A Comparative Analysis of DistilBERT and ALBERT

1. Introduction

The field of Natural Language Processing (NLP) has witnessed a big shift in recent years, moving away from sequential processing models like Recurrent Neural Networks (RNNs) and Long Short-Term Memory networks (LSTMs) toward parallelized architectures. While LSTMs were effective at capturing long-term dependencies, they suffered from slow training times and computational bottlenecks due to their sequential nature. The introduction of the Transformer architecture by Vaswani et al. in 2017 revolutionized the field by utilizing "Self-Attention" mechanisms, allowing models to process entire sequences simultaneously rather than word-by-word [1].

However, the original Transformer was designed primarily for sequence-to-sequence tasks like translation. For tasks requiring deep language understanding—such as classification, sentiment analysis, and question answering—researchers discovered that utilizing only the **Encoder** stack of the Transformer yielded superior results. This led to the creation of **BERT** (Bidirectional Encoder Representations from Transformers [2].

As BERT's size and computational requirements grew, the need for efficient, lightweight alternatives became critical for real-world deployment. This paper explores the architectural foundations of BERT and investigates two of its optimized variants: **DistilBERT** and **ALBERT**. We aim to analyze how these models reduce parameter size and inference latency while maintaining comparable accuracy to the original BERT.

---

2. The Foundation: Encoders and BERT

To understand the optimizations introduced in DistilBERT and ALBERT, one must first understand the architecture they are derived from: the Transformer Encoder and the BERT training methodology.

2.1 The Transformer Encoder

The Transformer architecture consists of two parts: an Encoder (which processes input text) and a Decoder (which generates output text). BERT utilizes only the **Encoder** stack.

The core innovation of the Encoder is the **Self-Attention Mechanism**. the Self-Attention mechanism allows every word in a sentence to "look at" every other word simultaneously. This

enables the model to understand context dynamically [1].

2.2 BERT: Bidirectional Encoder Representations from Transformers

BERT, introduced by Google in 2018, is essentially a stack of Transformer Encoders (12 layers for BERT-Base, 24 for BERT-Large). Its defining characteristic is **Bidirectionality**. Standard language models at the time read text either left-to-right (like GPT) or right-to-left. BERT reads the entire sequence at once, allowing it to learn the context of a word based on both its previous and next neighbors simultaneously [2].

To achieve this, BERT uses two novel pre-training tasks:

1. **Masked Language Modeling (MLM):**
   Instead of predicting the next word in a sequence, BERT randomly hides 15% of the words in the input (replacing them with a [MASK] token). The model must predict the hidden words based on the context of the visible words. This forces the model to develop a deep, bidirectional understanding of language structure.
   - *Example Input:* "The [MASK] sat on the mat."
   - *Target:* "cat"
2. **Next Sentence Prediction (NSP):**
   To understand the relationship between sentences (crucial for Question Answering), BERT is trained on pairs of sentences and must predict if Sentence B logically follows Sentence A.

By pre-training on a massive corpus (Wikipedia and BookCorpus) using these tasks, BERT learns rich linguistic patterns. However, this performance comes at a cost: BERT-Base contains **110 million parameters**, making it computationally expensive to train and slow to deploy.

---

3. DistilBERT: Efficiency through Knowledge Distillation

While BERT set new standards for accuracy, its computational cost makes it impractical for resource-constrained environments such as mobile devices or real-time applications. To address this, Sanh et al. (2019) introduced **DistilBERT**, a smaller, faster, and lighter version of BERT based on the principle of *Knowledge Distillation* [3].

3.1 The Concept of Knowledge Distillation

Knowledge Distillation is a compression technique in which a compact model is trained to reproduce the behavior of a larger, pre-trained model , rather than training from scratch on

the raw data alone.

In the case of DistilBERT:

- **The Teacher:** A fully pre-trained BERT-Base model (12 layers, 110M parameters).
- **The Student:** DistilBERT (6 layers, 66M parameters).

The key insight is that the Teacher model contains "dark knowledge" in its output probabilities. For an input like *"I saw a bark in the woods,"* a standard model might just predict the label **"Tree"** (Probability 1.0). However, a trained BERT model might output: **"Tree" (0.9)**, **"Dog" (0.09)**, and **"Car" (0.01)**.

The small probability assigned to "Dog" tells the Student that the word "bark" has a secondary linguistic connection to animals, even if "Tree" is the correct answer here. DistilBERT learns these rich, soft probability distributions, allowing it to generalize better than if it were trained on hard labels alone.

3.2 Architectural Optimizations

DistilBERT is not just a shrunken BERT; it features specific architectural decisions to maximize speed:

1. **Layer Reduction:** DistilBERT reduces the number of Transformer layers from 12 to 6. The authors found that variations in the Hidden Dimension size had a smaller impact on computation efficiency than the number of layers. Therefore, focusing on depth reduction gav the highest speed gains.
2. **Removal of Token-Type Embeddings:** Since the "Next Sentence Prediction" (NSP) task was removed from DistilBERT's training, the segment embeddings used to distinguish Sentence A from Sentence B were discarded.
3. **Triple Loss Function:** To effectively learn from the Teacher, DistilBERT uses a linear combination of three loss functions:
    - **Distillation Loss:** Forces the Student's soft probabilities to match the Teacher's.
    - **Masked Language Modeling (MLM) Loss:** Standard BERT training loss.
    - **Cosine Embedding Loss:** Aligns the hidden state vectors of the Student and Teacher, ensuring their internal representations of text are directionally similar.

3.3 Performance and Trade-offs

The results of this distillation process are significant for production environments. According to the GLUE benchmark (General Language Understanding Evaluation), DistilBERT achieves:

- **97%** of BERT’s performance accuracy.
- **40%** reduction in parameter size (dropping from 110M to roughly 66M).
- **60%** faster inference speed.

This trade-off implies that for the vast majority of commercial applications (e.g., chatbots, search ranking), the drop in accuracy is worth the massive gain in speed and reduced hosting costs.

---

4. ALBERT: A Lite BERT for Self-Supervised Learning

While DistilBERT focused on reducing the number of layers to increase speed, **ALBERT** (A Lite BERT), introduced by Lan et al. (2019), took a different approach. The authors observed that increasing the model size (hidden dimensions) of BERT improved performance but caused a parameter explosion, leading to GPU memory limitations. ALBERT was designed to drastically reduce the parameter count without shrinking the model's depth [4].

4.1 Factorized Embedding Parameterization

In the original BERT architecture, the size of the Word Embeddings (E) is tied directly to the size of the Hidden Layers (H).

- **BERT-Base:** E = H = 768.
- **The Problem:** The vocabulary size (V) is huge (30,000 words). This means the embedding matrix (V x E) contains millions of parameters that are rarely updated during training.

ALBERT decouples these two dimensions. It uses a small embedding size (e.g., E=128) to represent words and then projects them into the larger hidden space (H=768) using a small linear layer.

- *Mathematics:* Instead of one massive matrix (V x E), ALBERT decomposes it into two smaller matrices: (V x E) and (E x H).
- *Result:* This reduces the embedding parameters from **23 million** (in BERT) to just **3 million** (in ALBERT), freeing up significant memory.

4.2 Cross-Layer Parameter Sharing

This is ALBERT's most radical innovation. In BERT, Layer 1 has its own weights, Layer 2 has its own weights, and so on up to Layer 12.

ALBERT shares the parameters across all layers.

It uses the exact same Self-Attention and Feed-Forward Network weights for every single layer in the stack. Effectively, the data is looped through the same function 12 times.

- *Impact:* This prevents the model from growing in size as depth increases. It reduces the total parameter count of an ALBERT-Base equivalent to just **12 million** (compared to BERT's 110 million), an **89% reduction**.

4.3 Sentence Order Prediction (SOP)

The original BERT used "Next Sentence Prediction" (NSP) to learn relationships between sentences.

ALBERT replaces NSP with **Sentence Order Prediction (SOP)**.

> The model is given two consecutive sentences from a document, but sometimes their order is swapped. The model must determine if the order is correct. This forces the model to understand logical flow and coherence, not just topic similarity, leading to better performance on complex reading comprehension tasks.

4.4 The "Lite" Misconception

It is crucial to note a distinction between ALBERT and DistilBERT.

- **DistilBERT** is *faster* (lower latency) because it has fewer layers to compute.
- **ALBERT** is *smaller* (lower memory footprint) but **not faster**. Because it still has 12 layers (even though they share weights), the data must still pass through 12 mathematical operations. Therefore, ALBERT saves disk space and RAM, but inference time remains similar to the original BERT.

References

[1] Vaswani, A., et al. (2017). "Attention Is All You Need."

- *The original paper introducing the Transformer and Encoder architecture.*
- Link: https://arxiv.org/pdf/1706.03762

[2] Devlin, J., et al. (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding."

- *The original paper introducing BERT, MLM, and NSP.*
- Link: https://arxiv.org/pdf/1810.04805

[3] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter."

- *The original paper detailing the triple-loss function and distillation process.*
- Link: https://arxiv.org/pdf/1910.01108

[4] Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations."

- *The original introducing parameter sharing and factorized embeddings.*
- Link: https://arxiv.org/pdf/1909.11942