# Web Retrieval and Mining Final Project Report

## Sponsored Posts Recognizer for Restaurant Review

[1]鄭筱樺, [1]王藝霖, [1]林煦恩, [1]曹峻寧

[1]Dept. of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan

***Abstract* — Nowadays, restaurant reviews are not only written by professional food critics, but also by the general public. There are many food bloggers who write reviews after visiting a restaurant, however, some of these posts are sponsored by the restaurant owner and thus considered less convincing. People may want to know whether the post they are viewing is sponsored, but it is often hard to distinguish. In this project, we aim to build a sponsored posts recognizer that given a restaurant review on social media, it can help to recognize whether the post is sponsored.**

***Keywords — Neural Network; Sponsored Posts Recognition***

## I. Introduction

### *A. What is sponsored post*

Normally, a restaurant review which tells both pros and cons are written after consuming at the restaurant. While a sponsored post is sponsored by the restaurant owner as an advertisement to promote their products and services.

Since the bloggers are sponsored, they may not honestly expose all their thoughts about the food and services to conceal the weakness. Thus, sponsored posts are considered less reliable. Readers may want to known whether the post they are reading is sponsored or not to make sure the information is symmetry.

### *B. Motivation*

Before we choose a restaurant, we probably want to refer to some reviews on the internet to prevent from overvalued restaurants. Recently, there are few sponsored posts not telling the truth that they are sponsored, but pretend to be normal ones. A reader of a post should be informed if the post is a sponsored one. Since the sponsored review may not be that trustworthy, readers should have right to judge by themselves.

Those "fake reviews" are hard to recognized because they attend to disguise. Readers might be mislead and then visit the restaurant. Finally find out the restaurant is not as good as the "fake reviews" says. What we want to do is to prevent those situation from happening. The purpose of our work is to recognized sponsored post, which can help us make our decision.

## II. Methodology

To build a automatic recognizer, a straightforward way is to define useful rules based on the characteristics of sponsored posts. To define the rules, we try to extract the features potentially contain helpful information. Although there are many features can be used, such as posting time or tags, article comments, author information, etc. After observation, we think the key to recognizing sponsored posts lies in the post content, which includes both words and images. Therefore, we extract both linguistic and visual features to represent the post. However, it's hard to recognize sponsored posts even to human, so is to design how to utilize these features to build a recognizer by hand-crafted rules. Therefore, we decide to apply machine learning to learn how to do the recognition without defining hand-crafted rules.

### *A. Word2Vec*

To produce word vectors, Word2Vec is a well-developed model and is widely used in Natural Language Processing field nowadays. Thus, we choose to utilize the pre-trained word2Vec from Kyubyong's github [1] to get the word embeddings which are pre-trained on Wikipedia. After mapping each term in a post to its embedding, we average over the vector of words in a sentence. Therefore, each sentence has its vector representation and a post is composed of rows of sentences. We then put these vectors into a CNN (Convolutional Neural Network) to train the classifier.

### *B. Doc2Vec*

Besides Word2Vec, we also apply Doc2Vec to generate vector representation of each document, which is similar with Word2Vec. In fact, Doc2Vec is composed of 2 different methods: Distributed Memory (DM) and Distributed Bag of Words (DBOW), where DM predicts words given context and a paragraph vector and DBOW predicts a group of words given only a paragraph vector.

Unlike our approach on Word2Vec where each document is represented by a list of sentence vectors, we concate 2 paragraph vectors (or document vectors) trained by DM and DBOW to represent a document.

After generating the paragraph vectors, the classifier is

trained by several different methods, including SVC (Support Vector Classification), naive bayes, decision tree and linear regression model. We'll compare the performance of these methods in evaluation section.

### C. TFIDF

TFIDF is a general way for texts retrieval. The TFIDF value reflects how important a word is to a document. Firstly, we select k terms with highest document frequency, and then represent each document with a k-dimensional vector. Each dimension is either the TFIDF value or zero depending on whether the corresponding term is in the document. Then, we can view each document as a vector with TFIDF of important terms. We train the classifier by several different methods as in Doc2Vec.

$$TF(w,d) = f_{w,d} / \#\ of\ words\ in\ d$$

$$IDF(w) = \log N / |\{d \in D : w \in d\}|$$

### D. Linguistic-Visual Hybrid Features

#### 1) Visual Features

Many blog posts contain images of either the restaurant environment or the food, and some bloggers also put their selfie. Image is an important component of a post, so we extract some visual features to consider it. Since some posts contain so many images, we set a threshold = 10 to limit the number of crawled images for each post. And, apart from pure visual features, we also store the exact number of images in the post as a feature.

- Human Faces

Images in sponsored posts are often provided by the restaurant. Thus, it is less likely to contain selfie in a sponsored post. We perform human face detection using Haar feature-based cascade classifiers [2] provided by openCV. Taking the list of boolean results that whether each image of the post contains human face as features.

- Image Sharpness

Images in sponsored posts usually have better "quality". However, it is ambiguous on how to define image quality. We use sharpness to define the quality that can filter out some blurry images hit by unprofessional bloggers. We utilize the variation of the Laplacian by Pech-Pacheco et al [3] to compute the sharpness.

#### 2) Model

Firstly, we pre-train the CNN model and get the weights between each layer. Then we extract the output of max-over-time pooling layer, which is 32-dimensional vector. Since each dimension of the vector represents an hidden feature, we try to append 3 rule-based features that may be useful (number of images, average probability of faces appearing in the images of a post, and average sharpness of the images) to that vector. At last, using that vector as a new representation of a document, we train an NN to classify documents.

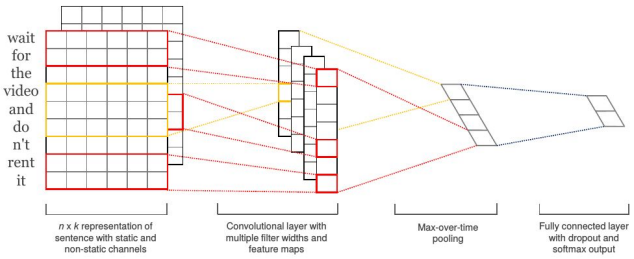


Fig1. an illustration of the CNN, from Yoon Kim's work. [4]

## III. EXPERIMENTS

### A. Data Set

#### 1) PTT (Training and Validation)

PTT is a popular BBS (Bulletin Board System) in Taiwan, with over million users and tens of thousands of new posts every day. Most posts are written in Chinese, and there are many boards for different topics, such as the Food board which talks about restaurants. According to the rule of the Food board, the author should specify whether the post is sponsored, so the posts on PTT are well labeled. Thus, we crawl 10000 no-sponsored posts and 5603 sponsored posts in the Food board to be our training data.

#### 2) PIXNET BLOG (Testing)

PIXNET is a biggest blogging platform in Taiwan. According to Alexa.com, PIXNET is ranked 2nd. Most posts are also written in Chinese. We crawl 8671 posts on category of food, but most of them is unlabeled. We developed a simple website for labeling, and make human labeling for 479 posts. Since the labeled data is not enough for training, we only use PIXNET data for testing.

kappa value = $\frac{6+1+1+4+1+2+1}{8+20+5} = 0.48$ (which is an accepted value)

TABLE I. KAPPA

| Rater A | Rater B | | | | Total |
|---|---|---|---|---|---|
| | 0 | 1-2 | 3 | 4-5 | |
| 0 | 8 | 6 | 0 | 0 | 14 |
| 1-2 | 1 | 20 | 1 | 4 | 26 |
| 3 | 0 | 1 | 0 | 0 | 1 |
| 4-5 | 0 | 2 | 1 | 5 | 8 |
| Total | 9 | 29 | 2 | 9 | 49 |

### B. Evaluation

#### 1) Cross Entropy Loss

$$p = (a,\ 1-a),\ q = (b,\ 1-b)$$

$$H(p,q) = -\Sigma_i p_i \log q_i$$

$$= -a\log b - (1-a)\log(1-b)$$

2) *Accuracy*

$$acc = \frac{number\ of\ correctly\ predicted\ posts}{number\ of\ ptt\ posts}$$

### C. Experiments

Validation set is composed of 30% of PTT posts and training set is the other 70%. On the other hand, testing data comes form 479 labelled PIXNET blog posts.

1) *Word2Vec*

After trying CNN models with different numbers of filters, different filter size, and dropout probability, we know that 1 layer of convolutional is enough to generate good result on validation data.

| model | cov layer num | filter num | filter size | cov layer num | valid acc | test: loss | test: acc |
|---|---|---|---|---|---|---|---|
| cnn | 1 | 28 | 3 | 0.3687 | 0.8512 | 0.7510 | 0.6285 |
| cnn | 1 | 32 | 3 | 0.3072 | 0.8778 | 0.7823 | 0.6239 |
| cnn | 1 | 32 | 3 | 0.3218 | 0.8752 | 0.7470 | 0.6285 |
| cnn | 1 | 40 | 3 | 0.3203 | 0.8747 | 0.7830 | 0.6376 |
| cnn | 2 | 32 | 3 | 0.2858 | 0.8843 | 0.8001 | 0.6421 |

2) *Doc2Vec*

The SVC model performs the best in this experiment. We can observe that the larger epoch leads to the higher accuracy. Besides, the size of paragraph vector also has a slightly impact on it.

| model_type | epoch | size | window | valid: loss | valid: acc | test: loss | test: acc |
|---|---|---|---|---|---|---|---|
| svc(rbf) | 1 | 100 | 8 | 10.2120 | 0.7043 | 15.6567 | 0.5490 |
| svc(rbf) | 1 | 200 | 8 | 10.2194 | 0.7041 | 15.4994 | 0.5513 |
| svc(rbf) | 1 | 128 | 8 | 10.0349 | 0.7086 | 15.4994 | 0.5513 |
| svc(rbf) | 1 | 64 | 8 | 9.7103 | 0.7195 | 15.1847 | 0.5604 |
| svc(rbf) | 1 | 64 | 10 | 9.6807 | 0.7223 | 14.9486 | 0.5672 |
| naive base | 1 | 64 | 8 | 10.4702 | 0.6969 | 17.4662 | 0.4943 |
| decision tree | 1 | 64 | 8 | 11.7245 | 0.6605 | 15.9713 | 0.5376 |

3) *TFIDF*

According to the following experiment, we discover that SVC with document frequency threshold 800 performs the best.

TABLE II. TFIDF

| model_type | df threshold | tfidf dim | valid: loss | valid:acc | test: loss | test: acc |
|---|---|---|---|---|---|---|
| svc(kernel=rbf) | 800 | 522 | 6.3922 | 0.8149 | 12.3504 | 0.6424 |
| svc(kernel=linear) | 800 | 522 | 6.3922 | 0.8149 | 12.3504 | 0.6424 |
| svc(kernel=rbf) | 900 | 456 | 6.3621 | 0.8158 | 11.7224 | 0.6606 |
| svc(kernel=linear) | 900 | 456 | 6.3771 | 0.8154 | 11.8271 | 0.6576 |
| svc(kernel=rbf) | 1000 | 370 | 4.8093 | 0.8608 | 11.0945 | 0.6788 |
| svc(kernel=linear) | 1000 | 370 | 4.8093 | 0.8608 | 11.0945 | 0.6788 |
| svc(kernel=rbf) | 1100 | 353 | 10.0406 | 0.7093 | 15.1763 | 0.5606 |
| svc(kernel=linear) | 1100 | 353 | 10.0255 | 0.7097 | 15.1763 | 0.5606 |
| naive bayes | 700 | 557 | 9.1963 | 0.7337 | 16.0135 | 0.5364 |
| naive bayes | 800 | 522 | 9.0757 | 0.7372 | 16.4322 | 0.5242 |
| naive bayes | 900 | 456 | 9.1209 | 0.7359 | 16.1182 | 0.5333 |
| decision tree | 700 | 557 | 8.5631 | 0.7521 | 16.0135 | 0.5364 |
| decision tree | 800 | 522 | 8.5631 | 0.7521 | 15.9089 | 0.5394 |
| decision tree | 900 | 456 | 8.8043 | 0.7451 | 16.3275 | 0.5273 |
| linear regression | 800 | 522 | 9.7240 | 0.7185 | 14.9669 | 0.5667 |
| linear regression | 900 | 456 | 9.5732 | 0.7228 | 14.7576 | 0.5727 |
| linear regression | 1000 | 370 | 9.7390 | 0.7180 | 14.7576 | 0.5727 |

4) *Linguistic-Visual Hybrid Features*

The result is not so well. We may need to verify whether the image features we define is useful.

| valid: loss | valid acc | test: loss | test: acc |
|---|---|---|---|
| 0.3684 | 0.8539 | 0.6713 | 0.6376 |

## IV. CONCLUSION

### A. Result Discussion

Due to the experiment results, we find that Word2Vec outperforms other methodologies. The result may imply that linguistic is more useful to recognize sponsored posts.

### B. Future Work

1) *Data Labeling*

Sponsored posts are hard to recognized even for human, consequently the label may be ambiguous and strongly affect our testing result. Also, we don't have enough budget nor human resource, as a result we can not label enough data. With both high quality and quantity data, our work might be improved.

2) *Extended Application*

At this work, we tend to deal with the problem of recognize wrong information out of a set of data, and this idea can be apply to not only sponsored posts but also other situations, such as fake news recognition or content farm detection.

3) *Model Improving*

Use okapi bm25 instead of TFIDF.

## REFERENCE


[1] Kyubyong Park, Pre-trained word vectors of 30+ languages, Available at: https://github.com/Kyubyong/wordvectors

[2] Paul Viola, Michael Jones, "Robust Real-Time Face Detection", International Journal of Computer Vision, 2004.

[3] J. Pech, G. Cristobal, J. Chamorro, J. Fernandez, "Diatom autofocusing in brightfield microscopy: a comparative study", Pattern Recognition, 2000. Proceedings. 15th International Conference on, vol. 3, pp. 314–317, IEEE.

[4] Yoon Kim, "Convolutional Neural Networks for Sentence Classification", EMNLP 2014.