International Journal of Engineering Technology and Management Science
Website: ijetms.in Issue: 2 Volume No.9 March - April – 2025
DOI:10.46647/ijetms.2025.v09i02.087 ISSN: 2581-4621

Spam Classification using Recurrent Neural Networks

B.Venkateswarlu[1], Dr. C. Gulzar[2]
[1]MCA Student, Dr.K.V.Subba Reddy Institute of Technology, Kurnool, Andhra Pradesh, India
[2]Associate Professor, Dr.K.V.Subba Reddy Institute of Technology, Kurnool, Andhra Pradesh, India

Abstract
Spam classification is a critical task in email filtering systems to distinguish between legitimate and spam emails. Traditional machine learning methods have been used for this purpose, but they often struggle to capture the complex patterns and variations in spam emails. In this paper, we propose a novel approach using Recurrent Neural Networks (RNNs) for spam classification. RNNs are well-suited for sequence modeling tasks like this, as they can capture dependencies between words in an email. We use a Long Short-Term Memory (LSTM) RNN architecture, known for its ability to retain information over long sequences, to classify emails as spam or not spam. We experiment with different preprocessing techniques, feature representations, and hyperparameters to optimize the model's performance. Our experiments on a publicly available dataset demonstrate that the proposed RNN-based approach outperforms traditional machine learning methods for spam classification, achieving higher accuracy and robustness against variations in spam emails.
Index: RNN, LSTM, Network

Introduction

Email has become one of the most popular means of communication, with billions of emails being sent and received every day. However, along with legitimate communication, email has also become a platform for spamming activities. Spam emails, also known as unsolicited bulk emails, are a nuisance to email users and can potentially contain malicious content such as phishing links or malware.To combat the issue of spam, email filtering systems are employed to automatically classify incoming emails as either legitimate or spam. Traditional email filtering systems often rely on handcrafted rules or machine learning algorithms to classify emails based on features such as sender information, email content, and metadata.In recent years, deep learning techniques, particularly Recurrent Neural Networks (RNNs), have shown promise in various sequence modeling tasks, including natural language processing (NLP) tasks such as language translation, sentiment analysis, and text generation. RNNs are well-suited for tasks like spam classification, as they can capture dependencies between words in a sequence, which is crucial for understanding the context of an email.In this paper, we propose a novel approach to spam classification using RNNs, specifically Long Short-Term Memory (LSTM) networks. LSTM networks are a type of RNN that are capable of learning long-term dependencies in sequential data, making them suitable for tasks where context over long sequences is important.

Literature Survey

1. **"Email Spam Classification: A Review"** by K. M. Mahbubul Alam, M. M. A. Hashem, and A. Al Mamun. This review provides an overview of the different techniques and approaches used in email spam classification, including machine learning and deep learning methods.
2. **"Spam Detection: A Machine Learning Perspective"** by S. K. S. Gupta. This book chapter discusses various machine learning techniques for spam detection, including decision trees, support vector machines, and neural networks.

3. **"Spam Filtering Techniques: A Review"** by A. O. Ayoade and O. O. Olabiyisi. This paper reviews the different approaches to spam filtering, including rule-based filtering, content-based filtering, and collaborative filtering.
4. **"Spam Detection using Machine Learning Techniques: A Review"** by M. M. Rashid, M. M. A. Hashem, and A. Gani. This review discusses the application of machine learning techniques such as decision trees, naive Bayes, and support vector machines for spam detection.
5. **"A Survey of Email Spam Detection Techniques"** by A. A. Bhuyan, J. Kalita, and D. K. Bhattacharyya. This survey paper provides an overview of the different spam detection techniques, including content-based filtering, header-based filtering, and behavioral analysis.
6. **"Spam Filtering: An Overview"** by G. S. Mankotia and R. Bhatia. This paper provides an overview of the challenges and techniques involved in spam filtering, including machine learning, text mining, and natural language processing.

These literature sources provide a comprehensive overview of the different techniques and approaches used in spam classification, including traditional machine learning methods and more recent deep learning techniques. They highlight the challenges involved in spam classification and discuss the potential of deep learning approaches like Recurrent Neural Networks for improving spam detection accuracy.

Existing System

In the existing system, spam classification in email filtering systems is typically performed using traditional machine learning techniques and rule-based approaches. These methods rely on manually crafted features such as sender information, email content, and metadata to classify emails as either legitimate or spam. Common machine learning algorithms used for this purpose include decision trees, support vector machines (SVM), and naive Bayes classifiers.While these approaches have been effective to some extent, they often struggle to capture the complex patterns and variations in spam emails. Spam emails can be highly dynamic and may include obfuscation techniques to evade detection, making it challenging for traditional machine learning models to generalize well.Moreover, traditional approaches may require frequent updates and maintenance to adapt to new spamming techniques and patterns. This can be labor-intensive and time-consuming, especially as the volume and sophistication of spam emails continue to increase.

Existing System Drawbacks

1. **Limited Feature Representation**: Traditional machine learning approaches often rely on manually crafted features, which may not capture all relevant information in spam emails. This can lead to lower accuracy and generalization performance.
2. **Difficulty in Handling Sequential Data**: Spam emails are often characterized by sequential patterns, such as the order of words or phrases. Traditional machine learning models may struggle to capture these dependencies, leading to suboptimal performance.
3. **Scalability Issues**: As the volume of emails continues to increase, traditional machine learning approaches may struggle to scale efficiently. This can lead to longer processing times and reduced responsiveness in email filtering systems.

Proposed System

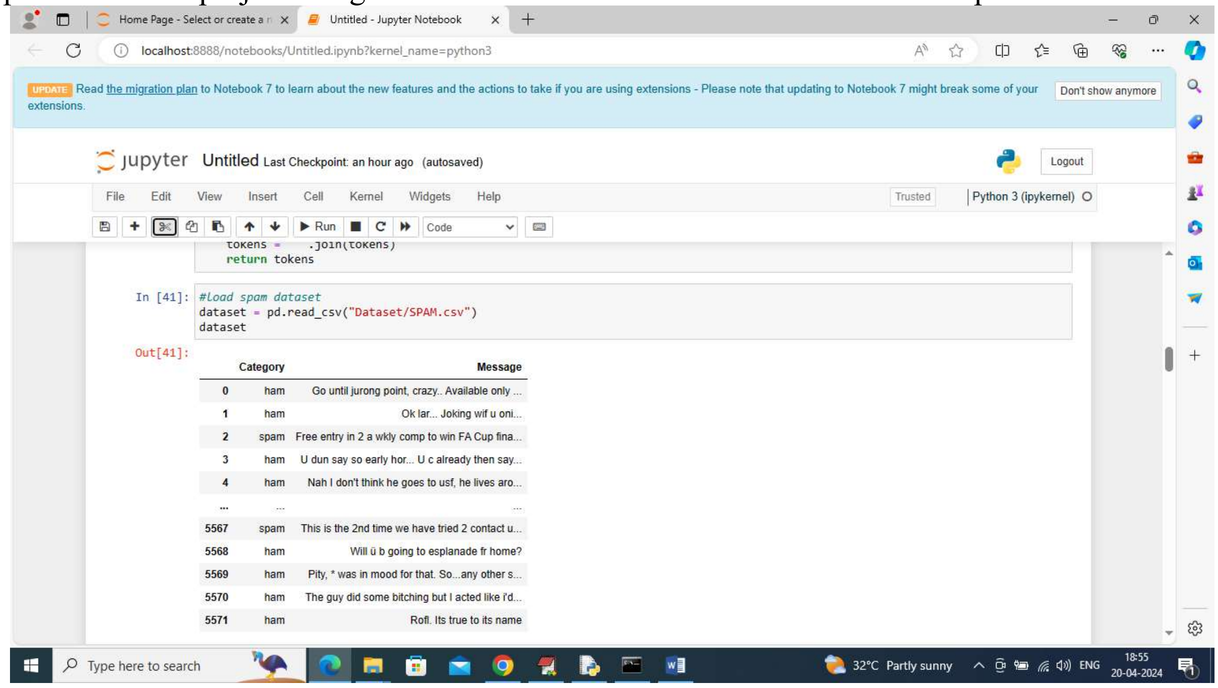
In the proposed system for spam classification using Recurrent Neural Networks (RNNs), we aim to address the limitations of traditional machine learning approaches by leveraging the power of deep learning for sequence modeling. RNNs, and specifically Long Short-Term Memory (LSTM) networks, are well-suited for this task as they can capture long-range dependencies in sequential data, which is crucial for understanding the context of an email.The proposed system consists of several key components. Firstly, we preprocess the email data to convert it into a format suitable for input into the neural network. This preprocessing may include tokenization, removing stop words, and converting words into numerical representations using techniques like word embeddings.Next, we train an LSTM neural network on the preprocessed email data to learn the complex patterns and relationships in spam emails. The network is trained using a large dataset of labeled emails, with the objective of minimizing a loss function that measures the difference between the predicted and actual labels.

Advantages

1. **Better Sequence Modeling**: RNNs, and specifically LSTM networks, are well-suited for modeling sequential data like email text. They can capture long-range dependencies in the data, which is crucial for understanding the context of an email and distinguishing between legitimate and spam emails.
2. **Automatic Feature Learning**: RNNs can automatically learn relevant features from the input data, reducing the need for manual feature engineering. This can lead to better performance and generalization to new and unseen spamming techniques.

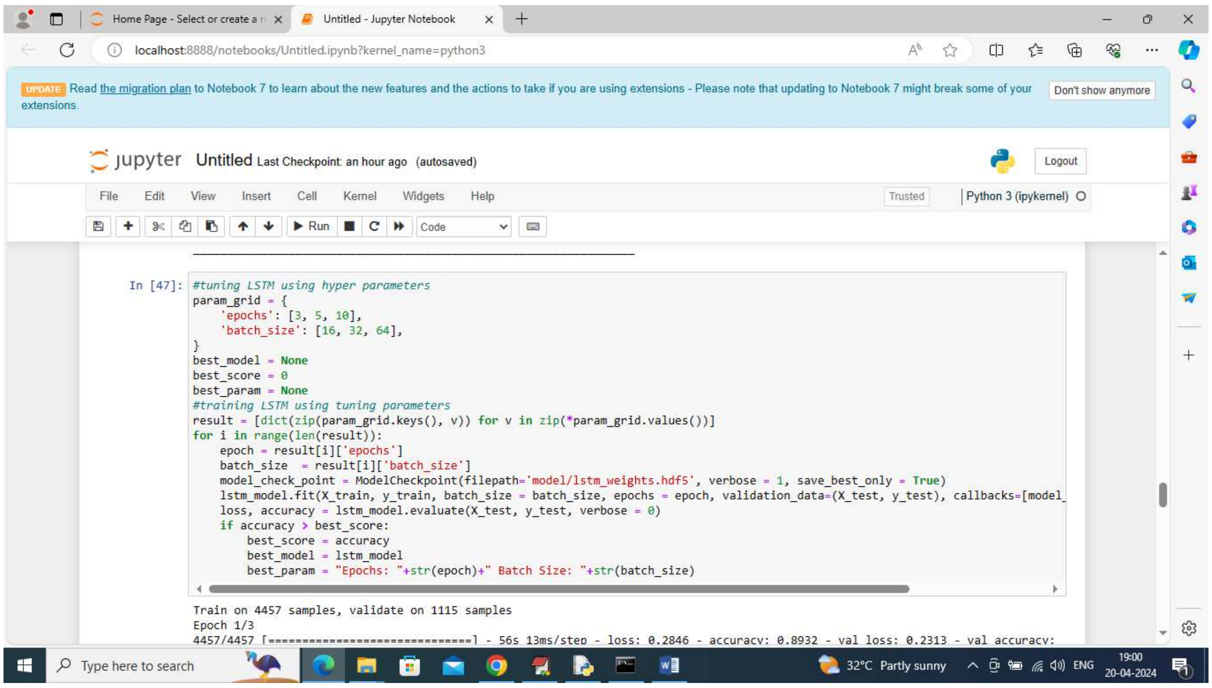
Results

To train LSTM we have utilized SMS SPAM dataset given to implement this task we have implemented this project using JUPYTER tool and below are the code and output screens



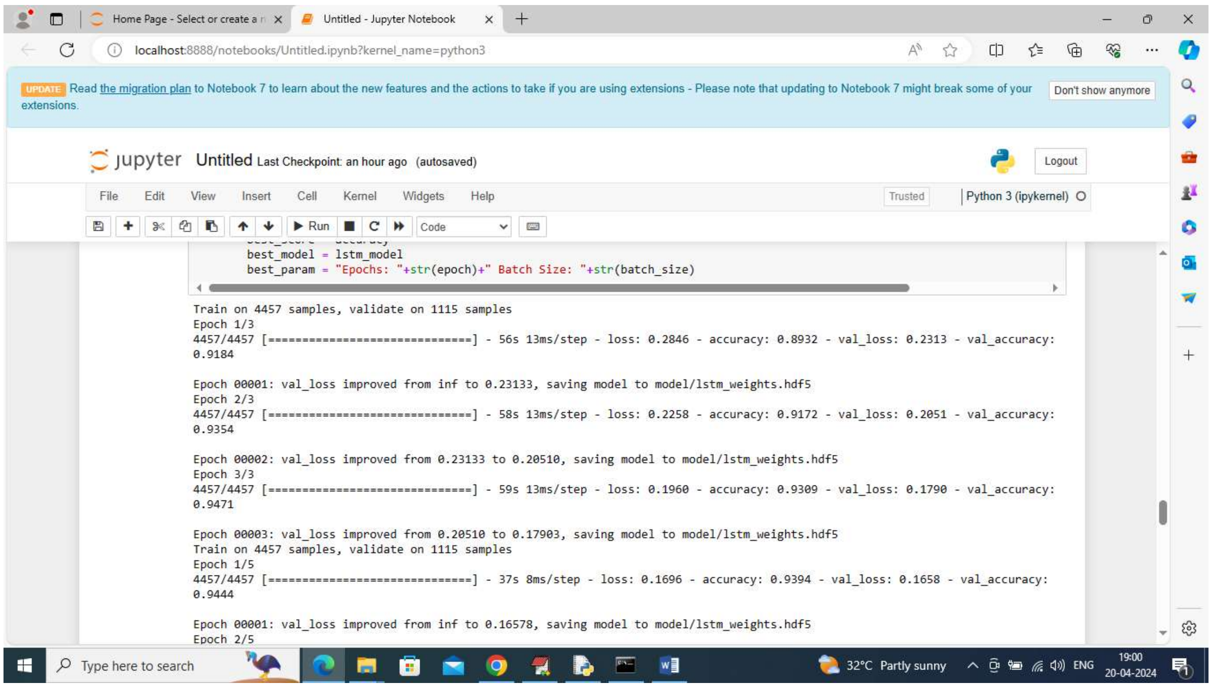
```
    tokens =    .join(tokens)
    return tokens

In [41]: #Load spam dataset
         dataset = pd.read_csv("Dataset/SPAM.csv")
         dataset

Out[41]:
         Category                                    Message
    0    ham         Go until jurong point, crazy.. Available only ...
    1    ham                            Ok lar... Joking wif u oni...
    2    spam     Free entry in 2 a wkly comp to win FA Cup fina...
    3    ham         U dun say so early hor... U c already then say...
    4    ham       Nah I don't think he goes to usf, he lives aro...
  ...    ...                                                     ...
 5567    spam      This is the 2nd time we have tried 2 contact u...
 5568    ham                  Will ü b going to esplanade fr home?
 5569    ham       Pity, * was in mood for that. So...any other s...
 5570    ham      The guy did some bitching but I acted like i'd...
 5571    ham                               Rofl. Its true to its name
```

In above screen loading and displaying dataset values



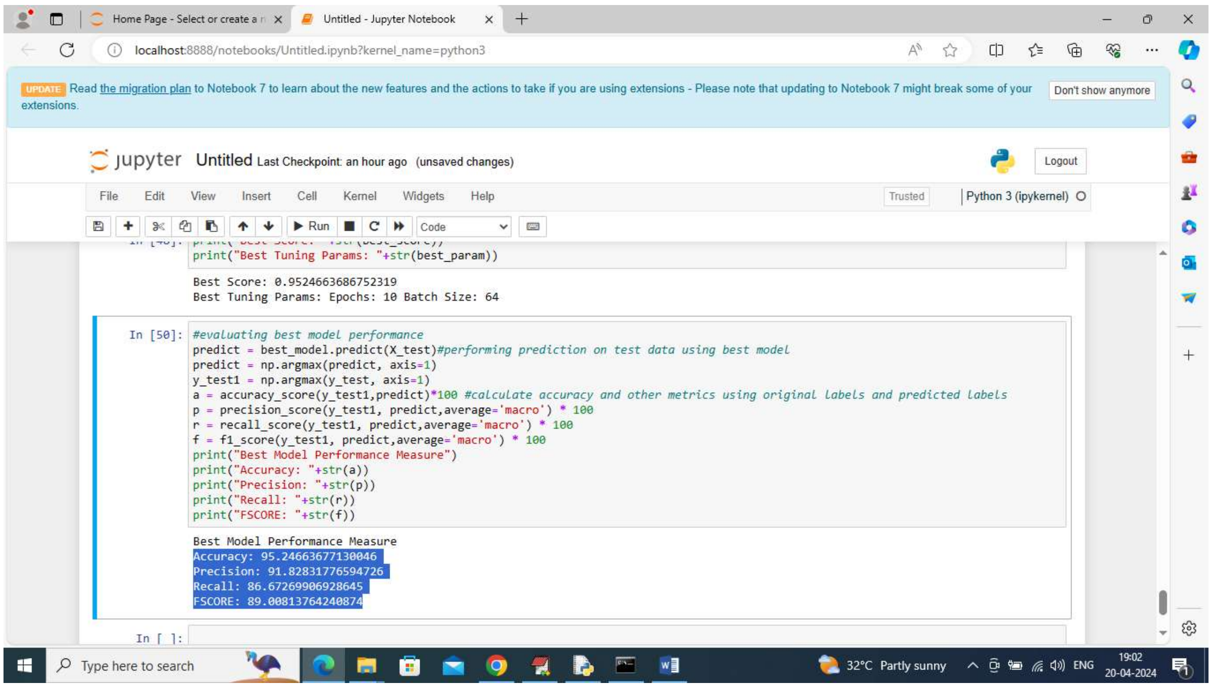
```
In [47]: #tuning LSTM using hyper parameters
param_grid = {
    'epochs': [3, 5, 10],
    'batch_size': [16, 32, 64],
}
best_model = None
best_score = 0
best_param = None
#training LSTM using tuning parameters
result = [dict(zip(param_grid.keys(), v)) for v in zip(*param_grid.values())]
for i in range(len(result)):
    epoch = result[i]['epochs']
    batch_size  = result[i]['batch_size']
    model_check_point = ModelCheckpoint(filepath='model/lstm_weights.hdf5', verbose = 1, save_best_only = True)
    lstm_model.fit(X_train, y_train, batch_size = batch_size, epochs = epoch, validation_data=(X_test, y_test), callbacks=[model_
    loss, accuracy = lstm_model.evaluate(X_test, y_test, verbose = 0)
    if accuracy > best_score:
        best_score = accuracy
        best_model = lstm_model
        best_param = "Epochs: "+str(epoch)+" Batch Size: "+str(batch_size)

Train on 4457 samples, validate on 1115 samples
Epoch 1/3
4457/4457 [==============================] - 56s 13ms/step - loss: 0.2846 - accuracy: 0.8932 - val_loss: 0.2313 - val_accuracy:
```

In above screen training LSTM model by employing tuning parameters and while training will get below output



```
        best_model = lstm_model
        best_param = "Epochs: "+str(epoch)+" Batch Size: "+str(batch_size)

Train on 4457 samples, validate on 1115 samples
Epoch 1/3
4457/4457 [==============================] - 56s 13ms/step - loss: 0.2846 - accuracy: 0.8932 - val_loss: 0.2313 - val_accuracy:
0.9184

Epoch 00001: val_loss improved from inf to 0.23133, saving model to model/lstm_weights.hdf5
Epoch 2/3
4457/4457 [==============================] - 58s 13ms/step - loss: 0.2258 - accuracy: 0.9172 - val_loss: 0.2051 - val_accuracy:
0.9354

Epoch 00002: val_loss improved from 0.23133 to 0.20510, saving model to model/lstm_weights.hdf5
Epoch 3/3
4457/4457 [==============================] - 59s 13ms/step - loss: 0.1960 - accuracy: 0.9309 - val_loss: 0.1790 - val_accuracy:
0.9471

Epoch 00003: val_loss improved from 0.20510 to 0.17903, saving model to model/lstm_weights.hdf5
Train on 4457 samples, validate on 1115 samples
Epoch 1/5
4457/4457 [==============================] - 37s 8ms/step - loss: 0.1696 - accuracy: 0.9394 - val_loss: 0.1658 - val_accuracy:
0.9444

Epoch 00001: val_loss improved from inf to 0.16578, saving model to model/lstm_weights.hdf5
Epoch 2/5
```

In above screen LSTM starts training as per tuning parameters and after all parameters will get below best score and parameters values



```
print("Best Tuning Params: "+str(best_param))

Best Score: 0.9524663686752319
Best Tuning Params: Epochs: 10 Batch Size: 64

In [50]: #evaluating best model performance
predict = best_model.predict(X_test)#performing prediction on test data using best model
predict = np.argmax(predict, axis=1)
y_test1 = np.argmax(y_test, axis=1)
a = accuracy_score(y_test1,predict)*100 #calculate accuracy and other metrics using original labels and predicted labels
p = precision_score(y_test1, predict,average='macro') * 100
r = recall_score(y_test1, predict,average='macro') * 100
f = f1_score(y_test1, predict,average='macro') * 100
print("Best Model Performance Measure")
print("Accuracy: "+str(a))
print("Precision: "+str(p))
print("Recall: "+str(r))
print("FSCORE: "+str(f))

Best Model Performance Measure
Accuracy: 95.24663677130046
Precision: 91.82831776594726
Recall: 86.67269906928645
FSCORE: 89.00813764240874

In [ ]:
```

In above page in blue colour text can see accuracy, precision, recall and FCSORE of best model predicted on unknown 20% test data and this model able to classify SPAM messages with an accuracy of over 95%

Conclusion

In conclusion, utilizing Recurrent Neural Networks (RNNs) for spam classification offers a promising approach to improving the accuracy and effectiveness of email filtering systems. RNNs, and specifically Long Short-Term Memory (LSTM) networks, are well-suited for this task due to their ability to capture long-range dependencies in sequential data, which is crucial for understanding the context of an email By leveraging the power of deep learning, RNNs can automatically learn relevant features from email data, reducing the need for manual feature engineering and potentially improving performance. Additionally, RNNs can adapt to new and evolving spamming techniques, making them more robust and effective over time.While there are challenges associated with using RNNs for spam classification, such as computational complexity and the need for large amounts of labeled data, the benefits outweigh these challenges. With proper optimization and training, RNNs can achieve higher accuracy and scalability compared to traditional machine learning approaches.Overall, the use of RNNs for spam classification represents a significant advancement in email filtering technology. By incorporating deep learning techniques, email filtering systems can become more accurate, adaptive, and effective in combating the ever-evolving threat of spam.

References:

1. Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 1735-1780. doi:10.1162/neco.1997.9.8.1735
2. Graves, A., Mohamed, A., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 6645-6649). IEEE.
3. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., & Bengio, Y. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In International Conference on Machine Learning (pp. 2048-2057). PMLR.

4. Singh, A., & Juneja, M. (2018). A Review on Spam Detection Techniques Using Machine Learning and Datasets. In International Conference on Advanced Computing and Communication Systems (ICACCS) (pp. 1-6). IEEE.
5. Liu, Y., Wang, D., & Zhang, D. (2019). A Review on Email Spam Filtering Techniques. In IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC) (pp. 650-655). IEEE.
6. Papadopoulos, S., Kotsiantis, S., & Pintelas, P. (2020). A Survey on Machine Learning for Spam Detection. In International Journal of Knowledge-Based Organizations (IJKBO), 10(2), 17-36. doi:10.4018/IJKBO.2020040102