BigBird NLP Tech report

September 2025

1 Introduction

In our experiments on long-sequence natural language processing tasks, we evaluated several classical machine learning and transformer-based approaches. Ultimately, Google's BigBird Base Roberta (400M parameters) was selected for its superior performance on long documents.

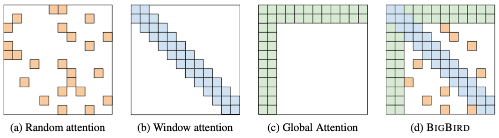
2 BigBird Architecture

BigBird is a transformer-based model inspired by BERT/Roberta, designed specifically for handling very long sequences efficiently. Its key innovation is a **sparse attention mechanism** that reduces the self-attention complexity from $O(n^2)$ to $O(n)$, enabling processing of sequences up to 8k–16k tokens.

The attention mechanism combines three types:

- **Global attention:** Selected important tokens attend to all tokens in the sequence.
- **Random attention:** Randomly selected tokens attend to other tokens, allowing global context propagation.
- **Local sliding window attention:** Each token attends to a fixed-size neighborhood, preserving sequential information.

BigBird's combination of global, random, and local attention captures long-range dependencies more effectively while remaining computationally efficient.



(a) Random attention (b) Window attention (c) Global Attention (d) BigBird

Figure 1: BigBird Attention Mechanisms

3 Experimental Methods

The following approaches were explored:

1. **TFIDF + XGBoost:** A baseline gradient boosting approach, effective on small datasets but limited in capturing semantic context.
2. **TFIDF + MLP:** Slight improvement over XGBoost due to non-linear modeling, but still constrained by the bag-of-words representation.
3. **Longformer:** Transformer with sparse attention; handles long sequences better than classical methods, yet underperforms BigBird on very long documents and is slightly larger taking more time to train.
4. **Distilled BigBird:** A smaller, faster variant providing a good speed-accuracy trade-off, with some accuracy loss compared to full BigBird.
5. **BigBird + LightGBM:** Using BigBird embeddings as features for gradient boosting; competitive but slightly worse than end-to-end fine-tuning.

4 Final Model Selection

After extensive experimentation, **Google's BigBird Base Roberta (400M)** was selected. Fine-tuning the full model on task-specific data leveraged the sparse attention mechanism fully, resulting in the best performance for long-sequence tasks.

5 Model and Training Pipeline

5.1 Model

We utilized the pre-trained transformer model `google/bigbird-roberta-base` from Hugging Face. BigBird extends the RoBERTa architecture by employing a sparse attention mechanism, allowing it to efficiently handle long input sequences.

5.2 Data Preprocessing

The dataset consisted of prompts associated with class labels. Using the following label-to-ID mapping:

```
CLASS_NAME_TO_ID = {'pikachu': 1, 'charizard': 2, 'bulbasaur': 3, 'mewtwo': 4}
```

and the JSON-formatted dataset, we created a pandas DataFrame containing only `image_id` and `prompt` columns.

The dataset was split into training, validation, and test sets in a 4:1:1 ratio. Tokenization was performed in batches using the tokenizer corresponding to the BigBird model. The objective of the model is to predict the correct class label given a text prompt.

5.3 Training and Evaluation

The model was trained using the following hyperparameters:

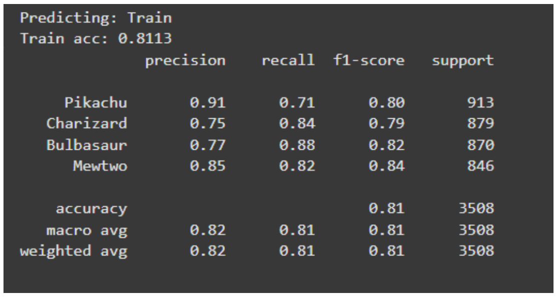
- Learning rate: 2×10^{-5}
- Number of epochs: 3
- Evaluation metrics: Accuracy, Precision, Recall, and F1 Score

During training, metrics were computed on the validation set at the end of each epoch to monitor performance.

6 Evaluation Results

6.1 Training Data

Achieved 81 % Training Accuracy



```
Predicting: Train
Train acc: 0.8113
              precision    recall  f1-score   support

     Pikachu       0.91      0.71      0.80       913
   Charizard       0.75      0.84      0.79       879
   Bulbasaur       0.77      0.88      0.82       870
      Mewtwo       0.85      0.82      0.84       846

    accuracy                           0.81      3508
   macro avg       0.82      0.81      0.81      3508
weighted avg       0.82      0.81      0.81      3508
```

Figure 2: Training Metrics (Accuracy, Precision, Recall, F1 Score).

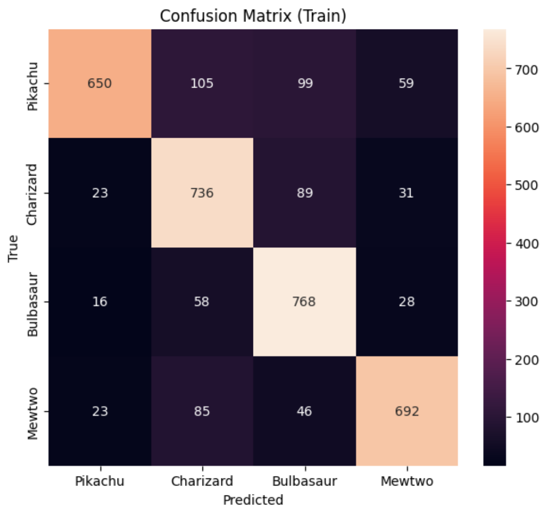
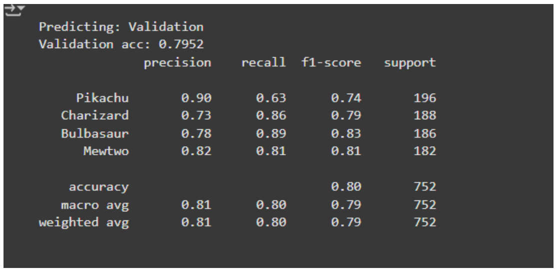


Figure 3: Confusion Matrix for the Training Data.

6.2 Validation Data

Achieved 79 % Validation Accuracy



```
Predicting: Validation
Validation acc: 0.7952
              precision    recall  f1-score   support

     Pikachu       0.90      0.63      0.74       196
   Charizard       0.73      0.86      0.79       188
   Bulbasaur       0.78      0.89      0.83       186
      Mewtwo       0.82      0.81      0.81       182

    accuracy                           0.80       752
   macro avg       0.81      0.80      0.79       752
weighted avg       0.81      0.80      0.79       752
```

Figure 4: Validation Metrics (Accuracy, Precision, Recall, F1 Score).

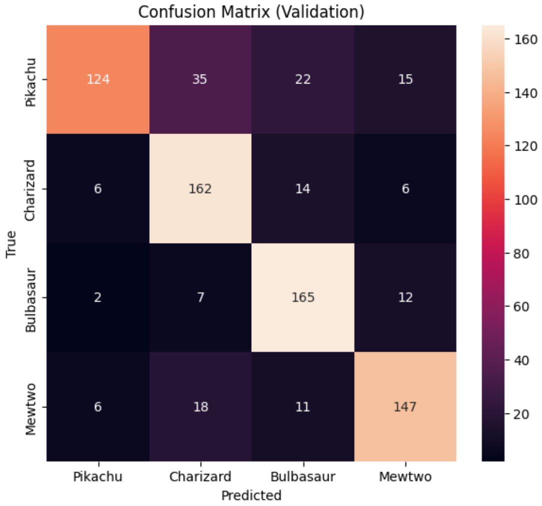
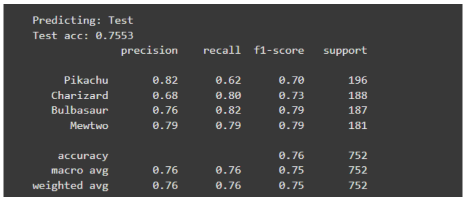


Figure 5: Confusion Matrix for the Validation Data.

6.3 Test Data

Achieved 75 % Test Accuracy



```
Predicting: Test
Test acc: 0.7553
              precision    recall  f1-score   support

     Pikachu       0.82      0.62      0.70       196
   Charizard       0.68      0.80      0.73       188
   Bulbasaur       0.76      0.82      0.79       187
      Mewtwo       0.79      0.79      0.79       181

    accuracy                           0.76       752
   macro avg       0.76      0.76      0.75       752
weighted avg       0.76      0.76      0.75       752
```

Figure 6: Test Metrics (Accuracy, Precision, Recall, F1 Score).

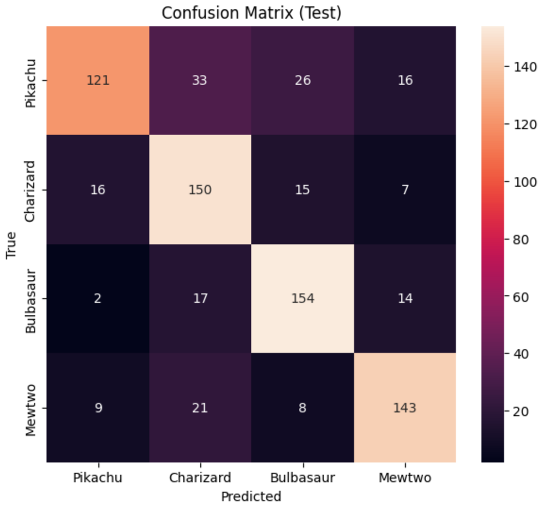


Figure 7: Confusion Matrix for the Test Data.