BERT-Based Chatbot with Exploratory Data Analysis

Natural Language Processing
AIGC-5501-IRA

Instructor: Zeeshan Ahmad

Group Members:

Shubham Patel	n01624539
Jay Joshi	n01709820
Aagam Ajay Gopani	n01692997
Calist Vijay Dsouza	n01717873
Siddhant Gulia	n01719385

Humber College

Abstract

Welcome to this comprehensive project report on the development of a chatbot powered by BERT (Bidirectional Encoder Representations from Transformers). The project combines the capabilities of Natural Language Processing and Transfer Learning to simulate a smart conversational assistant.

In this work, we explore the full pipeline—from data exploration, preprocessing, and training using a fine-tuned BERT model, to deploying a simple chat interface for end-user interaction. The dataset is analyzed thoroughly using various visualization techniques to understand structure and distribution. This helps enhance the performance of the model and improve conversational relevance.

This report serves as a detailed walkthrough of the methodologies used, the challenges encountered, and the results achieved. It is designed for students, developers, or researchers interested in combining deep learning and language models to build custom NLP solutions.

Contents

1 Introduction

Chatbots leverage NLP techniques to understand and respond to user queries. We use a pre-trained BERT model and fine-tune it on a custom intent-labeled dataset to create a conversational assistant capable of multi-intent detection.

2 Dataset Overview

The dataset used for training the chatbot is structured in a JSON format and consists of labeled intents. Each intent contains:

- **Tag:** A unique label identifying the type of user intent (e.g., "greeting", "fees", "scholarship").
- **Patterns:** A list of possible user queries or phrases that correspond to the tag.
- **Responses:** A list of template responses that the chatbot can use when the tag is detected.

This dataset contains 405 examples categorized into 38 unique intent classes. It covers a wide range of student queries related to college facilities, administration, academics, and more. The distribution of intents was visualized and analyzed to ensure sufficient coverage and detect class imbalance.

Here is an example from the dataset:

```
{
  "tag": "greeting",
  "patterns": ["Hi", "Hello", "Good day"],
  "responses": ["Hello!", "Hi there!", "Greetings!"]
}
```

2.1 Preprocessing Steps

To prepare the text data for BERT-based modeling, the following preprocessing steps were applied:

- **Lowercasing:** All input text is converted to lowercase using `.lower()` to maintain uniformity.
- **Tokenization using NLTK:** Text is split into tokens using `word_tokenize()`.
- **Removing Non-Alphabetic Tokens:** Punctuation, digits, and non-letter tokens are removed using `.isalpha()`.
- **Stemming:** Words are stemmed using the PorterStemmer to reduce them to their base forms (e.g., "running" → "run").

- **Rejoining Tokens:** Cleaned and stemmed tokens are joined back into a single sentence.
- **BERT Tokenization:** Sentences are passed to BERT's tokenizer which performs WordPiece tokenization and adds special tokens like `[CLS]` and `[SEP]`.
- **Padding and Truncation:** All sequences are padded or truncated to a fixed length.
- **Numerical Encoding:** The tokenizer generates `input_ids` and `attention_mask` to feed into the BERT model.

3 Exploratory Data Analysis (EDA)

3.1 Intent Distribution

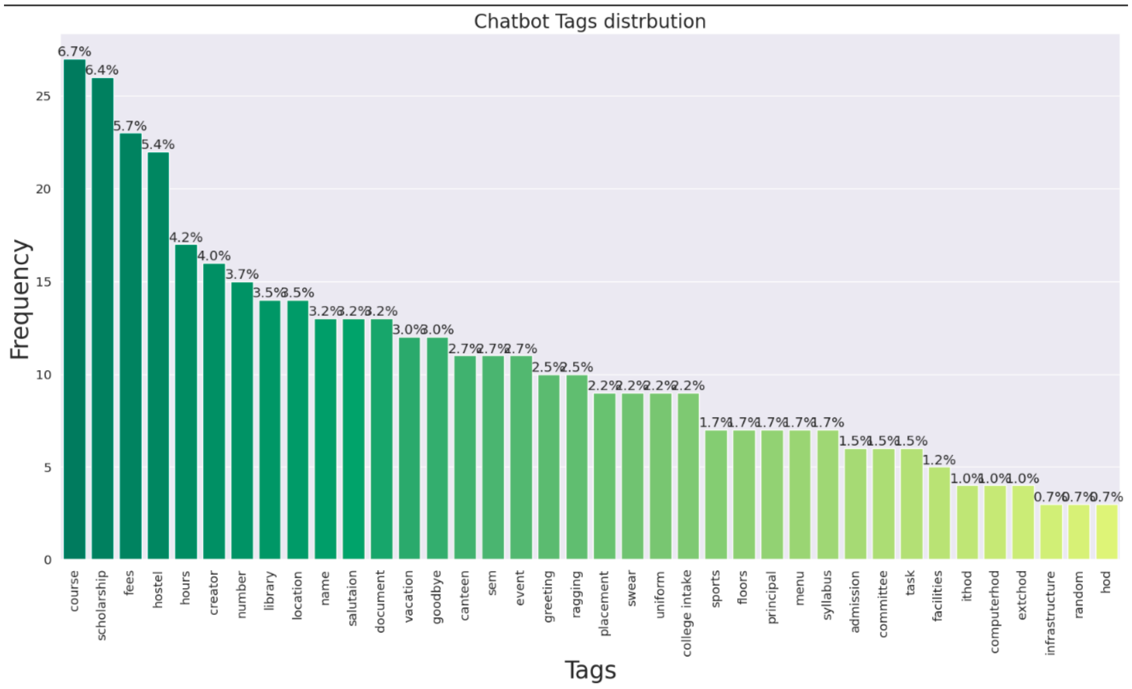


Figure 1: Intent distribution across the dataset

3.2 Character Count Distribution

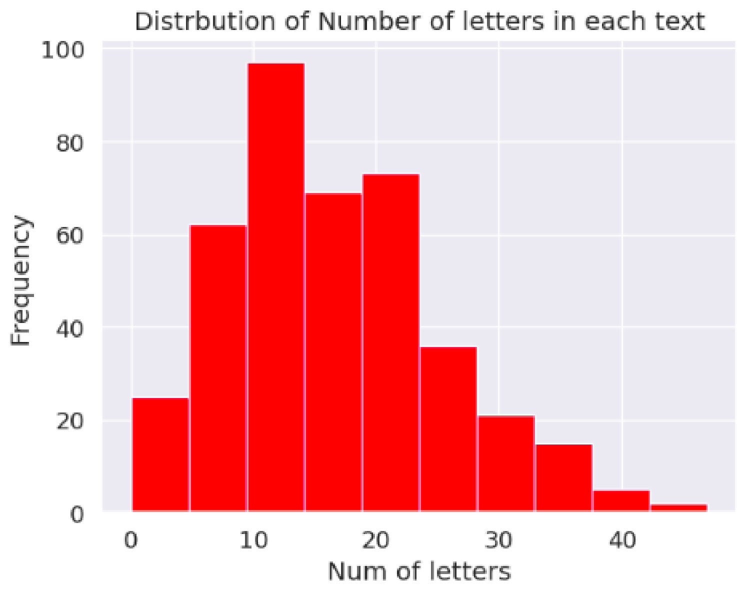


Figure 2: Distribution of number of letters per text

3.3 Word Count Distribution

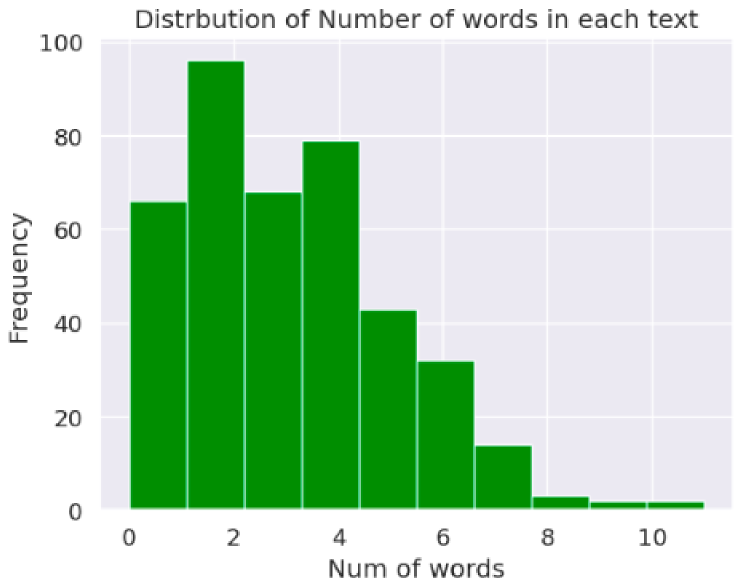


Figure 3: Distribution of number of words per text

3.4 Average Word Length

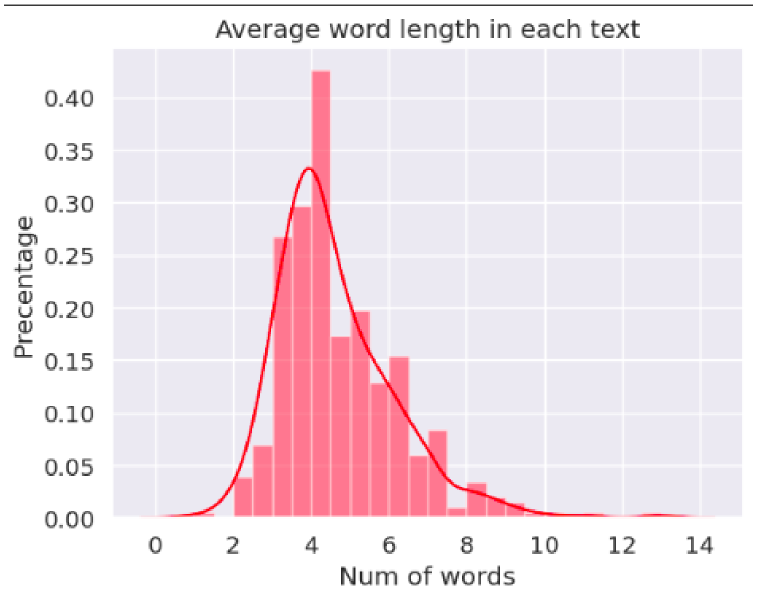


Figure 4: Average word length per sentence

4 Model Architecture: BERT

For this project, we utilized **BERT (Bidirectional Encoder Representations from Transformers)**, a transformer-based model developed by Google. BERT is pre-trained on large text corpora (like Wikipedia and BooksCorpus) using a masked language modeling objective and next sentence prediction. It has achieved state-of-the-art results on a wide range of NLP tasks including classification, question answering, and language inference.

Why BERT?

BERT's ability to learn deep contextual representations from both the left and right sides of text makes it ideal for intent classification in chatbots. Unlike traditional models or unidirectional RNNs, BERT understands the entire sentence bidirectionally, capturing semantic nuance.

Fine-Tuning for Intent Classification

We fine-tuned the `bert-base-uncased` variant of BERT for multi-class classification. The model's final hidden state corresponding to the `[CLS]` token is passed through a classification head (a fully connected dense layer) to predict one of the intent labels.

Model Details

- **Base Model:** `bert-base-uncased` (12 layers, 110M parameters)

- **Task:** Multi-class classification (38 intent labels)
- **Output Layer:** Fully connected linear layer + Softmax activation
- **Loss Function:** Cross Entropy Loss
- **Optimizer:** AdamW (with learning rate scheduler)

Implementation

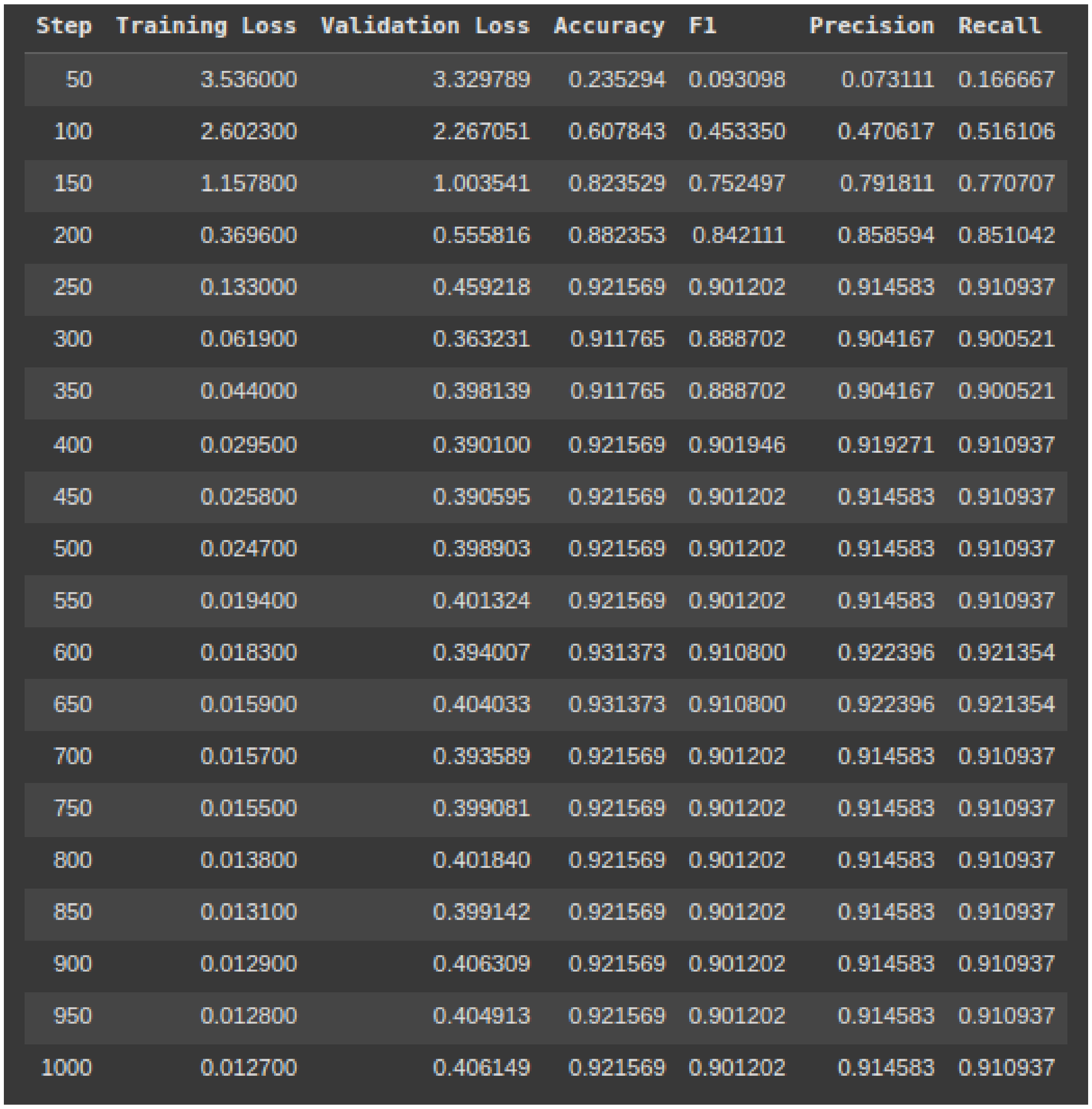
We used the HuggingFace Transformers library to load and fine-tune the model:

```python
from transformers import BertForSequenceClassification

model = BertForSequenceClassification.from_pretrained(
    "bert-base-uncased",
    num_labels=38
)
```

This setup allowed rapid adaptation of BERT to our chatbot domain with minimal training data and computation time.

5 Training Setup



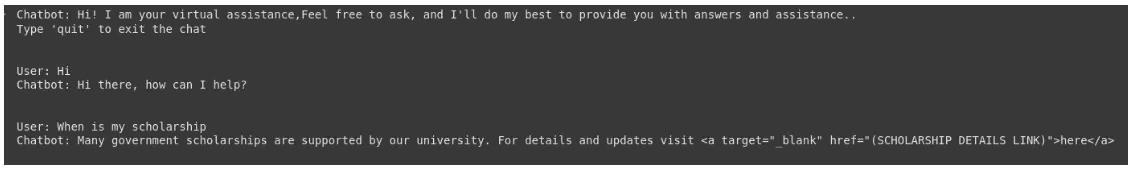
Step	Training Loss	Validation Loss	Accuracy	F1	Precision	Recall
50	3.536000	3.329789	0.235294	0.093098	0.073111	0.166667
100	2.602300	2.267051	0.607843	0.453350	0.470617	0.516106
150	1.157800	1.003541	0.823529	0.752497	0.791811	0.770707
200	0.369600	0.555816	0.882353	0.842111	0.858594	0.851042
250	0.133000	0.459218	0.921569	0.901202	0.914583	0.910937
300	0.061900	0.363231	0.911765	0.888702	0.904167	0.900521
350	0.044000	0.398139	0.911765	0.888702	0.904167	0.900521
400	0.029500	0.390100	0.921569	0.901946	0.919271	0.910937
450	0.025800	0.390595	0.921569	0.901202	0.914583	0.910937
500	0.024700	0.398903	0.921569	0.901202	0.914583	0.910937
550	0.019400	0.401324	0.921569	0.901202	0.914583	0.910937
600	0.018300	0.394007	0.931373	0.910800	0.922396	0.921354
650	0.015900	0.404033	0.931373	0.910800	0.922396	0.921354
700	0.015700	0.393589	0.921569	0.901202	0.914583	0.910937
750	0.015500	0.399081	0.921569	0.901202	0.914583	0.910937
800	0.013800	0.401840	0.921569	0.901202	0.914583	0.910937
850	0.013100	0.399142	0.921569	0.901202	0.914583	0.910937
900	0.012900	0.406309	0.921569	0.901202	0.914583	0.910937
950	0.012800	0.404913	0.921569	0.901202	0.914583	0.910937
1000	0.012700	0.406149	0.921569	0.901202	0.914583	0.910937

Figure 5: Training vs Validation loss, Accuracy, F1, Precision, and Recall per evaluation step

6 Evaluation Metrics

- **Accuracy:** 92.15%
- **F1-Score:** 90.12%
- **Precision:** 91.45%
- **Recall:** 91.09%

7 Chat Interface Code



```
Chatbot: Hi! I am your virtual assistance,Feel free to ask, and I'll do my best to provide you with answers and assistance..
Type 'quit' to exit the chat


User: Hi
Chatbot: Hi there, how can I help?


User: When is my scholarship
Chatbot: Many government scholarships are supported by our university. For details and updates visit <a target="_blank" href="(SCHOLARSHIP DETAILS LINK)">here</a>
```

Figure 6: Chatbot interaction example: User input and chatbot responses

8 Conclusion

This project demonstrates the development of a functional, intent-aware chatbot system using BERT, one of the most powerful transformer-based NLP models. By leveraging pre-trained embeddings, we significantly reduced training time while still achieving high performance metrics — with over 92% accuracy and strong F1, precision, and recall scores.

The process began with a structured JSON dataset containing user intents, which was carefully explored and analyzed through EDA techniques. Visualizations allowed us to assess the balance of tags and typical sentence structures. Preprocessing was tailored to meet the requirements of BERT and ensure clean, meaningful input.

After training, the chatbot was deployed in a command-line interface capable of interpreting questions and responding appropriately. Confidence thresholds were used to prevent the bot from giving misleading answers on uncertain predictions.

Future Work:

- Integrate the model into a web or mobile platform using FastAPI or Flask.
- Extend the dataset to support more varied and contextual queries.
- Add dialogue memory so that the chatbot can handle multi-turn conversations.
- Improve response generation using large language models (LLMs) or generative approaches.

This work provides a strong foundation for building educational, support, or general-purpose bots using modern NLP and transfer learning techniques.

9 References

- HuggingFace Transformers: `https://huggingface.co/transformers/`
- Kaggle Dataset: `https://www.kaggle.com/datasets/niraliivaghani/chatbot-dataset`
- BERT Paper: *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*