NLP From Scratch Without Large-Scale Pretraining: A Simple and Efficient Framework

Xingcheng Yao*, Yanan Zheng*, Xiaocong Yang, Zhilin Yang

*Equal Contribution
Code: https://github.com/yaoxingcheng/TLM

Outline

- Introduction: Background and Motivations
- Methods
- Experiments
- Discussion and Conclusion

Introduction: PLMs

- Pretrained Language Models (PLMs) have drawn much attention from the NLP community
  - based on the Transformer[1] architecture
  - BERT[2], RoBERTa[3], T5[4], GPT[5], XLNet[6]...
  - pretrained on large general corpora via SSL:
    - masked language modeling[2,3]
    - autoregressive language modeling[5]
    - permutation language modeling[6]
  - significantly improved the performance of many NLP tasks

[1] Attention is all you need. Vaswani et al., 2017.
[2] BERT: Pre-training of deep bidirectional transformers for language understanding. Devlin et al., 2019.
[3] RoBERTa: A robustly optimized BERT pretraining approach. Liu et al., 2019.
[4] Exploring the limits of transfer learning with a unified text-to-text transformer. Raffel et al., 2019.
[5] Language models are unsupervised multitask learners. Radford et al., 2018.
[6] XLNet: Generalized autoregressive pretraining for language understanding. Yang et al., 2019.

Introduction: PLMs

- One inconvenient issue: PLMs are usually computationally expensive
  - RoBERTa-Large comsumes a computational cost of 4.36e21 FLOPs.
  - Larger PLMs such as GPT-3 consume 50 times more FLOPs than RoBERTa-large.
- Why expensiveness is an issue:
  - Hard for many research groups with limited budgets to train their customized PLMs.
  - Creating a high barrier for NLP research.

[1] RoBERTa: A robustly optimized BERT pretraining approach. Liu et al., 2019.
[2] Language models are few-shot learners. Brown et al., 2020.

Introduction: Efficient NLP

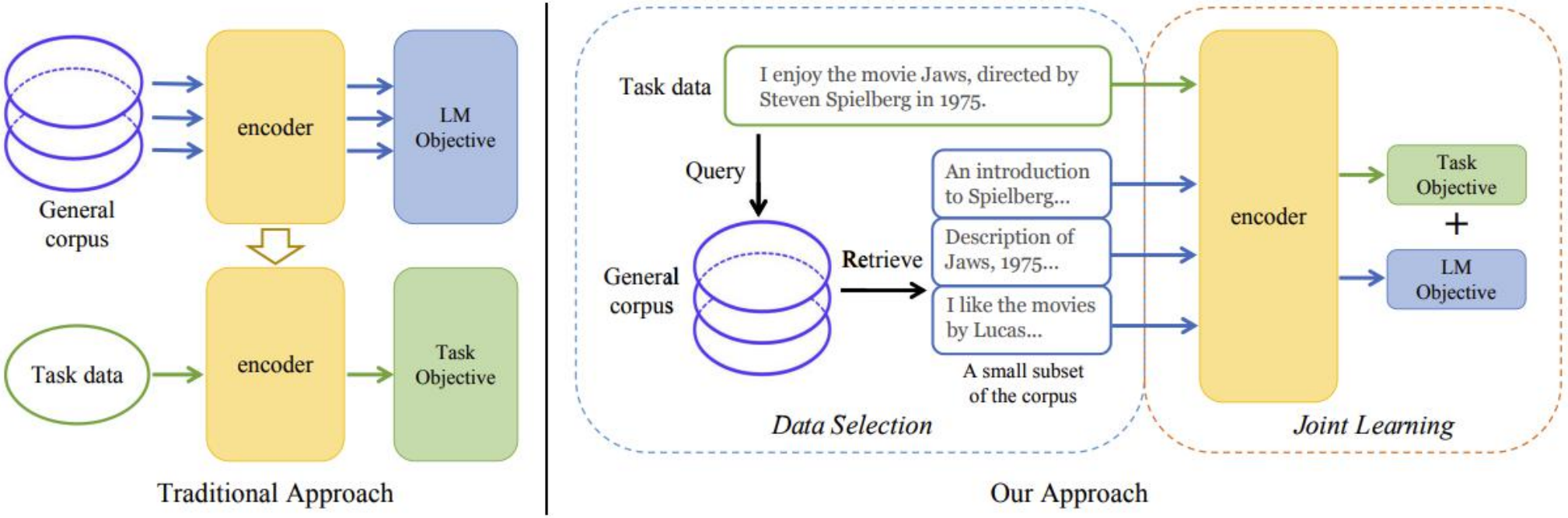
- Previous solutions to improve the efficiency for NLP:
  - During Training:
    - Data and Model parallelism, e.g. Megatron-LM[1]: no actual reduction in cost.
    - Identify efficient architecture[2,3] or incoporate manually designed tricks[4,5]: reduction within one-order of magnitude.
  - During Inference:
    - distillation, quantization, pruning
    - still relies on large-scale pretraining

[1] Megatron-LM: training multi-billion parameter language models using model parallelism. Shoeybi et al., 2019.
[2] Primer: Searching for efficient transformers for language modeling. So et al., 2021.
[3] EarlyBERT: Efficient BERT training via early-bird lottery tickets. Chen et al., 2021.
[4] ELECTRA: Pre-training text encoders as discriminators rather than generators. Clark et al., 2020.
[5] DeBERTa: Decoding_x0002_enhanced BERT with disentangled attention. He et al., 2021.

Introduction: TLM

- We propose Task-driven Language Modeling (TLM)
- Accelerate LM from the perspective of Data Efficiency
  - Inspiration from human beings:
    - We humans can master a certain task by using only a small portion of knowledge
    - Most of the knowledge is just useless if irrelevant to the task
    - e.g. Students cramming for an exam only need to review highlighted chapters
    - For NLP, there could be much redundency in the general corpus for a certain task
  - Supervision signals provided by labeled task data is much more efficient than general unlabeled data

Method: Overview



TLM consists of two steps:

1. Data Selection: Retrieve data from a general corpus using task data as queries
2. Joint Learning: Training a model from scratch by jointly optimizing the task objective and language modeling objective

Method: Data Selection

- Similarity-based data selection to prune the redundant data
  - We use each example in the task data as queries, and take the union top-k similar sequences retrieved from the general corpus as the selected data
- BM25 is adopted as the similarity measure for the retrieval.
  - Note that we don't depend on embedding based dense retrievers which might require pretrained encoders
- No human-curated data is required in this process

Method: Joint Training

- We term the selected data as $\mathcal{S}$, and task data as $\mathcal{T}$, TLM optimizes the following loss function:

$$\mathbb{E}_{x\sim\mathcal{S}}[\rho_1\mathcal{L}_{\text{mlm}}(x)]$$
$$+\mathbb{E}_{x,y\sim\mathcal{T}}\left[\rho_2\mathcal{L}_{\text{mlm}}(x)+\mathcal{L}_{\text{task}}(f(x),y)\right]$$

- Here:
  - $\mathcal{L}_{\text{mlm}}(x)$ is MLM loss and $\mathcal{L}_{\text{task}}(f(x),y)$ is the cross-entropy loss
  - ρ_1 controls ratio of selected external data
  - ρ_2 controls the weight of language modeling loss on task data

Experiments: Main Results

Model	#Param	FLOPs[1]	Data[2]	AGNews	Hyp.	Help.	IMDB	ACL.	SciERC	Chem.	RCT	Avg.
BERT-Base[3]	109M	2.79E19	16GB	93.50 ±0.15	91.93 ±1.74	69.11 ±0.17	93.77 ±0.22	69.45 ±2.90	80.98 ±1.07	81.94 ±0.38	87.00 ±0.06	83.46
BERT-Large[3]	355M	9.07E19	16GB	93.51 ±0.40	91.62 ±0.69	69.39 ±1.14	**94.76** ±0.09	69.13 ±2.93	**81.37** ±1.35	**83.64** ±0.41	**87.13** ±0.09	**83.82**
TLM *(small-scale)*	109M	2.74E18	0.91GB	**93.74** ±0.20	**93.53** ±1.61	**70.54** ±0.39	93.08 ±0.17	**69.84** ±3.69	80.51 ±1.53	81.99 ±0.42	86.99 ±0.03	83.78
RoBERTa-Base[3]	125M	1.54E21	160GB	**94.02** ±0.15	93.53 ±1.61	70.45 ±0.24	**95.43** ±0.16	68.34 ±7.27	81.35 ±0.63	82.60 ±0.53	87.23 ±0.09	84.12
TLM *(medium-scale)*	109M	8.30E18	1.21GB	93.96 ±0.18	**94.05** ±0.96	**70.90** ±0.73	93.97 ±0.10	**72.37** ±2.11	**81.88** ±1.92	**83.24** ±0.36	**87.28** ±0.10	**84.71**
RoBERTa-Large[3]	355M	4.36E21	160GB	94.30 ±0.23	**95.16** ±0.00	70.73 ±0.62	**96.20** ±0.19	**72.80** ±0.62	82.62 ±0.68	84.62 ±0.50	**87.53** ±0.13	85.50
TLM *(large-scale)*	355M	7.59E19	3.64GB	**94.34** ±0.12	**95.16** ±0.00	**72.49** ±0.33	95.77 ±0.24	72.19 ±1.72	**83.29** ±0.95	**85.12** ±0.85	87.50 ±0.12	**85.74**

- Extensive experiments on 8 different tasks from 4 domains show that:
  - TLM matches the performance of BERT-Large with approximately 1/33 of its FLOPS
  - TLM reduces the training FLOPs of RoBERTa by two orders of maginitude

Experiments: Ablations

	IMDB	SciERC	ChemProt
Random			
w/ $\mathcal{C}_{\text{BERT}}$	93.65±0.09	83.80±0.62	80.65±0.48
w/ $\mathcal{C}_{\text{RoBERTa}}$	94.04±0.22	83.10±1.54	80.73±0.46
BM25			
w/ $\mathcal{C}_{\text{BERT}}$	94.40±0.09	86.07±0.48	83.64±0.26
w/ $\mathcal{C}_{\text{RoBERTa}}$	**94.90**±0.06	**87.41**±0.36	**84.99**±0.72

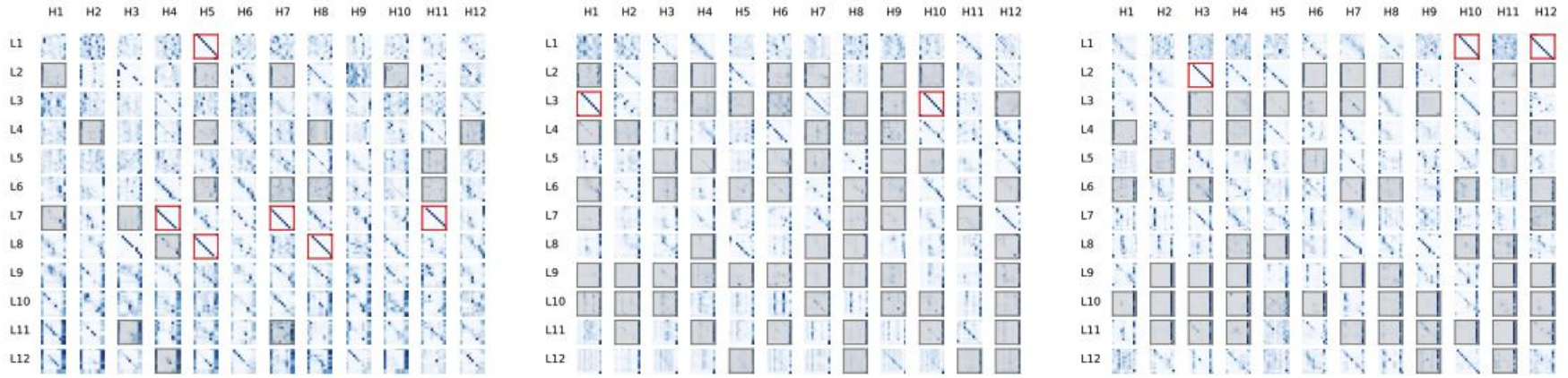
- Data quality counts:
  - Data selection with BM25 yields significant improvement over randomly selected data
  - Performance improves as the size of the general corpus scales up

	Helpfulness	SciERC	ChemProt
ρ_1 =1	**71.02**±0.51	80.72±3.32	73.27±0.30
ρ_1 =3	70.41±0.52	80.01±0.72	79.43±1.03
ρ_1 =99	69.56±0.23	84.95±0.57	83.30±0.30
ρ_1 =999	69.35±0.72	**86.07**±0.48	**83.64**±0.26
Ext only	69.76±0.50	85.66±1.58	82.50±0.27

- Joint training makes a difference:
  - High resource tasks requires a small ratio on external selected data, low resource requires a large ratio.
  - In general, only use external selected data is suboptimal.

Experiments: Analysis

- The following figure is the attention pattern of the sentence:
  “[CLS] crystallographic comparison with the structurally related. [SEP]”



(a) TLM (*Medium scale*) (b) BERT-Base (c) RoBERTa-Base

- “Verticle pattern”: encode few semantic or syntactic information[1]
- “Diagonal pattern”: vital contribution to final predictions of the modelp[2]
- TLM has much less verticle attention pattern and much more diagonal attention pattern than BERT or RoBERTa, indicating that it's more parameter efficient.

[1] Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. Voita et al., 2019.
[2] Revealing the dark secrets of BERT. Kovaleva et al., 2019.

Discussion and Conclusion

- TLM opens the possibility of reducing the heavy reliance on large-scale PLMs and training a model from scratch in an efficient manner, while not hurting the overall performance.
- We hope TLM will contribute to democratizing NLP and expediting its development by allowing most researchers to freely explore the architectures, loss functions, algorithms, and other design choices in the neighborhood of a state-of-the-art solution.
- Limitations:
  - TLM enjoys high effciency at the expense of low generality,
  - which could be solved by multi-task learning.

Thanks for listening!