Model Optimization and Tuning Phase in Crude Oil Price Prediction

In the **Crude Oil Price Prediction** project, the **Model Optimization and Tuning Phase** aims to refine the LSTM model for better accuracy and robustness. This phase involves systematically improving the model's hyperparameters, architecture, and training process to ensure optimal performance on the task of predicting crude oil prices.

---

1. Hyperparameter Tuning

Hyperparameters directly affect the performance of the LSTM model. In this project, the following parameters were tuned:

Hyperparameters Considered:

- **Number of LSTM Layers**: Experimented with 1 to 3 layers to balance complexity and performance.
- **Number of Neurons per Layer**: Tested values like 32, 50, and 100 to determine the best capacity for capturing patterns.
- **Batch Size**: Experimented with sizes like 16, 32, and 64 to optimize memory usage and training speed.
- **Learning Rate**: Fine-tuned using values like 0.001, 0.0005, and 0.0001.
- **Epochs**: Evaluated performance with 10, 20, and 50 epochs to find the ideal training duration.
- **Dropout Rate**: Tested dropout rates (e.g., 0.2, 0.3, 0.5) to prevent overfitting.

Optimization Techniques:

- **Grid Search**: Manually tried combinations of hyperparameters to identify the best set.
- **Random Search**: Randomly selected combinations for quicker exploration.
- **Keras Tuner**: Used to automate the search for the best hyperparameters efficiently.

---

2. Model Architecture Optimization

The LSTM model's architecture was optimized to balance complexity and performance:

Steps Taken:

1. **Number of Layers**:
    - Experimented with single-layer and multi-layer LSTM networks.
    - Found that a 2-layer LSTM provided the best balance between complexity and performance.
2. **Neurons per Layer**:
    - Added 50 neurons in each LSTM layer after testing with different neuron counts.
3. **Output Layer**:
    - Used a single dense layer with linear activation for regression.

Architecture Adjustments:

- Incorporated **Dropout Layers**: Added after LSTM layers to reduce overfitting.
- Tuned **Activation Functions**: Used tanh in LSTM layers for better gradient flow.

---

3. Loss Function and Optimizer Tuning

Loss Function:

- **Mean Squared Error (MSE)**: Selected as the primary loss function due to its suitability for regression tasks.
- Experimented with **Mean Absolute Error (MAE)** for comparison.

Optimizer:

- Chose **Adam Optimizer** for its efficiency in handling non-stationary data.
- Experimented with different learning rates to ensure smooth convergence.

---

4. Learning Rate Scheduling

- Applied **ReduceLROnPlateau** to dynamically reduce the learning rate if validation loss did not improve.

- Initial learning rate set to 0.001, reduced by a factor of 0.1 when the model plateaued.

---

5. Early Stopping

- Used **EarlyStopping** to monitor validation loss and stop training once improvement stagnated.
- Set a patience of 5 epochs to prevent overfitting while saving training time.

---

6. Cross-Validation

- Performed **k-fold cross-validation** on the training set to evaluate the model's consistency and generalization.
- Validated that the model performed well across multiple data splits.

---

7. Data Preprocessing Adjustments

- Experimented with sliding window sizes of n_steps = 5, 10, 15 to find the most effective window length.
- Normalized data using **Min-Max Scaling**, ensuring the scale matched training and testing data.

---

8. Performance Monitoring

Metrics evaluated during tuning:

1. **Training and Validation Loss**:
    - Monitored to detect overfitting or underfitting.
2. **MSE and RMSE**:
    - Key metrics to evaluate prediction accuracy.
    - Targeted low RMSE values for better real-world applicability.
3. **Training Time**:
    - Ensured training was computationally efficient without sacrificing performance.

Final Optimized Model

Configuration:

- **Number of LSTM Layers**: 2
- **Neurons per Layer**: 50
- **Dropout Rate**: 0.2
- **Batch Size**: 32
- **Epochs**: 20 (with Early Stopping)
- **Learning Rate**: Adaptive (starting at 0.001, reduced when performance plateaued)

Performance:

- **Training Loss (MSE)**: [Value]
- **Validation Loss (MSE)**: [Value]
- **Test RMSE**: [Value]

Conclusion of Optimization Phase

The optimization phase significantly improved the model's ability to generalize and predict crude oil prices with higher accuracy. The tuned parameters and refined architecture provided a robust foundation for real-world application through the web interface.

Reference link

https://github.com/ANSHJAISWAR/Crude-oil-Price-Prediction/blob/main/Project%20Excutable%20files/CRUDE%20OIL%20PROCE%20PREDICTION.pdf