Joint Fine-tuning and Conversion of Pretrained Speech and Language Models towards Linear Complexity

Mutian He[1,2], Philip N. Garner[1]
[1] Idiap Research Institute [2] EPFL
On ICLR 2025

Idiap Research Institute · EPFL · Swiss National Science Foundation

arXiv:2410.06846
Code available

Background

Transformers are so expensive!

- $O(L^2)$ time complexity
- $O(L)$ KV cache

…especially when handing speech

- few words ≈ 1sec = 16K samples = 50 frames

An ever-growing arsenal of transformer alternatives

- Low-rank attention: Linformer
- Restricted attention: Longformer, Big Bird, MoBA, Native Sparse Attention…
- RNNs (linear attention): RetNet, Mamba (2), DeltaNet …
- …still increasing!

Motivation

How to make use of these new archs?

- Pretrained parameters often unavailable, esp. on speech
- New models emerge rapidly

…redo the whole pretraining for each new arch?

- Computational costs & access to pretraining data

Convert/fine-tune pretrained transformers into the target arch on the target downstream task

- Use only the downstream target task data, avoid re-pretraining
- Without performance degradation

Methods: Cross-Arch Layerwise Distillation

Knowledge transfer from original transformer

- Unguided (a): Parameter transfer
  - Replace attention layers with, e.g., Mamba layers, then fine-tuning
  - Other parameters (e.g., MLPs) are reused
- Guided: Behavior transfer
  - Reproduce the original behavior (hidden states) by layerwise distillation

$$\mathcal{L}_{\mathrm{CE}}(\boldsymbol{y}^{(s)}, \boldsymbol{y}) = -\sum_i \boldsymbol{y}_i \log(\boldsymbol{y}_i^{(s)})$$

Cross-entropy loss for the target classification task

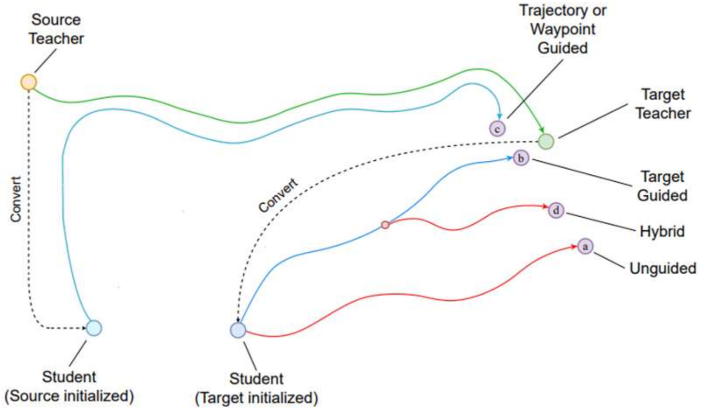
$$\mathcal{L}_{\mathrm{KD}}(\boldsymbol{y}^{(s)}, \boldsymbol{y}^{(t)}) = \sum_i \left(\frac{\boldsymbol{y}_i^{(t)}}{\beta}\right) \log\left(\frac{\boldsymbol{y}_i^{(t)}/\beta}{\boldsymbol{y}_i^{(s)}/\beta}\right)$$

KL-div between student (ours) and teacher (transformer) outputs

$$\mathcal{L}_{\mathrm{LD}}(\boldsymbol{H}^{(s)}, \boldsymbol{H}^{(t)}) = \frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{H}_i^{(s)} - \boldsymbol{H}_i^{(t)})^2$$

L2 loss between hidden states in each layer of the student and the teacher

$$\mathcal{L} = \alpha_{\mathrm{CE}}\mathcal{L}_{\mathrm{CE}} + \alpha_{\mathrm{KD}}\mathcal{L}_{\mathrm{KD}} + \alpha_{\mathrm{LD}}\mathcal{L}_{\mathrm{LD}}$$



What should be the teacher?

- Target-guided (b)
  - Directly distill from the fine-tuned transformer (*target teacher*)
- Trajectory/Waypoint guided (c)
  - Original pretrained transformer (*source teacher*) carries important knowledge that leads to downstream capabilities
    - Essential to fine-tuning, but will be lost in the end
    - Preserving the knowledge helps, as found by e.g. L2-SP
  - Can we reproduce the trajectory of transformer fine-tuning?
    - Simultaneously fine-tune transformer & target model
    - Distill from hidden states at each fine-tuning step
    - Approximation: distill from several checkpoints (*waypoints*) during transformer fine-tuning

When should we distill?

- Distillation loss terms are like splints, stabilizes optimization in the early stage but restrains it later
- Hybrid (d): stop distillation later and set the target loss free

Configuration

- RoBERTa → Linformer for NLP: QNLI, QQP, SST2, IMDB
- Wav2Vec2 → Bidirectional Mamba2 for speech tasks: TEDLIUM (ASR), SLURP (IC), VoxCeleb1 (Speaker ID)
- Pythia-1B → Mamba, on zero-shot LM tasks *

* An ancillary experiment, since there isn't a separate *target task* here; hence trajectory/waypoint guided conversion doesn't apply

Results

- Guided conversion matches standard transformer results
- Trajectory/Waypoint guided mode helps in the NLP case
  - Waypoint approximation works well
  - Hybrid mode performs worse

	QNLI	QQP	SST2	IMDB	Average
Pretrained Linformer	91.2%	90.8%	93.1%	94.1%	92.3%
Std. RoBERTa	92.4%	91.8%	95.3%	95.7%	93.8% +1.3
Unguided	69.4%	84.3%	83.6%	82.6%	80.0% -12.3
CALD					
- Target Guided	89.0%	91.8%	93.3%	92.3%	91.6% -0.7
- Trajectory Guided	91.2%	91.9%	94.0%	93.1%	92.5% +0.2
- Waypoint Guided	89.9%	91.9%	93.7%	92.8%	92.1% -0.2
- Hybrid	86.8%	90.8%	91.4%	90.5%	89.9% -2.4

Performance retained

- Works also on speech; hybrid mode works better

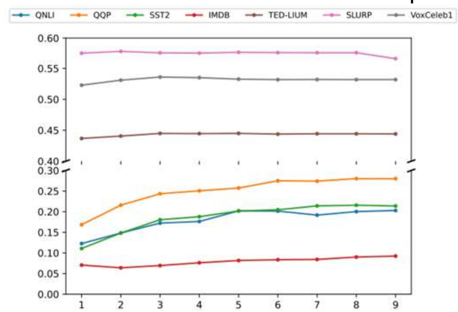
	ASR WER ↓	IC Acc. ↑	SID Acc. ↑
Std. Wav2Vec2	6.24	91.70	96.09
Unguided	11.29	79.68	84.24
CALD			
- Target Guided	6.56	90.43	**96.56**
- Waypoint Guided	6.92	90.32	96.16
- Hybrid	**6.41**	**91.23**	96.41

- Similar results on LM, converted using 0.5%/1%/2% Pile

Model	Lambada	PIQA	Winog.	WSC	ArcE	ArcC	SciQ	LogiQA	Avg.
Pythia-1B	0.562	0.707	0.535	0.670	0.570	0.244	0.839	0.221	0.544
0.5% Pile / 1.5B tokens									
Unguided	0.394	0.671	0.493	0.542	0.502	0.211	0.778	0.246	0.479
Tgt. Gd.	0.410	0.686	0.504	0.608	0.538	0.210	0.775	0.230	0.495
Hybrid	0.432	0.683	0.507	0.608	0.537	0.209	0.788	0.238	0.500
1% Pile / 3.0B tokens									
Unguided	0.453	0.673	0.514	0.571	0.525	0.217	0.794	0.217	0.495
Tgt. Gd.	0.449	0.689	0.518	0.626	0.535	0.220	0.798	0.218	0.507
Hybrid	0.479	0.693	0.520	0.648	0.531	0.224	0.808	0.247	0.519

Why difference between hybrid & trajectory modes?

- Hidden states shift gradually, epoch-by-epoch in NLP
- Significant shifts have occurred in a few steps in speech



Takeaway

- Pretrained transformers can be converted to linear-complexity (Linformer, Mamba) downstream models
  - Guided by layerwise distillation only on the target task data
- Alternative distillation modes help per task
  - Trajectory guided models, using knowledge from pre-fine-tuned pretrained transformer that will be lost later in FT
  - Time-hybrid of guided and unguided fine-tuning