**Performance Evaluation Report**

**1. Introduction**

This report evaluates the performance of different models used for duplicate question detection in the Quora dataset. The evaluation includes data preprocessing, exploratory data analysis (EDA), model training, and performance comparison between Artificial Neural Networks (ANN), Long Short-Term Memory (LSTM), and baseline machine learning models like Logistic Regression and Support Vector Machines (SVM).

**2. Data Preprocessing and Exploratory Data Analysis (EDA)**

* The dataset was loaded and checked for missing values.
* Missing values in the 'question1' and 'question2' columns were handled by filling them with placeholder text.
* The distribution of duplicate vs. non-duplicate questions was analyzed, showing that **36.92%** of the questions were duplicates.
* Various text-processing steps were performed, including tokenization, stopword removal, stemming, and lemmatization.
* TF-IDF vectorization and Word2Vec embeddings were used for feature extraction.
* Character length and word count distributions were analyzed, indicating that duplicate questions tend to have similar structures.
* Correlation heatmaps and boxplots were used to visualize the relationship between different features.
* A WordCloud was generated for duplicate and non-duplicate questions, revealing common keywords.

**3. Model Implementations and Performance Analysis**

**3.1 Artificial Neural Network (ANN)**

* A feedforward ANN model was implemented using TensorFlow.
* The architecture included multiple dense layers with ReLU activation and dropout for regularization.
* The model was trained for **5 epochs** using binary cross-entropy loss and Adam optimizer.
* The evaluation results:
  + **Accuracy:** 79.36%
  + **Precision:** 86.19%
  + **Recall:** 86.97%
  + **F1-Score:** 86.58%
  + **AUC-ROC:** 95.95%
* Confusion matrix analysis showed that the ANN model struggled with some borderline cases, but overall, it performed well in distinguishing duplicate and non-duplicate questions.

**3.2 Long Short-Term Memory (LSTM) Model**

* A bidirectional LSTM model was trained on tokenized and padded sequences.
* The architecture consisted of an embedding layer, bidirectional LSTM layers, and fully connected dense layers.
* The model used **128 LSTM units** and was trained for **10 epochs**.
* Performance Metrics:
  + **Accuracy:** 92.16%
  + **Precision:** 88.14%
  + **Recall:** 91.00%
  + **F1-Score:** 89.55%
  + **AUC-ROC:** 96.97%
* The ROC curve and confusion matrix indicated that LSTM had a higher recall, making it suitable for applications where identifying duplicates is crucial.

**3.3 Baseline Models (Logistic Regression & SVM)**

* A logistic regression model was trained but struggled with convergence, achieving an accuracy of **63.08%**.
* Support Vector Machine (SVM) was trained using a TF-IDF feature matrix but was computationally expensive.
* Baseline model comparison:
  + **Logistic Regression:** 63.08% accuracy (not ideal for this problem)
  + **SVM:** Slightly better than Logistic Regression but required significant computational power.

**4. Hyperparameter Optimization**

* Grid search was conducted on LSTM with different optimizers (Adam, SGD), batch sizes (32, 64), and epochs (5, 10).
* The best configuration was **Adam optimizer, batch size 64, and 10 epochs**, achieving the highest validation accuracy.
* Hyperparameter tuning results indicated that higher batch sizes led to faster convergence, but too high values caused overfitting.
* Regularization techniques like dropout (0.5) were added to improve generalization.

**5. Additional Performance Analysis**

* Feature importance analysis using SHAP values indicated that word similarity and sentence structure played a crucial role in duplicate detection.
* AUC-ROC curves for all models showed that LSTM had the highest performance, followed by ANN and SVM.
* Model latency was tested, and LSTM had a slightly higher inference time compared to ANN but was still feasible for real-time applications.
* Comparative analysis with Transformer-based models (like BERT) was suggested for future improvements.

**6. Conclusion**

* LSTM outperformed ANN and traditional machine learning models in accuracy, recall, and F1-score.
* ANN showed competitive performance but was slightly behind LSTM.
* Logistic Regression and SVM models underperformed compared to deep learning models.
* Feature engineering and preprocessing played a significant role in improving model performance.
* Future work could explore transformer-based models (e.g., BERT) for further improvements.

**7. Recommendations**

* Use the LSTM model for deployment due to its superior performance.
* Further optimize hyperparameters for better generalization.
* Consider integrating attention mechanisms or transformer-based architectures for improved accuracy.
* Explore ensemble models to combine ANN and LSTM predictions for better overall performance.