https://doi.org/10.31871/IJNTR.7.4.11
International Journal of New Technology and Research (IJNTR)
ISSN: 2454-4116, Volume-7, Issue-4, April 2021 Pages 35-40

An Collaborative and Early Detection of Email Spam Using Multitask Learning

Hariharan N, Kamaraj G, Ramanuja Babu R D

***Abstract*— Currently, E-mail is one of the most important methods of communication. The increasing of spam e-mails causes traffic congestion, decreasing productivity, which has become a serious problem for our society. The problem of Email spam has grown significantly over the past few years. It is not just a nuisance for users but also it is damaging for those who fall for scams and other attacks.The complexity intensification of Email spamming techniques which are advancing from traditional spamming (direct spamming) techniques to a more scalable and indirect approach of botnets for distributing Email spam message is the major reason for it.The aim of this research is to find an effective solution to filter possible spam e-mails.In this paper a hybrid solution which uses machine learning algorithms like Deep Neural Network, Convolution Neural Network are used to produce an improved result and efficiency compared to existing system.The experimental results show that the proposed algorithm has 92.8% accuracy.**

***Index Terms*— Deep Neural Network, Convolution Neural Network, Botnets.**

I. INTRODUCTION

Email spam, also referred to as junk email, is unsolicited messages sent in bulk by email (spamming).Based on a recent Internet Security Threat Report, published by Symantec Corporation , in 2012 and 2013, the estimated Global Email Spam Volume per day is about 30 and 29 Billion, and the global average spam rate was 69% and 66%, respectfully.The real cost of spam emails is more than one can imagine.Therefore, spam e-mail filtering is an important and meaningful topic. Spam email filtering methods can be categorized into several categories. For example, blacklists and whitelists, IP blocking, header-based filtering and content-based filtering approach. Blacklists, whitelists and IP blocking are relatively the fast way, as compared to other detection approaches, to identify spammers. However, blacklists and whitelists or IP blocking have potential issues that the spammer could change current email account(s) or one IP to another one, in order to escape detection. In this case, normal methods could not easily filter these spam emails. Poor performance and low accuracy are the result of using these approaches.

This paper proposes a content based spam email filtering approach based on . The proposed algorithm contains two main phases: one is the training phase and the other is classification phase.Individual users' emails' are extracted from training datasets in the training phase.

A spam and ham keywords corpus was built after the collection of email content which was used to compare with those keywords that were extracted from individual users' email. Before comparing those extracted words with the spam and ham keywords corpus, in order to improve the accuracy and handle more possible spam techniques, some content processing methods are applied that ; for example, HTML tags removing, insignificant words, and irrelevant words filtering.In order to improve the accuracy of classification,beside the above procedure a special scheme for keyword detection is applied. According to the experimental results, the proposed approach has 92.8% accuracy rate.

The rest of the paper is organized as follows. In section 2, Literature Review. Section 3 describes the methods for spam email filtering. Section 4 presents the experimental results of this work. Section 5 deals with discussion.Section 6 discusses the conclusion of this paper, the future work in spam filtering.Finally,Section 7 lists the references of this paper.

II. LITERATURE REVIEW

Hybrid Water Cycle Optimization Algorithm With Simulated Annealing for Spam E-mail Detection.

The methodology used in this study consists of groundwork, induction, improvement, evaluation and comparison quality. The cross-validation was used for training and validation dataset and seven datasets were employed in testing the spam classification proposed.Content based filtering and IP Blacklisting solutions are employed.But factual origins cannot be decontaminated, depends on extracted email features from eml file and involves high Computational cost.

A Study of the Personalization of Spam Content using Facebook Public Information

When In this paper, a passive or reactive approaches is employed.Feedback mechanism to distinguish between a proper output and an ambiguous output is utilized.And use of interleaved hybridization generated better optimal features.But higher prediction complexity with higher dimensions and very calculation intensive while training the model.

Hariharan N, Department of Computer Science, Thiagarajar College Of Engineering, Madurai
Kamaraj G, Department of Computer Science, Thiagarajar College Of Engineering, Madurai
Ramanuja Babu R, Department of Computer Science, Thiagarajar College Of Engineering, Madurai

Email Classification Research Trends: Review and Open Issue

This study comprehensively reviews articles on email classification published in 2006– 2016 by exploiting the methodological decision analysis in five aspects, namely, email classification application areas, datasets used in each application area, feature space utilized in each application area, email classification techniques, and use of performance measures.Based on the study , the content based email classification model is considered to be a feasible one.But require a plethora of Email spam messages to accurately identify botnet spam which may not be a practical solution,does not support a huge number of features and does not improve the accuracy of feature selection.

A Comprehensive Study of Email Spam Botnet Detection

This paper discusses the sources and architectures used by the spamming botnets for sending massive amount of email spam. It also presents the detailed chronicles of spamming botnets which systematically describes the timeline of events and notable occurrences in the advancement of these spamming botnets. A comprehensive analysis of Email spamming botnet detection techniques is performed and categorize them according to both their nature of defense and method of detection.From the analysis it is found that greater accuracy is achieved only when the dataset is huge.

III. METHODOLOGY

Dataset Preparation:

Here the dataset file consists of three columns where the first two columns are namely label and text which are the independent variables and the spam status which is the dependent variable. The spam status depends on the value of label and text. The dataset file consists of nearly SIX thousand rows of data.

	label	text	spam	length
0	ham	Go until jurong point, crazy.. Available only ...	0	111
1	ham	Ok lar... Joking wif u oni...	0	29
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	1	155
3	ham	U dun say so early hor... U c already then say...	0	49
4	ham	Nah I don't think he goes to usf, he lives aro...	0	61
5	spam	FreeMsg Hey there darling it's been 3 week's n...	1	148
6	ham	Even my brother is not like to speak with me. ...	0	77
7	ham	As per your request 'Melle Melle (Oru Minnamin...	0	160
8	spam	WINNER!! As a valued network customer you have...	1	158
9	spam	Had your mobile 11 months or more? U R entitle...	1	154

Data pre-processing:

Removing stop words from the text which is in the message column of datasets.Stop words are available in abundance in any human language. We can do more focus on the important information instead of doing it on less level information by removing those irrelevant words.

Train Test Split:

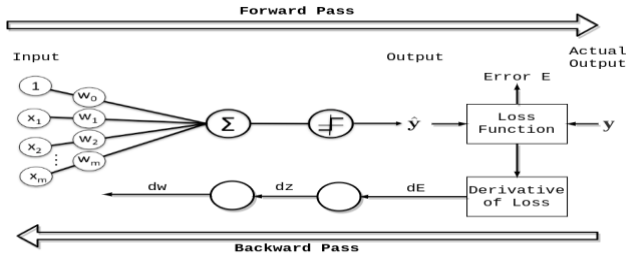
```
In [6]: pad = 'post'
        max_len = 25
        embedding_size = 100
        batch_size = 20
        sequences = pad_sequences(sequences, maxlen=max_len, padding=pad, truncating=pad)
        sequences.shape

        X_train, X_test, y_train, y_test = train_test_split(sequences, y, test_size = 0.2, random_state= 0)

In [7]: model = Sequential()
        model.add(Embedding(input_dim=vocab_size, output_dim=embedding_size, input_length=max_len))
        model.add(Dropout(0.8))
        model.add(LSTM(140, return_sequences=False))
        model.add(Dropout(0.8))
        model.add(Dense(1, activation='sigmoid', name='Classification'))
        model.summary()
```

Importing necessary packages for data model creation and training and testing.

Data model creation using Deep Neural Network algorithm

N number of epoch creation for improving performance :

An epoch is the process of making the neural network to practice with all the data which is separated for training purpose for a single cycle. In this single epoch we used all the data exactly once. Single pass is considered as either in a forward way or backward way. This process is mainly created for one or more cycles, where we used this as a part of dataset to train our proposed neural network.



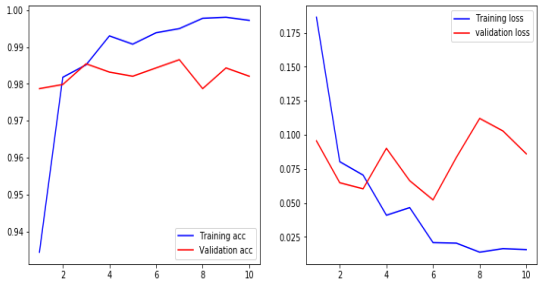
Why Epoch?

Epoch to train our model

```
In [15]: n_epochs = 10
         results = model.fit(X_train, y_train, epochs=n_epochs, batch_size=batch_size, validation_split=0.2, verbose=1)

Train on 3568 samples, validate on 892 samples
Epoch 1/10
3568/3568 [==============================] - 25s 7ms/step - loss: 3.4495e-05 - accuracy: 1.0000 - val_loss: 9.7276e-07 - val_ac
curacy: 1.0000
Epoch 2/10
3568/3568 [==============================] - 24s 7ms/step - loss: 2.6669e-05 - accuracy: 1.0000 - val_loss: 6.2979e-07 - val_ac
curacy: 1.0000
Epoch 3/10
3568/3568 [==============================] - 23s 6ms/step - loss: 2.1317e-05 - accuracy: 1.0000 - val_loss: 4.4765e-07 - val_ac
curacy: 1.0000
Epoch 4/10
3568/3568 [==============================] - 24s 7ms/step - loss: 1.6358e-05 - accuracy: 1.0000 - val_loss: 3.3641e-07 - val_ac
curacy: 1.0000
Epoch 5/10
3568/3568 [==============================] - 23s 6ms/step - loss: 1.2568e-05 - accuracy: 1.0000 - val_loss: 2.6120e-07 - val_ac
curacy: 1.0000
Epoch 6/10
3568/3568 [==============================] - 24s 7ms/step - loss: 9.3257e-06 - accuracy: 1.0000 - val_loss: 2.1335e-07 - val_ac
curacy: 1.0000
Epoch 7/10
3568/3568 [==============================] - 24s 7ms/step - loss: 1.0213e-05 - accuracy: 1.0000 - val_loss: 1.6826e-07 - val_ac
curacy: 1.0000
Epoch 8/10
3568/3568 [==============================] - 24s 7ms/step - loss: 1.0144e-05 - accuracy: 1.0000 - val_loss: 1.3072e-07 - val_ac
curacy: 1.0000
Epoch 9/10
3568/3568 [==============================] - 23s 6ms/step - loss: 6.6946e-06 - accuracy: 1.0000 - val_loss: 1.0837e-07 - val_ac
curacy: 1.0000
```

No.of sample data taken for training : 3568
No.of sample data taken for testing : 892
No.of samples taken for training : 4460 (80%)

Training Accuracy/Loss and validation Accuracy/Loss:



Fetching realtime messages from mail:

Subject: Taylor Swift, Porcupine Tree or Black Sabbath? Pick one
From: "Anoushka Mehra" <anoushka.mehra@hackerearth.com>
https://www.surveymonkey.com/r/KB33CJL

Which one of these would you listen to while coding?

I'm running the Developer Survey 2021 from HackerEarth; this is one of our questions: What kind of music helps you code bette
r?

For me, it's Steven Wilson all the way - not just his tracks from PT, but even some of his music from Blackfield. It's amazin
g!

But I digress - I'm writing to you to ask for your help. I'm running a survey for developers. And I'd love your inputs.

If you can help me fill this up, I'll send you a report with inputs from over 30K developers across the globe - you'll get to l
earn more about your global peers: their salaries, challenges, 2021 goals, learning plans and skills (and favorite music).

So go ahead, help me with 12 minutes of your time :) I will be super grateful .

Best

Anoushka Mehra

Survey Link https://www.surveymonkey.com/r/KB33CJL

Ohh...before I forget, 10 lucky developers get a chance to win 50$ Amazon vouchers!

Real time message 1

Subject: Critical security alert
From: Google <no-reply@accounts.google.com>
[image: Google]
Sign-in attempt was blocked

kamaraj1732000@gmail.com
Someone just used your password to try to sign in to your account from a
non-Google app. Google blocked them, but you should check what happened.
Review your account activity to make sure no one else has access.
Check activity
<https://accounts.google.com/AccountChooser?Email=kamaraj1732000@gmail.com&continue=https://myaccount.google.com/alert/nt/16191
00140000?rfn%3D27%26rfnc%3D1%26eid%3D-6601883628501479423%26et%3D0%26anexp%3Dnret-fa>
You can also see security activity at
https://myaccount.google.com/notifications
You received this email to let you know about important changes to your
Google Account and services.
© 2021 Google LLC, 1600 Amphitheatre Parkway, Mountain View, CA 94043, USA

Real time message 2

IV. RESULTS

1. Text Processing

Data usually comes from the variety of many sources and often in the different formats. For this purpose, transforming our data is more essential. Eventhough, this transformation is not a easy process, text data may have contained redundant and repetitive words in itself. So, processing the text data is our first and foremost steps towards the solution.

Those fundamental steps involved in the text preprocessing are:

A. Cleaning the raw data

B. Tokenizing the cleaned data

A. Cleaning the Raw Data

This process involves the in the deletion of words or characters that does not add any value in main information. Some of the standard cleaning steps are listed as below:

- Lowering case
- Removal of special characters
- Removal of stopwords
- Removal of hyperlinks
- Removal of numbers
- Removal of whitespaces
- decreasing the size of the vocabulary

B. Tokenizing the Cleaned Data

Tokenization is splitting the text into small amount of chunks. Each token is considered as an input to the machine learning algorithm as a feature of itself.

When we tokenized the text, we may collect a massive amount of dictionary of words, and they won' t all play an essential role in this process. We can set the ' maximum features' to select the top frequented words that we want to consider.

2. Text Sequencing

a. Padding

Process of making the tokens for all emails in an equal size is described as padding.

We have to send inputs in batches of data points. Information may have lost when inputs are of different sizes. So, we have to make them to the same size using the padding process, and that eases the batch updates regularly.

The length of all the tokenized emails post-padding is set using by the ' max_len' value which helps you to set the tokens an equal size.

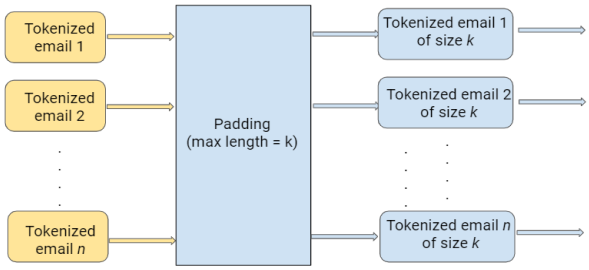


Figure--All Tokenized emails are converted to the same size in the ' Padding' stage.

Embedding

Text data can be easily interpreted by humans. But for the machines, reading, analyzing is a very complex task. To accomplish this task, we need to convert our texts into a machine-understandable format.

Embedding is the process which is converting the formatted text data into numerical values/vectors which a machine can interpret by itself.

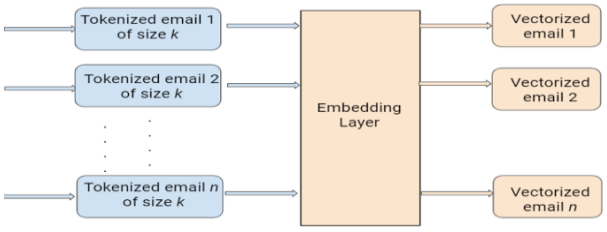


Figure : All tokenized emails are converted into vectors in the embedding phase

Model Summary :

```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_3 (Embedding)      (None, 25, 100)           873400
_________________________________________________________________
dropout_5 (Dropout)          (None, 25, 100)           0
_________________________________________________________________
lstm_3 (LSTM)                (None, 140)               134960
_________________________________________________________________
dropout_6 (Dropout)          (None, 140)               0
_________________________________________________________________
Classification (Dense)       (None, 1)                 141
=================================================================
Total params: 1,008,501
Trainable params: 1,008,501
Non-trainable params: 0
_________________________________________________________________
```

n_epochs = 10

results = model.fit(X_train, y_train, epochs=n_epochs, batch_size=batch_size, validation_split=0.2, verbose=1)

y_predict = [1 if o>0.5 else 0 for o in model.predict(X_test)]

```
Train on 3568 samples, validate on 892 samples
Epoch 1/10
3568/3568 [==============================] - 27s 7ms/step - loss: 0.1863 - accuracy: 0.9344 - val_loss: 0.0955 - val_accuracy:
0.9787
Epoch 2/10
3568/3568 [==============================] - 29s 8ms/step - loss: 0.0802 - accuracy: 0.9818 - val_loss: 0.0647 - val_accuracy:
0.9798
Epoch 3/10
3568/3568 [==============================] - 26s 7ms/step - loss: 0.0704 - accuracy: 0.9851 - val_loss: 0.0603 - val_accuracy:
0.9854
Epoch 4/10
3568/3568 [==============================] - 28s 8ms/step - loss: 0.0409 - accuracy: 0.9930 - val_loss: 0.0900 - val_accuracy:
0.9832
Epoch 5/10
3568/3568 [==============================] - 25s 7ms/step - loss: 0.0465 - accuracy: 0.9908 - val_loss: 0.0662 - val_accuracy:
0.9821
Epoch 6/10
3568/3568 [==============================] - 26s 7ms/step - loss: 0.0208 - accuracy: 0.9938 - val_loss: 0.0522 - val_accuracy:
0.9843
Epoch 7/10
3568/3568 [==============================] - 28s 8ms/step - loss: 0.0204 - accuracy: 0.9950 - val_loss: 0.0834 - val_accuracy:
0.9865
Epoch 8/10
3568/3568 [==============================] - 29s 8ms/step - loss: 0.0138 - accuracy: 0.9978 - val_loss: 0.1120 - val_accuracy:
0.9787
Epoch 9/10
3568/3568 [==============================] - 25s 7ms/step - loss: 0.0164 - accuracy: 0.9980 - val_loss: 0.1028 - val_accuracy:
0.9843
Epoch 10/10
3568/3568 [==============================] - 25s 7ms/step - loss: 0.0157 - accuracy: 0.9972 - val_loss: 0.0860 - val_accuracy:
0.9821
```

Confusion Matrix:

A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. The confusion matrix shows the ways in which your classification model is confused when it makes predictions.

It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made.

	Class 1 Predicted	Class 2 Predicted
Class 1 Actual	TP	FN
Class 2 Actual	FP	TN

- ☐ true positives (TP): These are cases in which we predicted yes (there will be fire and smoke) and they do have fire and smoke.
- ☐ true negatives (TN): We predicted no, and there will be no fire and smoke.
- ☐ false positives (FP): We predicted yes, but there will be no fire and smoke.
- ☐ false negatives (FN): We predicted no, but there will be fire and smoke.

The testing accuracy for the algorithm is calculated from the below formula:

$$\textbf{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Now the confusion matrix is created to determine the correct and incorrect predictions. Below is the code for it:

cf_matrix =confusion_matrix(y_test,y_predict)

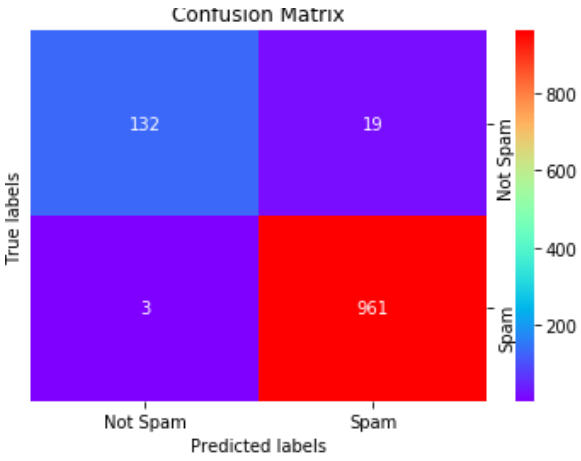
cf_matrix

```
array([[132,  19],
       [  3, 961]], dtype=int64)
```

tn,fp,fn,tp= confusion_matrix(y_test,y_predict,labels=[0,1]).ravel()

tn, fp, fn, tp

```
(132, 19, 3, 961)
```



From the above Figures the values in the colored boxes are the one where the actual spam status and the predicted spam status is same. They are the correctly predicted values. Other values in the matrix lead to the incorrect predictions because the actual spam status and the predicted spam status is not same.

The accuracy is calculated as follows:

= Total no. of correct predictions / Total no. of predictions

= 4428 / 4450

= 0.9972 (Testing Accuracy of Random Forest)

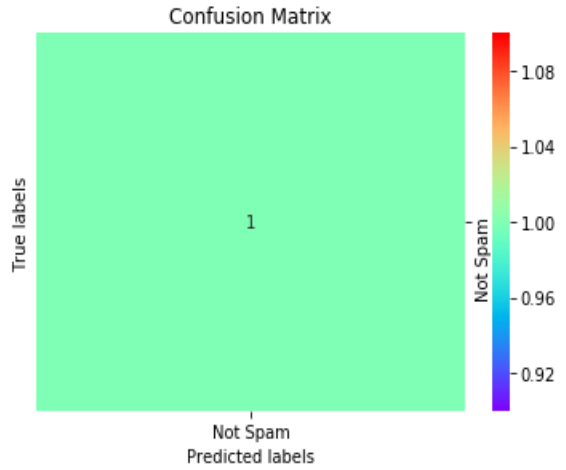
Output :

```
msg = [input()]
```

Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...

```
print (classification_report(ye,y_prediction))
```

	precision	recall	f1-score	support
1	1.00	1.00	1.00	1
accuracy			1.00	1
macro avg	1.00	1.00	1.00	1
weighted avg	1.00	1.00	1.00	1

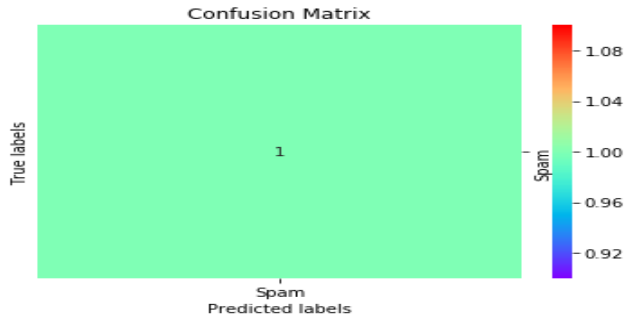


True label is not spam and Predicted label is not spam

```
msg = [input()]
```

Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's

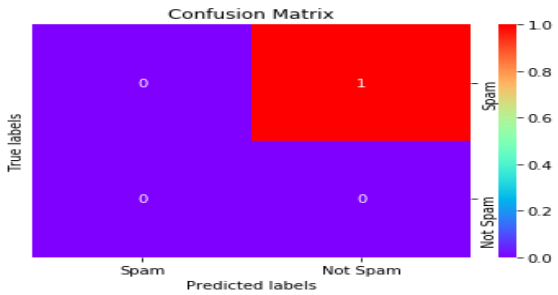
	precision	recall	f1-score	support
0	1.00	1.00	1.00	1
accuracy			1.00	1
macro avg	1.00	1.00	1.00	1
weighted avg	1.00	1.00	1.00	1



True label is spam and Predicted label is spam

SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6days, 16+ TsandCs apply Reply HL 4 info

	precision	recall	f1-score	support
0	0.00	0.00	0.00	1.0
1	0.00	0.00	0.00	0.0
accuracy			0.00	1.0
macro avg	0.00	0.00	0.00	1.0
weighted avg	0.00	0.00	0.00	1.0



True label is spam and Predicted label is not spam

Accuracy:

ENSEMBLE DECISION FOR SPAM DETECTION USING TSP APPROACH

PERFORMANCE COMPARISON OF THE EDTSP WITH CURRENT APPROACHES ON ENRON-SPAM

Approach	Precision(%)	Recall(%)	Accuracy(%)	F_1(%)
BoW-NB	77.74	98.41	87.45	86.10
BoW-C4.5	82.88	97.07	90.33	89.02
BoW-SVM	89.64	98.74	94.63	93.86
BoW-AdaBoost	89.13	98.78	94.25	93.57
BoW-RF	91.46	99.28	96.06	95.11
BoW-MLP	92.70	98.71	96.23	95.54
BoW-SAE	94.90	98.95	97.49	**96.84**
CFC-SVM	91.48	97.81	95.62	94.39
LC-FL-SVM	94.07	98.00	96.79	95.94
LC-VL-SVM	92.44	97.81	96.02	94.94
EDTSP-SVM	94.74	98.88	**97.58**	96.71

Our Approach :
print (classification_report(y_test,y_predict))

	precision	recall	f1-score	support
0	0.98	0.87	0.92	151
1	0.98	1.00	0.99	964
accuracy			0.98	1115
macro avg	0.98	0.94	0.96	1115
weighted avg	0.98	0.98	0.98	1115

ALGORITHM	ACCURACY
ENSEMBLE DECISION FOR SPAM DETECTION USING TSP APPROACH	97.58%
CNN	99.72%

Fig: Accuracy of the classification algorithm

First TSP APPROACH algorithm is implemented and it gives 97.58% accuracy and then to improve the accuracy better CNN algorithm is used which provides 99.72%. Hence the accuracy is improved.

V. Discussions

The major finding is spam email detection using machine learning techniques like Deep Neural Network, Convolution Neural Network.Machine Learning techniques are used since number of available training data is increasing, the effectiveness and efficiency of deep learning becomes more pronounced. .They have the ability to recognise distinctive characteristics of the contents of emails. Lack of effective strategy to handle the threats to the security of the spam filters. Such an attack can be causative or exploratory, targeted or indiscriminate attack is the major drawback need to be addressed in future.

Conclusion

In this paper, we proposed a content-based spam email filtering approach. The system uses keyword-based corpus that were built from training datasets to classify new incoming email message. Based on this finding, it can be concluded that the content classification performance will be improved with enhancements as a feature selection. The second finding is that the use of the interleaved hybridization generated better optimal features for the classifier than using all the features From this observation, it can be stated that content classification can be better performed using all the optimal features generated by the interleaved hybridization.

Future work

Our future work targets the botnet phenomena in mobile devices and its detection comprehensively.

References

[1] S. Wasi, S. Jami and Z. Shaikh, "Context-based email classification model", Expert Systems, vol. 33, no. 2, pp. 129-144, 2015.

[2] Alsmadi and I. Alhami, "Clustering and classification of email contents", Journal of King Saud University - Computer and Information Sciences, vol. 27, no. 1, pp. 46-57, 2015.

[3] S. Sayed, "Three-Phase Tournament-Based Method for Better Email Classification", International Journal of Artificial Intelligence & Applications, vol. 3, no. 6, pp. 49-56, 2012.

[4] D. Patil and Y. Dongre, " A Clustering Technique for Email Content Mining," Int. J. Comput. Sci. Inf. Technol., vol. 7, no. 3, pp. 73– 79, 2015.

[5] H. He, A. Tiwari, J. Mehnen, T. Watson, C. Maple, Y. Jin, and B. Gabrys, " Incremental information gain analysis of input attribute impact on rbf-kernel svm spam detection," in Evolutionary Computation (CEC), 2016 IEEE Congress on, pp. 1022– 1029, IEEE, 2016.

[6] A.-Z. Ala' M, H. Paris, et al., " Spam profile detection in social networks based on public features," in Information and Communication Systems (ICICS), 2017 8th International Conference on, pp. 130– 135, IEEE, 2017.

[7] A.-Z. AlaM, H. Faris, M. A. Hassonah, et al., " Evolving support vector machines using whale optimization algorithm for spam profiles detection on online social networks in different lingual contexts," Knowledge-Based Systems, vol. 153, pp. 91– 104, 2018.

[8] Barushka and P. Hájek, " Spam filtering using regularized neural networks with rectified linear units," in Proc. AI*IA Adv. Artif. Intell. 15th Int. Conf. Italian Assoc. Artif. Intell., Genoa, Italy, Nov./Dec. 2016, pp. 65– 75.

[9] H. Faris, I. Aljarah, and B. Al-Shboul, " A hybrid approach based on particle swarm optimization and random forests for e-mail spam filtering," in International Conference on Computational Collective Intelligence, pp. 498– 508, Springer, 2016.

[10] F. Wang, T. Xu, T. Tang, M. Zhou, and H. Wang, ' ' Bilevel feature extraction-based text mining for fault diagnosis of railway systems,' ' IEEE Trans. Intell. Transp. Syst., vol. 18, no. 1, pp. 49– 58, Jan. 2017.

[11] S. Gupta, A. Khattar, A. Gogia, P. Kumaraguru, and T. Chakraborty, ' ' Collective classification of spam campaigners on Twitter: A hierarchical meta-path based approach,' ' in Proc. World Wide Web Conf., Apr. 2018, pp. 529– 538.

[12] M. Chakraborty, S. Pal, R. Pramanik, and C. R. Chowdary, " Recent developments in social spam detection and combating techniques: A survey," Inf. Process. Manag., vol. 52, no. 6, pp. 1053– 1073, 2016.

[13] H. Fu, X. Xie, and Y. Rui, " Leveraging careful microblog users for spammer detection," in Proc. 24th Int. Conf. World Wide Web, 2015, pp. 419– 429.

[14] S. K. Trivedi and P. K. Panigrahi, ' ' Spam classification: A comparative analysis of different boosted decision tree approaches,' ' J. Syst. Inf. Technol., vol. 20, no. 3, pp. 105– 298, Aug. 2018.

[15] Y. Ren and Y. Zhang, " Deceptive opinion spam detection using neural network," in Proc.

[16] COLING 26th Int. Conf. Comput. Linguist. Conf. Tech. Papers, Dec. 2016, Osaka, Japan, 2016, pp. 140– 150. [Online]. Available: http://aclweb.org/anthology/C/C16/C16-1014.pdf

[17] D. Bahdanau, K. Cho, and Y. Bengio, " Neural machine translation by jointly learning to align and translate," CoRR, vol. abs/1409.0473, Sep. 2014. [Online]. Available: http://arxiv.org/abs/1409.0473

[18] S. Zhao, Z. Xu, L. Liu, and M. Guo, " Towards accurate deceptive opinion spam detection based on word order-preserving CNN," CoRR, vol. abs/1711.09181, 2017. [Online]. Available: http://arxiv.org/abs/1711.09181

[19] G. Jain, M. Sharma, and B. Agarwal, ' ' Spam detection on social media using semantic convolutional neural network,' ' Int. J. Knowl. Discovery Bioinf., vol. 8, no. 1, pp. 12– 26, Jan. 2018.