Weakly Supervised Spam-Label Classification

Garrett T. Birch
Data Science Undergraduate
University of California - San Diego
La Jolla, 92093
gbirch@ucsd.edu

Ke Xu
Data Science Undergraduate
University of California - San Diego
La Jolla, 92093
k6xu@ucsd.edu

Zairan Xiang
Data Science Undergraduate
University of California - San Diego
La Jolla, 92093
zaxiang@ucsd.edu

1 Abstract

We are interested in identifying spam emails and messages in general given a set of seed words and classes in the weakly supervised text classification setting. We are mostly interested in this because of the emergence of spam messages that continue to improve and bypass current spam filter detection systems. Furthermore, we want to be able to classify what kind of spam messages are coming in to get an idea of the pattern of each type of spam message. We are hoping detecting patterns could leave to money saving and seeing how these messages evolve can provide some insight into how to do so. Phishing and other forms of scams cost United States citizens billions of dollars a year as companies try to keep up against these attacks. The Federal Trade Commission recently published that consumers lost nearly 8.8 billion dollars to scams in 2022 (2). So, we want to be able to help non-tech-savvy people by limiting their interactions with these potentially harmful messages.

2 Introduction

2.1 Narrow Problem Statement and Methods

Previous works have attempted to implement effective spam filtering tools and some are successfully being used in E-mail services such as Gmail. These filters are good at finding most spam emails, but they don't tell the user what specific spam they are receiving, so we are interested in providing this information to the users by further classifying the types of spam that a particular user receives, hoping to facilitate users to realize what a typical message of a certain type of spam can look like.

The investigation into spam email filtering is closely related to the technology being introduced in Quarter 1. We will utilize the NLP machine learning framework learned in Quarter 1, ConWea, to implement this NLP application(1). Moreover, we will reproduce the models TF-IDF and Word2Vec implemented in the Quarter 1 project as our baseline models for the Quarter 2 project. After doing so, we will look at improving these models with better word embeddings using FastText. Next, we will be using large, more complex language models such as ConWea, BERT, and potentially more to see if these large language models improve upon those baselines - showing if they are worth using or not. We are hopeful that ConWea, with the provided contextualization of words, will lead to a better classifier.

Preprint. Under review.

Furthermore, the setting of weakly supervised text classification will alleviate the human burden of annotation future massive datasets filled with spam messages. The trends found from a subset of documents that were labeled should be representative of future documents the classifier may see.

2.2 Deliverable

Our group aims to deliver a report, discussing these details further and diving deeper into a discussion about these models and their performance. The report will also include a model deployment section that gives a user guide to use the spam filter tool with special consideration on the potential data ethics issue. We would also like to put out a website as a platform for model deployment, after model training, that would allow a user to input a message and receive feedback into the message - such as if it was spam or not and if it is spam; what kind of spam it is. Our interactive website with instructions on how to set it up can be found here. We do not host the interactive website 24/7 due to hosting costs.

3 Dataset

To do so, we will be leveraging the most reliable data set in terms of email messaging known as Enron. This data set contains 6 different datasets, each has around 6000 emails, and 1/4 of them are spams [1]. We had to annotate all instances of spam and classify them in a certain way to fine-grain the data into more detail about which spam it is. The leveraging of our own time was heavily devoted to the annotation process, but still human errors are bound to exist within the annotations. We will do all of this by considering models that range from baseline models to complex language models. Only one of these data sets was annotated fully to use in training, while the others are used primarily for testing our models prediction accuracy on completely unseen messaging patterns. The topics extracted from the spam messages were split up and annotated into following categories;

1. Medical Sales
2. Investment
3. Phishing
4. Sexual
5. Insurance
6. Software
7. Other

The Enron data set provides an already preprocessed version of the text into their text files, but due to the nature of spam emails a lot more preprocessing techniques must be put into place to extract the actual information from messages. More specifically, people are trying to bypass current detection systems by adding filler words that have nothing to do with the message, changing out letters for numbers like I for ! or L, and patterns similar to that. By using strong preprocessing techniques such as lemmatizing and stemming words, we are able to extract the true meaning behind each message and further classify them into the categories listed before.

4 Seed Words

Through our model of weakly supervised text classification, we will use a variety of seed words that correspond to each label of spam.
Mathematically, we are given a set of n messages $M = \{M_1, M_2, ..., M_n\}$, a set of i labels $L = \{L_1, L_2, ..., L_i\}$ and finally we are constructing a set of j seed words $S = \{S_1, S_2, ..., S_j\}$.
To construct the set S, we look at the manually annotated set of messages from the data and find common words. Of course to not overfit to this dataset only, we also made sure the seed words generated all made sense in an overall picture of this label. An example of this is for the medical sales label, a consesus was agreed upon by both us as humans and what the data told us through common words that seed words like *drug, pain, pill* would work well.

[1] https://www2.aueb.gr/users/ion/data/enron-spam/

The current seed words are being used. However, they may be changed in future versions as preprocessing techniques develop and change. Note: Other does not have seed words as it means the spam could not be classified further into our categories.

Category	Seed Words												
Insurance	credit	cash	home	mortgage	rate	loan	refinance						
Investment	stock	market	price	invest	interest	statement	secur	base	risk	trade			
Medical Sales	pill	buy	medication	drug	prescription	med	doctor	viagra	pain	effect			
Phishing	lottery	win	bank	award	money	confidenti	agent	winner	prize				
Software Sales	software	price	microsoft	adobe	office	system	photoshop	window	offer	download	server		
Sexual	sex	horny	date	free	woman	girl	video	porn	adult	cheat	hottest	teen	cum

Table 1: Seed Words

5 Models

5.1 IR-TF-IDF

Prior to fitting the models, we changed all sentences to be lower-cased. Removed punctuation, trailing white spaces and stop words using NLTK.
We started with baseline models to get an idea of how well they will perform in relation to some of the more complex models. First, we leveraged a classical method known as IR-TF-IDF - which put simply refers to counting up the number of seed words in each class and picking the class that has the highest occurrence in the message.

- TF-IDF calculation: For each seedword, calculate its TF-IDF value with respect to a specific document using the following formula;

  $$tfidf(t,d) = tf(t,d) * idf(t)$$

  Where

  $$tf(t,d) = \text{\# of times } t \text{ appears in } d$$

  and

  $$idf(t) = log(\frac{\#\ documents}{\#\ documents\ where\ t\ appears}).$$

  For each document, sum up all TF-IDF values for all seedwords within a label, and assign the document with the label that has the highest TF-IDF sum.
- Micro/Macro F1 calculation: Use sklearn.metrics.f1_score to derive Micro and Macro F1 scores, respectively.

5.2 Word2Vec

As our next baseline model, we implemented another classical technique Word2Vec.

- Word2Vec Model Training: Initialize a Word2Vec vector from gensim and specify size = 110, window = 5, min_count = 1. Use 4 workers and train for 800 epochs.
- Cosine similarity calculation: For each seedword, fetch the corresponding vector from Word2Vec model. Take the average of all vectors within a label and use that as the final word vector. For a single document, simply take the average of all word vectors within that document. Compute the cosine similarity between a document and a label using the following formula

  $$cos(doc, label) = \frac{doc \cdot label}{||doc||||label||}$$

  Where doc represents the word vector for a document, and label represents the word vector for a label. Assign the document with the label that has the highest cosine similarity.
- Micro/Macro F1 calculation: Use sklearn.metrics.f1_score to derive Micro and Macro F1 scores, respectively.

5.3 FastText

We used the word vector representations trained by unsupervised FastText for prediction. FastText (3), created by Facebook AI, is a lightweight tool that takes in to account the structure of the words - producing better and more efficient word embeddings compared to Word2Vec. We then use these word embeddings in the same way as we did above for Word2Vec.

Model	Micro-F1	Macro-F1
IR-TF-IDF	0.69	0.67
Word2Vec	0.70	0.72
FastText	0.71	0.72
ConWea	0.75	0.75

Table 2: Prediction results.

- FastText Model Training: Initialize a FastText model and specify lr=0.05, wordNgrams=5, loss='hs', dim=50. Train for 600 epochs.
- Cosine similarity calculation: The calculation and prediction procedure is similar to that of Word2Vec, the only difference is that we use different word embeddings.
- Micro/Macro F1 calculation: Use sklearn.metrics.f1_score to derive Micro and Macro F1 scores, respectively.

5.4 ConWea

Our final model is ConWea, which is a weakly supervised classification model that automatically assigns labels to the training data, which was developed by our mentor Jingbo Shang. As we know with human language, context is a huge factor in how a word should be portrayed. ConWea looks to solve this problem by providing contextualized weak supervision for text classification. ConWea uses BERT vectors along with a Hierarchical Attention Network (HAN) classifier to make predictions.

- ConWea Model Contextualization: Encode both the word occurrences and the user provided seed words to create a contextualized corpus using BERT vectors (Mekala & Shang, ACL 2020). This means different meanings of the same word can be used for different labels.
- ConWea Model training: Unlike traditional text classification methods, ConWea is not fed with labels during training. Instead, it generates its own labels with the contextualized documents and train a HAN classifier with the labels to proceed with text classification (Mekala & Shang, ACL 2020).
- Micro/Macro F1 calculation: In the training stage, ConWea runs for 6 iterations and outputs the Micro/Macro F1 scores for each iteration. Each iteration contains a model with different input layers. The model with the best performance has 4 layers and we recorded its Micro/Macro F1 scores correspondingly.

For more extended details about the ConWea model, refer to the paper mentioned. (1)

6 Results

Prediction results are shown in Table 1. The overall performance of these models are similar (around 0.7) but do increase as we use bigger models. We expect both micro and macro F1 scores to be higher if we apply more large and complex models such as BERT and ConWea. It now falls on the company or user to decide if the increase of F1 scores is worth the computational costs of running larger models.

7 Conclusion

As we can see from the results, as we increase our model's complexity we are achieving a better accuracy even on unseen data. Experiments showed that combining Word2Vec with better word embeddings through FastText provided the best results so far. We expect in the future as our data preprocesing techniques improve along with more advanced models that we will produce better results further down the road. It comes down to whether or not a company thinks that approximately 3 percent better accuracy across classes matters, and for big companies that use Spam Filters like Google we'd assume they would probably invest in these bigger models for the sake of their consumers. For other simple scenarios, we believe that just using FastText word embeddings provide enough accuracy at little computational cost. Some limitations we faced included; the dataset. The Enron dataset is one of very few public email datasets available, and the data included in the Enron dataset follows a certain pattern since these messages were only collected from a certain period of time. Thus, our models pick up on these patterns and produce results based on this. Moreso, with no dataset having

labels of what time of spam it is we had to humanly annotate 4500 documents and use only those as our training data. Human annotation error is present in this dataset, but is kept as minimal as possible. However, in real world scenarios spam is endlessly evolving and trying to get around these filters so we can expect some messages to be misclassified, especially if a user inputted message is much different than what the model has seen with the data. Most of our limitations is due to what data is available for this type of work. We assume that in practice, places like Google with G-mail, have access to a lot more data that they can train on and explore newer layouts of spam. In conclusion, we are satisfied that we were able to given just a set of labels with a small subset of words that represent each later, we are able to classify most documents as a label with a balanced accuracy of about 75 percent.

References

[1] Mekala, Dheeraj, and Jingbo Shang. "Contextualized weak supervision for text classification." Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020.

[2] Staff, the Premerger Notification Office, and Stephanie T. Nguyen. “New FTC Data Show Consumers Reported Losing Nearly $8.8 Billion to Scams in 2022.” Federal Trade Commission, 23 Feb. 2023, https://www.ftc.gov/news-events/news/press-releases/2023/02/new-ftc-data-show-consumers-reported-losing-nearly-88-billion-scams-2022.

[3] A. Joulin, E. Grave, P. Bojanowski, T. Mikolov, Bag of Tricks for Efficient Text Classification. https://arxiv.org/abs/1607.01759