Model Optimization and Tuning Phase Template

Date	15 October 2024
Team ID	739743
Project Title	Spooky Author Identification Using Deep Learning
Maximum Marks	10 Marks

Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining neural network models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

Hyperparameter Tuning Documentation (8 Marks):

Model	Tuned Hyperparameters

Keras

```python
import tensorflow as tf
from tensorflow.keras.regularizers import l2

# Manually define a single set of hyperparameters to tune
embedding_dim = 100  # Example embedding dimension
filters = 128        # Example number of Conv1D filters
kernel_size = 5      # Example kernel size
dense_units = 128    # Example dense layer units
dropout_rate = 0.3   # Example dropout rate
l2_reg = 0.001       # Example L2 regularization strength
learning_rate = 0.001  # Example learning rate

# Build the model with the chosen hyperparameters
model = tf.keras.Sequential([
    tf.keras.layers.Embedding(
        input_dim=vocab_size,
        output_dim=embedding_dim,
        input_length=sequence_length
    ),
    tf.keras.layers.Conv1D(
        filters=filters,
        kernel_size=kernel_size,
        activation='relu'
    ),
    tf.keras.layers.GlobalMaxPooling1D(),
    tf.keras.layers.Dense(
        units=dense_units,
        activation='relu',
        kernel_regularizer=l2(l2_reg)
    ),
    tf.keras.layers.Dropout(dropout_rate),
    tf.keras.layers.Dense(
```

	Hyperparameter tuning optimizes the performance of a machine learning model by testing different combinations of parameters such as the number of neurons, learning rate, and more. The code uses Keras Tuner to automate this by defining a search space and testing two configurations (limited by max_trials=2). The model includes tunable parameters like the number of neurons in each dense layer and the learning rate. The tuner evaluates these configurations based on validation accuracy or loss, helping find the best combination for your dataset. After tuning, the best parameters are used to build and train the final model. This approach ensures efficient experimentation without manual trial and error.

Final Model	**Reasoning**

Keras	Keras was chosen as the final optimized approach for spooky author identification due to its simplicity, flexibility, and robust capabilities for building and fine-tuning neural network models. Keras provides an intuitive interface for designing and training deep learning architectures, enabling experimentation with various model designs and hyperparameter configurations. Integrated seamlessly with TensorFlow, Keras offers tools for scalability, distributed training, and deployment, ensuring efficient handling of complex tasks. Its ability to process text data and effectively manage categorical features makes it an excellent fit for author identification tasks. Furthermore, Keras's proven performance in natural language processing and text classification scenarios underscores its suitability for accurately identifying spooky authorship.