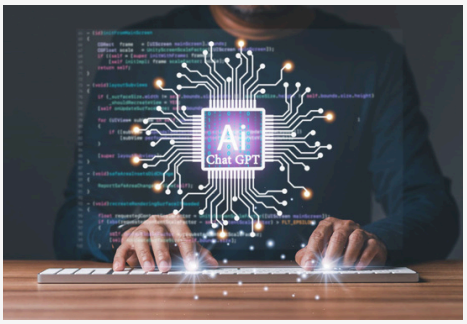
AI Generated Text Detection

ML CHALLENGE 2.0 SUBMISSION

PROBLEM STATEMENT

The task was to classify given text as either human-written or AI-generated.



MODEL'S DESIGN AND DEVELOPMENT PROCESS

- **Data Preparation**: The dataset contained labeled text samples. Preprocessing included tokenization and text cleaning, followed by the generation of embeddings using the BERT model, which captures contextual relationships in the text.
- **Model Architecture**: A sequential model was designed using TensorFlow Keras. It comprised dropout layers to prevent overfitting and dense layers for classification. The final output layer utilized a sigmoid activation function to output probabilities for binary classification.
- **Training and Validation**: The model was trained on a training set, with a validation split for hyperparameter tuning. The training process optimized for binary cross-entropy loss, focusing on metrics like accuracy, precision, recall, and F1 score.
- **Hyperparameter Tuning**: Parameters such as learning rate, batch size, and dropout rate were adjusted to improve model performance.
- **Evaluation**: On the test set, the model achieved a macro F1 score of 0.88, demonstrating its effectiveness in distinguishing between human and AI-generated content.

KEY DECISIONS AND STRATEGIES

Choosing BERT for Text Representation

One of the most crucial decisions was the selection of BERT (Bidirectional Encoder Representations from Transformers) for generating text embeddings. Unlike traditional methods like TF-IDF or word2vec, BERT provides contextual embeddings by considering both the left and right context of a word in a sentence. This bidirectional approach ensures a more comprehensive understanding of the text, which is critical in distinguishing nuanced differences between human and AI-generated content. The pre-trained BERT model was fine-tuned on the dataset, allowing it to learn task-specific patterns and improve classification performance.

Text Preprocessing and Embedding Conversion

Before feeding the data into the model, the text underwent preprocessing to remove noise and standardize input. This included steps such as removing special characters, lowercasing, and tokenization. Tokenization was particularly important because BERT operates on tokenized sequences. Each input text was then padded to a fixed length to ensure uniformity in batch processing. Once tokenized, the text was converted into embeddings using BERT's transformer layers, resulting in a high-dimensional vector representation for each sample. These embeddings capture semantic relationships and contextual nuances, enabling the model to better differentiate between human and AI writing styles.

Model Architecture and Training Strategies

The model was designed as a sequential neural network in TensorFlow Keras. It consisted of dense layers with ReLU activations for feature extraction and a final sigmoid layer for binary classification. Dropout layers were incorporated to prevent overfitting, which was critical given the relatively small dataset size. Binary cross-entropy loss was chosen as the loss function due to the binary nature of the classification problem. Metrics such as precision, recall, and F1 score were prioritized during training to ensure the model performed well on imbalanced data.

Hyperparameter Tuning and Evaluation

Hyperparameter tuning involved optimizing learning rate, batch size, and dropout rate. A stratified validation split ensured balanced class distribution during training, and early stopping was employed to prevent overfitting. The model achieved a macro F1 score of 0.88 on test data, indicating robust performance.
This strategy ensured the model could effectively handle the complexity of distinguishing between human and AI-generated text, contributing to enhanced content authenticity verification.

INSIGHTS, AND LEARNINGS

In this project, I focused on building a robust model to distinguish between human-written and AI-generated text. A key decision was using BERT for text embeddings due to its ability to capture contextual relationships, going beyond simple word vectors to understand sentence-level meaning. This was crucial given the subtle distinctions between human and AI-generated content.
Preprocessing was another critical aspect. Cleaning the text, removing unnecessary symbols, and standardizing formats enhanced the quality of the data. Experimenting with various preprocessing techniques helped me understand how minor changes impacted the embeddings and model performance.
Hyperparameter tuning also played a significant role. Adjusting parameters such as learning rate, batch size, and dropout rates helped optimize the balance between model complexity and generalization. To address the challenge of imbalanced datasets, I explored different threshold values and evaluation metrics, focusing on the macro F1 score.
Ultimately, this project emphasized the importance of balancing precision and recall, teaching me to adapt and continually refine my approach for improved outcomes.

click here to view the project

Thank you

Submitted by:
Harsh kumar
22f3002198@ds.study.iitm.ac.in