Efficient Transformer

余阳 2021.11.25

Personal Information

- 姓名：余阳
- 年级：研一
- 导师：刘淇老师
- 研究兴趣：推荐系统，联邦学习等
- 工位：信智楼B702
- 邮箱：yflyl613@mail.ustc.edu.cn

Outline

- Background
- General Methods
  - Quantization / Mixed Precision
  - Knowledge Distillation
- Transformer-specific Methods
  - Weight Sharing
  - Complexity reduction of self-attention
- Conclusions

Background

- Transformer-based models have achieved great success in multiple fields
  - Very powerful
  - Large model size
  - High computational overhead
- What do we want ?
  - Powerful
  - Small model size
  - Fast inference speed

Trade-off

TECHNICAL WALKTHROUGH

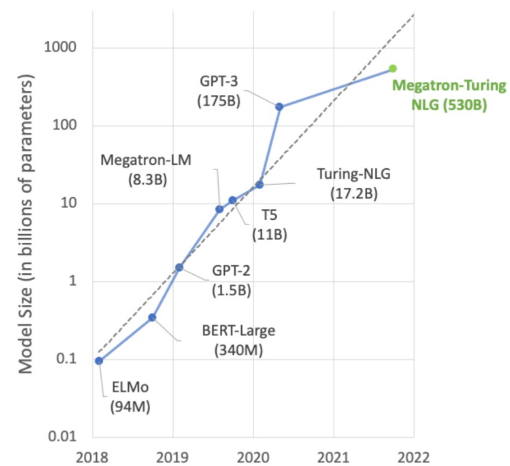
Oct 11, 2021 English

Using DeepSpeed and Megatron to Train Megatron-Turing NLG 530B, the World's Largest and Most Powerful Generative Language Model

By Paresh Kharya and Ali Alvi

Discuss (0) Share 0 Like

Tags: Conversational AI / NLP, DGX SuperPOD, featured, HPC / Supercomputing, Megatron, Technical Walkthrough



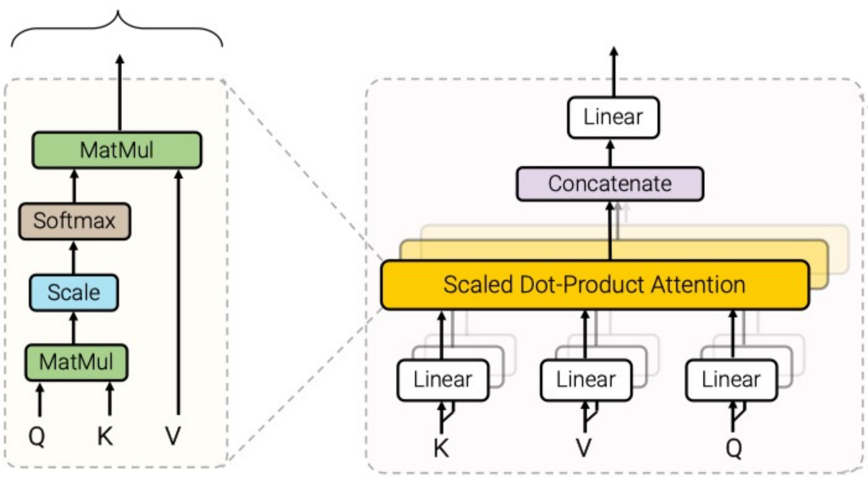
Background

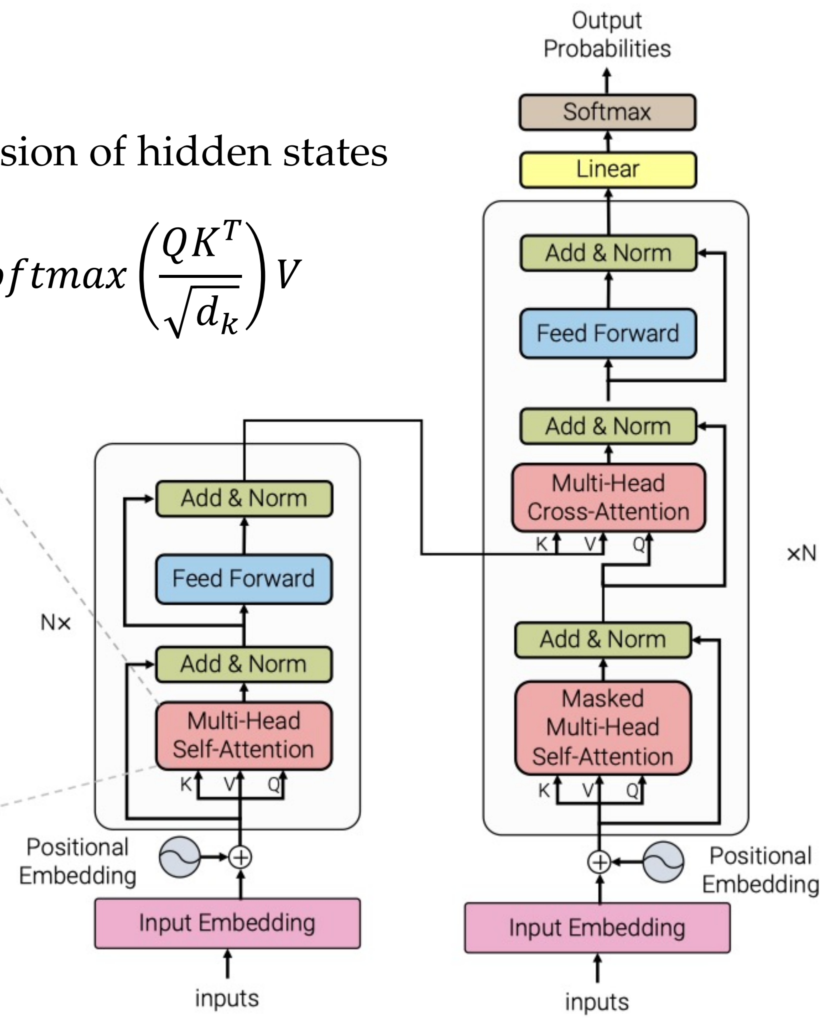
Transformer (encoder) architecture

n: Sequence length d: Dimension of hidden states

Computational Complexity $O(n^2d)$

$$Attention(Q,K,V) = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$



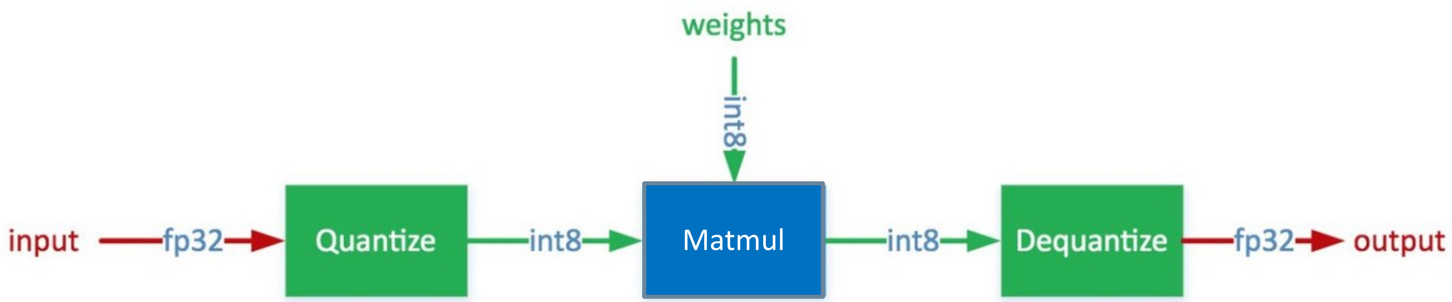


Total complexity: $O(n^2dN)$

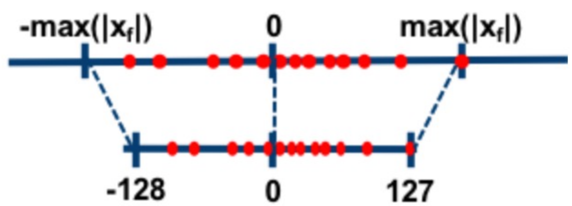
Outline

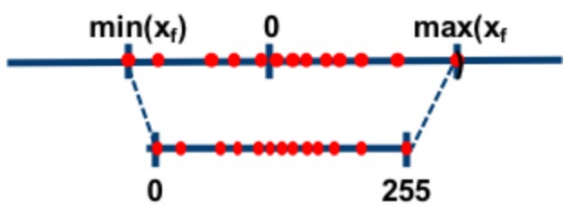
- Background
- General Methods
  - Quantization / Mixed Precision
  - Knowledge Distillation
- Transformer-specific Methods
  - Weight Sharing
  - Complexity reduction of self-attention
- Conclusions

Quantization / Mixed Precision



- Linear Quantization
  - Quantize: $q = Round(S(f - Z))$ Dequantize: $f = \frac{q}{S} + Z$
  - Symmetrical: $S = \frac{2^{n-1}-1}{\max(|x|)}$ $Z = 0$
  - Asymmetrical: $S = \frac{2^{n-1}-1}{\max(x)-\min(x)}$ $Z = min(x)$
- Non-linear Quantization
- Tools
  - TensorFlow Lite
  - torch.quantization
  - Nvidia Apex





Outline

- Background
- General Methods
  - Quantization / Mixed Precision
  - Knowledge Distillation
- Transformer-specific Methods
  - Weight Sharing
  - Complexity reduction of self-attention
- Conclusions

Knowledge Distillation

- **Goal:** Compress a heavy teacher model into a lightweight student model while maintaining its performance.
- **Method:** Let the student model imitate the teacher model.

$$L_{KD} = \sum_{\boldsymbol{x} \in \mathcal{X}} L(f^{(s)}(\boldsymbol{x}), f^{(t)}(\boldsymbol{x}))$$

KD Layers	$\boldsymbol{f}(\boldsymbol{x})$	$\boldsymbol{L}(\cdot)$
Embedding-layer distillation	output of the embedding layer	MSE
Attention-layer distillation	query-key / value-value matrices	MSE / KL
Hidden-layer distillation	output hidden states of corresponding sub-layers	MSE / Cos
Prediction-layer distillation	soft labels	CE / MSE

- **Tools:** TextBrewer (https://github.com/airaria/TextBrewer)

Knowledge Distillation

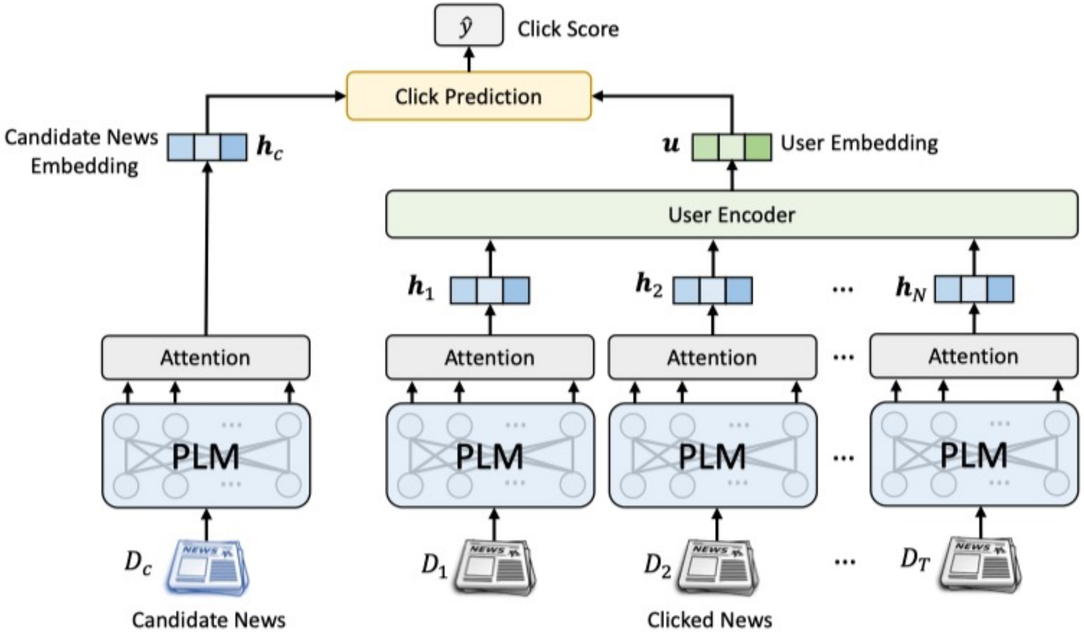
KD Methods	KD at Pre-training Stage					KD at Fine-tuning Stage			
	Embed	Query-Key	Value-Value	Hidden	Prediction	Embed	Query-Key	Hidden	Prediction
BERT-PKD								MSE	CE
DistilBERT				Cos	CE				
TinyBERT	MSE	MSE		MSE		MSE	MSE	MSE	CE
MobileBERT		KL		MSE	MSE				
MiniLM		KL	KL						

[1] https://zhuanlan.zhihu.com/p/273378905

Tiny-NewsRec

Background

- Learning accurate news representations from news texts is the prerequisite for high-quality news recommendation.
- Pre-trained language models (PLMs) are powerful in text modeling and have been employed for news understanding in news recommendation.
  - E.g. PLM-NR (SIGIR 2021)



Tiny-NewsRec

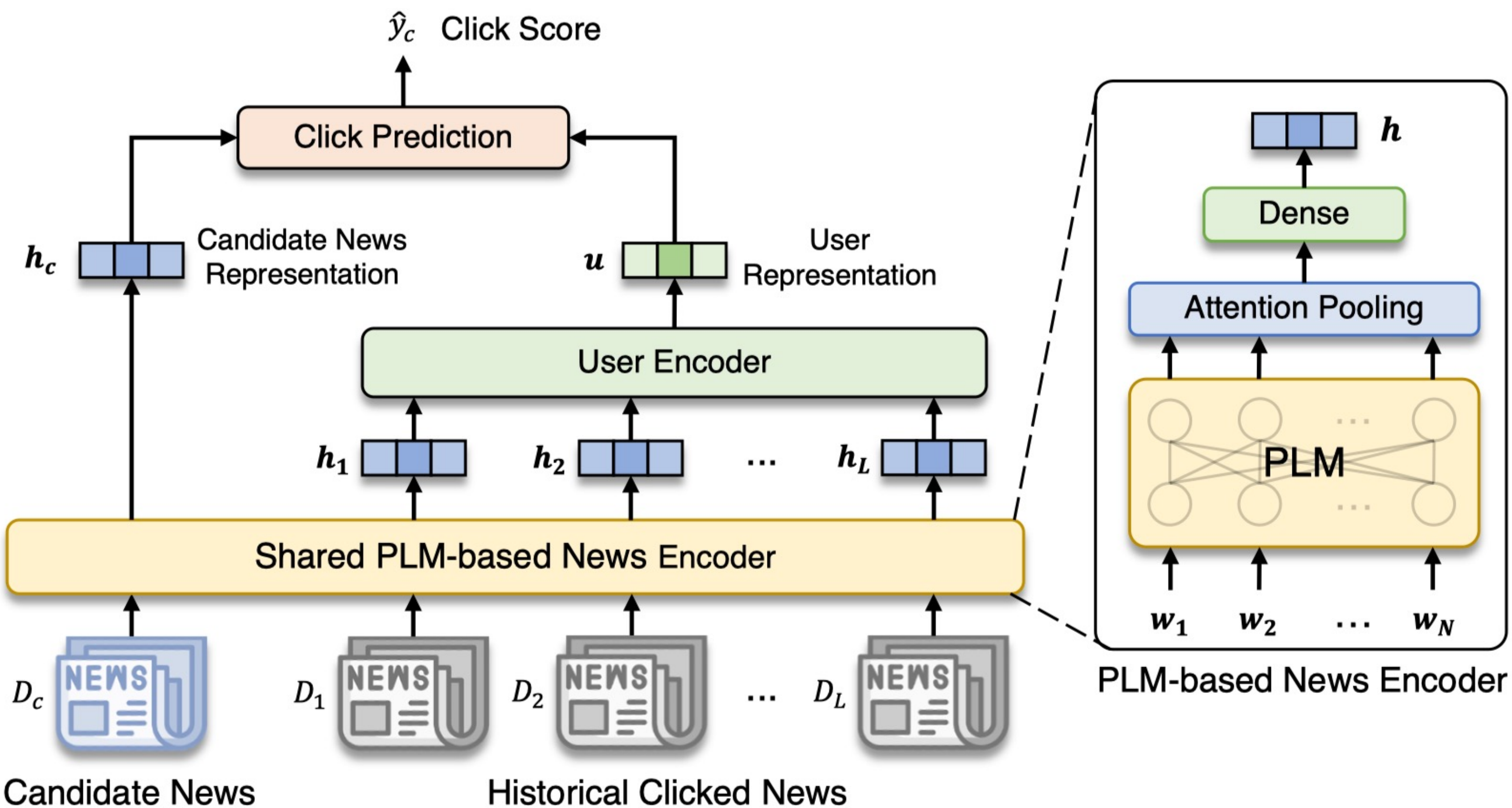
- ## Motivation
  - **[Effectiveness]** PLMs are usually pre-trained on general corpus (e.g. BookCorpus, Wikipedia), which have some gaps with the news domain. Directly finetuning PLMs with the news recommendation task may be sub-optimal for news understanding.
  - **[Efficiency]** Deploying these PLM-based news recommendation models to provide low-latency online services requires extensive computational resources.

Tiny-NewsRec

- Approach
  - **[Effectiveness]** A self-supervised domain-specific post-training method.
  - **[Efficiency]** A two-stage knowledge distillation method with multiple teachers.

Tiny-NewsRec

- PLM-based News Recommendation Model



Tiny-NewsRec

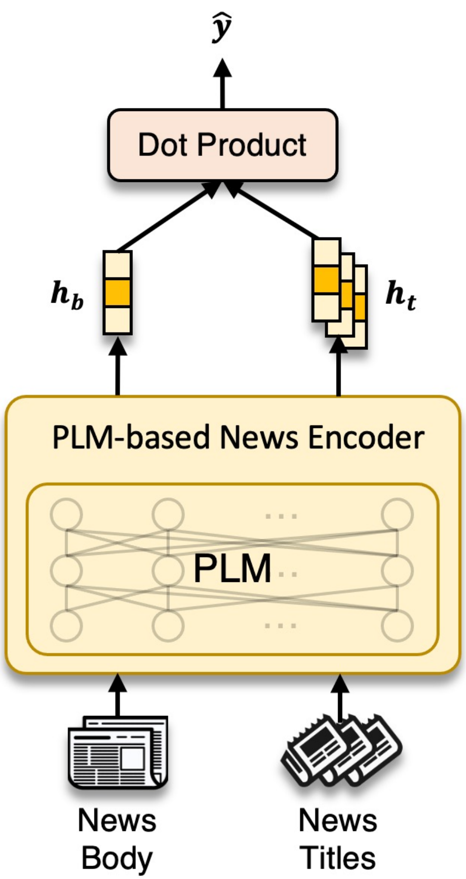
Domain-specific Post-training (before finetuning)

- A self-supervised matching task between news titles and news bodies.
- Contrastive learning
  - Anchor: news body
  - Positive sample: the corresponding news title
  - Negative samples: randomly select N other news titles

$$p_i = \frac{\exp(\hat{y}^+)}{\exp(\hat{y}^+) + \sum_{j=1}^{N} \exp(\hat{y}_j^-)}$$

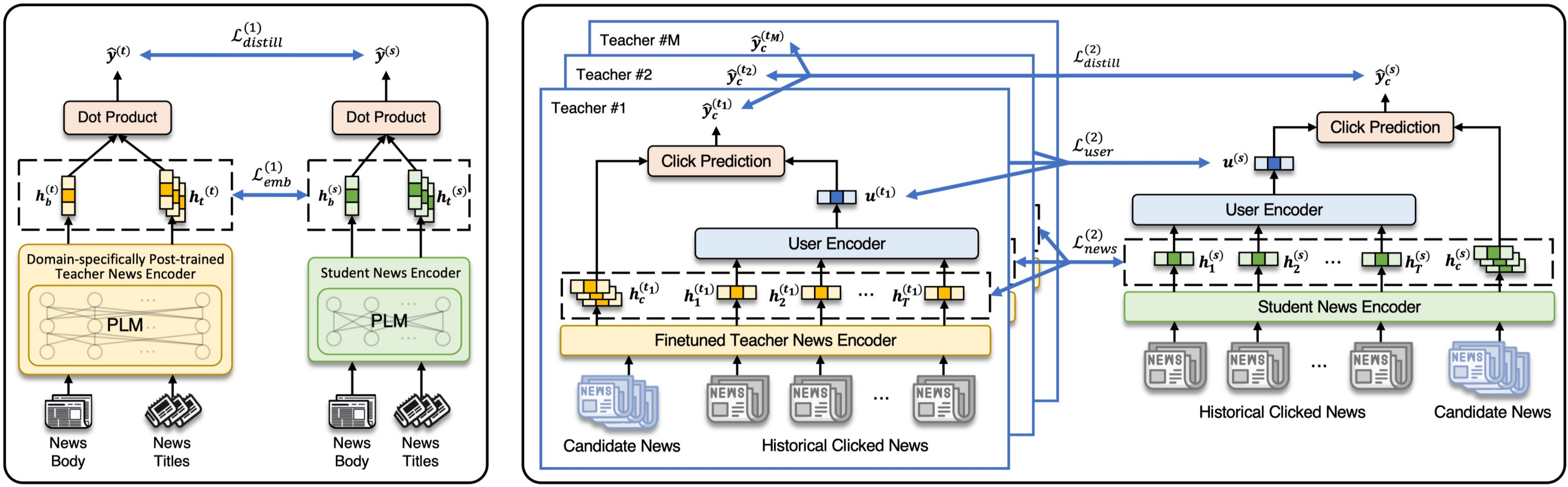
$$\mathcal{L}_{match} = -\sum_{i \in \mathcal{T}} \log(p_i)$$

- The news encoder can generate more discriminative news representations.



Tiny-NewsRec

- Two-stage Knowledge Distillation with Multiple Teachers



(a) First stage knowledge distillation.

(b) Second stage knowledge distillation.

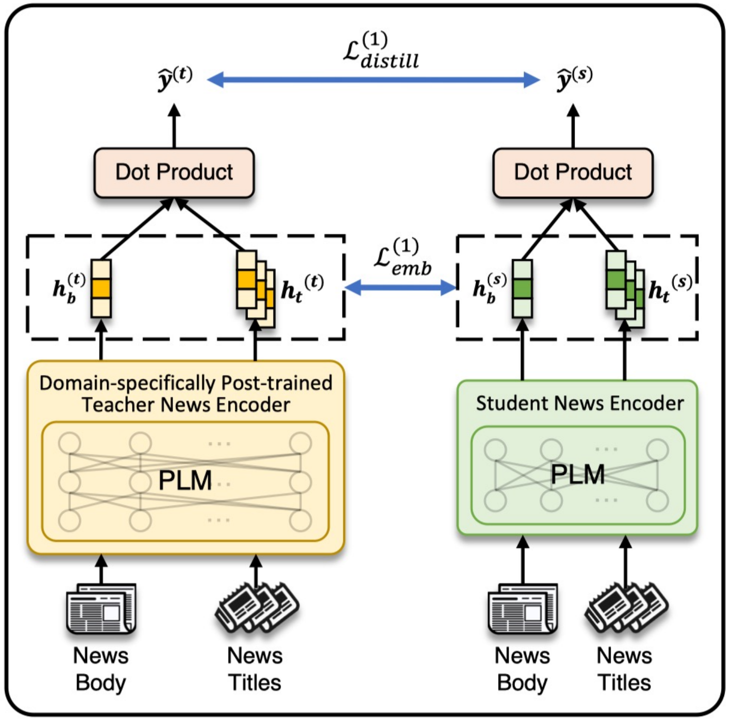
Tiny-NewsRec

- First-stage Knowledge Distillation
  - Force the student news encoder imitate the domain-specifically post-trained teacher news encoder in the matching task between news titles and news bodies.

$$\mathcal{L}_{distill}^{(1)} = T_1^2 \cdot \mathrm{CE}(\hat{\mathbf{y}}^{(t)}/T_1, \hat{\mathbf{y}}^{(s)}/T_1)$$

$$\mathcal{L}_{emb}^{(1)} = \mathrm{MSE}(\mathbf{h}_t^{(t)}, \mathbf{h}_t^{(s)}) + \mathrm{MSE}(\mathbf{h}_b^{(t)}, \mathbf{h}_b^{(s)})$$

$$\mathcal{L}^{(1)} = \mathcal{L}_{distill}^{(1)} + \mathcal{L}_{emb}^{(1)} + \beta_1 \cdot \mathrm{CE}(\hat{\mathbf{y}}^{(s)}, \mathbf{y})$$



(a) First stage knowledge distillation.

Tiny-NewsRec

Second-stage Knowledge Distillation

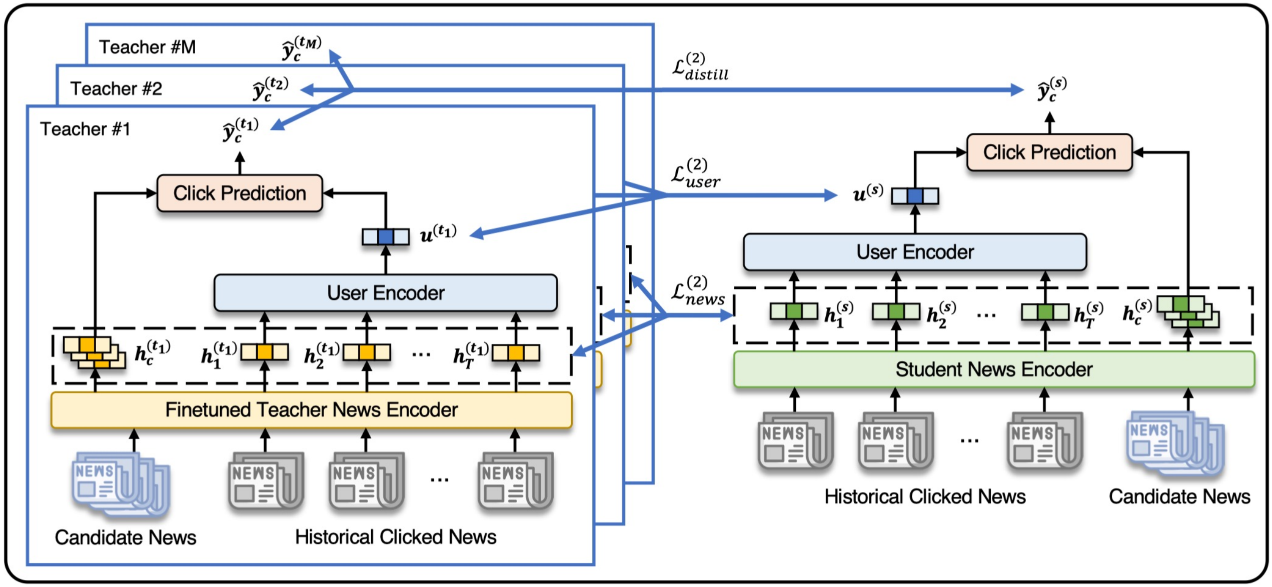
- Use multiple domain-specifically post-trained teacher models to transfer more comprehensive knowledge to the student during finetuning.
- For each training sample, assigning a weight to each teacher according to its performance.

$$w_i = \frac{\exp(-\mathrm{CE}(\hat{\mathbf{y}}_c^{(t_i)}, \mathbf{y}_c) \times \omega)}{\sum_{j=1}^{M} \exp(-\mathrm{CE}(\hat{\mathbf{y}}_c^{(t_j)}, \mathbf{y}_c) \times \omega)}$$

$$\mathcal{L}_{distill}^{(2)} = T_2^2 \cdot \mathrm{CE}(\sum_{i=1}^{M} w_i \cdot \hat{\mathbf{y}}_c^{(t_i)}/T_2, \hat{\mathbf{y}}_c^{(s)}/T_2)$$

$$\mathcal{L}_{emb}^{(2)} = \sum_{i=1}^{M} w_i \cdot (\mathcal{L}_{news_i}^{(2)} + \mathcal{L}_{user_i}^{(2)})$$

$$\mathcal{L}^{(2)} = \mathcal{L}_{distill}^{(2)} + \mathcal{L}_{emb}^{(2)} + \beta_2 \cdot \mathrm{CE}(\hat{\mathbf{y}}_c^{(s)}, \mathbf{y}_c)$$



(b) Second stage knowledge distillation.

Tiny-NewsRec

- Experiments
  - Datasets
    - MIND (https://msnews.github.io), Feeds
    - News
  - Baselines:
    - PLM-NR (**F**ine**t**une)
    - PLM-NR (**F**urther **P**re-train)
    - TinyBERT
    - NewsBERT

MIND			
# News	161,013	# Users	1,000,000
# Impressions	15,777,377	# Clicks	24,155,470
Avg. title length	11.52		
Feeds			
# News	377,296	# Users	10,000
# Impressions	320,925	# Clicks	437,072
Avg. title length	11.93		
News			
# News	1,975,767	Avg. title length	11.84
Avg. body length	511.43		

Table 1: Detailed statistics of *MIND*, *Feeds* and *News*.

Tiny-NewsRec

Performance Comparison

Model	MIND				Feeds			
	AUC	MRR	nDCG@5	nDCG@10	AUC	MRR	nDCG@5	nDCG@10
PLM-NR-12 (FT)	69.72±0.15	34.74±0.10	37.99±0.11	43.71±0.07	67.93±0.13	34.42±0.07	37.46±0.09	45.09±0.07
PLM-NR-12 (FP)	69.82±0.14	34.90±0.11	38.17±0.09	43.83±0.07	68.11±0.11	34.49±0.12	37.58±0.07	45.11±0.08
PLM-NR-12 (DP)*	**70.20±0.10**	**35.27±0.08**	**38.54±0.07**	**44.20±0.08**	**68.71±0.08**	**35.10±0.09**	**38.32±0.06**	**45.83±0.08**
PLM-NR-4 (FT)	69.49±0.14	34.40±0.10	37.64±0.10	43.40±0.09	67.46±0.12	33.71±0.11	36.69±0.08	44.36±0.09
PLM-NR-2 (FT)	68.99±0.08	33.59±0.14	36.81±0.11	42.61±0.11	67.05±0.14	33.33±0.09	36.15±0.10	43.90±0.12
PLM-NR-1 (FT)	68.12±0.12	33.20±0.07	36.29±0.09	42.07±0.10	66.26±0.10	32.55±0.12	35.22±0.07	42.99±0.09
TinyBERT-4	69.77±0.13	34.83±0.09	38.02±0.11	43.69±0.09	67.73±0.11	34.00±0.08	37.03±0.10	44.59±0.12
TinyBERT-2	69.44±0.17	34.11±0.07	37.55±0.08	43.14±0.07	67.35±0.13	33.69±0.05	36.59±0.08	44.21±0.09
TinyBERT-1	68.42±0.12	33.55±0.10	36.69±0.09	42.35±0.08	66.53±0.10	32.81±0.07	35.61±0.11	43.29±0.09
NewsBERT-4	69.85±0.17	34.91±0.09	38.19±0.09	43.84±0.08	68.34±0.13	34.58±0.06	37.69±0.09	45.27±0.08
NewsBERT-2	69.62±0.09	34.67±0.12	37.86±0.11	43.54±0.11	67.90±0.07	34.26±0.09	37.29±0.10	44.86±0.11
NewsBERT-1	68.67±0.11	33.95±0.07	37.05±0.14	42.74±0.13	67.00±0.10	33.24±0.11	36.09±0.08	43.80±0.07
Tiny-NewsRec-4*	**70.40±0.05**	**35.43±0.08**	**38.76±0.05**	**44.43±0.04**	**68.93±0.06**	**35.21±0.09**	**38.43±0.08**	**45.97±0.10**
Tiny-NewsRec-2	70.28±0.07	35.32±0.07	38.65±0.07	44.28±0.08	68.58±0.03	34.82±0.07	38.02±0.09	45.57±0.07
Tiny-NewsRec-1	69.85±0.03	34.93±0.08	38.21±0.09	43.84±0.09	68.14±0.05	34.53±0.07	37.61±0.08	45.14±0.08

Table 2: Performance comparisons of different models. (FT=Finetune, FP=Further Pre-train, DP=Domain-specific Post-train) *Improvements over other baselines are significant at $p < 0.01$ (by comparing the models with the same number of layers).

Tiny-NewsRec

- Effectiveness of Multiple Teacher Models

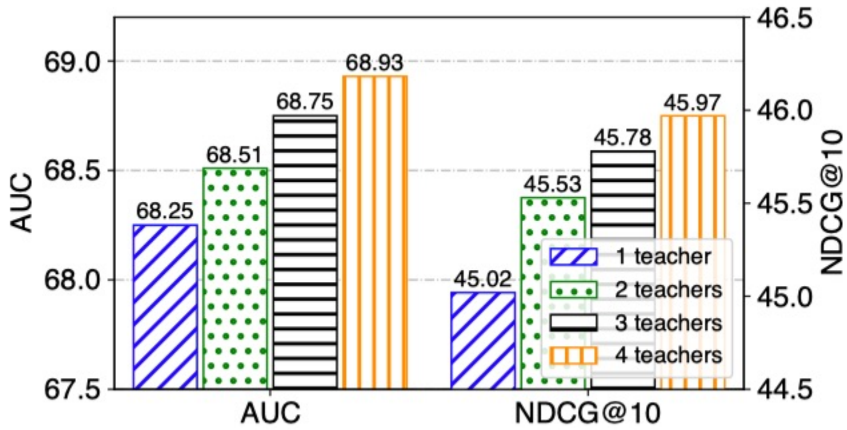


Figure 3: Impact of different number of teacher models.

- Effectiveness of Two-stage Knowledge Distillation

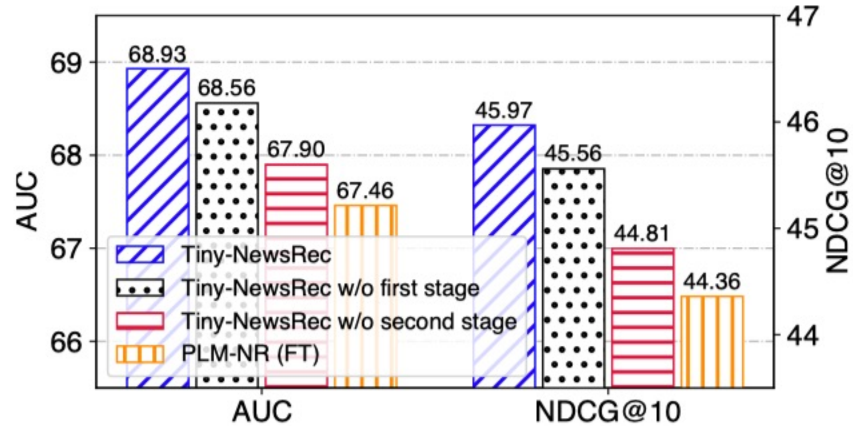


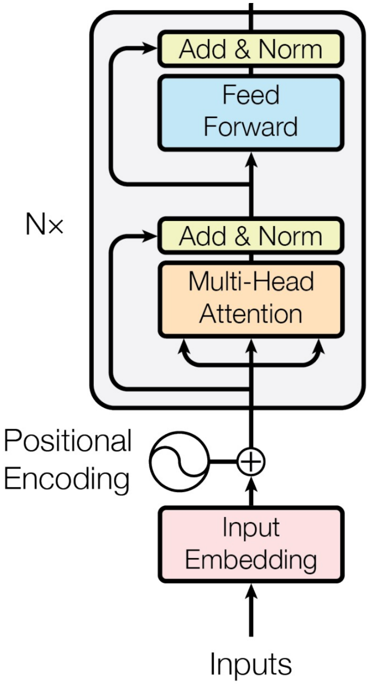
Figure 4: Effectiveness of each stage in our framework.

Outline

- Background
- General Methods
  - Quantization / Mixed Precision
  - Knowledge Distillation
- **Transformer-specific Methods**
  - **Weight Sharing**
  - Complexity reduction of self-attention
- Conclusions

Weight Sharing

- Sharing parameters across layers
  - A form of regularization
- What to share ?
  - Parameters in Attention
  - Parameters in FFN
  - Both
- Who to share ?
  - Share across all layers
  - Share across every N/M layers



Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT base E=768	all-shared	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8
	shared-attention	83M	89.9/82.7	80.0/77.2	84.0	91.4	67.7	81.6
	shared-FFN	57M	89.2/82.1	78.2/75.4	81.5	90.8	62.6	79.5
	not-shared	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT base E=128	all-shared	12M	89.3/82.3	80.0/77.1	82.0	90.3	64.0	80.1
	shared-attention	64M	89.9/82.8	80.7/77.9	83.4	91.9	67.6	81.7
	shared-FFN	38M	88.9/81.6	78.6/75.6	82.3	91.7	64.4	80.2
	not-shared	89M	89.9/82.8	80.3/77.3	83.2	91.5	67.9	81.6

Outline

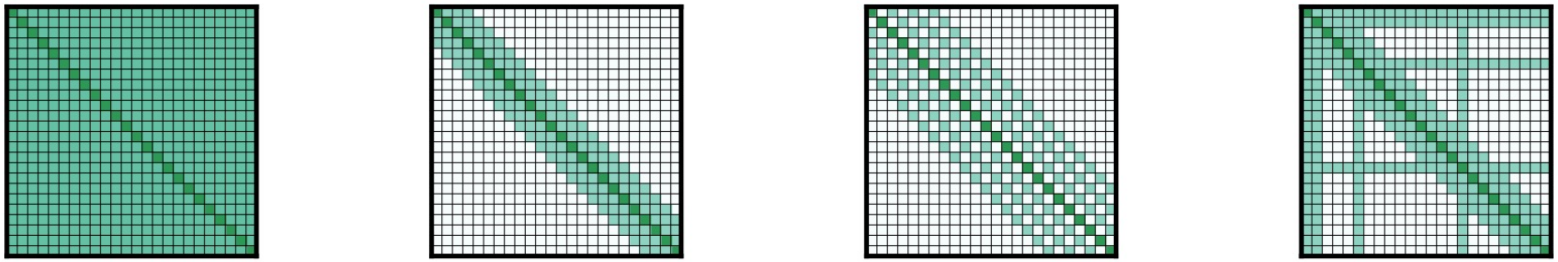
- Background
- General Methods
  - Quantization / Mixed Precision
  - Knowledge Distillation
- **Transformer-specific Methods**
  - Weight Sharing
  - **Complexity reduction of self-attention**
- Conclusions

Complexity Reduction of Self-attention

- **General idea:** Approximate the quadratic-cost self-attention mechanism
  - Sparse attention
  - Low-rank approximation
  - Kernelization
  - Additive attention
  - …

Complexity Reduction of Self-attention

LongFormer

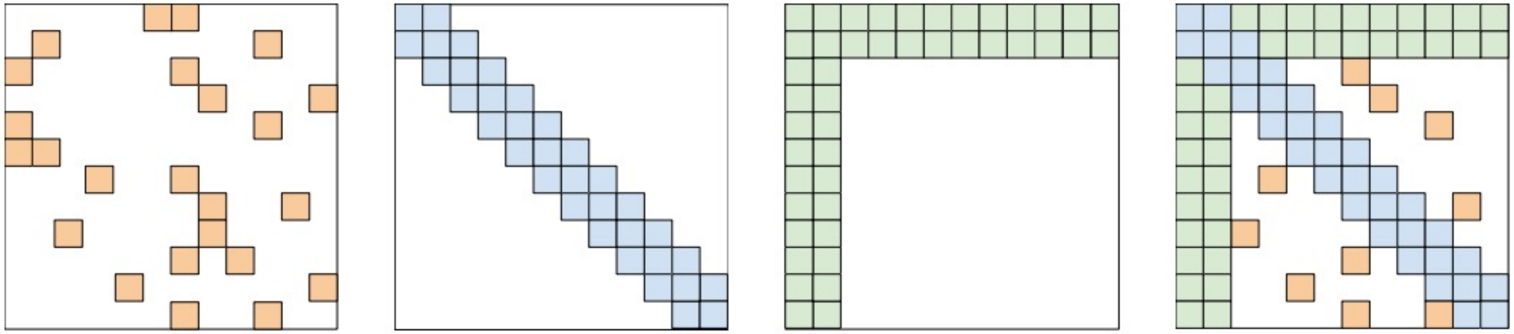


(a) Full n^2 attention (b) Sliding window attention (c) Dilated sliding window (d) Global+sliding window

- Sliding window: each token only attend to $\omega/2$ tokens on each side -> reception field $N\times\omega$
- Dilated sliding window: the window has gaps of size c -> reception field $N\times c\times\omega$
- Global attention: few pre-selected input locations
- Complexity: $O(n\omega d)$

Complexity Reduction of Self-attention

- Big Bird



(a) Random attention (b) Window attention (c) Global Attention (d) BIGBIRD

  - Random attention: each token attends over r random keys
  - Window attention: each token only attends to $\omega/2$ tokens on each side
  - Global attention: g global tokens attending on the entire sequence
  - Complexity: $O(n(\omega + r)d)$

Complexity Reduction of Self-attention

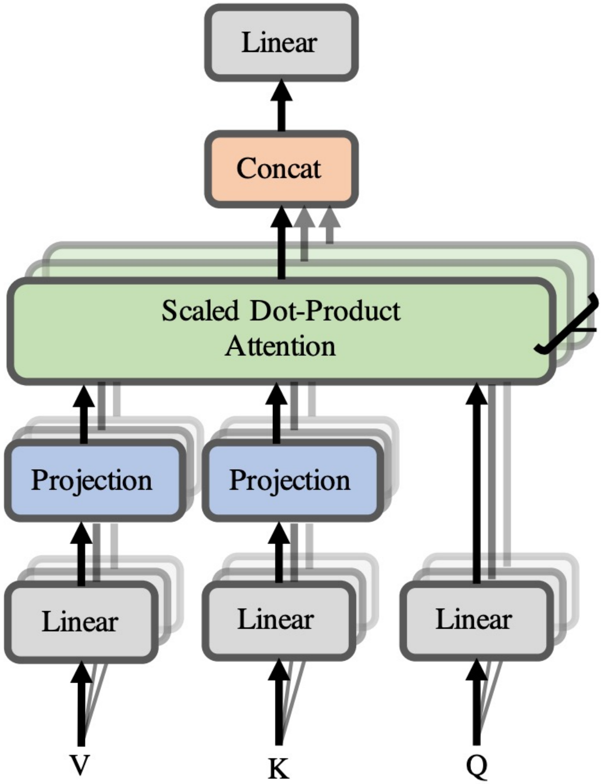
Linformer

- Add two projection matrices $E_i, F_j \in \mathbb{R}^{k \times n}$

$$\overline{\text{head}_i} = \text{Attention}(QW_i^Q, E_iKW_i^K, F_iVW_i^V)$$
$$= \underbrace{\text{softmax}\left(\frac{QW_i^Q(E_iKW_i^K)^T}{\sqrt{d_k}}\right)}_{\bar{P}:n\times k} \cdot \underbrace{F_iVW_i^V}_{k\times d}$$

$QW_i^Q: n\times d$ $\quad E_iKW_i^K: k\times d$ $\quad F_iVW_i^V: k\times d$

- When $k = O(d/\epsilon^2)$, one can approximate $P \cdot VW_i^V$ using linear self-attention with ϵ error.
- Complexity: $O(nkd)$



Complexity Reduction of Self-attention

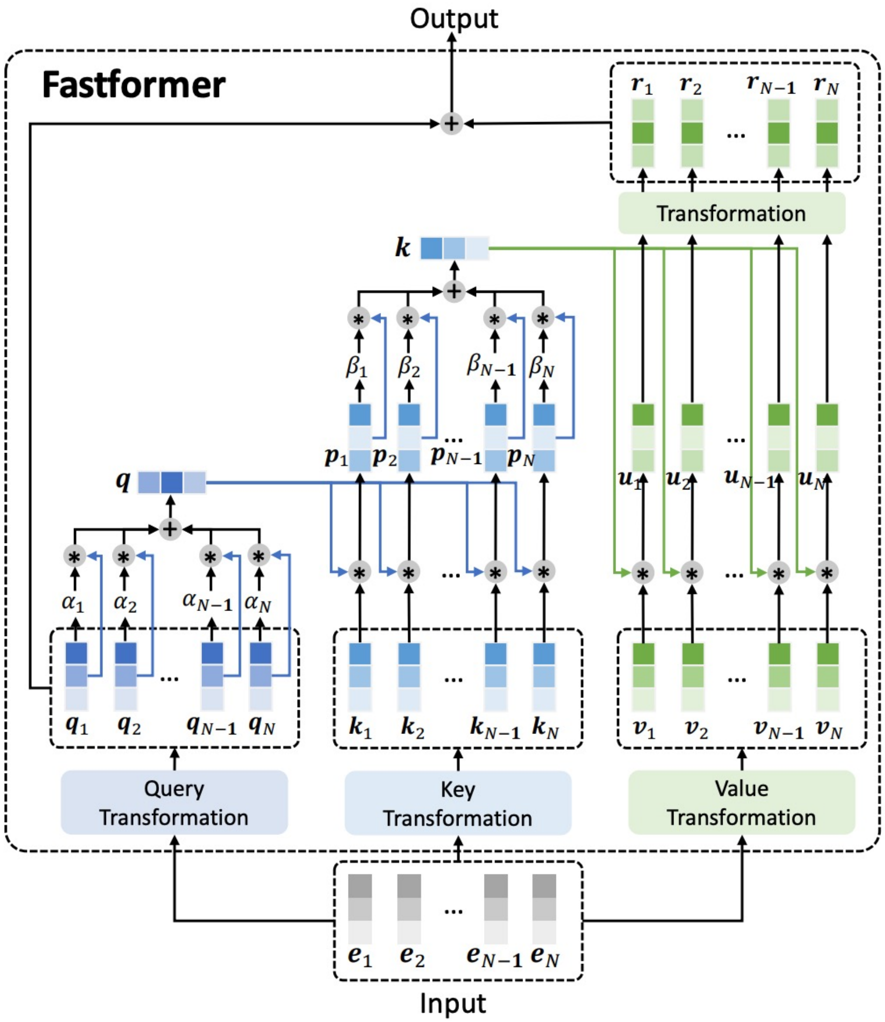
- Fastformer
  - Replace self-attention with additive attention.
  - q: global query vector

$$\alpha_i = \frac{\exp(\mathbf{w}_q^T \mathbf{q}_i / \sqrt{d})}{\sum_{j=1}^{N} \exp(\mathbf{w}_q^T \mathbf{q}_j / \sqrt{d})} \qquad \mathbf{q} = \sum_{i=1}^{N} \alpha_i \mathbf{q}_i$$

  - k: global key vector

$$\beta_i = \frac{\exp(\mathbf{w}_k^T \mathbf{p}_i / \sqrt{d})}{\sum_{j=1}^{N} \exp(\mathbf{w}_k^T \mathbf{p}_j / \sqrt{d})} \qquad \mathbf{k} = \sum_{i=1}^{N} \beta_i \mathbf{p}_i$$

  - Use element-wise product to model the interaction between q and attention keys / k and attention values.
  - Complexity: $O(nd)$



Complexity Reduction of Self-attention

Effectiveness Comparison

Text classification

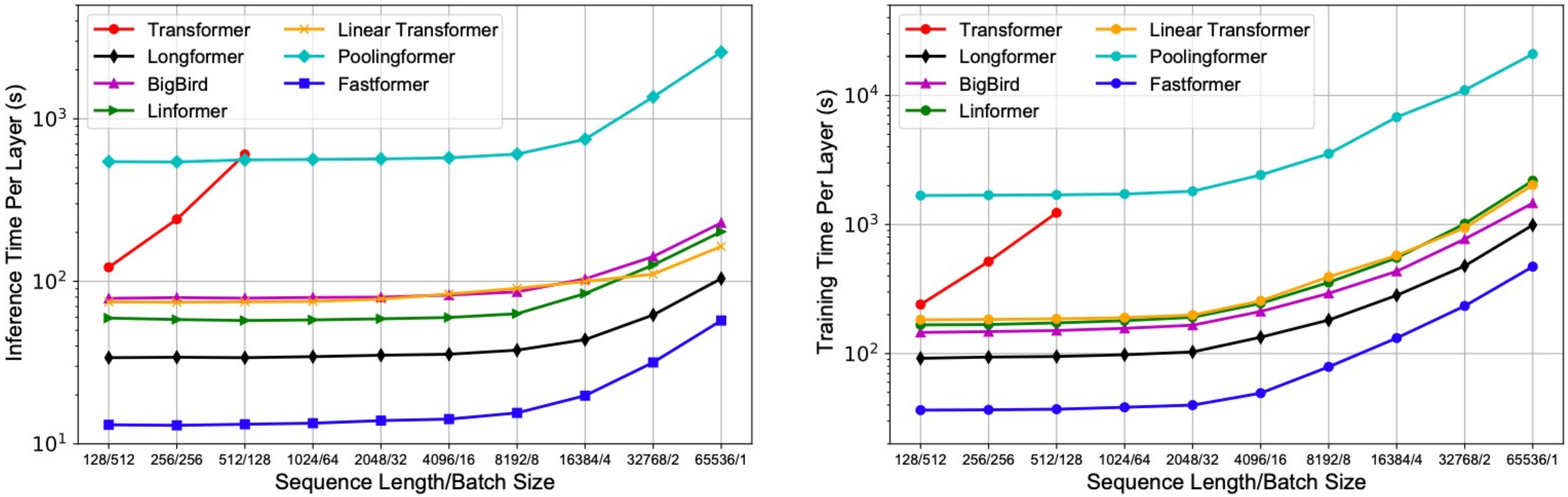
Methods	Amazon		IMDB		MIND	
	Accuracy	Macro-F	Accuracy	Macro-F	Accuracy	Macro-F
Transformer	65.32±0.35	42.31±0.33	52.04±0.50	42.69±0.47	80.90±0.20	60.02±0.21
Longformer	65.45±0.39	42.48±0.44	52.21±0.36	43.36±0.38	81.36±0.21	62.59±0.23
BigBird	66.14±0.42	42.96±0.40	53.23±0.46	44.03±0.44	81.93±0.24	63.58±0.26
Linformer	**66.20**±0.49	43.13±0.48	53.17±0.59	44.34±0.57	82.16±0.28	63.77±0.30
Linear Transformers	66.12±0.42	43.04±0.44	53.09±0.47	44.30±0.49	82.25±0.23	63.81±0.22
Poolingformer	66.05±0.44	43.00±0.45	53.78±0.51	44.52±0.50	**82.46**±0.24	**64.10**±0.26
Fastformer	66.13±0.29	**43.23**±0.30	**54.10**±0.42	**44.65**±0.44	82.34±0.19	63.89±0.20

Text summarization

Method	CNN/DailyMail			PubMed		
	R-1	R-2	R-L	R-1	R-2	R-L
Transformer	38.52	16.04	35.87	34.26	11.88	31.64
Longformer	37.89	15.46	35.19	36.92	14.34	33.75
BigBird	38.31	15.78	35.60	37.73	14.99	34.51
Linformer	37.96	15.58	35.34	37.22	14.48	34.02
Linear Transformer	37.24	14.87	34.64	36.43	13.80	33.21
Poolingformer	**38.58**	16.16	36.17	37.82	15.15	34.63
Fastformer	38.54	**16.22**	**36.21**	**38.09**	**15.44**	**34.81**

Complexity Reduction of Self-attention

- Efficiency Comparison



(a) Inference. (b) Training.

Outline

- Background
- General Methods
  - Quantization / Mixed Precision
  - Knowledge Distillation
- Transformer-specific Methods
  - Weight Sharing
  - Complexity reduction of self-attention
- **Conclusions**

Conclusions

- Quantization / Mixed Precision
- Knowledge Distillation
  - How to reduce the training cost of multiple teacher models?
  - Can we add feedback from the student model to the teacher model?
- Weight Sharing
- Complexity reduction of self-attention

Q&A

References

- https://zhuanlan.zhihu.com/p/273378905
- Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020. Efficient Transformers: A Survey. arXiv preprint arXiv:2009.06732.
- Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. 2020. arXiv preprint arXiv:2004.05150.
- Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. 2020. arXiv preprint arXiv:2007.14062.
- Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Selfattention with linear complexity. 2020. arXiv preprint arXiv:2006.04768.
- Chuhan Wu, Fangzhao Wu, Tao Qi, Yongfeng Huang, and Xing Xie. Fastformer: Additive Attention Can Be All You Need. 2021. arXiv preprint arXiv: 2108.09084.

References

- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. 2017. arXiv preprint arXiv: 1706.03762.
- Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. 2020. arXiv preprint arXiv: 1909.11942.
- Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. Empowering News Recommendation with Pre-Trained Language Models. 2021. In SIGIR, 1652–1656.