DistilBERT and ALBERT

Abstract

This paper reviews two popular BERT-derived models designed for efficiency: DistilBERT and ALBERT. We describe their motivations, architectural changes compared with BERT, training objectives, empirical performance on standard benchmarks, typical use cases, and trade-offs. Practical notes for fine-tuning, deployment, and suggested experiments are provided so that readers can both understand the methods and reproduce comparative results.

1 Introduction

Large pretrained Transformer language models such as BERT have delivered substantial improvements across many NLP tasks. However, their large size and inference cost limit deployment in resource-constrained environments. DistilBERT and ALBERT are two approaches that reduce size or memory usage while attempting to preserve performance. DistilBERT applies knowledge distillation to create a smaller student model; ALBERT reduces parameter redundancy through factorized embeddings and cross-layer parameter sharing.

2 Background: BERT in brief

BERT (Bidirectional Encoder Representations from Transformers) is a stack of Transformer encoder layers pretrained with masked language modeling and next-sentence prediction objectives. The common BERT_base configuration uses 12 layers, hidden size 768, 12 attention heads and about 110M parameters. Many efficient variants build on BERT by compressing the model, changing parameterization, or modifying training objectives.

3 DistilBERT

3.1 Motivation and approach

DistilBERT is a compressed version of BERT obtained using knowledge distillation during the pretraining phase. Instead of starting from a randomly initialized small model, DistilBERT uses the larger BERT as a teacher and trains a smaller student network to match the teacher's behavior on pretraining tasks.

3.2 Architecture and training

Key points:

- The student typically has fewer layers than the teacher (e.g., initialize by taking every other layer from the teacher).
- The pretraining objective is a triple-loss combining: masked language modeling loss, distillation loss (soft-target cross-entropy against the teacher logits), and a cosine embedding loss that aligns intermediate hidden representations.
- This yields a model that is substantially smaller, faster at inference, and cheaper to pretrain compared to the original BERT.

3.3 Empirical results and trade-offs

DistilBERT commonly reduces parameter count (roughly 40% smaller than BERT_base in the original work) and reports retaining the majority of BERT's language understanding capabilities while improving inference speed. The trade-off is a modest drop in task accuracy for gains in latency and memory usage.

4 ALBERT

4.1 Motivation and approach

ALBERT (A Lite BERT) addresses parameter inefficiency in BERT by introducing two main techniques: factorized embedding parameterization and cross-layer parameter sharing. Additionally, ALBERT introduces an inter-sentence coherence loss (sentence-order prediction or SOP) during pretraining to better model relationships between sentences.

4.2 Architecture and training

- **Factorized embeddings:** The large token embedding matrix is factorized into a smaller embedding dimension followed by a projection to the model hidden size. This significantly reduces parameters in the embedding layer.
- **Cross-layer parameter sharing:** Instead of learning separate parameters for every Transformer layer, ALBERT shares parameters across layers (e.g., all attention parameters shared), allowing very deep models without linear increases in parameters.
- **Inter-sentence loss:** Replaces BERT's next-sentence prediction with a sentence-order prediction (SOP) task that focuses on fine-grained coherence between adjacent sentences.

4.3 Empirical results and trade-offs

ALBERT demonstrates that with parameter-reduction techniques it is possible to scale depth while keeping the total parameter count low, and that such models achieve strong results on benchmarks like GLUE, RACE and SQuAD. The trade-offs include greater engineering complexity and potential sensitivity to optimization and hyperparameters when scaling.

5 Comparative analysis

5.1 Compression strategy

- DistilBERT compresses by distillation (student-teacher), reducing layers and training to mimic a teacher.
- ALBERT compresses by changing parameterization (factorized embeddings, cross-layer sharing) to reduce redundant parameters while allowing depth to increase.

5.2 When to choose which

- **Choose DistilBERT** when you need a straightforward, smaller, and faster model that closely follows BERT behavior and when you want minimal changes to fine-tuning recipes.
- **Choose ALBERT** when you want to explore larger depth with fewer total parameters, especially if you aim for highest possible accuracy under a parameter budget and can invest in careful tuning.

6 Practical notes for fine-tuning and deployment

- Both models can be fine-tuned using the same task-specific heads as BERT (classification, extraction, sequence labeling).
- Use learning-rate warmup, weight decay, and small batch sizes if memory constrained; ALBERT may require more careful hyperparameter tuning.
- For on-device or low-latency inference, DistilBERT often gives the simplest gains; additional quantization (8-bit) and pruning can further reduce footprint for both models.

7 Suggested experiments (reproducible)

To compare the models in a short study:

1. Select benchmark tasks: e.g., SST-2 (classification), SQuAD v1 (QA), and a small named-entity dataset.

2. Use publicly available pretrained checkpoints (Hugging Face) and run identical fine-tuning recipes (optimizer, lr schedule, epochs) for each model.

3. Measure: validation accuracy/F1, inference latency (on CPU), peak memory, and model size on disk.

4. Report results with a brief discussion of the trade-offs.

8 Conclusion

DistilBERT and ALBERT represent two effective, complementary strategies to make Transformer-based language models more efficient. Distillation emphasizes simplicity and speed, while ALBERT emphasizes parameter-efficient scaling. Choice depends on application constraints (latency, memory, accuracy) and engineering resources.

Acknowledgments

This document follows publicly available research by the original authors of DistilBERT and ALBERT and implementation resources such as Hugging Face and Google Research.

References

[1] V. Sanh, L. Debut, J. Chaumond, T. Wolf. ”DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.” arXiv preprint arXiv:1910.01108, 2019.

[2] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, R. Soricut. ”ALBERT: A Lite BERT for Self-supervised Learning of Language Representations.” arXiv preprint arXiv:1909.11942, 2019.

[3] Hugging Face Transformers documentation: DistilBERT and model hub.