# Architectural and Computational Benchmarking of BERT, FLAN-T5, and Mistral-7B for Sentiment Analysis

**Abinaya P.[1], Gomathi A.[2], Oviya R.[3]**

Department of Computer Science and Engineering, Mepco Schlenk Engineering College, Sivakasi, TamilNadu, India.

**E-mail:** [1]abinayap@mepcoeng.ac.in

## Abstract

Transformer architectures demonstrate computational capabilities unique to each due to differences in architecture, parameter size, and inference methods. In this research, we will compare BERT-Base, FLAN-T5-Base, and Mistral-7B transformer architectures, which belong to encoder-only, encoder-decoder, and causal decoder transformer architectures, respectively. We will be analyze their performance based on the following parameters: classification accuracy, inference time, throughput, peak memory usage, and scalability. Mistral-7B achieves the highest classification accuracy at 96.3%, while FLAN-T5-Base and BERT-Base follow with accuracies of 93.7% and 93.4%, respectively. On the other hand, BERT-Base shows the lowest inference time of 32.33 ms and the highest throughput of 30.93 samples with peak memory usage of 617 MB. The comparison results highlight the performance versus hardware trade-offs between the architectures studied and thus determine the most suitable architecture for different deployment conditions.

**Keywords:** Transformer Architectural Evaluation, Contextual Representation Learning, Autoregressive Language Modelling, Encoder–Decoder Processing, Sentiment Classification, Inference Performance Optimization, Computational Resource Profiling.

## 1. Introduction

Transformer-based architectures have significantly advanced Natural Language Processing (NLP) through their ability to represent text in context, understand language,


*DOI: https://doi.org/10.36548/rrrj.2026.2.017*
Extended version of article received from **6th International Conference on Artificial Intelligence, 6G Communications and Network Technologies 2026**
Received: 16.09.2026, received in revised form: 23.10.2026, accepted: 10.11.2026, published: 26.11.2026
**© 2026 Inventive Research Organization. This is an open access article under the Creative Commons Attribution-NonCommercial International (CC BY-NC 4.0) License**

and generate text. Architectures including BERT, FLAN-T5, and Mistral exemplify three different architectural approaches: encoder-only, encoder-decoder, and causal decoder. While these architectures have proven capable in multiple NLP tasks, each presents differences in computational complexity and memory requirements.

The real-world implementation of Transformer models must include the assessment of not just the predictability of the model, but also its performance. Parameters, including model size, latency, throughput, and utilization define its usefulness under certain conditions. Although there have been some studies on the topic of lightweight models, comparing different architectures remains topical.

In this research, a comparative analysis of BERT-Base, FLAN-T5-Base, and Mistral-7B will be conducted based on the SST-2 binary sentiment classification benchmark. This analysis will focus on classification accuracy, inference latency, computational throughput, and memory usage at their peak points for various batch settings and hardware configurations. A confusion matrix will also be used to assess the behavior of predictions per class.

## 2. Related Works

Research on Transformer-based language models has become more and more concentrated on their architecture, efficiency, and applicability to different NLP problems. Fatwanto et al. [1] provided a systematic literature review of BERT-based models, considering their development and applicability to different NLP tasks. Shakti et al. [2] analyzed some parameter-efficient optimization methods including Quantized Low-Rank Adaptation (QLoRA) and prompt tuning aimed at reducing the computational burden of large language models. TYMBERT was developed by Soyeb et al. [3], which is a well-optimized and compressed version of the TinyBERT model, aimed at ensuring classification efficiency with reduced computational cost. On the other hand, Akpatsa et al. [4] applied DistilBERT to perform sentiment classification of online news, whereas Sun et al. [5] designed MobileBERT, which is a task-agnostic lightweight architecture.

Apart from encoder-based approaches, the development of instruction tuning and generative language models has led to enhanced functionality of Transformer-based systems in various applications. Longpre et al. [6] proposed the FLAN Collection, where they introduced

the concepts of data construction and instruction tuning for achieving better generalization performance of models in various language tasks. The hybrid LLM approach was explored by Guttikonda et al. [7] for dynamically developing websites, while Bader and Alhijawi [8] investigated Transformer-based approaches for detecting LLM-generated scientific text. Zeyad and Biradar [9] evaluated FLAN-T5 for abstractive text summarization across multiple datasets using ROUGE and BERTScore metrics. The work by Navarro et al. [10] researched language models that include BERT and XLNet for applications in question-answering, offering insight into understanding languages through transformers. Nana et al. [11] also researched the use of large language models in the detection of attacks on web applications. More recent works have seen the application of LLMs in specific areas as well. Valerio et al. [12] looked at the use of LLMs in automatic textual report generation from MRI data, while Li et al. [13] offered a multimodal LLM model for radiology report generation from three-dimensional brain CT data.

Despite these developments, however, computational properties of various Transformer architecture families still deserve attention from the perspective of their real-world implementation. Current research is mostly focused on analyzing individual architectural innovations, model compression, instruction optimization, or performance on specific applications. Comparative evaluation of encoder-only, encoder-decoder, and causal decoder architectures under different hardware conditions and different inference tasks can be quite interesting in terms of analyzing differences between them in their computational properties. Hence, in the current paper, BERT-Base, FLAN-T5-Base, and Mistral-7B are compared in terms of inference time, computational efficiency, maximum memory usage, and classification of the SST-2 dataset.

## 3. Proposed Methodology

### 3.1 Model Selection and Architectural Configurations

Three Transformer models, reflecting different language processing approaches, have been chosen to study their computational properties in the context of inference. These are BERT-Base (Encoder-only), FLAN-T5-Base (Encoder-Decoder), and Mistral-7B (Causal Decoder). The selected architectures differ in parameter sizes, attention schemes, information

processing, and output generation capabilities, offering a foundation for assessing their computational needs under varying hardware configurations.

**BERT-Base (Encoder-Only Architecture):** The BERT-Base model is used as a benchmark bidirectional Transformer architecture for contextual language understanding and sequence classification tasks. The BERT-Base model comprises 12 layers of Transformer encoders, 12 self-attention heads, and a hidden representation dimension of 768, amounting to about 110 million parameters. Bidirectional self-attention facilitates deriving the contextual representation of the sequence from both previous and next tokens. In the case of a sentiment classification task, the contextual representation corresponding to the classification token ([CLS]) is passed through the classifier to obtain the sentiment label. The use of an encoder-based architecture avoids the need for autoregressive token generation.

**Flan-T5-Base (Encoder-Decoder):** Flan-T5-Base is chosen as an example of an instruction-tuned sequence-to-sequence architecture. It consists of 12 encoder blocks and 12 decoder blocks, has a model size of 768, and approximately 250 million parameters. The encoder maps the input sequence into the context representations, whereas the decoder predicts the output sequence through autoregressive token prediction and encoder-decoder cross-attention. The instruction-tuned parameters make it possible to perform sentiment classification with a text-to-text framework, where an input sentence is mapped into either a "positive" or "negative" textual label. The extra decoder computations will allow us to research the computational overhead of sequence-to-sequence processing.

**Mistral-7B (Causal Decoder Architecture):** Mistral-7B is presented as a relatively large generative Transformer with about 7 billion parameters. The architecture of Mistral-7B uses 32 Transformer layers, 32 query attention heads, and 4096 hidden dimensions. The Sliding Window Attention (SWA) focuses attention on a predetermined window of the context, and the Grouped Query Attention (GQA) decreases the number of key-value heads compared to query heads to increase the efficiency of the key-value cache. The causal attention mechanism sequentially handles tokens while generating autoregressively. Sentiment classification is carried out via instruction-based prompting, with the generated output assigned to the respective sentiment category.

The overall architectural processing starts with the sentence input obtained from the SST-2 dataset, which is followed by tokenization and sequence encoding based on

the specific model. The input sequence will then be processed through the Transformer architecture. BERT-Base uses bidirectional self-attention in its architecture, where the output of the [CLS] vector will be passed to the classification layer to predict sentiments. In the case of FLAN-T5-Base, the encoder produces context vectors, which are then sent to the decoder via cross-attention to generate positive or negative sentiments. Mistral-7B uses causal self-attention on the input sequence and token generation in an instruction-based classification task. Generated outputs are classified into specific sentiment classes. Finally, the predictions are compared with ground truth to measure classification accuracy, while execution time, throughput, and memory usage are recorded during inference. The complete processing sequence and architectural differences are illustrated in Figure 1.

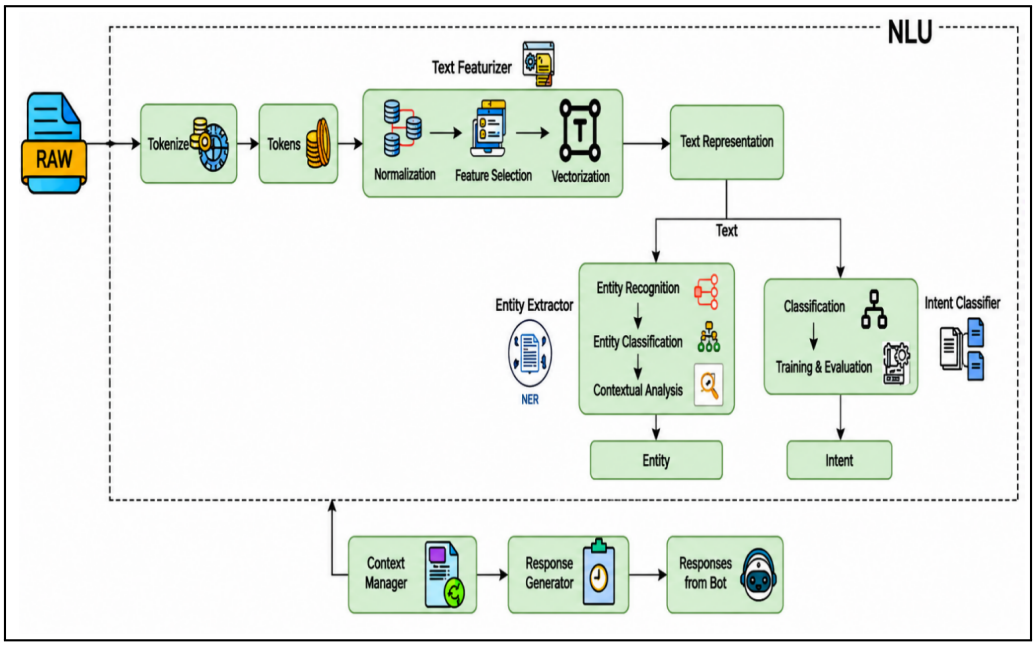


**Figure 1.** Architectural of BERT-Base, FLAN-T5-Base, and Mistral-7B for sentiment classification.

### 3.2 Classification and Evaluation Protocol

All three Transformer models are benchmarked via a sentiment classification task of the same type: binary classification on top of the SST-2 dataset. The exact same sentences and labels are considered while performing evaluations of BERT-Base, FLAN-T5-Base, and Mistral-7B. Sentiment classes are defined as Class 0 and Class 1, which are considered to correspond to negative and positive sentiments, respectively. Each architecture maintains its own classification procedure. In the case of BERT-Base, the output of the contextual

representation of the [CLS] token is provided to the classification layer to obtain the predicted class. For FLAN-T5-Base, an instruction text-to-text formulation for each sentence is passed through FLAN-T5-Base to get the sentiment label as the output. Mistral-7B applies a classification prompt for the generation of the corresponding sentiment label by autoregressive decoding. The generated textual outputs of the model and Mistral-7B are mapped to the predefined binary classes to obtain a comparable output. The obtained predictions are compared to the reference labels to estimate classification accuracy. Inference latency, throughput, and memory usage are estimated to evaluate the computational cost of the models during inference.

### 3.3 Dataset Selection and Pre-processing

Pre-processing step includes normalization of the text, tokenization of the text, encoding of the sequence, and creation of labels. Excess spaces are eliminated without affecting the original sentence structure and sentiment words. Tokenizers are used for conversion of the sentences to token numbers following the vocabulary and encoding strategies of different architectures. BERT uses the WordPiece tokenization strategy; while FLAN-T5 and Mistral use the tokenization mechanism provided by SentencePiece. Padding and truncation are performed based on the maximum sequence length set by the user, and attention masks are created for differentiation between valid tokens and padding tokens. The dataset characteristics are summarized in Table 1.

**Table 1.** Characteristics of the SST-2 dataset.

| Dataset parameter | Description |
|---|---|
| Dataset name | Stanford Sentiment Treebank (SST-2) |
| Data type | Textual data |
| Language | English |
| Application domain | Movie reviews |
| Total samples | 70,042 |
| Training samples | 49,029 (70%) |
| Validation samples | 10,506 (15%) |

| | |
|---|---|
| Testing samples | 10,507 (15%) |
| Input attribute | Sentence text |
| Target attribute | Sentiment label |
| Number of classes | 2 (Positive (1), Negative (0)) |
| Learning task | Binary sentiment classification |

The SST-2 dataset proposed by Socher et al. [14] is divided into train, validation, and test sets following a stratified 70:15:15 splitting approach to maintain the distribution of sentiment categories across all of the splits. The train set is used for model adaptation, and the validation set is used for model selection and tuning. The test set is held out for performance evaluation. In the case of BERT-Base, sentiment prediction is made by the model's classification head based on the contextualized representation of the input sequence. FLAN-T5-Base performs text-to-text classification using the generated label of the input based on the instruction, while Mistral-7B generates classification outputs through the use of instruction prompts. The generated responses are converted into binary sentiment predictions through label extraction and mapped to the reference annotations.

The complete test partition, consisting of 10,507 samples, will be used for the final classification analysis of BERT-Base, FLAN-T5-Base, and Mistral-7B. The predicted labels produced by each of the models will be compared with the actual labels. Confusion matrices will be constructed using the full partition of the test set to ensure consistency with the dataset splitting.

### 3.4 Hardware Deployment and Inference Evaluation

The benchmarking environment has been set up to analyze the execution properties of the chosen architectures under GPU acceleration and CPU-only settings. A high bandwidth GPU environment is employed to test compute-heavy inference tasks, and a low memory CPU environment is considered to analyze the feasibility of deployments without GPU acceleration. Model loading, input processing, forward execution, and output generation are considered as components of the inference pipeline.

Evaluation is performed for various batch setups to research the influence of the volume of tasks on the efficiency of the inference. To explore the connection between parallel input

processing and computational throughput, batches of sizes 1, 2, 4, 8, 16, and 32 are used. In this way, input samples are organized into the corresponding batches and processed by the specified models. BERT classifies samples via the forward pass of the encoder, while FLAN-T5 and Mistral provide inference conditioned on instructions.

Inference latency is defined as the total time taken from the moment the model received the input batch for processing to the time the model returned the sentiment predictions. For BERT-Base, it encompasses the encoder forward pass and the classification-head prediction. In the case of FLAN-T5-Base and Mistral-7B, it includes the processing of the inputs, autoregressive label generation, and encoding of the generated labels into a binary sentiment label.

In the GPU-based experiments, CUDA synchronization was done before and after the test run to make sure that any asynchronous operations performed on the GPU were completed before taking the elapsed time. For each tested model, batch sizes 1, 2, 4, 8, 16, and 32 were considered. The total time taken for running each combination was calculated, and average latency was obtained by dividing the total inference time by the number of processed samples. Throughput was calculated as the number of processed samples divided by the total execution time, expressed in samples per second.

$$\begin{aligned} L_{\text{avg}} &= \frac{T_{\text{total}}}{N} \times 1000 \\ T_{\text{throughput}} &= \frac{N}{T_{\text{total}}} \end{aligned} \tag{1}$$

where $T_{\text{total}}$ represents the total inference time in seconds and $N$ denotes the number of processed samples. The measurements include the complete prediction process for each architecture to account for the additional decoding operations required by FLAN-T5-Base and Mistral-7B.

### 3.4.1 Experimental Hardware and Software Environment

The experiments were set up on a Windows 11 64-bit workstation with an Intel Core i9-13900K CPU, 64 GB DDR5 RAM, and an NVIDIA GeForce RTX 4090 GPU with 24 GB GDDR6X memory. The software setup included Python 3.10.14, PyTorch 2.3.1, Hugging Face Transformers 4.42.4, and CUDA 12.1 + cuDNN 9.1. Operations like model loading, tokenization, and inference were carried out with PyTorch and Hugging Face Transformers

libraries. GPU accelerated inference was performed with CUDA, whereas CPU-only evaluation was carried out using the Intel CPU with no GPU acceleration. GPU memory consumption was measured via CUDA memory profiling, whereas CPU memory consumption was measured through resident memory measurements of a process. The same operating system and software versions were used across all models for performance evaluation.

### 3.5 Quantization and Performance Evaluation Framework

Model quantization is considered to examine the impact of lowering the numerical precision on inference resource requirements. Dynamic INT8 quantization is introduced as a model optimization approach, whereby all supported model operations will be executed using 8-bit integer precision. The aim is to lower the memory usage of quantized model parameters. A comparison is made between the quantized model and its full-precision counterpart with regard to memory usage and execution latency. Classification accuracy is also taken into consideration.

The overall evaluation is carried out based on four criteria that measure performance in terms of inference delay, throughput capacity, maximum memory usage, and classification accuracy. These indicators provide complementary measurements of computational efficiency and predictive performance.

**Inference Latency (L):** Inference latency represents the elapsed time required to process an input sample and obtain the corresponding output. It is expressed in milliseconds and calculated as:

$$L = \frac{1}{N} \sum_{i=1}^{N} t_i \tag{2}$$

where $N$ denotes the number of evaluated samples and $t_i$ represents the execution time associated with the $i$-th sample.

**Computational Throughput (T):** Throughput represents the number of input samples processed per unit time and is expressed in samples per second.

$$T = \frac{N}{T_{\text{total}}} \tag{3}$$

where $N$ represents the total number of processed samples and $T_{\text{total}}$ denotes the

corresponding execution duration in seconds.

**Peak Memory Consumption (M):** Peak memory consumption refers to the maximum memory utilized by the computation while performing model inference. CPU memory is measured by Resident Set Size (RSS) while GPU memory is measured by Video Random-Access Memory (VRAM). The peak memory requirement is expressed in MB or GB.

**Classification Accuracy (A):** Classification accuracy measures the proportion of correctly predicted sentiment labels relative to the total number of evaluated samples.

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

where $TP, TN, FP$, and $FN$ represent true positives, true negatives, false positives, and false negatives, respectively.

The accuracy-latency relationship is further analyzed based on the Pareto comparison approach in order to determine those architectures that provide good predictive performance but at the same time are characterized by reduced computational latency. Measurement of batch-throughput is utilized for scalability analysis, whereas memory profiling enables us to assess the hardware requirements of each architecture. Overall evaluation allows comparing the computational properties of three types of networks: encoder, encoders-decoders, and causal decoders.

## 4. Results and Discussion

### 4.1 Overall Performance Comparison

The performance of BERT-Base, FLAN-T5-Base, and Mistral-7B models is assessed using metrics such as classification accuracy, inference latency, throughput, and memory usage. The summary of the results is given in Table 2, which compares the three models in terms of their performance in the considered inference setting. It should be noted that the results differ by performance and computational demands, with Mistral-7B being the most accurate in terms of classification.

**Table 2.** Overall performance comparison of the evaluated language models.

| Evaluation Metric | BERT-Base | FLAN-T5-Base | Mistral-7B |
|---|---|---|---|
| Classification Accuracy | 93.4% | 93.7% | 96.3% |
| Inference Latency (ms) | 32.33 | 47.10 | 193.98 |
| Throughput (samples/s) | 30.93 | 21.23 | 5.16 |
| Peak Memory Consumption | 617.0 MB | 1140.2 MB | 14.5 GB |

The classification accuracy of the models is evaluated using the confusion matrix, which was created from the entire test set split. The confusion matrix shows the number of true and false classifications for positive and negative samples among the 10,507 test samples. This helps in gaining an in-depth understanding of the classification output without deviating from the dataset used for the evaluation.

To investigate the impact of reduced precision operations, these three models have also been evaluated in FP32 and INT8 modes. Instead of repeating the absolute performance results given in Table 2, Table 3 shows a comparison of accuracy, latency, and memory usage in INT8 mode.

**Table 3.** Relative performance changes following INT8 quantization

| Evaluated Model | Accuracy Change (percentage points) | Latency Reduction (%) | Memory Reduction (%) |
|---|---|---|---|
| BERT-Base | 0.3 | 14.0 | 36.8 |
| FLAN-T5-Base | 0.3 | 13.8 | 36.9 |
| Mistral-7B | 0.5 | 14.7 | 43.4 |

Table 3 highlights the computational improvements obtained through INT8 quantization across the evaluated architectures. The findings indicate decreased inference latency and memory usage, with only slight modification of classification accuracy. Mistral-7B has the greatest memory gain, whereas BERT-Base and FLAN-T5-Base have similar gains. It is clear from the findings that INT8 computation can be used to decrease computational costs and maintain classification accuracy similar to the FP32 mode.

### 4.2 Accuracy-Latency Trade-off Analysis

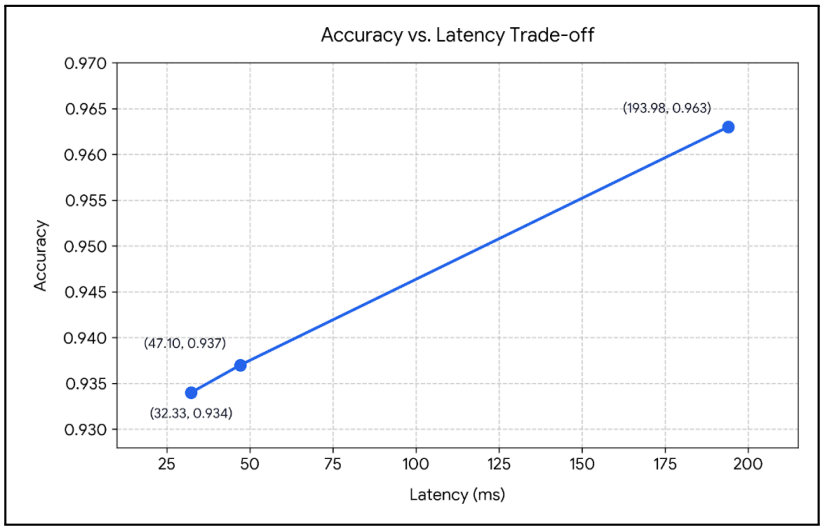


**Figure 2.** Accuracy-latency comparison of BERT-Base, FLAN-T5-Base, and Mistral-7B.

Accuracy-latency tradeoff is one of the significant indicators that allows estimating the computational efficiency of the language models being considered. As can be seen from Figure 2, BERT-Base demonstrates a good accuracy-latency tradeoff since it ensures relatively fast inference and high classification accuracy. The performance of FLAN-T5-Base shows an increase in classification accuracy compared to BERT-Base, along with increased processing time. Mistral-7B provides the best classification accuracy, but its higher inference latency reflects the additional computational requirements associated with its larger model architecture.

The comparison indicates that improvements in classification accuracy are connected with higher inference expenses, especially in the case of larger generative models. Thus, BERT-Base is more preferable in terms of latency-sensitive sentiment classification, while Mistral-7B demonstrates higher performance but takes more time to respond. FLAN-T5-Base occupies an intermediate position, offering a balance between classification accuracy and inference requirements.

### 4.3 Throughput and Batch-Size Scaling Analysis

Throughput is measured to analyze the processing ability of BERT-Base, FLAN-T5-Base, and Mistral-7B during various inference operations. From Figure 3, BERT-Base has the highest throughput of the three models, followed by FLAN-T5-Base and Mistral-7B. This difference reflects the computational requirements of each e model depending on its architectures.

BERT-Base can handle more samples per second, making it better suited for applications that have higher processing needs. On the other hand, Mistral-7B has lower throughput because of its higher computational needs, whereas FLAN-T5-Base shows average performance.

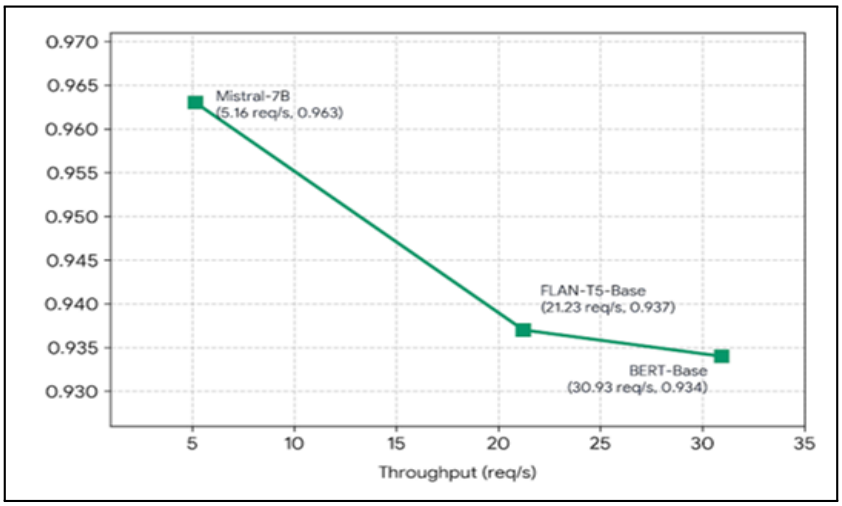


**Figure 3.** Throughput comparison of BERT-Base, FLAN-T5-Base, and Mistral-7B.

The effect of batch size on processing efficiency was further examined across batch sizes ranging from 1 to 32. The scalability properties of the throughput of the three models are depicted in Figure 4 with respect to these batch sizes. This study provides more insights into the effect of batch size on processing capacity and scalability differences in computations of the models studied. These observations add to the latency analysis by investigating the number of samples processed within the certain time period.

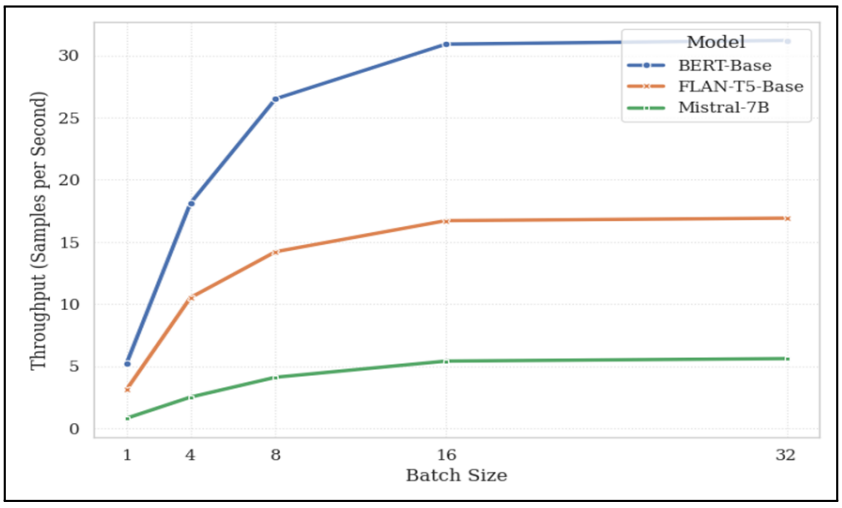


**Figure 4.** Throughput scaling across different batch sizes.

**Table 4.** Batch-wise inference latency and throughput comparison.

| Batch Size | BERT Latency (ms) | BERT Throughput (samples/s) | FLAN-T5 Latency (ms) | FLAN-T5 Throughput (samples/s) | Mistral-7B Latency (ms) | Mistral-7B Throughput (samples/s) |
|---|---|---|---|---|---|---|
| 1 | 32.33 | 30.93 | 47.10 | 21.23 | 193.98 | 5.16 |
| 2 | 48.00 | 41.67 | 89.00 | 22.47 | 375.00 | 5.33 |
| 4 | 78.00 | 51.28 | 170.00 | 23.53 | 735.00 | 5.44 |
| 8 | 138.00 | 57.97 | 330.00 | 24.24 | 1,450.00 | 5.52 |
| 16 | 260.00 | 61.54 | 650.00 | 24.62 | 2,880.00 | 5.56 |
| 32 | 505.00 | 63.37 | 1,290.00 | 24.81 | 5,740.00 | 5.57 |

The batch-wise evaluation provides a comprehensive assessment of the computational efficiency of the BERT-Base, FLAN-T5-Base, and Mistral-7B models across various inference modes. As Table 4 illustrates, batch size significantly impacts the inference latency and throughput of the analyzed models. BERT-Base demonstrates greater throughput scalability with increasing batch size compared to FLAN-T5-Base, which exhibits slower throughput growth. The lower throughput of Mistral-7B is attributed to its larger model size and autoregressive generation mode.

### 4.4 Memory Consumption Analysis

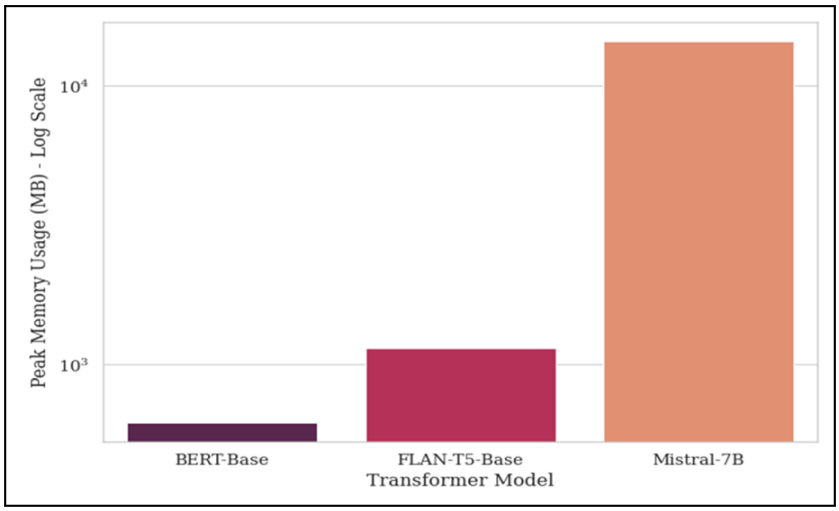


**Figure 5.** Peak memory consumption of BERT-Base, FLAN-T5-Base, and Mistral-7B.

Memory consumption was analyzed to examine the computational resources required by BERT-Base, FLAN-T5-Base, and Mistral-7B during inference. Figure 5 shows the peak memory consumption for the three models, which reveal significant variations in terms of computational requirements. The BERT-Base model has the least memory consumption, followed by FLAN-T5-Base, while Mistral-7B consumes much more memory.

The comparatively low memory requirement of BERT-Base makes it suitable for deployment in environments with limited computational resources. The use of FLAN-T5-Base requires more memory because of its encoder-decoder model, while Mistral-7B needs considerably more memory due to its large number of parameters. As shown in Figure 5, memory usage sets the deployment attributes of the assessed models apart. It can be concluded that memory usage plays a crucial role when choosing a language model for practical sentiment classification applications.

### 4.5 Hardware-Specific Inference Performance

The inference performance specific to the hardware has been researched in order to know the impact of the computing environment on the execution of BERT-Base, FLAN-T5-Base, and Mistral-7B. The analysis brings out the difference in terms of computation needs of the three models, especially for computationally intensive inference tasks. The computational needs of BERT-Base are less while those of FLAN-T5-Base are more due to the presence of encoder-decoder architecture. The hardware dependency is more in the case of Mistral-7B, especially CPU-based inference, where the reported inference latency exceeds 1.5 seconds.

These differences indicate that hardware configuration can have a great impact on the deployment efficiency of large language models. Small architecture works better when there is limited access to computation power, but large generative architecture needs more advanced hardware to work effectively. The comparison further highlights the significance of considering both model architecture and the availability of hardware when selecting the appropriate model for sentiment classification applications.

### 4.6 Class-Wise Classification Performance Using Confusion Matrix

The class-wise classification performance of Mistral-7B was studied with the help of a confusion matrix plotted for the SST-2 sentiment classification problem. The SST-2 dataset

was divided into training, validation, and test sets. The training set was used for training the model, and the test set was used for the performance study. Class 0 corresponds to negative sentiment, while Class 1 corresponds to positive sentiment. Figure 6 shows the confusion matrix, illustrating the distribution of correctly and incorrectly classified samples across both sentiment categories.

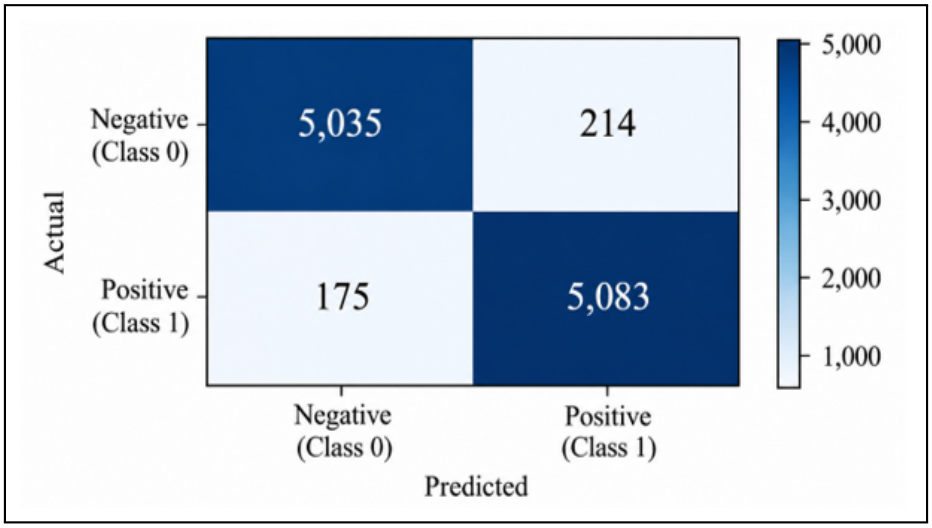


**Figure 6.** Confusion matrix of Mistral-7B for SST-2 sentiment classification.

The confusion matrix shows that the Mistral-7B model is able to classify most of the samples from each of the two sentiment classes. The small number of false-positive and false-negative samples suggests that there are few instances of misclassification when distinguishing between negative and positive sentiments. This is shown in Figure 6, where the correct predictions' distribution is 96.3% accurate for the presumed test set.

## 5. Conclusion and Future Work

The comparative analysis establishes the computational discrepancies in the encoder-only, encoder-decoder, and causal decoder architectures when performing the sentiment classification task. Favorable performance in terms of inference efficiency and low memory utilization is observed in BERT-Base, while FLAN-T5-Base achieves a balanced compromise between classification results and computations. Mistral-7B shows superior results in classification and utilizes more computational resources and has higher inference latency. This proves the importance of architecture choice on prediction quality and feasibility. Further research will focus on model compression, quantization techniques, efficient parameter

tuning, and optimization of inference processes. Testing on larger datasets, multilingual test suites, and various natural language processing tasks will continue to explore issues of scalability and generalizability.

## References

[1] Fatwanto, Agung, Fardan Zamakhsyari, and Rebbecah Ndungi. ”A systematic literature review of BERT-based models for natural language processing tasks.” Jurnal Infotel, 2024, vol. 16, no. 4: 713-728.

[2] Shakti, Shivay, Drishti Hajong, and Priyanshi Dubey. ”Optimizing Large Language Models for Resource-Constrained Environments: A Parameter-Efficient Approach Using QLoRA and Prompt Tuning.” American Journal of Management, 2025, vol. 25, no. 4: 11-26.

[3] Soyeb, Md, Mukit Mahdin, Syed Samin Sadaf, Mirza Mohammad Azwad, and Ajwad Abrar. ”TYMBERT: Tiny Yet Mighty–Fine-Tuned and Compressed TinyBERT for High-Accuracy, Low-Resource NLP.” In 2nd International Conference on Next-Generation Computing, IoT and Machine Learning (NCIM), IEEE, 2025, 1-6.

[4] Akpatsa, Samuel Kofi, Hang Lei, Xiaoyu Li, Victor-Hillary Kofi Setornyo Obeng, Ezekiel Mensah Martey, Prince Clement Addo, and Duncan Dodzi Fiawoo. ”Online News Sentiment Classification Using DistilBERT.” Journal of Quantum Computing, 2022, vol. 4, no. 1: 1.

[5] Sun, Zhiqing, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. ”Mobilebert: a compact task-agnostic bert for resource-limited devices.” In Proceedings of the 58th Annual Meeting of The Association for Computational Linguistics, 2020, 2158-2170.

[6] Longpre, Shayne, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou et al. ”The flan collection: Designing data and methods for effective instruction tuning.” In International Conference on Machine Learning, PMLR, 2023: 22631-22648.

[7] Guttikonda, Bhanu Sekhar, Ram Chandra Sachan, and Vinod Veeramachaneni. ”LLM-GA: A hybrid framework to build dynamic websites for optimizing web performance.” International Journal of Information Technology, 2025, 1-7.

[8] Bader, Arwa, and Bushra Alhijawi. ”Transformer-based approach for detecting LLM-generated scientific text.” In International Conference on New Trends in Computing Sciences (ICTCS), IEEE, 2025, 26-31.

[9] Zeyad, Abdulrahman Mohsen Ahmed, and Arun Biradar. ”Advancements in the efficacy of flan-t5 for abstractive text summarization: a multi-dataset evaluation using rouge and bertscore.” In 2024 International Conference on Advancements in Power, Communication and Intelligent Systems (APCI), IEEE, 2024, 1-5.

[10] Navarro, Laura M. Parra, Evelyn CS Batista, and Marco Aurélio C. Pacheco. ”Exploring state-of-the-art LLMs from BERT to XLNet: a study over question answering.” In IEEE Latin American Conference on Computational Intelligence (LA-CCI), IEEE, 2024, 1-6.

[11] Nana, Sidwendluian Romaric, Didier Bassolé, Désiré Guel, and Oumarou Sié. ”Large language models’ adaptation for web applications attacks detection.” In Conference on Information Communications Technology and Society (ICTAS), IEEE, 2025, 1-6.

[12] Valerio, Alberto G., Katya Trufanova, Salvatore De Benedictis, Gennaro Vessio, and Giovanna Castellano. ”From segmentation to explanation: Generating textual reports from MRI with LLMs.” Computer Methods and Programs in Biomedicine, 2025, vol. 270: 108922.

[13] Li, Cheng-Yi, Kao-Jung Chang, Cheng-Fu Yang, Hsin-Yu Wu, Wenting Chen, Hritik Bansal, Ling Chen et al. ”Towards a holistic framework for multimodal LLM in 3D brain CT radiology report generation.” Nature Communications, 2025, vol. 16, no. 1: 2258.

[14] Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. ”Recursive deep models for semantic compositionality over a sentiment treebank.” Dataset. Available: https://huggingface.co/datasets/stanfordnlp/sst2