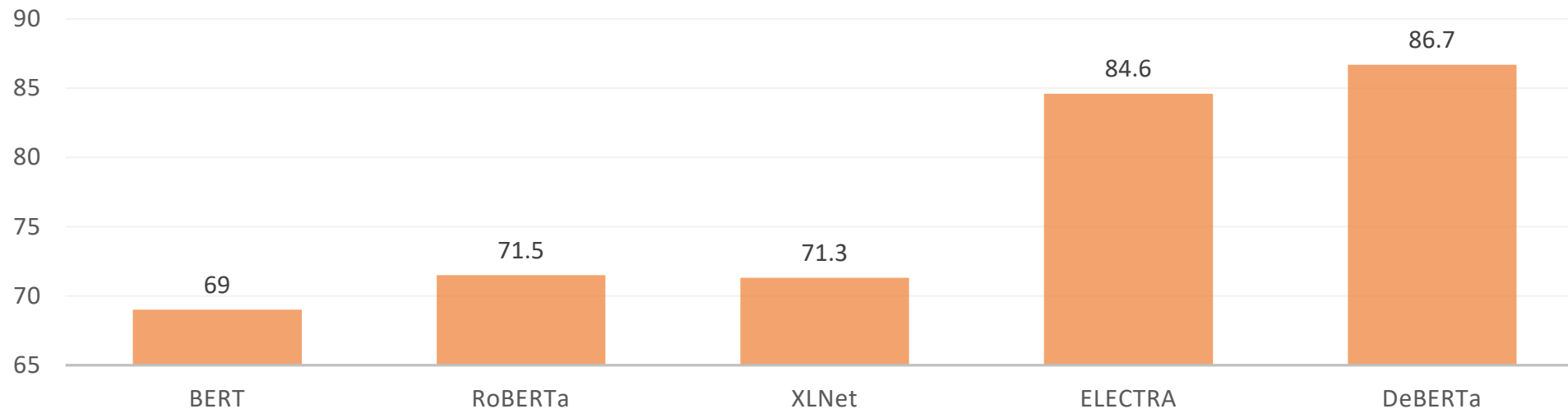


Fig 5. Performance of large models on GLUE Dev

Experiments – Scale up with 1.5B Parameters

- We scale up our model with 1.5B parameters
 - 48 layers, 1536 hidden size, 24 heads
 - The first single model to surpass **human performance** on **SuperGLUE**.
 - The **SOTA** model on **GLUE** and **SuperGLUE** as of 1/6/2021.

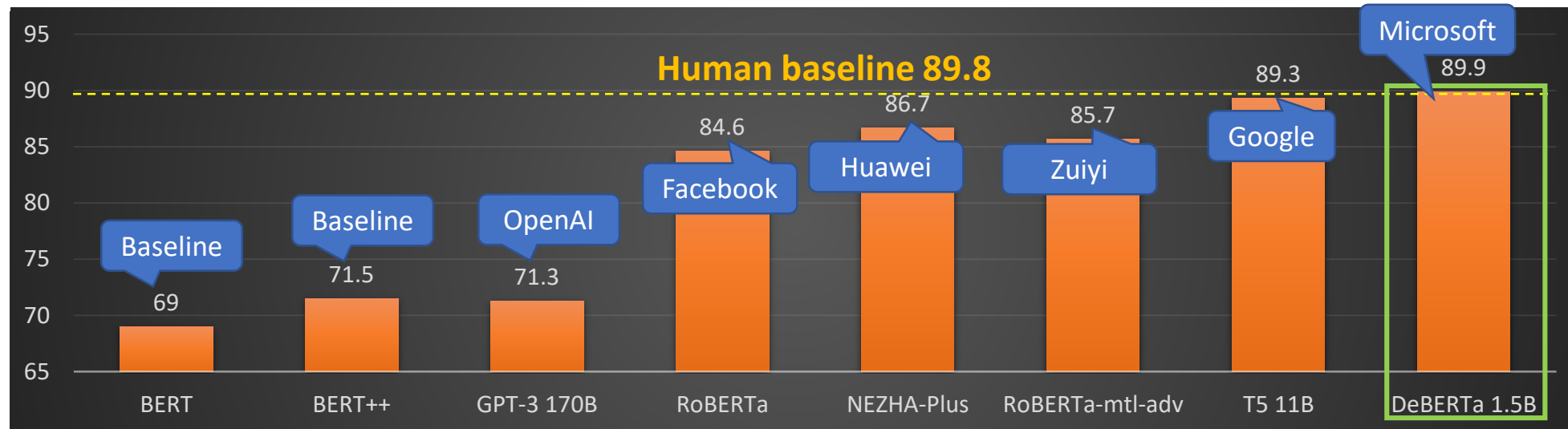


Fig 6. Performance on SuperGLUE leaderboard

Summary

- Three techniques is proposed to improve model performance,
 - **Disentangled Attention** to improve position encoding in transformers
 - **Enhanced Mask Decoder** to overcome ambiguity issue with MLM
 - **Scale Invariant Fine-tuning** to solve instability issue with adversarial training on large models
- [Code](#) & [Blog](#)

