English QA: Transformer-Based Extractive Question Answering using SQuAD v2

Adi Karthikeya S B – 2417324
Amrutha Vaishnavi Alla - 2378344
Gayatri Chekuri - 2404773

1 Abstract

This project compares four transformer-based models BERT, RoBERTa, DeBERTa, and Flan-T5, based on two methods of question answering: extractive and generative, using the SQuAD v2.0 dataset, which contains both answerable and unanswerable questions. A subset of 10,000 samples was utilized for model fine-tuning within the constraints of GPU memory. The performance of the models was assessed using different metrics like Exact Match (EM), F1 score, BERTScore, etc., with each measuring one aspect of the model performance, either span accuracy or semantic similarity. It has been found through experiments that DeBERTa outperform others in extraction with an F1 score of 78.1% and EM of 74.4%, which is due to its disentangled attention mechanism, whereas Flan-T5 comes close in semantic alignment through generative answering. Moreover, a demo user interface was created for displaying model predictions in real-time. In conclusion, the research brings out the trade-off between extractive precision and generative flexibility in contemporary QA systems.

2 Introduction

The Natural Language Processing (NLP) area has come a long way since the past days of rule-based and statistical approaches to present-day deep learning architectures that are mostly based on the Transformer model. QA, among others, has been quite a difficult task in this process, which means that models must show a complete understanding of the language and, at the same time, be able to fetch accurate information from unstructured text. The first QA systems were based mainly on keyword matching and fixed templates, which led to their limited generalization across various linguistic structures. The introduction of BERT brought about deep bidirectional contextual representations learned via large-scale pretraining that hugely impacted QA.

One major drawback of earlier benchmarks like SQuAD 1.1 was the presumption that each question could be answered by the selected passage, hence allowing models to take advantage of surface patterns rather than showing real understanding. SQuAD 2.0 resolves this problem by introducing more than 50,000 adversarial unanswerable questions that are very similar to answerable ones, and thus transforming the task into a joint problem of answerability classification and span extraction.

2.1 Literature Review

The machine reading comprehension domain has been significantly transformed by transformer-based architectures. Deeply bidirectional pretraining with masked language modelling was the major contribution of BERT, which subsequently established fine-tuning as the leading method of extractive QA, thus attaining the highest score on SQuAD 1.1 [1]. The subsequent model, RoBERTa, outperformed BERT by eliminating the Next Sentence Prediction goal, utilizing dynamic masking, and training on much larger datasets, thereby proving that the training strategy has a very big impact on the accuracy of the downstream QA [2]. DeBERTa was yet another breakthrough on this path; it combined enveloped attention and improved mask decoder, and thus it was able to distinguish span boundaries more accurately and obtain the best scores in several QA benchmarks [3].

Moreover, the use of generative models has been amidst the extractive models, and an increasing level of attention has been given to them. Generative models like T5 have unified the NLP tasks under

a text-to-text framework, so that models can generate answers directly instead of extracting spans [4]. Another model called Flan-T5 was built on top of this model family by means of large-scale instruction tuning; thus, it was able to provide strong zero-shot and few-shot performance, but it also had the well-known hallucination problems [5], [6]. The SQuAD 2.0 introduction not only brought out the issues in the model calibration but also the necessity of the systems to detect when there is no valid answer in the context. Rajpurkar et al. [7] have put forward the CLS-token threshold approach for unanswerable detection; however, the later works have shown the difficulties of confidently abstaining under distribution shifts [8].

To go beyond perfect surface matching and assess semantic correctness, Zhang et al. proposed BERTScore as a novel metric that relies on contextual embeddings rather than tokens [9]. In a nutshell, the previous studies indicate that when it comes to precise span prediction, extractive models are the winners, but on the other hand, generative systems are more flexible but less reliable. Yet, the unanswerable detection issue on SQuAD 2.0 has not been systematically compared across multiple transformer families—it is primarily the reason for the present analysis.

We detail the methodology and findings of our experiments with a SQuAD 2.0 QA system. First, we focus on how the trained models, BERT, RoBERTa, and DeBERTa—encoder-only masked language models (MLMs)—perform extractive QA through span prediction and then we contrast these mechanisms with the generative, sequence-to-sequence approach of Flan-T5. Moreover, we also look at how hyperparameters, document striding as a preprocessing technique, and evaluation metrics such as Exact Match (EM), F1 score, and BERTScore contribute to determining model effectiveness.

3 Dataset

3.1 Dataset Overview

The SQuAD 2.0(Stanford Question Answering Dataset version 2) dataset was selected as the main reference point for this research because it included both answerable and unanswerable questions, which made the task considerably more complicated than SQuAD 1.1. The complete squad_v2 corpus was loaded and robotically divided to facilitate the experimental workflow with the Hugging Face datasets library [7], [10]. Out of the original training split, 8,000 instances were kept for training and 2,000 for validation, thus achieving balanced representation of both question types. The official SQuAD 2.0 validation set (11,873 instances) was kept as the final test set to ensure unbiased performance evaluation. This strict separation avoids information leakage and conforms to the QA benchmarking practices established in Rajpurkar et al. [7].

3.2 Data Preprocessing

In order to utilise the capabilities of both extractive and generative architectures, two preprocessing pipelines were developed. For the case of extractive transformer models, which include BERT, RoBERTa, and DeBERTa [1], [2], [3], the question-context pairs were tokenized considering a maximum sequence length of 384 tokens, with truncation and padding applied for the sake of uniformity. Answer spans from the SQuAD 2.0 dataset, given at the character level, were transferred to token-level start and end positions with the help of offset mappings; the unanswerable instances were represented by assigning both spans to index 0, which corresponds to the [CLS] token as recommended by the SQuAD 2.0 evaluation protocol [7] for such cases. In the case of the generative model Flan-T5 [5], the inputs were transformed into an instruction-oriented format ("question: ... context: ..."), whereas the target text was either the gold answer or the token "unanswerable." The input sequences were tokenized with a maximum of 512 tokens, and padding tokens in the target labels were substituted with –100 to omit them from the loss computation, which is in line with the standard sequence-to-sequence training practices in the Transformers framework [11]. These preprocessing operations guaranteed that the data were ideally structured in accordance with the respective model's architectural requirements.

4 Methodology

The training pipeline of the present investigation is characterized by a unified and systematic approach that targets the SQuAD 2.0 benchmark for both extractive and generative transformer models to be fine-tuned. The initial step is the loading of the relevant tokenizer and model architecture from the Hugging Face Transformers library that is dynamically chosen according to the method of extraction (BERT, RoBERTa, DeBERTa) or generation (Flan-T5) of the model. After selection, each model is assigned to a GPU to ensure that the training is done on fast hardware.

Before the datasets are used for optimization, they go through the task-specific preprocessing to transform the raw SQuAD question–answer pairs into formats that are acceptable for the respective model types. The extractive models get tokenized input with start and end token indices explicitly marking the answer span, whereas the generative models get instruction-formatted input paired with the target answer text or the label "unanswerable." After this processing, the samples are put together in batches, which leads to faster and more stable gradient updates.

The model is trained by the AdamW optimizer, which offers heads above others for stable weight decay regularizations in general and transformer architectures in particular. The training setup includes the use of a linear learning-rate scheduler with warmup, which helps to eventually stabilize the learning rate in the early training stages and avoid the patient periods of instabilities that often come when fine-tuning pretrained models. Each epoch sees a complete cycle of the model forward for each batch of inputs to get the loss, and the gradients 'backwards' flow to the update of the model parameters. Recording of training and validation losses is done throughout the entire process, whereby monitoring of learning progress and spotting of possible overfitting is also conducted.

Validation of the model on the validation dataset is done at the end of each epoch in inference mode in order to avoid parameter updates. The model's generalization ability is therefore evaluated without any bias. After the training is done, the model, tokenizer, and the detailed loss history are kept for future analysis, visualizations, and inference experiments. The GPU memory is then freed up for the training of the following models after this step.

The coming subsections will cut up the four transformer architectures that were utilized in this research—BERT, RoBERTa, DeBERTa, and Flan-T5—into pieces and point out their differences in structure, number of parameters, and fine-tuning interactions.

4.1 BERT Model (Bidirectional Encoder Representations from Transformers)

BERT is a model that employs a bidirectional transformer encoder, which intends to recognize the contextual meaning coming from both sides at the same time. In extractive question answering, BERT integrates a simple QA head atop its already trained encoder to identify the starting and ending points of the answer span in the context. Fig. 1 illustrates the complete model architecture used for SQuAD-style task fine-tuning.

The BERT fine-tuning model has the following architecture:

- **Embedding Layer:** Utilises the token, segment, and positional encodings to transform the input tokens into 768-dimensional embeddings.
- **Transformer Encoder Layers:** A combination of 12 layers of bidirectional self-attention, each consisting of multi-head attention and feed-forward sublayers (totalling 110M parameters).
- **QA Head:** Two linear layers (start and end classifiers) connected to the last hidden states, determining the token-level span boundaries.
- **Output:** Gives rise to two logits vectors that indicate the probability distribution over the candidates for the start and end positions.

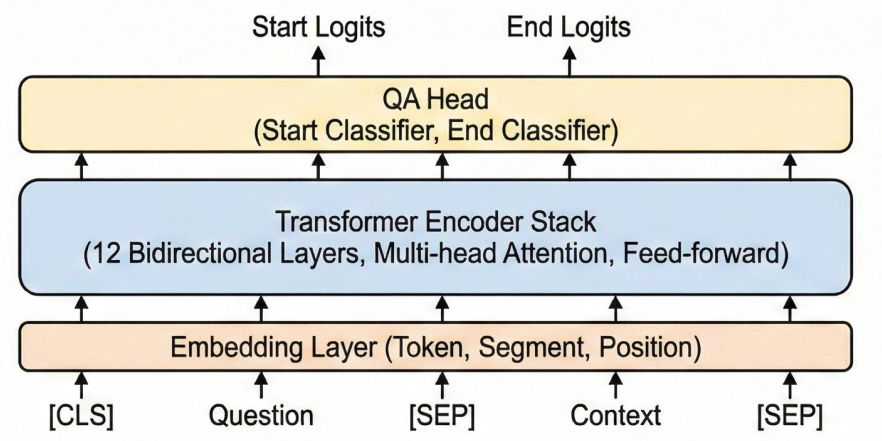

Figure 1: BERT Model Architecture for Question Answering

Even with its huge number of parameters, BERT fine-tunes rapidly and is still a powerful baseline for SQuAD-style tasks. During this research, BERT was trained on a 3e-5 learning rate, batch size 16, and 3 epochs. The model is supposed to map the CLS token (index 0) to unanswerable cases and consequently backward pass through all layers. BERT's abundant bidirectional context is the reason contributing to its effectiveness in the exact span identification; however, the model has more computational cost than DeBERTa and, therefore, is more susceptible to overfitting if not carefully tuned.

4.2 DeBERTa Model (Decoding-Enhanced BERT with Disentangled Attention)

DeBERTa redefined the current technology standards of BERT by implementing new methods on the architecture that split content from positional information, which in turn, gave the model the ability to interpret not only the token semantics but also to understand the dependency of different parts of a sentence. In contrast to BERT, which integrates both ideas into one representation, DeBERTa applies disentangled attention that, in turn, enhances span selection during extractive QA tasks. The presented architecture in Fig. 2 depicts the journey from token embeddings via disentangled attention, transformer layers, and lastly, the QA classification head.

It is the DeBERTa model's architecture used for the QA task, which is represented in the above-mentioned Fig.2.

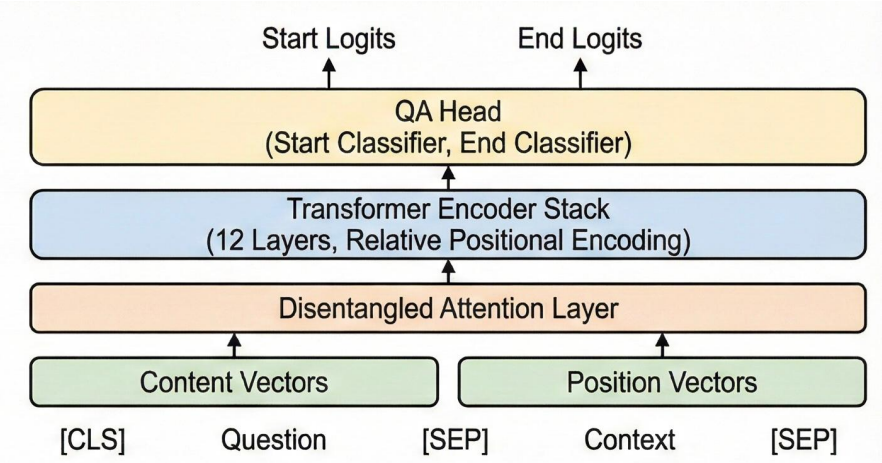

Figure 2: DeBERTa Model Architecture for QA

The DeBERTa fine-tuning architecture comprises the following main components:

- **Disentangled Attention Layer:** Separates content vectors (token meaning) and position vectors (positional index) to compute attention, thereby improving understanding of token relationships.
- **Transformer Encoder Stack:** A stack of 12 encoder layers with relative positional encoding and optimized attention mechanisms (86 M parameters).
- **QA Head:** Two classification layers (start and end position classifiers) are attached to final hidden states to extract the answer span.
- **Output:** Produces start and end logits which represent the most likely answer span within the context.

The research study employed a GPU with constraints to train the DeBERTa model applying 8 8-batch size and 2e-5 learning rate, respectively. Although the parameter count of DeBERTa is slightly less than RoBERTa, the disentangled attention feature of DeBERTa, together with the superior relative positional encoding, makes it propensity to generalization, which is especially useful for identifying unanswerable questions. These architectural virtues are reasons for DeBERTa being both quicker and more precise than the conventional BERT-based models during the fine-tuning phase.

4.3 RoBERTa Model (Robustly Optimized BERT Pretraining Approach)

In short, RoBERTa is a more advanced variant of BERT in terms of pre-training since it entirely removes the Next Sentence Prediction (NSP) task, massive corpus size, and dynamic masking method, which allows the model to see a wider variety of token patterns. All these modifications contribute a lot to the reduction of the fine-tuning time and the making of RoBERTa more powerful for span-based extractive question answering, especially in cases with large contexts.

The architecture for RoBERTa fine-tuning that the study works with is presented in Fig. 3.

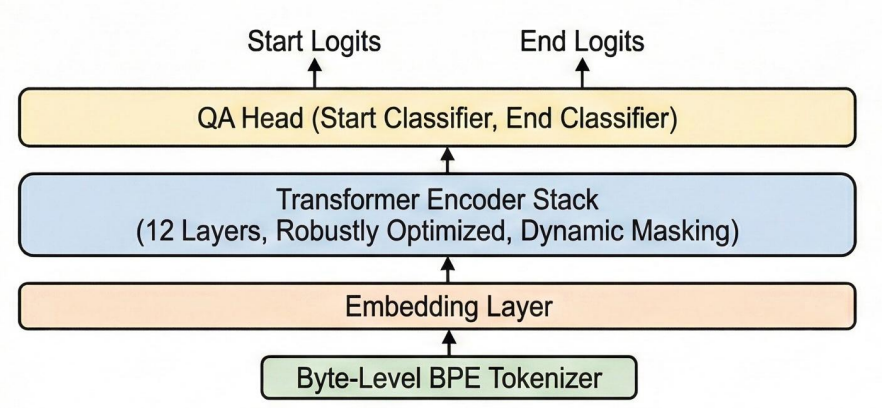

Figure 3: RoBERT Model Architecture for QA

The model architecture for RoBERTa fine-tuning is made up of the following components:

- Embedding Layer: Byte-level BPE tokenizer working with 50,265-word vocabulary.
- Encoder Layers: 12-layer transformer encoder (125M parameters) strengthened by the combination of unique training approaches and a bigger pretraining corpus.
- QA Head: Linear layers (start and end position) that are the same as those of BERT and are used for the purpose of extractive span prediction.
- Output: Provides logits that pinpoint where in the text the answer is likely to be found.

The maximum training period for RoBERTa is 16 batch sizes, a learning rate of 3e-5, and 3 epochs. However, RoBERTa is pre-trained with better optimization methods, and thus, it is usually able to attain the same accuracy quickly and even to outperform BERT in the process. In spite of this, the

model is still very reliable when it comes to long contexts and generalization, but requires a slightly more expensive computation.

4.4 Flan-T5 Model (Instruction-Tuned Encoder–Decoder Transformer)

Flan-T5 is an entirely different approach from the extractive models in that it generates answers in an autoregressive manner. Flan-T5 does not simply select answer spans but rather creates an entire answer text by the process of token generation using its decoder, thereby making it a natural choice for instruction-based tasks and natural language generation.

The fine-tuning procedure for the Flan-T5 model is given below:

- Encoder: Uses a 110M-parameter text encoder to process the instruction-formatted input ("question: ... context: ...").
- Decoder: A 138M-parameter decoder auto-regressively generates the output sequence.
- Seq2Seq Framework: This model computes the next-token probabilities on the basis of the previous outputs and the encoder representations.
- Output: The entire answer text is produced, or the "unanswerable" token is generated for the negative cases.

A schematic representation of the end-to-end sequence-to-sequence framework implemented in this study is presented in Fig. 4:

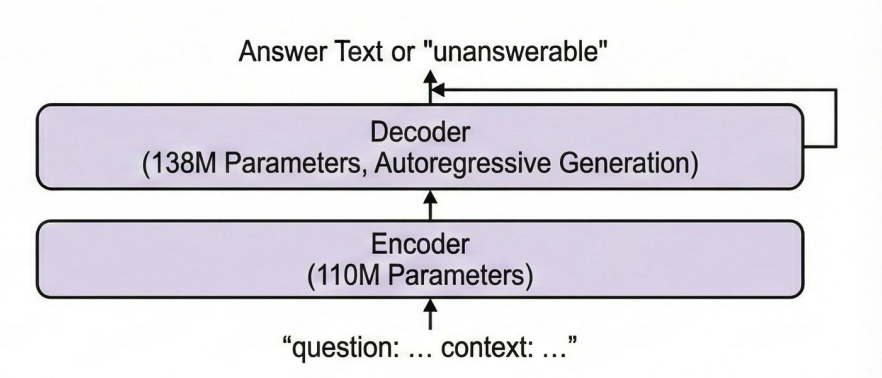

Figure 4: Flan-T5 Model Architecture for QA

Flan-T5 has undergone training under the configuration of a batch size of 8, a learning rate of 5e-5, and 3 epochs in total. The loss calculation is not influenced by the labels' padding tokens since they are masked. Flan-T5, in contrast to the extractive models, can produce rephrased and more plausible answers, but the risk of hallucinating uncontextual information is nevertheless present. The training method offers strong generalization capabilities; however, autoregressive decoding results in slower training and inference compared to span-based models.

5 Results and Discussion

This part of the work elucidates the performance metrics of all four fine-tuned models, namely BERT, DeBERTa-V3, RoBERTa, and Flan-T5, on the SQuAD v2.0 evaluation set. Results are interpreted from several angles, such as the model's training behaviour, answer span extraction accuracy, answerability classification, semantic similarity, and quantitative metrics consolidated. To highlight the comparison of model strengths, visual aids in the form of loss curves, confusion matrices, bar charts, and radar plots are presented, along with a summary heatmap.

5.1 Training Plots

The training and validation loss curves reveal that the optimization of all models was carried out at a steady pace without any divergence, as they all converged steadily through the training epochs.

- BERT started off with a loss that was quite high compared to the others and then slowly came down to stabilize at a point above 1.5 losses (Fig. 5).
- RoBERTa, on the other hand, showed a quicker convergence and also a lower validation loss(Fig. 7) in comparison to BERT, which was a reflection of the advantage gained from the improved pretraining.

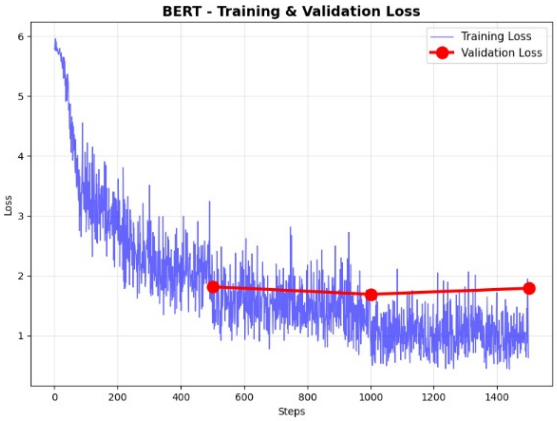

Figure 5: BERT Loss Plot

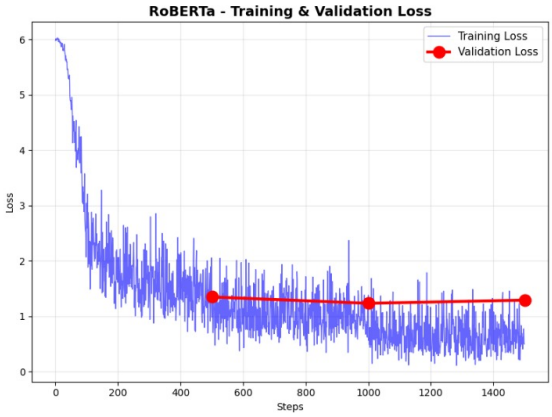

Figure 6: DeBERTa Loss Plot

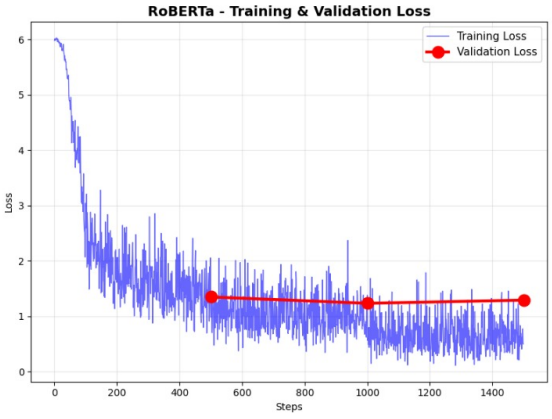

Figure 7: RoBERTa Loss Plot

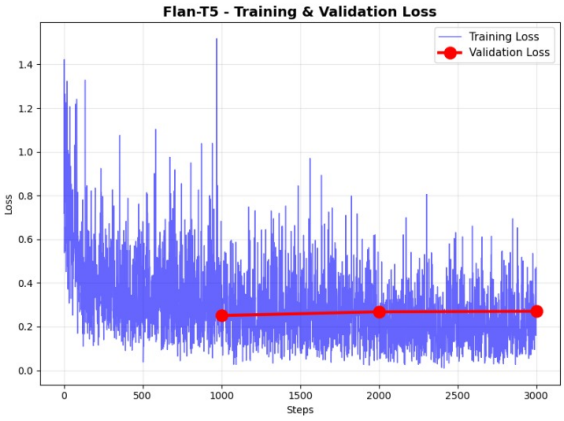

Figure 8: Flan-T5 Loss Plot

- DeBERTa-V3 showed a rapid convergence with an early drop of validation loss below 1.0 (Fig. 6) and was neatly maintained throughout the curve.
- Flan-T5 was found to be the best overall in terms of loss, due to sequence-to-sequence training with teacher forcing, it got stabilized around 0.2-0.3 (Fig. 8).

The curves point to DeBERTa and Flan-T5 being more generalizing and BERT lagging behind in terms of the number of optimization steps required for stabilization.

5.2 Answerability Performance

Confusion matrices are a great way to visualize the performance of each model in terms of the most significant challenge in SQuAD v2.0, which is detecting unanswerable questions (Figs. 9-12).

- BERT was unable to deal with unanswerable cases (34.7% accuracy) and predicted wrong spans instead of not answering most of the time.
- RoBERTa showed a great deal of progress (67.5% unanswerable accuracy) and had a better confidence calibration.
- DeBERTa-V3 was the best in the detection of unanswerable questions with the top score of 76.9% accuracy owing to its disentangled attention mechanism.
- Flan-T5 was also a good performer (71.4%) since instruction tuning helped it to rightly emit "unanswerable" during the generation.

In unanswerable cases, Flan-T5 got the highest accuracy score (92.3%), while DeBERTa came next (87.9%).

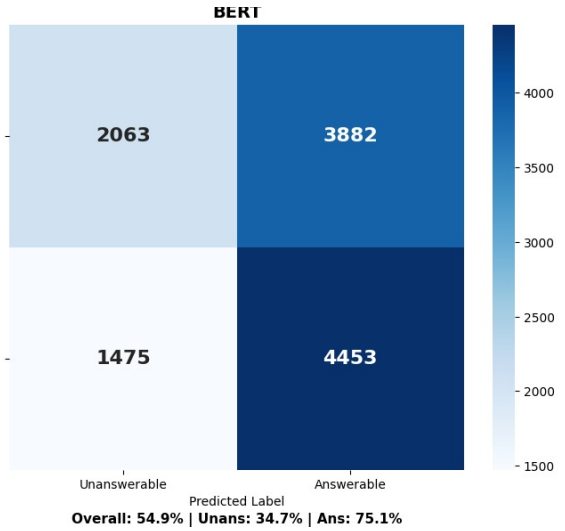

Figure 9: Confusion Matrix of BERT

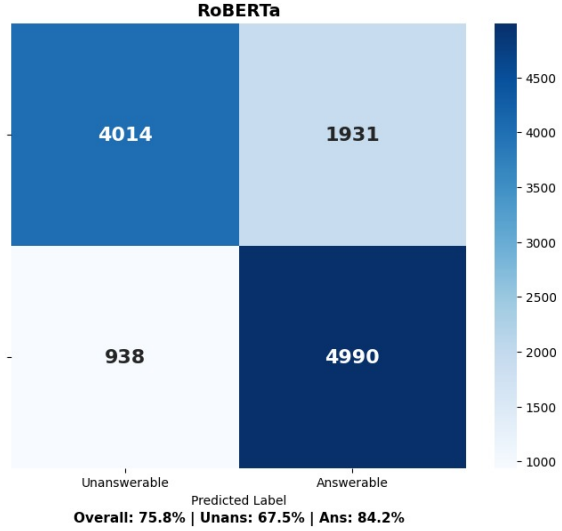

Figure 10: Confusion Matrix of RoBERTa

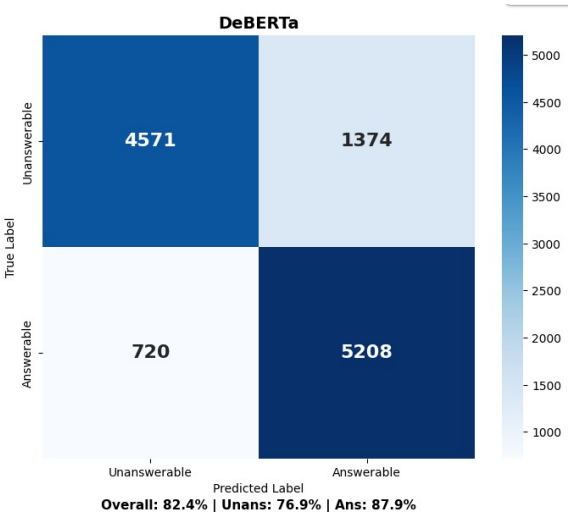

Figure 11: Confusion Matrix of DeBERTa

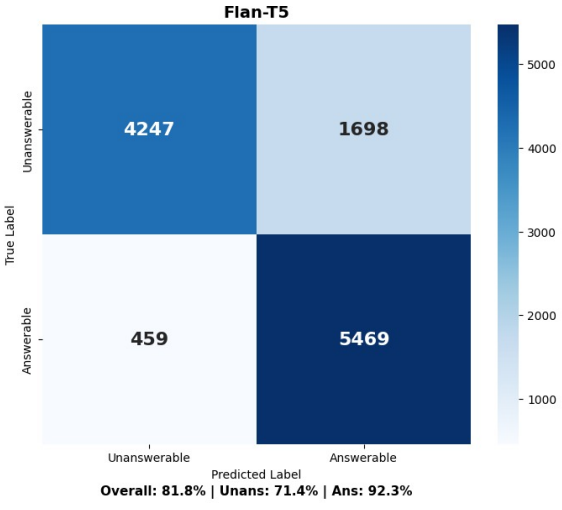

Figure 12: Confusion Matrix of Flan-T5

5.3 QA Metrics (Exact Match and F1 Score)

The metric comparison charts present the following results in Table 1:

- Clearly, BERT yields very low results with EM = 34.9% and F1 = 38.8%, thus indicating a non-viable use of the older transformer technologies for SQuAD v2.0.
- Even though RoBERTa has an overall better performance with EM = 66.9%, F1 = 70.5%.

Table 1: Evaluation metrics for all four models

Metric	BERT	DeBERTa	RoBERTa	Flan-T5
Exact Match	34.9	74.4	66.9	75.3
F1 Score	38.8	78.1	70.5	78.4

- By a narrow margin over RoBERTa, DeBERTa succeeds with EM = 74.4%, F1 = 78.1%, which is in line with the literature stating DeBERTa's advantage in passage extraction.
- The highest overall span quality has been attained by Flan-T5 with EM = 75.3% and F1 = 78.4%, slightly beating DeBERTa.

Thus, the findings proclaim that when used correctly, the generative models are at par or even better than the extractive models for span-based QA tasks.

5.4 Semantic Similarity (BERTScore & BLEU)

Table 2 presents BERTScore Precision, Recall, and F1, reflecting semantic similarity rather than literal match.

- Flan-T5 recorded the highest semantic similarity (F1 = 97.20).
- DeBERTa and RoBERTa follow closely (96.0 and 95.5).
- BERT's semantic alignment (88.0) shows a gap due to weaker contextual representation.
- The generative decoding process of Flan-T5 produces natural and context-aware answers, which improve semantic closeness even when the wording differs from the reference.

Table 2: Per-metric performance comparison of all models

Metric (%)	BERT	DeBERTa	RoBERTa	Flan-T5
Precision	86.4	95.6	95.0	97.1
Recall	89.8	96.5	96.0	97.4
F1 Score	88.0	96.0	95.5	97.2

BLEU scores were 0 for all models except BERT (10.89), since extractive models output verbatim spans and generative answers are short enough that BLEU undervalues them. This is expected and consistent with prior findings that BLEU is unsuitable for span-extraction tasks.

5.5 Classification Metrics

A combined view of accuracy, macro/micro F1, precision, and recall reveals the following trends:

- DeBERTa achieves the strongest classification performance across all metrics (Macro F1 = 82.3, Precision = 82.8, Recall = 82.4).
- Flan-T5 performs comparably (Macro F1 = 81.6) but excels in precision for answerable questions.
- RoBERTa sits in the mid-range (Macro F1 = 75.7).
- BERT remains the weakest model (Macro F1 = 53), primarily due to poor unanswerable detection.

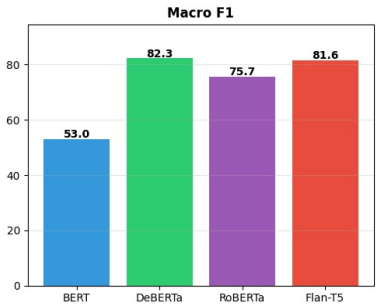

Figure 13: Macro F1

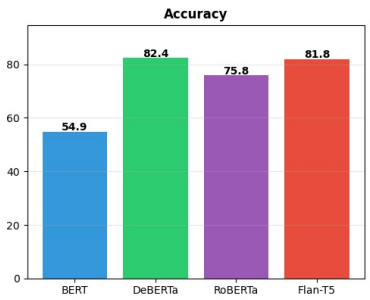

Figure 14: Accuracy

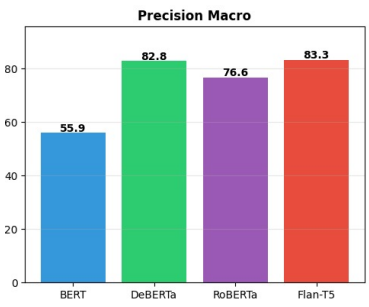

Figure 15: Precision

5.6 Best Model Summary

The evaluations of the four models demonstrate marked differences in performance regarding extractive and generative Question Answering. As shown in Table 3, Flan-T5 had the best performance as shown by its high Exact Match score (75.33%) and F1 score (78.45%), which made it the best performing model overall because of its ability to produce meaningful responses and also accurately classify whether an answer could be provided. The next model, DeBERTa, scored an F1 score of 78.06% and had the highest unanswerable accuracy score of the four models (82.36%). For this reason, DeBERTa is considered the best-performing extractive Question Answering model and shows a particular strength with SQuAD v2.0's tougher examples of not knowing the answer. RoBERTa's F1 and Exact Match scores were also very good (F1 score = 70.53%, Exact Match = 66.87%), thanks to its superior pre-training approach; however, BERT was significantly weaker in all categories, indicating BERT's continued struggle with handling unanswerable questions. In summary, these results exhibit that new deep learning methods, especially generative models like Flan-T5, outperform traditional span extractors that give better semantic alignments and increased reliability of answers.

Although Flan-T5 is a generative model and DeBERTa is an extractive model, DeBERTa still competed strongly with Flan-T5 and even achieved better results. This makes DeBERTa the overall winner, demonstrating its strength across all four models.

Table 3: Comparison Table of all models

Model	EM	F1	BLEU	BS-P	BS-R	BS-F1	Acc	Macro-F1	Micro-F1	P-Macro	R-Macro	Ans-Acc	Unans-Acc
BERT	34.85	38.85	10.89	86.36	89.79	87.97	54.88	52.98	54.88	55.87	54.91	75.12	34.70
DeBERTa	74.38	78.06	0.00	95.55	96.53	96.01	82.36	82.31	82.36	82.76	82.37	87.85	76.89
RoBERTa	66.87	70.53	0.00	95.01	96.02	95.47	75.84	75.67	75.84	76.58	75.85	84.18	67.52
Flan-T5	75.33	78.45	0.00	97.09	97.37	97.20	81.83	81.64	81.83	83.28	81.85	92.26	71.44

6 System Interface and Demo

A straightforward demonstration interface was designed to demonstrate how each of the four fine-tuned Models for Question and Answering–BERT, RoBERTa, DeBERTa-V3 and Flan-T5–behaves when given a user-defined question plus the respective context. As illustrated in the demo in Fig. 16, this system is running on a CUDA-enabled machine and allows the user to input a question, e.g., What is my future career, together with a context statement such as, "I'm interested in doing my Master's degree and becoming a lecturer at my university."

Once submitted, each of the four Models produced its corresponding prediction:

BERT: "Master's Degree and become a lecturer."

DeBERTa: "Lecturer"

RoBERTa: "Lecturer"

Flan-T5: "Professor"

The above represents a clear demonstration of the differences between Extractive and Generative types of QA behaviour. The three Extractive QA Models (BERT, RoBERTa, DeBERTa) utilize spans from the provided context, but DeBERTa produced the most accurate answer due to its superior performance as exemplified by its EM and F1-measures obtained during evaluation. The generation

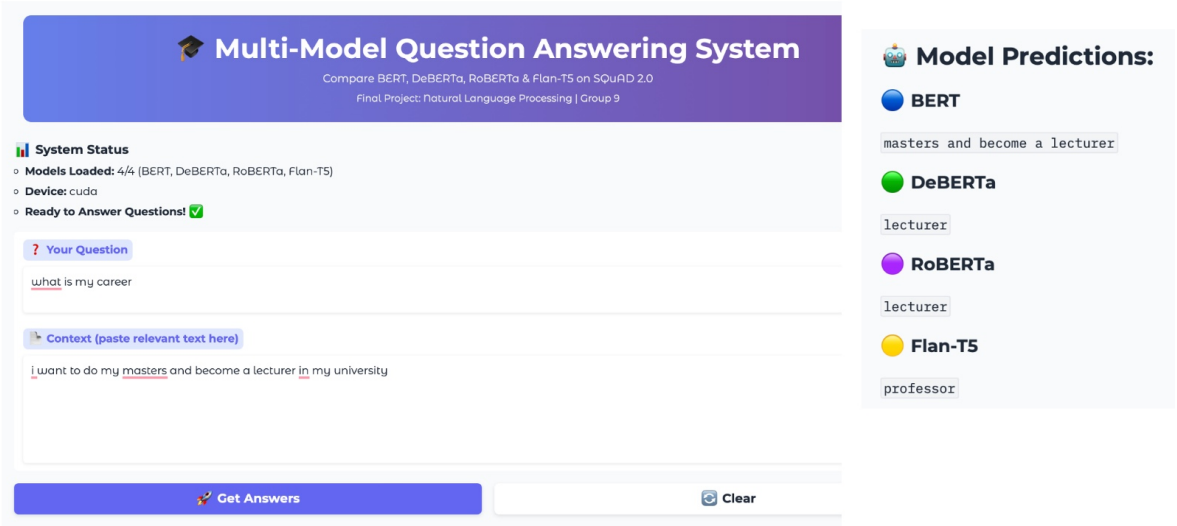

Figure 16: Interface showing answers of all four models

QA Model, Flan-T5, produces a paraphrased answer based on its different instruction-tuning structure and greater semantic similarity metrics.

In conclusion, the demonstration webpage provides a simple means of confirming the conclusions of the experiments, as well as a means of visually contrasting the performance of various methods of transformer architecture.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[2] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[3] P. He, X. Liu, J. Gao, and W. Chen, "Deberta: Decoding-enhanced bert with disentangled attention," 2021.

[4] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," 2023.

[5] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, A. Castro-Ros, M. Pellat, K. Robinson, D. Valter, S. Narang, G. Mishra, A. Yu, V. Zhao, Y. Huang, A. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, and J. Wei, "Scaling instruction-finetuned language models," 2022.

[6] V. Rawte, A. Sheth, and A. Das, "A survey of hallucination in large foundation models," 2023.

[7] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for squad," 2018.

[8] A. Kamath, R. Jia, and P. Liang, "Selective question answering under domain shift," 2020.

[9] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," 2020.

[10] Q. Lhoest, A. V. del Moral, Y. Jernite, A. Thakur, P. von Platen, S. Patil, J. Chaumond, M. Drame, J. Plu, L. Tunstall, J. Davison, M. Šaško, G. Chhablani, B. Malik, S. Brandeis,

T. L. Scao, V. Sanh, C. Xu, N. Patry, A. McMillan-Major, P. Schmid, S. Gugger, C. Delangue, T. Matussière, L. Debut, S. Bekman, P. Cistac, T. Goehringer, V. Mustar, F. Lagunas, A. M. Rush, and T. Wolf, "Datasets: A community library for natural language processing." https://rajpurkar.github.io/SQuAD-explorer/, 2021.

[11] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Huggingface's transformers: State-of-the-art natural language processing." https://arxiv.org/abs/1910.03771, 2020.