Performance Evaluation Report

Syed Niamul Kazbe Raiyan
01719087571
kazbeahmed@gmail.com

Introduction

In natural language processing (NLP), understanding the semantic relationship between pairs of sentences is a fundamental task with applications in question answering, text summarization, and information retrieval. This project focuses on developing a machine learning model to classify pairs of sentences into one of three categories: Contradiction, Neutral, or Entailment. The classification is based on the semantic relationship between the sentences, where:

- Contradiction (0): The sentences have opposite meanings.
- Neutral (1): The sentences are related but do not imply each other.
- Entailment (2): One sentence logically follows from the other.

The dataset provided, train.csv, contains labeled training data with the following columns:

- id: Unique identifier for each sentence pair.
- sentence1: The first sentence in the pair (Premise).
- sentence2: The second sentence in the pair (Hypothesis).
- label: The relationship classification (0, 1, or 2).

The goal is to preprocess the data, extract meaningful features, and train a machine learning model to accurately classify the semantic relationship between sentence pairs. The model will be evaluated using metrics such as accuracy, precision, recall, F1-score, and AUC-ROC.

Approach

Data Preprocessing:

- Tokenization: Split sentences into words or subwords.
- Lowercasing: Convert text to lowercase.
- Remove stop words, special characters, and punctuation.
- Stemming/Lemmatization: Normalize words to their root form.
- Feature Extraction: Convert text into numeric representations using techniques like TF-IDF, Word2Vec, or Transformer embeddings (e.g., BERT).

Model Development:

- Baseline Models: Train traditional machine learning models like Random Forest, Decision Tree, and XGBoost.
- Neural Networks: Implement a custom Artificial Neural Network (ANN).
- Advanced Models: Train LSTM/GRU models for sequence-based learning.
- Transformer-Based Models: Fine-tune BERT or XLM-R for contextual understanding.

Model Evaluation:

- Compute accuracy, precision, recall, and F1-score.
- Plot a confusion matrix to analyze misclassifications.
- Generate an AUC-ROC curve to evaluate classification performance.

Model Tuning and Optimization:

- Experiment with different optimizers (Adam, SGD, etc.) and activation functions.
- Adjust learning rate, batch size, and number of epochs.
- Use Grid Search or Random Search for hyperparameter tuning.

Evaluation Metrics

1. Accuracy: Measures the proportion of correctly classified instances.
2. Precision: Indicates the proportion of true positives among predicted positives.
3. Recall: Measures the proportion of true positives identified correctly.
4. F1-Score: Balances precision and recall, providing a single metric for model performance.
5. AUC-ROC: Evaluates the model's ability to distinguish between classes, with higher values indicating better performance.

Results

Model	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Random Forest	0.3464	0.3503	0.3464	0.3292	0.5008
Decision Tree	0.3581	0.3621	0.3581	0.3533	0.5247
XGBoost	0.3595	0.3573	0.3595	0.3477	0.5140
LSTM	0.3523	0.3728	0.3523	0.1973	0.5352
ANN	0.3200				

The evaluation results for the models show varying performance across different metrics. The XGBoost model achieved the highest accuracy of 0.3595, closely followed by the Decision Tree model with an accuracy of 0.3581. The LSTM model had an accuracy of 0.3523, while the Random Forest and ANN models performed slightly worse with accuracies of 0.3464 and 0.3200, respectively. In terms of precision, the LSTM model outperformed the others with a score of 0.3728, indicating fewer false positives, whereas the Random Forest had the lowest precision at 0.3503. The XGBoost model had the highest recall of 0.3595, suggesting it identified the most true positives, while the LSTM model had a slightly lower recall of 0.3523. The Decision Tree model achieved the best F1-score of 0.3533, indicating a better balance between precision and recall, whereas the LSTM model had the lowest F1-score of 0.1973, highlighting its poor balance between the two metrics. For the AUC-ROC score, the LSTM model performed the best with a score of 0.5352, indicating better class separation, while the Random Forest model had the lowest score of 0.5008, which is close to random guessing. Overall, the XGBoost and Decision Tree models show the

most balanced performance, while the LSTM model, despite its high AUC-ROC score, struggles with precision and recall balance. The ANN model performed the worst, suggesting a need for architectural improvements or more data. Further tuning and addressing potential class imbalance could help improve the performance of these models.

Key Observations

The XGBoost and Decision Tree models performed consistently well across most metrics, making them strong candidates for further tuning and deployment. The LSTM model, despite having the highest AUC-ROC score, struggled with precision and recall balance, as reflected in its low F1-score. The ANN model performed poorly across all metrics, suggesting a need for architectural improvements or additional training data. The Random Forest model showed moderate performance but had the lowest AUC-ROC score, indicating limited ability to distinguish between classes.

Recommendations

1. Focus on XGBoost and Decision Tree: These models showed the most balanced performance and should be prioritized for further tuning and optimization. Experiment with hyperparameter tuning and feature engineering to improve their performance.
2. Address LSTM's Imbalance: The LSTM model's low F1-score suggests potential issues with class imbalance or model architecture. Consider techniques like oversampling, undersampling, or using class weights to address imbalance.
3. Improve ANN Performance: The ANN model's poor performance indicates a need for architectural changes, such as adding more layers, using different activation functions, or increasing the number of epochs. Alternatively, consider using pre-trained embeddings or transformer-based models for better feature representation.
4. Explore Advanced Models: Experiment with advanced models like BERT, Transformer-based architectures, or ensemble methods to achieve better performance.
5. Address Class Imbalance: If class imbalance is present, apply techniques like SMOTE, class weighting, or data augmentation to improve model performance.

Conclusion

The XGBoost and Decision Tree models demonstrated the most balanced performance across all metrics, making them the best candidates for further development. The LSTM model, while showing promise in terms of AUC-ROC, requires improvements in precision and recall balance. The ANN model performed poorly and needs significant architectural changes or additional data. Addressing class imbalance and experimenting with advanced models could further enhance performance.