CS 344 Spring 2018
Project Proposal
Andrew Li, Benjamin McGarvey, Joshua Rappaport

Introduction

Paul Grahams 2003 essay on Bayesian email spam filters laid the foundation of modern day spam filter programs by popularizing the use of Naive Bayes algorithm in conjunction with Bag of Words feature selection [2]. While the Naive Bayes classification algorithm is rather lightweight and comparatively easy to implement over other algorithms, research papers we read have demonstrated that the accuracy at which Naive Bayes detects spam ranges between 70% to 90%, fluctuating depending on the attributes and spam data set used [6]. While the substantial majority of spam is eliminated, the failure rate is still considerably high and led us to speculate whether there were more accurate algorithms that could be implemented in a similar amount of time as Naive Bayes.

Enter neural networks. They provide exceptional accuracy after training and show great promise in filtering content, albeit at the cost of a very high runtime [4,5]. Furthermore, while the concept of deep learning was formally defined in 1986, only recently has implementing most deep learning techniques become feasible by relying modern graphics processing units (GPU) to perform the bulk of the work. As a result, deep neural networks could be trained up to 10 to 20 times faster than before [1]. Due to the current pace that GPU architectures are progressing, many tech giants are especially optimistic about the future prevalence and feasibility of deep learning in the coming decade [1].

Due to the aforementioned hardware constraints making deep learning only recently possible paired with hardware limitations making neural networks possible but cumbersome, there has been very little research in the avenue of utilizing neural networks to detect and filter out spam [1,5]. We would like to seize this opportunity in an attempt to do something relatively new, as well as cultivate our shared love for machine learning on artificial intelligence after all having taken courses on the subject. We noticed that many of our sources noted that recurrent neural networks was a power deep learning method for text analysis, but we could not find a paper comparing recurrent neural networks to Naive Bayes when it came to comparing spam [2,3]. This leads to our desire to utilize this project to research the following question: Are recurrent neural networks more accurate in detecting spam emails than the more traditional Naive Bayes method?. We hypothesize that a recurrent neural network (RNN) will be more accurate than Naive Bayes at detecting spam emails, but the runtime may make Naive Bayes the better current option. While this may be true, we hope that we can prove that RNNs are more accurate than Naive Bayes, meaning that eventually faster GPUs will make RNN the better option in the near future.

Background readings

1. Accelerating AI with GPUs: A New Computing Model (2016) This comes from Nvidias website and was primarily utilized as a source for some of the points we made in our introduction. This official blog post by the company provides a history on the abrupt popularity of deep learning, primarily stemming from the advent of faster GPUs performing a great bulk of the work. This source also emphasizes the capabilities and advances made in deep learning and neural networks

over the previous decade, along with future implications of deep learning on humanity.

2. Machine Learning for E-mail Spam Filtering: Review, Techniques and Trends (2016) This article discusses the trends and patterns of email spam filters throughout its short history. It notes the prevalence and accuracy of filters based on the Naive Bayes algorithm. Naive Bayes classifies emails based on the frequency of specific features within. These features are, in other words, the individual words in the email body. Combined with non-machine learning features such as blacklisting certain email addresses, Naive Bayesian spam filters are powerful tools in combating unwanted email. The article does not discuss the possibility of using Neural Networks for spam filtering, indicating that the articles reviewed here did not bring up the concept at all. This is interesting, considering that the paper itself was written in 2016. Neural Networks have been used in Natural Language Processing since at least 2012.

3. Towards Accurate Deceptive Opinion Spam Detection based on Word Order-preserving CNN Siyuan Zhao, et. al. describe a modified CNN agent that detects what they termed as opinion spam within messages. Deceptive opinions are any opinions that preach or defame a product to mislead consumers points of views and behaviors with fictitious opinions, deliberately written to sound authentic. They found that an agent that takes word order into account performs better than an agent that does not. It also exposed a flaw with using Convolutional networks over Recurrent ones, namely that the former requires inputs to be of a fixed size. This renders CNNs inefficient for our purposes, as the inputs into the network are, by design, going to be of different lengths. Simply put, some emails are longer than others.

4. Recurrent neural network based language model (2010) Mikolov Et. Al. demonstrates the use of a RNN to model sequential language data. In particular, the article touts RNNs ability to handle arbitrarily long sets of sequential data. We studied this paper to understand how the input, hidden, and output layers of an RNN functioned.

5. End-To-End Spam Classification With Neural Networks (2016) This paper focused on using Convolutional Neural Networks for spam classification and found that they are very good at filtering spam. One problem that it notes is the fixed length of the input. In the case of this classifier, they used the first n_c characters, where n_c was either 100, 500, or 1000. We believe that examining the whole length of the email will make the classifier more accurate.

6. ”Analysis of Nave Bayes Algorithm for Email Spam Filtering across Multiple Datasets” and ”Email spam detection: A method of metaclassifiers stacking”

These two sources were studies done in which Naive Bayes was used to classify spam emails using the WEKA tool. The accuracies referenced in our introduction for Naive Bayes come from here. The first article used Spam Data (9324 e-mails and 500 attributes) and SPAMBASE (4601 email messages and 58 attributes). The second article also used SPAMBASE, but also other decision trees along with Naive Bayes. Varying on the data set and stop words when extracting attributes, the two papers had various trials that fluctuated between 70% and 90%.

Methodology

The input for our classifiers will be the training set of emails. We will preprocess the emails by stemming them and eliminating stop words. Thus, we will have two sets of preprocessed emails: Spam, and non-Spam. We will label these as such. After, we will create two bag-of-words models out of the preprocessed email sets. These models will contain a count for every unique word in the set. Thus, we will have the frequencies of every word in a spam and non-spam context.

We plan on building two spam classifiers: one utilizing Naive Bayes (our baseline) through the Python scikit-learn package, the other utilizing a recurrent neural network of our own design. Naive Bayes functions by looking at each word, and calculating the probability of that word being either in the spam, or non-spam set. These probabilities are then combined to create the full probability of the email being either spam or non-spam. The most likely classification is then output. The training input for the Naive Bayes classifier will consist of a 75% of the spam and non-spam emails in the preprocessed data set. It will then be tested on the other 25% of emails.

The Recurrent Neural Network, however, requires a bit more finesse. Using a one-hot-encoding of each stemmed word, we will pass them through the network. One of the benefits of RNN is that it accepts inputs of any size. The network will parse the sequence, and will eventually output a vectorized classification as spam or non-spam (either 0 or 1). We will use some form of backpropagation to train the network based on accuracy.

To test both of these networks, we will perform two analyses: One will be based on how well they do on the test set in SpamAssassin. The other will be based on their success at classifying an unrelated email set. This should give us an idea as to how much the network learned, and how applicable that knowledge is to new spam.

This is what we plan our results table to look like:

	SpamAssassin	TREC 2007 Public Corpus
Naive Bayes	XX%Accuracy	XX%Accuracy
RNN	XX%Accuracy	XX%Accuracy

Timeline

4/5 : Finish structure of project and background research.
4/24 : Finish data gathering, begin writing report.
5/1 : Finishing touches on report.
5/8 : Presentation.

For how we will be splitting up the different tasks, we will have Ben and Andrew on the Bayes classifier, Ben and Joshua on the RNN classifier, finally Joshua and Andrew on the preprocessing of data.

Andrew will have the responsibility of running the whole enterprise, and we all will have each other to keep working on each task that we are a part of.

References

1. Huang, J ”Accelerating AI with GPUs: A New Computing Model” (2016),
   https://blogs.nvidia.com/blog/2016/01/12/accelerating-ai-artificial-intelligence-gpus/

2. Bhowmick, A ”Machine Learning for E-mail Spam Filtering: Review, Techniques and Trends” (2016)
   https://arxiv.org/pdf/1606.01042.pdf

3. Zhao, S Et. Al. ”Towards Accurate Deceptive Opinion Spam Detection based on Word Order-preserving CNN”
   https://arxiv.org/pdf/1711.09181.pdf

4. Mikolov Et. Al. ”Recurrent neural network based language model” (2010)
   http://www.fit.vutbr.cz/research/groups/speech/publi/2010/mikolov_interspeech2010_IS100722.pdf

5. Lennan C. ”End-To-End Spam Classification With Neural Networks” (2016)
   https://amor.cms.hu-berlin.de/ jaehnicp/project/spam-filter/cnn_report.pdf

6. Nurul Fitriah Rusland et al ”Analysis of Nave Bayes Algorithm for Email Spam Filtering across Multiple Datasets”
http://iopscience.iop.org/article/10.1088/1757-899X/226/1/012091/pdf

Mi ZhiWei, Manmeet Mahinderjit Singh, & Zarul Fitri Zaaba. (2017). ”Email spam detection: A method of metaclassifiers stacking”
http://icoci.cms.net.my/PROCEEDINGS/2017/Pdf_Version_Chap16e/PID200-750-757e.pdf