FIT5186 Intelligent Systems

Detecting Spam Emails Using Neural Networks

Group 1

Yang Yan 2759****
Xianqiang Gao 2731****

27 May 2016

Southeast University – Monash University

Joint Graduate School

Detecting Spam Emails Using Neural Networks

Abstract- 'Spam' is the word for unwanted emails. Detecting spam emails before they reach the user is quite important. Various machine-learning techniques can be used for spam filtering. We choose neural network to improve the detection accuracy by changing the network models, architectures or the parameters. After changing the parameters or the architectures, we will compare the accuracy level from the dimensions of spam messages classified as legitimate and legitimate messages classified as spam. Furthermore, we will provide the classification results of other traditional machine learning techniques, such as Naïve Bayes, Support Vector Machine, Logistic Regression.

1. Introduction

Electronic mail has become a dominant mean for personal and business communication. However, spam emails are used for marketing of products and services, spreading rumors and other fraudulent advertisements. Spam does not only annoy email users, but also exhausts network resources (bandwidth and storage), slows down email servers and provides a medium for phishing attacks and distributing harmful malicious codes. Also spam-filtering method falls into two broad categories: non-machine-learning based and machine-learning-based. Machine-learning based techniques can analyze the message content and classify it accordingly (El-Alfy et al., 2011).

Recently, various machine-learning techniques have been used for spam filtering, including support vector machines, memory-based learning, neural networks, Bayesian classifiers and fuzzy logic. Among these methods, we choose neural networks to do the experiments to find how to improve the detection accuracy. Neural network is able to classify the occurrence of certain words and phrases in terms of how and where they appear in the email message, not by their existence alone (Sivanadyan and Thiagarajan, 2003).

2. Data Sets

The data used for the project was obtained from the University of California – Irvine Machine Learning Repository. This database has been created in 1999 by M. Hopkins, E. Reeber, G. Forman, and J.Suermondt at Hewlett-Packard Labs. It consists of 4601 instances of legitimate and spam email messages with 39.4% being spam. Each instance is characterized by 57 input attributes and is labeled as spam (represented as 1) or legitimate (represented as 0). Among these 57 input attributes, attributes 1–48 give the percentage of words in the email message for the respective keyword indicated in the attribute name. Attributes 49–54 give the percentage of characters in the email message for the respective character indicated in the attribute name. Attributes 55 and 56 give the average and maximum lengths, respectively, of uninterrupted sequences of capital letters in the message. Attribute 57 gives the total number of capital letters in the message.

The attribute number will be used as the variable number for models in NeuroShell2. Attribute

number 58 in the dataset is the true email class. In our experiments, we can also use two output neurons and classify legitimate mails as 1 0 and spam mails as 0 1, so we add two columns at the end of the dataset to handle two output neurons. The dataset has no missing attribute values.

3. Training Issues and Results

Experient1:

In the first experiment, we choose all 57 variables as inputs, and construct a MFNN architecture with 57 inputs, 90 hidden neurons and 2 output neurons. The default number of hidden neurons in NeuroShell 2 is $(N+K)/2 + P^{1/2}$ while N is the number of input neurons, K is the number of output neurons and P represents the number of patterns in the training set. So the default number of hidden neurons in this experiment is 90. 20 percent of the 4601 instances are randomly extracted to form the test set. Also the scale function in slab 1 is linear [-1,1], the activation function in slab 2 and slab 3 are logistic function which are all default. The learning rate is set to 0.1 and the momentum is 0.1 while the initial weight is 0.3.

Our pattern selection is rotation and weight updates is momentum. The network performance will be measured on the test set every 200 epochs. When the average error is less than 0.01 in the training set or the test set error is not improved within 20,000 epochs, then training will be terminated to prevent memorization of the training data.

After 5 minutes learning, the output statistic of R square is shown in Figure 1. The R square of the first experiment is over 0.71.

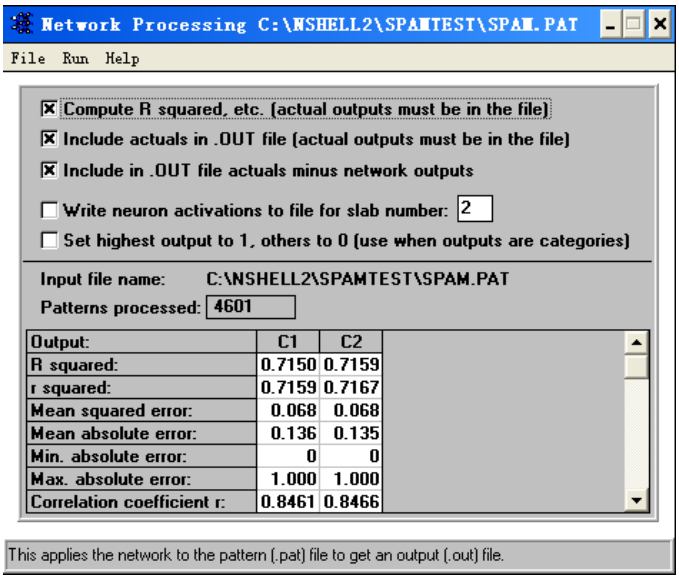

Figure 1 Output statistic by using MFNN with 57-90-2 architecture

Evaluating the performance across the entire data set, the neural network is able to correctly

classify 2651 of the 2788 known legitimate mails and 1556 of the 1813 known spam mails. These decisions have been made by choosing the maximum firing neuron, if neuron 1 fires more than neuron 2 for a given input pattern, then the input pattern is classified to belong to class 1(legitimate mails), and vice versa. The results are summarized in Table 1 showing the accuracy levels of the decisions.

Table 1 Classification accuracy using MFNN with 57-90-2 architecture

	Classified legitimate	**Classified spam**	**Row accuracy**
Actually legitimate	2651	137	0.9509
Actually spam	257	1556	0.8582
Column accuracy	0.9116	0.9191	

Experiment 2:

In further experiments, we vary the number of the hidden neurons from 90 to try 100, 80 and 70 neurons, still with 57 input features. The results are shown in Table 2, Table 3 and Table 4 for 100, 80 and 70 hidden neurons respectively. From the results we can find that with less number of hidden neurons, the accuracy of the classifying both legitimate and spam mails are all improved. And among this, 57-80-2 architecture achieves the highest accuracy in classifying actually spam mails and has the best performance. So with 57 input neurons and 2 output neurons, we will choose 80 hidden neurons.

Table 2 Classification accuracy using MFNN with 57-100-2 architecture

	Classified legitimate	**Classified spam**	**Row accuracy**
Actually legitimate	2641	147	0.9473
Actually spam	237	1576	0.8693
Column accuracy	0.9177	0.9147	

Table 3 Classification accuracy using MFNN with 57-80-2 architecture

	Classified legitimate	**Classified spam**	**Row accuracy**
Actually legitimate	2677	111	0.9602
Actually spam	198	1615	0.8908
Column accuracy	0.9311	0.9357	

Table 4 Classification accuracy using MFNN with 57-70-2 architecture

	Classified legitimate	Classified spam	Row accuracy
Actually legitimate	2681	107	0.9616
Actually spam	203	1610	0.8880
Column accuracy	0.9296	0.9377	

Experiment 3:

Looking to the contribution each variable is making to the decision making process in Figure 2, we can find that some variables are extremely significant to the output of the network, while others have little effect. We have 57 variables, so we can remove 6 variables which have little effect to the final output. Now we will have 51 input variables. We still choose the standard nets with three layers and according to the NeuroShell2, the default number of hidden neurons in slab2 is 80. Also we set the learning rate 0.1, momentum rate 0.1 and initial weight 0.3, which are all the same with the previous experiments. The parameters in pattern selection are also set as before.

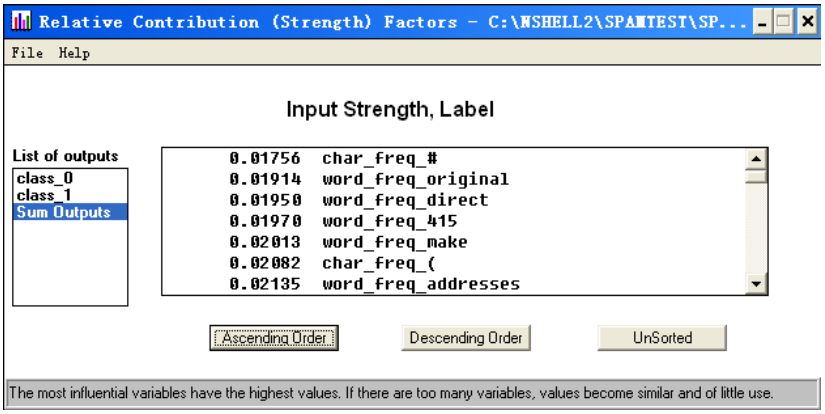

Figure 2 variable contribution

Table 5 Classification accuracy using MFNN with 51-80-2 architecture

	Classified legitimate	Classified spam	Row accuracy
Actually legitimate	2658	130	0.9534
Actually spam	201	1612	0.8891
Column accuracy	0.9297	0.9254	

From the results summarized in Table 5, we can find that there is not significant improvement compared to the 57-80-2 MFNN architecture, but the percentage of spam emails that are detected is slightly improved compared with 57-90-2 architecture and 57-100-2 architecture. But when compared with 57-70-2 architecture, the column accuracy level of classifying an email as a spam mail has declined.

Experiment 4:

Based on experiment 3, we try to change the activation function in the hidden layer. So, we have 51 input neurons and 80 hidden neurons, the output neurons are 2. Here we change the activation function in slab 2 from logistic to Gaussian function, the other parameters stay the same. Figure 3 shows the training set average error. The error tends to be stable at the error level of 0.3, which is quite high compared to all the previous experiments. The results are summarized in Table 6, and it seems that classifying the spam mails with this dataset is not suitable to be solved with Gaussian activation function.

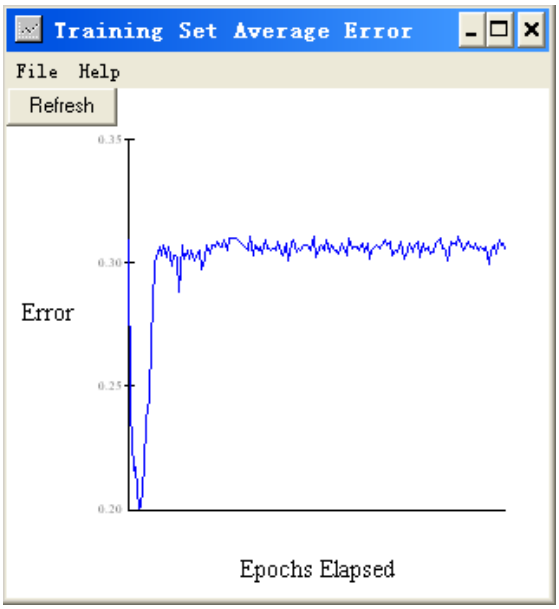

Figure 3 Training set average error of experiment 4

Table 6 Classification accuracy using MFNN with 51-80-2 architecture and Gaussian activation function in hidden layer

	Classified legitimate	**Classified spam**	**Row accuracy**
Actually legitimate	2623	165	0.9408
Actually spam	336	1477	0.8147
Column accuracy	0.8864	0.8995	

Experiment 5:

After changing the inputs and architectures, we can then alter the architecture to contain only a single output neuron which is attempting to classify the mail as either a ‘0’ (legitimate) or a ‘1’ (spam). The other parameters stay the same as the previous experiments. We decide the mail

is likely to be spam if the output of the network is greater than 0.5 and legitimate if the output is less than 0.5. From the results shown in Table 7, we can find that only 158 of the 1813 known spam mails are incorrectly classified as legitimate, which reach the highest accuracy level.

Table 7 Classification accuracy using MFNN with 57-80-1 architecture and decision threshold of 0.5

	Classified legitimate	**Classified spam**	**Row accuracy**
Actually legitimate	2670	118	0.9577
Actually spam	158	1655	0.9129
Column accuracy	0.9441	0.9334	

Based on this architecture, we try to change the value of the learning rate to see the influence on classifying results. If the learning rate is set too high, the algorithm can oscillate and become unstable. If the learning rate is too small, then the neural network will take long time to converge. We set the learning rate from 0.1 to 0.3 and 0.5, and other parameters remain unchanged. The average accuracy level of learning rate of 0.3 and 0.5 actually do not improve the results of learning rate of 0.1. Table 8 shows the results of the learning rate of 0.5 which have better performance than learning rate of 0.3. In conclusion, the learning rate of 0.1 is better.

Table 8 Classification accuracy using MFNN with 57-80-1 architecture and learning rate of 0.5

	Classified legitimate	**Classified spam**	**Row accuracy**
Actually legitimate	2643	145	0.9480
Actually spam	191	1622	0.8946
Column accuracy	0.9326	0.9179	

Since changing learning rate do not improve the performance, we try to change the momentum rate. The network which has the same network layers with higher momentum rate may give better performance (Sivanadyan and Thiagarajan, 2003). First, we keep the learning rate as 0.1 and the initial weight as 0.3, then we set the momentum to 0.5 and 0.8. Comparing with the situation that the value of momentum is set as 0.5, the value of the momentum set as 0.8 has a better performance, we show the results in Table 9. Higher momentum rate actually has improved the accuracy level in the dimension of classifying 1632 spams from the whole 1813 spam mails. But the momentum rate of 0.1 still has the best performance.

Table 9 Classification accuracy using MFNN with 57-80-1 architecture and momentum rate of 0.8

	Classified legitimate	Classified spam	Row accuracy
Actually legitimate	2670	118	0.9577
Actually spam	181	1632	0.9002
Column accuracy	0.9365	0.9326	

Experiment 6:

Probabilistic neural network is a powerful model which employs a genetic learning algorithm. PNN may take longer time to train but have the advantage of generalizing well on external data. In our experiment, the input neurons are 57. First we choose the iterative and non-adaptive training criteria. Then we set the maximum upper value for search to 1.6. We get the results in Figure 4. Then we change the criterion and choose genetic and adaptive learning method. It will train for 75 generations before it is stopped by the algorithm due to lack of performance improvement. This process takes an excessively long time. The results are shown in Figure 5. From the true positive proportion, we find that the accuracy of classifying legitimate mails are extremely high which are 0.9702 and 0.9889, but the performance of classifying spam emails is not satisfying. The percentage of the legitimate mails (60.6%) is higher. So PNN is not quite suitable for decreasing the number of false positives (legitimate messages classified as spam).

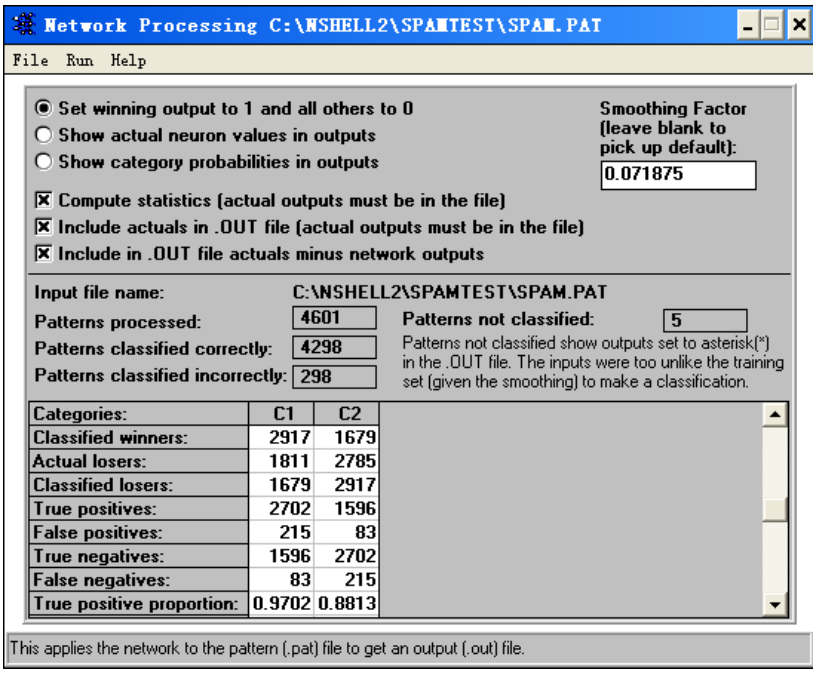

Figure 5 Classification accuracy using no-adaptive PNN with 57-4601-2 architecture

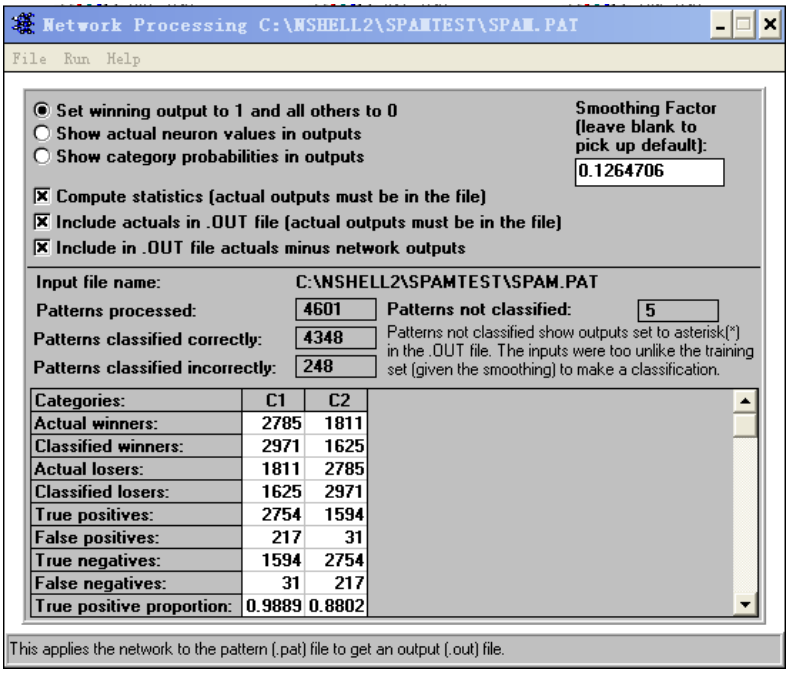

Figure 6 Classification accuracy using adaptive PNN with 57-4601-2 architecture

Experiment 7:

Expect using neural network model to classify the spam, there are still other machine learning algorithms can be used to solve classification problem like Bayesian classifiers (Yang et al., 2007). We choose the Naïve Bayes, SVM (Support Vector Machine), Logistic Regression, Random Forest to do a contrast experiment. As with the process mentioned above. We split the data set randomly with 80% as train set and 20% as test set. The input features are all 57 attributes, the two outputs are 0 as legitimate and 1 as spam. The experiment results are as follows.

Table 10 Naïve Bayes

	Classified legitimate	**Classified spam**	**Row accuracy**
Actually legitimate	2598	190	0.9319
Actually spam	339	1474	0.813
Column accuracy	0.8846	0.8858	

Table 11 SVM

	Classified legitimate	**Classified spam**	**Row accuracy**
Actually legitimate	2665	123	0.9559
Actually spam	213	1600	0.8825
Column accuracy	0.9261	0.9286	

Table 12 Logistic Regression

	Classified legitimate	**Classified spam**	**Row accuracy**
Actually legitimate	2666	122	0.9562
Actually spam	199	1614	0.8902
Column accuracy	0.9305	0.9297	

Table 13 Random Forest

	Classified legitimate	**Classified spam**	**Row accuracy**
Actually legitimate	2779	9	0.9968
Actually spam	32	1781	0.9823
Column accuracy	0.9886	0.9950	

4. Limitations

Neural Network is the one of the most popular and successful technique for pattern recognition, but its computation complexity is quite high due to its large amount of hidden neurons and propagation process. What's more, compared with Random Forest, Neural Network does not show enough strong generalization ability. Due to the amount of the instances and limitation of the features, neural network cannot learn the text features and relationship of different categories in spam emails.

5. Conclusion

From the experiments, we can find that, the row accuracy of actual legitimate classification is all high compared with the accuracy of actual spam, which indicates that NN can perform well in detecting legitimate mails due to sufficient learning instances. Among these, PNN achieve the highest accuracy level in row accuracy of actual legitimate classification. Except random forest, other machine learning techniques do not have better performance than NN in the row accuracy of actual spam classification.

References

El-Alfy, E. S. M., & Abdel-Aal, R. E. (2011). Using GMDH-based networks for improved spam detection and email feature analysis. *Applied soft computing*, *11*(1), 477-488.

Sivanadyan, T. (2003). *Spam? not any more! detecting spam emails using neural networks*. Technical report, University of Wisconsin.

Yang, Y., & Elfayoumy, S. (2007). Anti-spam filtering using neural networks and Baysian classifiers. *IEEE International Symposium on Computational Intelligence in Robotics and Automation* (CIRA 2007), 272-278.