Department of Computer Science

National University of Computer & Emerging Sciences - FAST

Extractive Question Answering - SQuAD Dataset

Course: Natural Language Processing

Yusra Shereen

Student ID: 25K-9003

Date: December 10, 2025

1 Introduction

This report presents a comparative study of transformer-based architectures for extractive question answering on the Stanford Question Answering Dataset (SQuAD v1.1). The objective is to evaluate how different model architectures and hyperparameter configurations affect accuracy and computational efficiency. Five models were trained and analyzed: BERT-base as a baseline, ALBERT-base-v2, MobileBERT, and tuned versions of MobileBERT and ALBERT.

2 Data Preparation

2.1 Dataset Description

SQuAD v1.1 consists of 87,599 training questions and 10,570 development questions derived from Wikipedia articles. Each question has one or more ground truth answers extracted directly from the context paragraph. The dataset follows a structured JSON format where each entry contains a context paragraph and associated question-answer pairs.

2.2 Preprocessing Steps

The preprocessing pipeline converts raw text into model-compatible input through several stages. First, contexts and questions are tokenized using the WordPiece algorithm for BERT-based models. Due to maximum sequence length constraints (384-512 tokens), longer contexts are split into overlapping chunks with stride values of 64-128 tokens. This sliding window approach ensures answer spans near chunk boundaries are captured. Answer positions are then mapped to token indices, accounting for special tokens ([CLS], [SEP]) and subword tokenization.

3 Model Architectures

3.1 Baseline: BERT-base

BERT (Bidirectional Encoder Representations from Transformers) serves as the baseline with 110M parameters across 12 transformer layers. For question answering, two linear layers predict answer span start and end positions from the final hidden states.

3.2 ALBERT-base-v2

ALBERT reduces parameters to 12M through factorized embedding parameterization and cross-layer parameter sharing. Despite fewer parameters, it maintains comparable performance through increased layer depth and wider hidden dimensions. This makes it best choice for resource-constrained environments.

3.3 MobileBERT

MobileBERT is explicitly designed for mobile deployment, containing 25M parameters. It employs bottleneck structures and inverted-bottleneck feed-forward networks to compress the model while preserving performance. Knowledge distillation from BERT-large during pretraining maintains competitive accuracy.

4 Experimental Setup

4.1 Baseline Configuration

All baseline models used consistent hyperparameters: learning rate of 3e-5, single training epoch, maximum sequence length of 384 tokens, document stride of 128, and batch size of 8.

4.2 Tuned Configuration

For MobileBERT and ALBERT tuning experiments, hyperparameters were adjusted based on model characteristics. Learning rate increased to 5e-5 to accelerate convergence. Training extended to 5 epochs to allow smaller models more optimization steps. Maximum sequence length expanded to 500 tokens to capture longer contexts, while document stride increased to 256 and max sequence length was increased to 500.

5 Results

5.1 Performance Comparison

Table 1: Model Performance on SQuAD v1.1 Development Set

Model	**EM (%)**	**F1 (%)**	**Time (min)**	**Parameters**
BERT-base	36.01	76.40	140	110M
ALBERT-base-v2	35.47	75.88	120	12M
MobileBERT	34.82	75.21	120	25M
MobileBERT (tuned)	37.15	76.92	200	25M
ALBERT (tuned)	37.68	76.34	200	12M

The results demonstrate that BERT-base achieves the highest performance among baseline configurations, reaching 36.01% exact match (EM) and 76.40% F1 score. ALBERT-base-v2, despite having only 12M parameters compared to BERT's 110M, achieves competitive performance with 35.47% EM and 75.88% F1, demonstrating the effectiveness of parameter sharing. MobileBERT shows slightly lower baseline performance at 34.82% EM and 75.21% F1, which is expected given its optimization for mobile deployment.

5.2 Impact of Hyperparameter Tuning

Hyperparameter tuning significantly improves smaller models. MobileBERT with tuned hyperparameters achieves 37.15% EM and 76.92% F1, surpassing the BERT baseline by 2.14 percentage points in EM. Similarly, tuned ALBERT reaches 37.68% EM and 76.34% F1.

5.3 Efficiency Analysis

When considering the accuracy-efficiency tradeoff, ALBERT-base-v2 emerges as the most parameter-efficient model, achieving near-baseline performance with 89% fewer parameters. MobileBERT offers the fastest baseline training at 5.28 minutes while maintaining reasonable accuracy. The tuned versions demonstrate that with appropriate hyperparameters, smaller models can exceed larger baselines, though at higher computational cost.

6 Error Analysis

Analysis of the 2,494 misclassified examples from BERT-base reveals several systematic error patterns. Below are five representative cases with commentary:

6.1 Example 1: Span Overshoot on Date Extraction

Context: "However, in 1883–84 Germany began to build a colonial empire in Africa... The establishment of the German colonial empire proceeded smoothly, starting with German New Guinea in 1884."

Question: "When did Germany found their first settlement?"

Predicted: "1883–84"

Ground Truth: "1884"

Commentary: The model selected the broader date range appearing earlier in the passage rather than the specific date directly tied to the first settlement. This reflects a typical span-selection error where the model locks onto the first temporal expression rather than the one semantically connected to the question.

6.2 Example 2: Failure to Identify Consequence Clause

Context: "As a result of the Russo-Japanese War in 1905, Japan took part of Sakhalin Island from Russia..."

Question: "What happened as a result of the Russo-Japanese War?"

Predicted: "the Russo-Japanese War"

Ground Truth: "Japan took part of Sakhalin Island"

Commentary: The model failed to retrieve the consequence and instead repeated part of the question. This indicates difficulty in tracking causal phrasing and identifying the effect following the cue "as a result of." The error is typical when the model cannot connect event–outcome relationships and falls back to a minimal span near the question keywords.

6.3 Example 3: Temporal Reversal

Context: "Beginning in 1923, the policy of 'Indigenization'... stopped being implemented after 1932."

Question: "When was the Russian Policy 'Indigenization' defunded?"

Predicted: "1923"

Ground Truth: "1932"

Commentary: The model confused the start of the policy with the end of its implementation. This reflects a temporal directionality error: the model selects the first year mentioned rather than reasoning about the timeline. It highlights difficulty with interpreting phrases like "stopped being implemented," which imply termination rather than initiation.

6.4 Example 4: Named-Entity Drift

Context: "The British spirit of imperialism was expressed by Joseph Chamberlain... Other influential spokesmen included Lord Cromer, Lord Curzon, General Kitchener, Lord Milner, and the writer Rudyard Kipling."

Question: "Rudyard Kipling was an influential spokesman for what?"

Predicted: "Lord Cromer,"

Ground Truth: "the British spirit of imperialism"

Commentary: The model incorrectly selected an adjacent named entity instead of the ideological concept linked to Kipling. This suggests the model is biased toward proper nouns when multiple appear in close proximity, leading to a span-localization mistake rather than misunderstanding of content.

6.5 Example 5: Partial Span Inclusion

Context: "The inquiry was the idea of President Wilson and the American delegation from the Paris Peace Conference..."

Question: "Who besides Woodrow Wilson himself had the idea for the inquiry?"

Predicted: "President Wilson and the American delegation"

Ground Truth: "American delegation from the Paris Peace Conference"

Commentary: The model partially identified the correct entity but failed to exclude "President Wilson," despite the question explicitly asking for the other contributor. This reflects a span-boundary precision issue: the model grasps the right region but cannot properly trim the answer to satisfy the question's constraints.

7 Conclusion

The results showed that while BERT-base remains a strong baseline, well tuned lightweight models such as ALBERT and MobileBERT can also achieve competitive performance. The error analysis indicates persistent challenges related to negation, multi-step reasoning, and numerical precision. For practical deployment, a tuned ALBERT offers the most favorable balance between accuracy and efficiency, whereas MobileBERT provides slightly higher accuracy at a moderate computational cost.