Group 12: Email Spam Classification

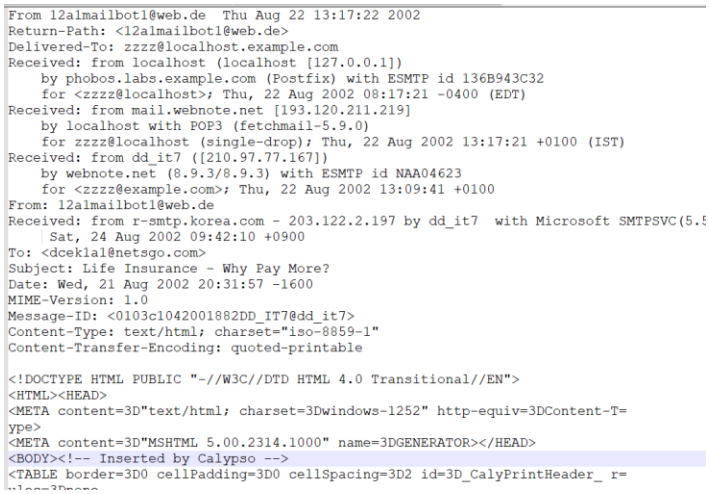
GONG Yanming	57335161
HE Jingrao	57394554
ZHOU Ruoyu	57414348
ZHOU Yifan	57638497

1.Introduction

Every day, people receive a large number of unwanted bulk emails that are sent to their inboxes. According to the statistics, the threat of spam emails is on the rise, and it's responsible for over 77% of the worldwide email traffic up to now [1]. These emails contain information that users did not ask for and some even dangerously contain phishing links. For this reason, recipients may find it very irritating, and even suffer incalculable economic losses. In this case, antispam filters are urgently needed. Therefore, we try to compare different classification algorithms and apply Neural Language Programming to mail classification to identify whether an email is spam or not and analyze it.

2. Data Preprocessing

2.1 Data Source



```
From 12almailbot1@web.de  Thu Aug 22 13:17:22 2002
Return-Path: <12almailbot1@web.de>
Delivered-To: zzzz@localhost.example.com
Received: from localhost (localhost [127.0.0.1])
    by phobos.labs.example.com (Postfix) with ESMTP id 136B943C32
    for <zzzz@localhost>; Thu, 22 Aug 2002 08:17:21 -0400 (EDT)
Received: from mail.webnote.net [193.120.211.219]
    by localhost with POP3 (fetchmail-5.9.0)
    for zzzz@localhost (single-drop); Thu, 22 Aug 2002 13:17:21 +0100 (IST)
Received: from dd_it7 ([210.97.77.167])
    by webnote.net (8.9.3/8.9.3) with ESMTP id NAA04623
    for <zzzz@example.com>; Thu, 22 Aug 2002 13:09:41 +0100
From: 12almailbot1@web.de
Received: from r-smtp.korea.com - 203.122.2.197 by dd_it7  with Microsoft SMTPSVC(5.5
     Sat, 24 Aug 2002 09:42:10 +0900
To: <dceklal@netsgo.com>
Subject: Life Insurance - Why Pay More?
Date: Wed, 21 Aug 2002 20:31:57 -1600
MIME-Version: 1.0
Message-ID: <0103c1042001882DD_IT7@dd_it7>
Content-Type: text/html; charset="iso-8859-1"
Content-Transfer-Encoding: quoted-printable

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML><HEAD>
<META content=3D"text/html; charset=3Dwindows-1252" http-equiv=3DContent-T=
ype>
<META content=3D"MSHTML 5.00.2314.1000" name=3DGENERATOR></HEAD>
<BODY><!-- Inserted by Calypso -->
<TABLE border=3D0 cellPadding=3D0 cellSpacing=3D2 id=3D_CalyPrintHeader_ r=
```

Figure 1: The Example of Non-spam Emails

The dataset we use is named SpamAssassin public mail corpus from the Kaggle website. It contains 3302 original email data, of which 501 are spam, and 2801 are ham (non-spam) email data. Figure

1 and Figure 2 show examples of two different types of emails, respectively. Most of the spam emails we used consist of text in TML and CSS. We will implement email spam classification based on this dataset.

```
  <TR>
    <TD colSpan=3D3>
      <HR color=3Dblack noShade SIZE=3D1>
    </TD></TR></TD></TR>
  <TR>
    <TD colSpan=3D3>
      <HR color=3Dblack noShade SIZE=3D1>
    </TD></TR></TBODY></TABLE><!-- End Calypso --><!-- Inserted by Calypso=
 --><FONT
color=3D#000000 face=3DVERDANA,ARIAL,HELVETICA size=3D-2><BR></FONT></TD><=
/TR></TABLE><!-- End Calypso --><FONT color=3D#ff0000
face=3D"Copperplate Gothic Bold" size=3D5 PTSIZE=3D"10">
<CENTER>Save up to 70% on Life Insurance.</CENTER></FONT><FONT color=3D#ff=
0000
face=3D"Copperplate Gothic Bold" size=3D5 PTSIZE=3D"10">
<CENTER>Why Spend More Than You Have To?
<CENTER><FONT color=3D#ff0000 face=3D"Copperplate Gothic Bold" size=3D5 PT=
SIZE=3D"10">
<CENTER>Life Quote Savings
<CENTER>
<P align=3Dleft></P>
<P align=3Dleft></P></FONT></U></I></B><BR></FONT></U></B></U></I>
<P></P>
<CENTER>
<TABLE border=3D0 borderColor=3D#111111 cellPadding=3D0 cellSpacing=3D0 wi=
dth=3D650>
  <TBODY></TBODY></TABLE>
<TABLE border=3D0 borderColor=3D#111111 cellPadding=3D5 cellSpacing=3D0 wi=
dth=3D650>
  <TBODY>
  <TR>
```

Figure 2: The Example of Spam Emails

2.2 Data Preprocessing

2.2.1 Label

Because our dataset has already been correctly classified, we first read all the text content into the *dataframe* and set the *label* attribute to identify all spam emails as 1 and non-spam emails as 0.

2.2.2 Parse Emails

We first extracted the email text information we require from our dataset, which is a file object in MIME format. The *Email.Parser* package was used for this procedure.

The standard syntax parser offered by *Email.Parser* can comprehend the structure of most email documents, including MIME documents. The root *EmailMessage* instance corresponding to any object structure we send to this parser, whether it be a byte string, string, or file object, will be returned. We utilized the *BytesParser* function provided by *Email.Parser* to parse the contents of the message because our data is a MIME message, and then we used the *get_payload()* method to return the contents of the instance, which contains the email text that we require. We were able to get the email text we needed in this method, and Figure 3 displays some of the results we were able to extract.

	email	label
0	b'From exmh-workers-admin@redhat.com Thu Aug ...	0.0
1	b'Return-Path: <Online#3.19578.34-UgGTgZFN19NA...	0.0
2	b'Return-Path: <Online#3.19584.83-p1SYlJ1blFvQ...	0.0
3	b'From Steve_Burt@cursor-system.com Thu Aug 2...	0.0
4	b'Return-Path: <Online#3.19586.b5-9w0blztbvHPd...	0.0
...	...	...
3297	b'From biz2biz2446@Flashmail.com Mon Oct 7 2...	1.0
3298	b'From cna@insiq.us Tue Oct 8 00:10:39 2002\...	1.0
3299	b'From bounce2@u-answer.com Tue Oct 8 11:02:...	1.0
3300	b'From beautyinfufuxxxmeb13mxy@aol.com Tue Oc...	1.0
3301	b'From evtwqmigru@datcon.co.uk Tue Oct 8 11:...	1.0

Figure 3: The Content of Emails

2.2.3 Preprocessing

In this part, we excluded certain extraneous information from the email text that could have a negative impact on the accuracy of our classification, such as punctuation, numbers, and stopwords.

We used Python's regular expressions to convert all '\n,' '\t', and integers in the text to spaces, then used *string.punctuation* to remove any punctuation. Only words were left in the text. Some of the data that we've processed is shown in Figure 4.

label	content
0.0	date wed aug from...
0.0	html head title cable companies cra...
0.0	html head title shopper newsletter alerts ...
0.0	martin a posted tassos papadopoulos the greek...
0.0	cnet download dispatch mac edition july ...
...	...
1.0	there is no stumbling on to it the greatest w...
1.0	None
1.0	html head meta http equiv content language...
1.0	html body tr valign d top td heig...
1.0	uncommon exotic pleasure botanicals feeling ma...

Figure 4: Processed Results

We also took the stopwords out of the text to improve the results. Stopwords are worthless words in natural language processing. These terms are used frequently throughout the text but have no meaningful meaning. These words are also useless for our email categorization. Therefore, we

filtered the text content using the stopwords library offered by *nltk* in order to improve the accuracy of our classification results. Several instances of stopwords are shown in Figure 5.

```
['i','me','my','myself','we','our','ours','ourselves',
 'you',"you're","you've","you'll","you'd",'your','yours',
 'yourself','yourselves','he','him','his','himself','she',
 "she's",'her','hers',...,"weren't",'won',"won't",'wouldn',"wouldn't"]
```

Figure 5: Partial Stopwords

Figure 6 shows some of the data results after removing the stopwords.

label	content
0.0	date wed aug chris garrigues cwg dated fa deep...
0.0	html head title cable companies cracking wi fi...
0.0	html head title shopper newsletter alerts titl...
0.0	martin posted tassos papadopoulos greek sculpt...
0.0	cnet download dispatch mac edition july vol us...
...	...
1.0	stumbling greatest way marketing century undou...
1.0	None
1.0	html head meta http equiv content language con...
1.0	html body tr valign top td height bgcolor ffff...
1.0	uncommon exotic pleasure botanicals feeling ma...

Figure 6: Processed Results without Stopwords

2.2.4 Stemming

The word stems of the contents' words were taken out and presented in this part. A whole phrase is typically made up of numerous words with various lexical forms, such as nouns, verbs, and adjectives. These words' morphologies will also have many expressions depending on the syntax, tense, and other factors. For instance, nouns become plural, verbs become past tense, and so forth. We employed *stemmer* to achieve stem extraction in order to enable these words to be counted more accurately. For instance, following stemmer processing, working, worked, and works will be changed into work, allowing our model to be trained more accurately because it will be able to identify them as the same word. The partially processed data is shown in Figure 7.

label	content
0.0	date wed aug chri garrigu cwg date fa deepeddi...
0.0	html head titl cabl compani crack wi fi titl h...
0.0	html head titl shopper newslett alert titl hea...
0.0	martin post tasso papadopoulo greek sculptor b...
0.0	cnet download dispatch mac edit juli vol use m...
...	...
1.0	stumbl greatest way market centuri undoubtedli...
1.0	None
1.0	html head meta http equiv content languag cont...
1.0	html bodi tr valign top td height bgcolor ffff...
1.0	uncommon exot pleasur botan feel marvel mood s...

Figure 7: Processed Results after Stemming

2.2.5 Extract Valid Data

All of the pre-processing procedures have been finished. We eliminated any rows with empty values and renumbered the data because since the data has been censored, it is possible that the processed data may be empty and that such data has no practical meaning for classification purposes. In the end, we collected 3063 pieces of valid data. All the data were exported to a CSV file for model training. Some of the final data we collected is shown in Figure 8.

label	Email content
1	html font color strong h hello h font p p font color ff h dmm dis
1	html bodi head bodi bgcolor ffffff tabl width height tr valign to
1	import domain inform new domain name final avail gener public dis
1	html head head bodi p align center font style font size pt face c
1	html head titl titl meta http equiv content type content text htm
1	html head titl teen titl head bodi bgcolor ffffff text link ff al
1	html head head bodi viagra xenic vioxx zyban propecia reg br offe
1	immedi help need fortun compani grow tremend rate per year simpli
1	help want year old fortun compani isgrow tremend rate look indivi
1	hi http club tfox com get email everi day offer show make money e
1	html head titl tell send health card titl meta http equiv content
1	new account zzzz exampl com adult club offer free membership best
1	p e c l r e p r reliabl gener hundr lead prospectseveri week rese
1	protect financi well purchas extend auto warranti car today click
1	mr vincent nnaji standard trust bank ltd lago nigeria dear sir mr
1	dear sir due respect humil write letter believ youwouldb great as
1	multipart mime messag multipart boundari content type text plain
1	lowest rate avail term life insur take moment fill onlin form see
1	bodi topmargin leftmargin tabl cellspac cellpad width border tbod
1	html font size ptsize famili sansserif face arial lang mortgag se
1	tabl id autonumb style border collaps collaps bordercolor cellspa

Figure 8: Part of the Valid Dataset

2.2.6 Split Training Set and Test Set

We used *train_test_split* to randomly divide the training set and test set from the data samples, and finally, the ratio we selected of the training set to test set is 7:3.

3. Classification Model

3.1 Classical Classification Models

We tried to classify the sample data by the following four classical classification models and adjusted the parameters to obtain maximum accuracy.

3.1.1 SVM

The binary classification model SVM (support vector machine) has a linear classifier as its fundamental model, defined by the greatest interval on the feature space. The fundamental goal of SVM learning is to identify the separating hyperplane with the largest geometric interval and the ability to accurately split the training data set. For linearly divisible datasets, there are an infinite number of such hyperplanes, but there is only one that is geometrically maximally spaced. We tried to classify the samples using *LinearSVC* provided by *sklearn*.

3.1.2 KNN

A new input instance is classified into a class using the K-nearest neighbor approach, which locates the K training dataset instances that are closest to the new instance. Most of these K instances must belong to the same class. We tried to classify the samples using *KNeighborsClassifier* provided by *sklearn*.

3.1.3 Decision Tree

A decision tree is a graph that uses a branching strategy to show all potential decisions' consequences. Each leaf node in the tree corresponds to the object represented by the route from the root node to the leaf node value, and each node in the tree represents a possible attribute value. Each node in the tree represents an item. We tried to classify the samples using *DecisionTreeClassifier* provided by *sklearn*.

3.1.4 Random Forest

A group of decision trees that operate on the output are produced by the random forest method. The random forest is a forest made up of randomly generated decision trees as opposed to the decision tree. The output result of the method integrates the output of a single decision tree to get the final output result. The technique employs numerous decision trees to make choices. As a result, random trees should be more accurate than choice trees. We tried to classify the samples using *RandomForestClassifier* provided by *sklearn*.

3.2 Bert-Based Model

We combined deep learning with Email Spam Classification and designed a new Bert-Based Model. The figure below shows the workflow of the Bert-Based Model. The embedded Email contents are first inputted into the encoder to generate encoded vectors, and then go to the decoder. After being processed by the full link layer, we calculate the loss function at the end of each epoch. Finally, we get the prediction.

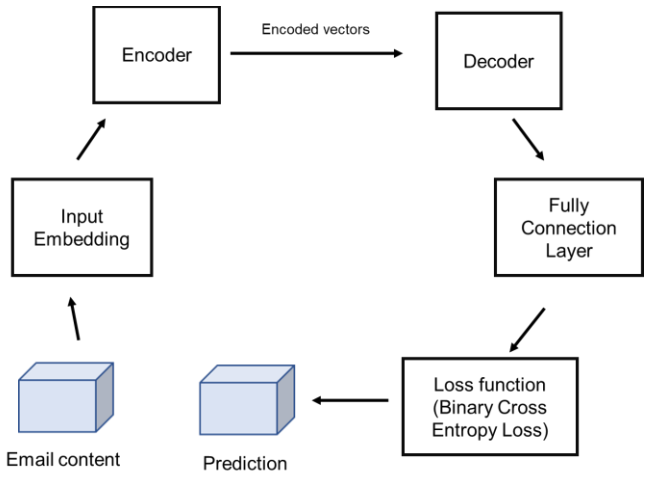


Figure 9: Workflow of Bert-Based Model

This model consists of two key parts. One is the encoder and the other is the decoder. We chose Flair, which is based on BERT, as the encoder. Flair encodes a sequence of text into vectors and solves the problem that the length of each sentence is not the same. Bi-LSTM is the decoder. It can get the content before and after the current text simultaneously. Therefore, Bi-TSTM can find the relationship between sentences and sentences.

4. Experiment Result

4.1 SVM

The impact of various *c* parameters on the SVM model's accuracy is depicted in Figure 10. The regularization parameter is denoted by c. We can see that the model's accuracy reduces as the value of *c* rises. The SVM model's accuracy at its maximum, when $c = 0.1$, is almost 98.2%.

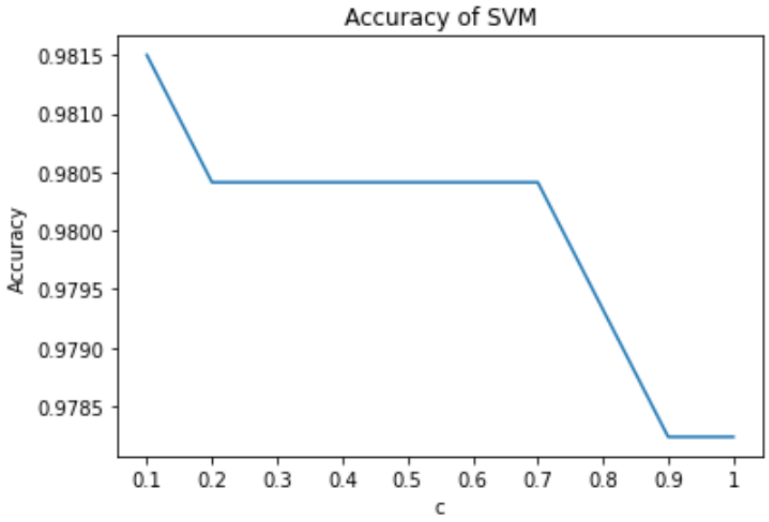


Figure 10: The Accuracy of SVM

4.2 KNN

Figure 11 shows the effect of different parameters *n* on the accuracy of the KNN model. *n* means that a sample belongs to a certain class if the majority of its n nearest neighbors in the feature space also belong to that class. We can see that the accuracy of the model is related to the change in the value of n. When *n*=3, the SVM model has the highest accuracy of 97.1s%.

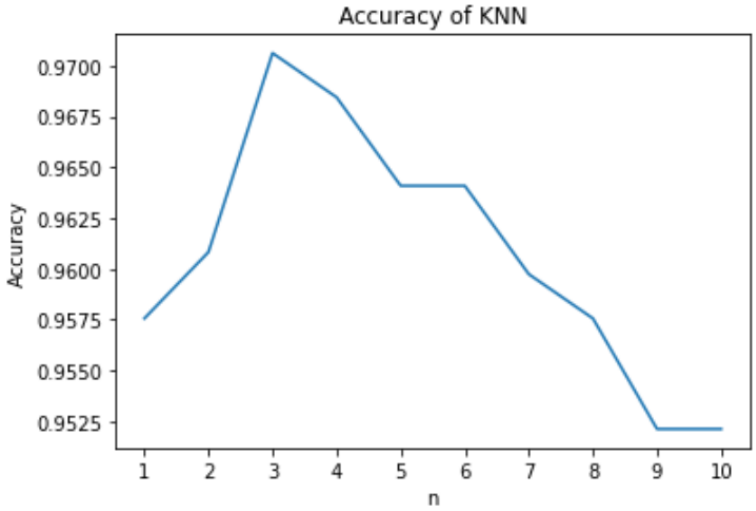


Figure 11: The Accuracy of KNN

4.3 Decision Tree

The impact of different *max_depth* settings on the decision tree model's accuracy is shown in Figure 12. *max_depth* refers to the decision tree's deepest level. We can observe that the variance

in the *max_depth* value is connected to the model's accuracy. The decision tree model has a 97.1% accuracy rate when *max_depth* is 60.

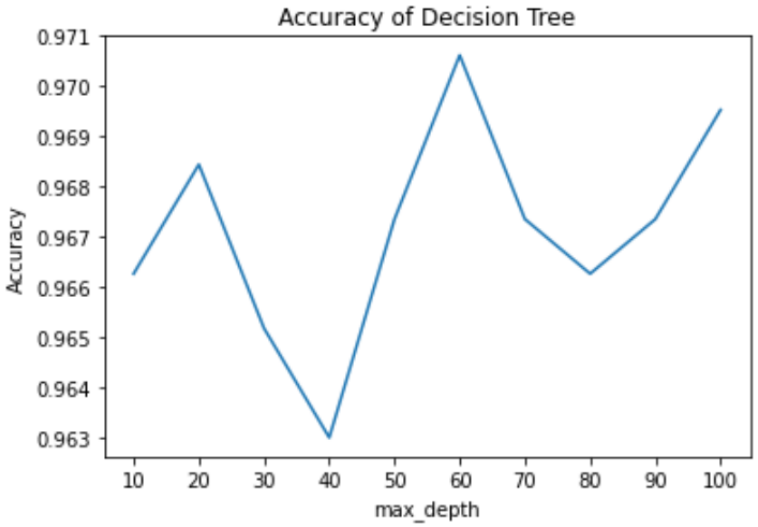


Figure 12: The Accuracy of Decision Tree

4.4 Random Forest

The impact of various *n_estimators* settings on the Random Forest model's accuracy is depicted in Figure 13. The number of trees in the model's forest is specified by the *n_estimators* parameter. As can be seen, *n_estimators* value variation and model accuracy are connected. The Random Forest model has a 97.5% accuracy rate when *n_estimators* is 30.

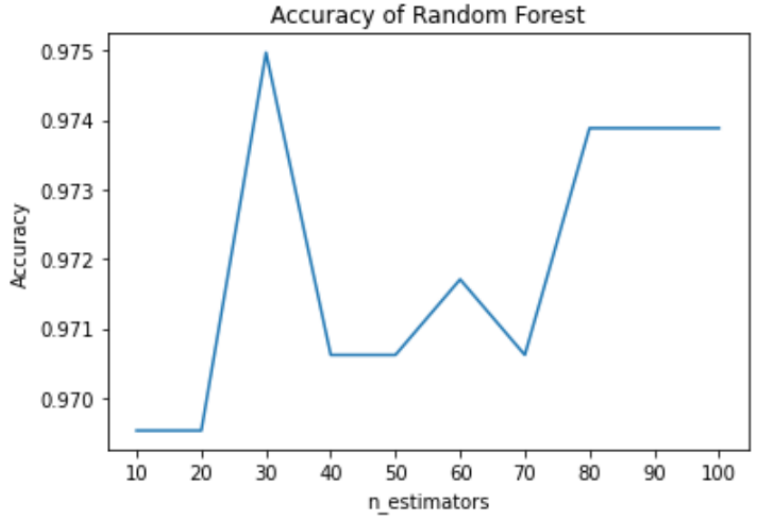


Figure 13: The Accuracy of Random Forest

4.5 Bert-Based Model

Since training a Bert-based model requires a lot of memory, we can only set batch_size = 128 and use binary cross entropy loss for training. The result is shown below.

test_accuracy	test_precision	test_recall	f1 score
0.9891186071817193	0.993734335839599	0.993734335839599	0.9587

4.6 Confusion Matrix

We output the images of each of the five confusion matrices, as shown in Figure 14.

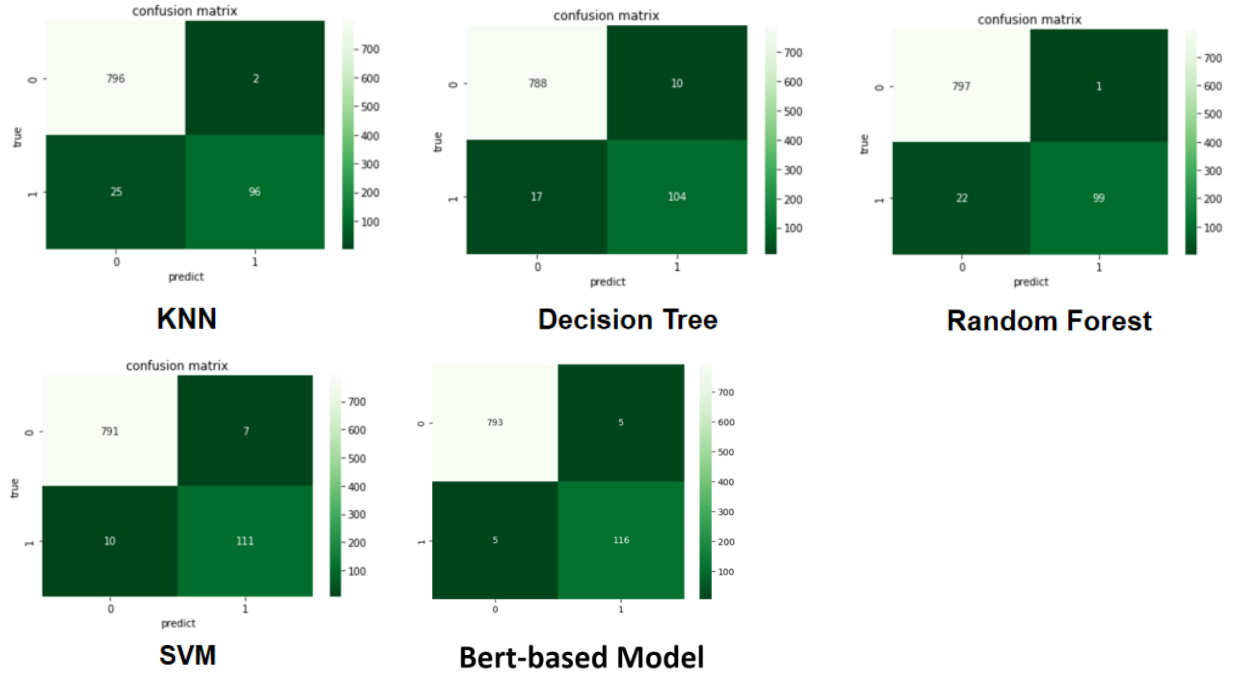


Figure 14: Confusion Matrix

4.7 Accuracy Score and F1 Score

The accuracy of the five models, SVM, KNN, Decision Tree, Random Forest, and Bert, are shown and plotted in the following bar chart. It can be clearly seen that each model obtained an accuracy of more than 97%. All classical models, in our opinion, exhibit high accuracy, and the Bert-Based model we implemented also works well.

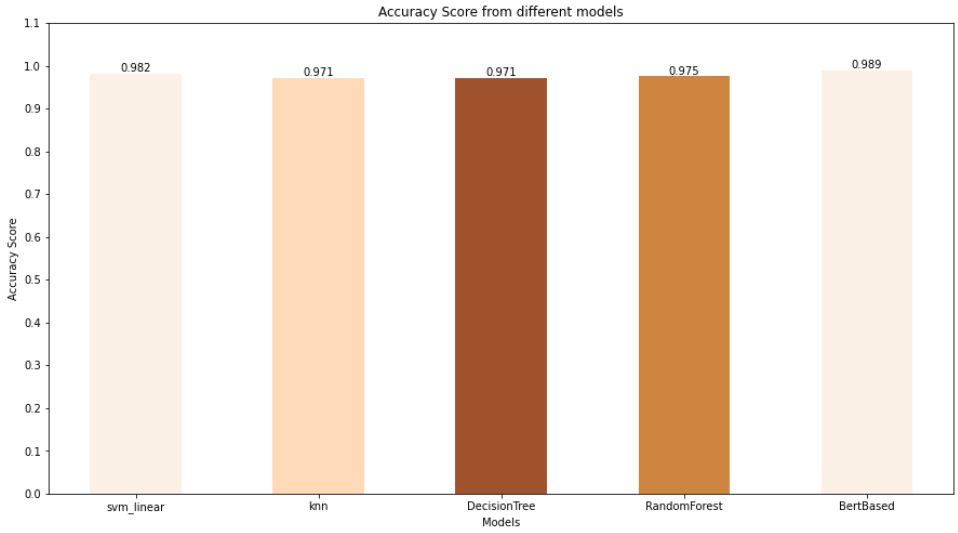


Figure 15: Accuracy Score

We calculated the F1 score of the five models respectively and plotted the following bar chart. All five models received scores of 88% or higher, except for KNN, which performed a little poorly.

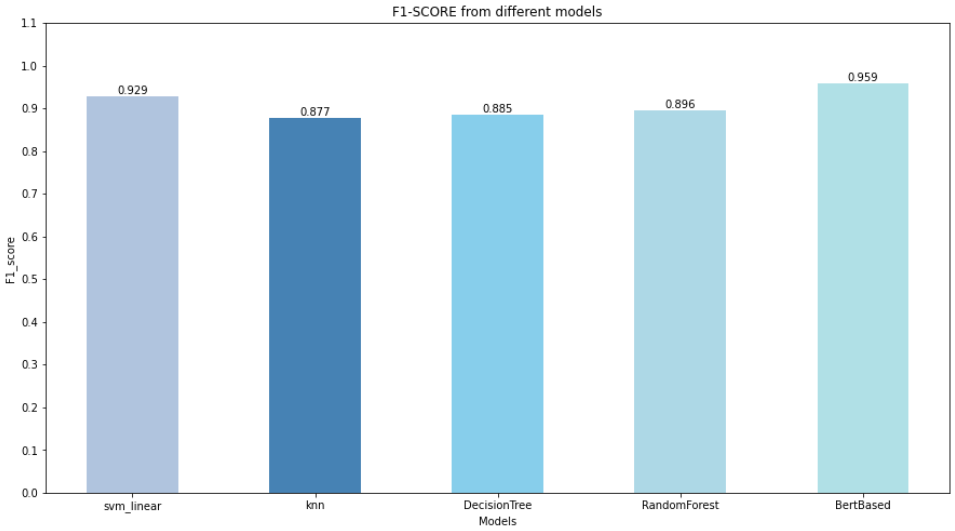


Figure 16: F1 Score

5. Discussion

5.1 Specific Types of Emails

Our classification approach trains the model using data from our existing dataset rather than focusing on emails from a particular domain. Hence, the majority of emails that people use to

communicate on a daily basis are the basis for our classification results. Most of the spam emails we used consist of text in HTML and CSS.

For domain-specific emails, such as filtering interview emails and advertising emails from Job Hunter, we need to collect relevant job email datasets and complete the classification by some specific words. For instance, if the sender is clearly a corporation or the emails clearly state the receiver's name, interview time, and place, we can recognize them as non-spam emails; yet, if the email is from a well-known recruitment website or contains more terms like "recommend," "try," and "deliver," for example, we can classify it as a spam email.

After obtaining a sufficient number of datasets, we can use these datasets to train our model and obtain the classification of emails for a specific domain.

5.2 Misclassified Analysis

We examined the SVM algorithm's outcomes as an illustration. As a result of the SVM algorithm, 17 emails have been incorrectly classified, 7 of which are non-spam emails but are labeled as spam, and 10 of which are spam emails but are labeled as non-spam. We took these incorrectly classified emails and examined them. Several of the misclassified email messages are displayed in Figure 17.

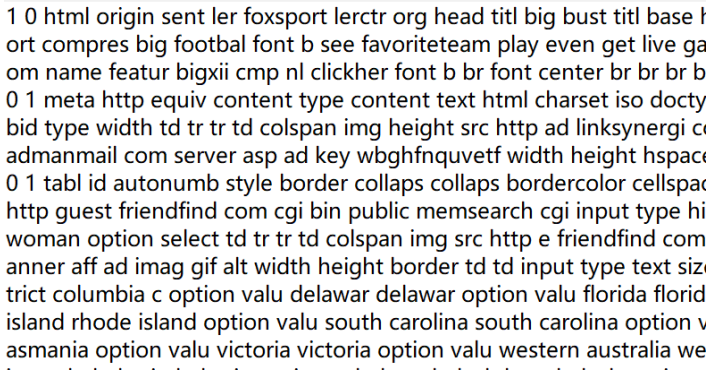


Figure 17: Misclassified emails

In the first instance, we discovered that certain words or letters, such p, repeatedly appeared in the text that was wrongly labeled as spam. Once more, the letter p is a telltale sign of spam because the spam we utilized is made up of HTML. P is a widely popular paragraph tag in HTML.

In the second instance, the content that was mistakenly categorized as non-spam mail included a lot of common terms. These words are text messages in HTML but can also be considered as words used in non-spam emails.

For the reasons outlined above, we think that these emails may be incorrectly categorized. Also, the algorithm for extracting word stems can also be further improved. We consider that the

extracted content can be filtered again in the future to avoid the problem that the extracted results are meaningless.

In addition, we believe we have a theory that may be supported in the future. Based on the kind of email content that needs to be classified, we may be able to identify some terms or phrases, which depend on the types of the emails and make it difficult to distinguish between the two kinds of emails and eliminate them during the preprocessing stage to increase the precision of our classification.

5.3 Model Accuracy Analysis

From the analysis of the accuracy scores, the Bert-based model obtained the highest accuracy but did not improve much compared to the second-place SVM. So, we analyzed it as follows.

Firstly, the possible reason is that the sample size is small. From the perspective of statistical significance, machine learning is difficult to compare on a large amount of data, so the limited sample size leads to a discount in statistical significance. In other words, Bert's model structure is complex. Our data volume is small, so SVMs perform about as well as neural networks on our dataset. However, only when the amount of data is large, the advantages of Bert may appear. We guess that Bert may perform better when dealing with large data sets. Secondly, SVM is good enough (accuracy score:98%), and the Bert-based network is excellent, so there is little to improve. Finally, Bert is 0.2% higher than SVM. It is possible that Bert has pre-training and prior knowledge of the network structure, so it is slightly higher than SVM after our training and fitting. Or, in some ways, it's because of the effects of noise.

In the future, we may be able to improve this by conducting multiple training tests, averaging their accuracy, testing both models with a large training set, and comparing their evaluation metrics.

6. Conclusion

In this project, we first preprocessed the dataset, excluded irrelevant information that could negatively affect classification accuracy, and then implemented stemming extraction for individual words to facilitate model training. Finally, we obtained more than 3000 pieces of valid data and divided them into training and test sets in a ratio of 7:3.

Five models are used separately to classify emails, including four classical models (SVM, KNN, Decision Tree, Random Forest) and the Bert-based model, which consists of a BiLSTM layer and two stacked dense layers. We tested the effectiveness of the models. The results showed that all models obtained an accuracy of over 94% and a relatively acceptable F1 score. In the future, we intend to further analyze and compare our spam classification algorithms in multiple aspects and extend them to larger problems.

Reference

[1] Kaspersky Lab Spam Report (2017)
https://www.securelist.com/en/analysis/204792230/Spam_Report_April_2012

[2] https://www.kaggle.com/datasets/cesaber/spam-email-data-spamassassin-2002

[3] Shuaib M, Osho O, Ismaila I, et al. Comparative analysis of classification algorithms for email spam detection[J]. International Journal of Computer Network and Information Security, 2018, 12(1): 60.

[4] Yaseen Q. Spam email detection using deep learning techniques[J]. Procedia Computer Science, 2021, 184: 853-858.