Spam Ham Email Classification

Name: Ritik Patel

Abstract:

This project classifies email messages as Spam or Ham using both traditional machine-learning and deep-learning approaches. Two TF-IDF–based models Logistic Regression and Decision Tree were compared with a Deep Learning LSTM model using GloVe word embeddings.
Each model was trained on the same dataset to ensure fairness, and their performances were evaluated through Accuracy, Precision, Recall, and F1-score.
Logistic Regression achieved the best overall results (≈ 98 % accuracy), demonstrating that simpler linear models can outperform deeper networks on keyword-driven datasets.

Project Objective:

- Pre-process and clean a real-world spam/ham email dataset.
- Implement and train multiple models: Logistic Regression, Decision Tree, and LSTM GloVe.
- Compare and analyze model performance on identical data splits.
- Provide interpretation of results, explain model behavior, and highlight limitations.

Dataset Description:

Source: 190K+ Spam | Ham Email Dataset for Classification - Kaggle

Property	Description
Size	193852 emails
Lables	Ham (0), spam (1)
Balance	App. 52% Ham
Avg. Length	300-2000 characters (removed extreme outliers)

Data Cleaning and Preparation

1. Lower-cased text.
2. Removed URLs, HTML tags, punctuation, digits, and extra spaces.
3. Encoded labels (Spam = 1, Ham = 0).
4. Performed EDA to ensure balance and reasonable email length distribution.
5. Used a single train/test split (80 / 20) with random_state = 42 for consistency.

Model Implementation

1. Feature Engineering:

- Vectorizer: TfidfVectorizer(max_features = 50 000, ngram_range = (1, 2))
- Reasoning: Captures unigrams and bigrams to model keyword patterns like "free offer" or "click here."

2. Traditional Machine Learning Models:

a) Logistic Regression (TF-IDF)

Parameter	Value	Reason
max_iter	2000	Ensures convergence on high-dimensional sparse data.
class_weight	"balanced"	Adjusts for minor label imbalance

Logistic Regression learns from the learn boundaries between spam and ham based on term weights.

b) Decision Tree (TF-IDF)

Parameter	Value	Reason
max_depth	40	Limits overfitting
min_samples_leaf	2	Avoid splitting of tiny nodes
class_weight	"balanced"	Adjusts for minor label imbalance

Decision Tree learns from non-linear baseline capturing interactions between tokens.

3. Deep Learning Model GloVe + LSTM:

Component	Description
Embedding	Pre-trained GloVe (6B, 100 d) vectors loaded into Keras Embedding layer.
LSTM (64)	Captures sequential and contextual dependencies in email text.
Dropout (0.2)	Regularization to reduce overfitting.
Dense(1, sigmoid)	Binary output layer predicting spam probability.
Epochs	6
Batch size	256
Optimizer	Adam
Loss	Binary Cross-Entropy
EarlyStopping	Patience = 2 -> Stops when validation loss plateaus.

Increasing epochs allowed LSTM to learn more context from longer emails without overfitting.

Tools And Libraries Used:

- Python (Jupyter Notebook)
- Numpy and Pandas for data handling.
- Matplotlib and Seaborn for visualization
- Tensorflow and keras for deep learning model building
- Sckit-learn for TF-IDF and ML models

How To Run:

There are two ways to run:

1. Jupyter Notebook:
    - Launch Anaconda Navigator and launch Jupyter Notebook in it.
    - Open “spam_ham_email_classification.ipynb” in Jupyter.

- Setup the environment if needed. Instructions are already provided in "spam_ham_email_classification.ipynb" files' first cell.
- Then go to Run-> Run All Cells
- Ensure glove.6B.100d.txt and spam_Emails_data.csv are in the same directory.
- Results (including graphs and metrics) will display within the notebook.

2. Python Script:

- Open the "spam_ham_email_classification_rl.py" using "Edit with IDLE"-> "Edit with IDLE 3.10".
- Make sure all the essential libraries and dataset are installed. Instructions to download dataset and libraries are provided at the beginning of the script in the form of comments.
- Then go to Run-> Run Module
- The plots will be saved in the 'outputs/' folder.

Results Summary:

Model	Accuracy	Precision	Recall	F1
Logistic Regression (TF-IDF)	97.98 %	97.24 %	98.59 %	97.91 %
Decision Tree (TF-IDF)	94.63 %	91.74 %	97.54 %	94.55 %
Deep Learning (GloVe + LSTM)	93.80 %	92.92 %	94.19 %	93.55 %

Discussion and Analysis:

- Simplicity performed best because spam emails are highly keyword-driven rather than context-driven. TF-IDF features capture the statistical importance of these keywords across the corpus — for example, words like "free," "offer," "win," and "click" appear frequently in spam but rarely in legitimate messages. Logistic Regression, being a linear classifier, assigns high positive or negative weights to these indicative tokens, enabling it to separate spam and ham emails efficiently in a high-dimensional feature space. Its

simplicity, regularization, and ability to generalize from sparse data make it particularly strong for text classification problems.

- Decision Tree achieved strong recall but lower precision, meaning it detected most spam messages but also incorrectly marked some legitimate emails as spam. This behavior is typical for trees trained on sparse, high-dimensional text data. The model tends to memorize specific token combinations (like "free trial offer"), leading to overfitting on the training set. Although limiting the max_depth and min_samples_leaf parameters helped, Decision Trees lack the regularization strength of linear models and do not scale well when feature dimensions exceed tens of thousands (as in TF-IDF).
- LSTM + GloVe captured semantic relationships that the TF-IDF models could not — for instance, recognizing that "promotion" and "discount" carry similar meanings even if one word was unseen during training. However, the dataset's structure limited its advantage: most spam emails are short and rely on repetitive tokens rather than complex language patterns. In such scenarios, sequential models like LSTM cannot fully leverage their contextual memory, and the benefits of word embeddings become less significant. Additionally, training LSTMs on large corpora requires more epochs, tuning, and computational resources to outperform simpler baselines.
- Increasing epochs to 6 improved performance slightly, allowing the model to learn richer patterns, but beyond that, overfitting began to appear. Validation loss plateaued while training loss continued to decrease, showing that the model started memorizing the data rather than generalizing. Therefore, 6 epochs represented an optimal trade-off between learning depth and generalization.
- Overall, all models achieved over 93 % accuracy and F1 ≥ 0.93, proving the robustness of the preprocessing pipeline and the strength of text-based features for spam classification. The results also align with previous research findings — that simpler, linear models often outperform deep learning architectures on short, keyword-focused text datasets, while LSTMs become more useful for longer, context-rich text.

Critical Insights and Learning:

1. Simplicity can outperform complexity: Complex models like LSTMs require large, context-rich datasets to demonstrate their advantage. For structured datasets such as spam detection, where word frequency and presence are

strong indicators, simpler models like Logistic Regression often generalize better. This highlights a key principle in machine learning — choose the simplest model that fits the problem well.

2. Feature representation defines performance: TF-IDF focuses on statistical relevance (term frequency vs. document frequency), which directly suits spam detection since the problem relies on the presence of discriminative keywords. In contrast, pre-trained embeddings like GloVe encode semantic meaning but may dilute the discriminative power of specific spam-related terms if those embeddings were trained on general text (Wikipedia, news, etc.).
3. Model interpretability matters: Logistic Regression provides easily interpretable coefficients showing which words contribute most to the spam classification. This transparency is useful for practical applications like spam filters, where users or organizations need explainable decisions.
4. Training duration and overfitting trade-off: The experiment reinforced that longer training does not guarantee improvement. After six epochs, the LSTM's accuracy plateaued, and validation loss began to rise. Regularization techniques like dropout helped mitigate overfitting, but monitoring validation performance remains crucial for efficient training.
5. Domain-specific embeddings outperform general ones: Using GloVe trained on Wikipedia data is a limitation — spam emails use unique slang and promotional phrasing. A custom embedding trained on email or marketing corpora could enhance the LSTM's performance, as it would capture domainspecific semantics.
6. Reproducibility and randomness: Minor variations in metrics are expected due to random initialization of weights, data shuffling, and hardware-level parallel computations. Setting random seeds can improve reproducibility, but exact matching results may still vary slightly — a normal behavior in stochastic optimization.
7. Computational efficiency and scalability: Logistic Regression and Decision Tree models trained in seconds, while LSTM required longer training time and GPU acceleration. This experiment shows that deep learning models are computationally heavier and not always necessary for tasks where simpler models perform equally well.

Conclusion:

All three models successfully classified emails as Spam or Ham with high accuracy, confirming the reliability of the preprocessing and feature-engineering pipeline. The Logistic Regression model proved most effective (≈ 98 %) because it aligns well with the sparse, high-dimensional nature of TF-IDF features. Its linear decision boundary generalizes effectively for keyword-based datasets, where certain terms (e.g., “free,” “offer,” “click”) are strong indicators of spam.

The Decision Tree model achieved strong recall but slightly lower precision, showing that while it can capture non-linear patterns, it tends to overfit on rare token combinations. The GloVe + LSTM model demonstrated solid semantic understanding and contextual learning, but its advantage was limited by the short, repetitive structure of emails and by the fact that the GloVe embeddings were pretrained on general Wikipedia and news data rather than email text.

Overall, the results reinforce a key principle in machine learning: model selection should depend on data characteristics rather than model complexity. For keywordrich tasks like spam detection, simpler linear models can outperform deeper architectures while being far more computationally efficient.

Reflection:

Working on this project helped me understand how data representation (TF-IDF vs. embeddings) affects model performance and why deep learning is not always superior for simple NLP tasks. I also learned to handle imbalanced datasets, apply consistent train/test splits, and interpret evaluation metrics critically. The experience strengthened my understanding of machine-learning workflow from data cleaning to model interpretation and comparison.

References:

1. Kaggle Dataset – 190K + Spam | Ham Email Dataset for Classification
2. Stanford NLP – GloVe: Global Vectors for Word Representation
3. Scikit-learn Documentation – LogisticRegression, DecisionTreeClassifier, TfidfVectorizer
4. TensorFlow Keras API – Tokenizer, pad_sequences, LSTM, Embedding
5. Medium Article – “Using Pre-Trained GloVe Vectors in Python”