Predicting Restaurant Inspection Scores Based on Yelp Data and Sanitary Inspection Reports

Filipe Flores Monteiro

Abstract

The main goal of this work was detecting hygiene problems and violations of sanitary standards in restaurants, cafes, and bars, using information provided on location-based social networks. This is a serious and important problem due to the limited number of inspectors compared to the number of establishments. Inspectors are rarely helped by formal complaints, although many complaints are actually reported in the form of comments on social platforms, such as Yelp, TripAdvisor, or Zomato. This M.Sc. project explored machine learning techniques to build models capable of detecting establishments that violate health and hygiene standards, based on textual comments collected from social platforms (in this case, from the social platform named Yelp). In particular, the project explored text classification mechanisms based on deep neural networks, taking inspiration in recent work within the Natural Language Process (NLP) field. This project used an existing data set provided by the Yelp platform. Naturally, some data were collected by me to increase the number of reviews from each establishment. So, I used the Yelp API, which allowed me to search for as many reviews as possible (for each establishment), and helped support the training and evaluation parts of the methods to be developed.

1 Introduction

The systematic collection of health inspection data is essential for evaluating a restaurant, and is also a basis for preventing a number of food poison diseases. For these and other purposes, inspectors have to write sanitary reports containing the location and other data of the restaurant, together with textual descriptions for the reasons that influenced the evaluation of the restaurant, and an health inspection score. The analysis of restaurants also involves the classification given by the costumers. The main goal of Yelp is to inform new customers if a place that they have never been to has a good classification or not, based on information given by other customers. In particular, an automatic classification of Yelp data associated to sanitary reports would allow people to better know if some place has a good classification or not, and decide if this certain place deserves their time and money.

In this paper, I report the development of a method for the automatic classification of the reviews given by customers on Yelp, in order to attribute a health inspection score to each establishment. The project explored text classification mechanisms based on deep neural networks, taking inspiration on recent works that have advanced robust models, capable of obtaining good results in terms of the accuracy of the classifications and simultaneously offering mechanisms that allow interpreting the obtained results. With the introduction of deep learning techniques in the area of natural language processing (NLP), the representation of the texts to be processed is no longer based on sparse vectors encoding into a binary form the presence of words and/or n-grams of words. Instead, the representation is done through dense vectors (e.g., word embedding) that capture the semantic meaning of words. This work evaluated the application of modern deep learning methods, on the specific text mining problem of prediction health inspection scores.

I compared different approaches, based on Recurrent Neural Networks (RNN) or Convolutional Neural Networks (CNN). The final goal of this work was to understand how these methods for automatic classification would perform over these data envisioning near real-time monitoring of health inspection scores for
all types of establishments. The RNN model follows the hierarchical structure adapted from a proposal by Yang et al. (2016), which considers the input as sequences of words from different sentences, sequences of sentences from different reviews, and sequences of reviews from different establishments. In turn, the CNN model follows the deep pyramid structure of Johnson and Zhang (2017), as referred in Section ???. In this model, the input is a vector representation of words (e.g. word embeddings), which enters in conventional layers, goes through pooling layers (e.g. max pooling) and finally reaches a fully connected layer to make the final prediction. In order to evaluate the results, I used a classical regression metrics to compare the achieved prediction with the desired score. The Root Mean Squared Error (RMSE) measures the average magnitude of error and gives a quadratic scoring rule. RMSE will give a higher weight to large errors, which will be later reflected in a greater penalization in the final outcome. The Mean Absolute Error (MAE), also measures the average magnitude of the error in a set of predictions, without considering their direction. It is the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight.

The rest of this paper is organized as follows: Section 2 surveys important concepts and previous related work. First, an overview of fundamental concepts related to representing textual documents, and briefly introduces neural network models for text processing. Then, previous related work is discussed in detail, considering text mining methods for inferring health inspection scores, as well as deep learning methods for text classification. Section 3 presents the proposed approach, describing the architectures of the deep neural networks that were considered for the classification task. Section 4 presents the results of evaluation experiments. The section starts by presenting the data sets used in the experiments, together with the experimental methodology and evaluation metrics. Next, the section gives a detailed analysis of the results obtained in the experiments. Finally, Section 5 overviews the most important aspects of my research, summarizing the main findings of this work, and also presenting possible developments for future work.

2 Fundamental Concepts and Related Work

Text classification through supervised learning requires the use of techniques to represent text into parameters that can be an input to the learning algorithms. A common approach involves transforming text documents into vectors which represent them. Each position of one such vector reflects the frequency of a specific word in a document. TF-IDF represents how important a term \( x \) is to a document \( d \) in a collection. We combine the two definitions to produce a composite weight for each term in each document.

The networks give us the possibility to minimize the errors in the output (results), by directly training all input parameters. We have a feedback loop mechanism, known as back-propagation, which redefines the weights of the neurons (Goldberg, 2016). We can think of RNN as a type of network that works recursively. In addition to the initial input received in each neuron, RNN also includes, in each node, the hidden state that determined the last classification in a sequential series. This state is then combined with the new input to produce a new state. Such action is named back-propagation through time and it is performed during the training stage of the model. A feed-forward neural network has a function that is often used as a classifier that takes a vector \( x \) which represents (for example) words, as an input and predicts one or more output vectors that match with the input. The input vector propagates through the network (layer by layer), until it reaches the output node. A feed-forward neural network with one hidden-layer can be represented by:

\[
y = g(g'(xW^1 + b^1)W^2 + b^2)
\]  

(1)

In the previous equation, \( x \) is a vector of inputs and \( y \) a vector of outputs. The matrix \( W^1 \) and the vector \( b^1 \) represent, respectively, the weights matrix and the bias vector of the first layer, while \( W^2 \) and \( b^2 \) are the weight matrix and the bias vector of the second layer. Finally, the function \( g \) can be seen as an activation function. This model helps us to improve the training step as it will match equal features with similar vectors, thus the information is always shared between the same features. In recurrent neural
networks (RNN) we can observe four variables that make up the functions. Beginning with the input that is, an ordered list of vectors $X_{i:n}$ together with an initial state vector $S_0$, and returns an order list of state vectors $S_{i:n}$, as well as an ordered list of output vectors $Y_{i:n}$ (corresponding state of $S_{i:n}$), as we see in the Figure 1. The output vector $Y_i$ is then used for further prediction. In my master project, I will use the Gated Recurrent Unit (GRU). The advantage of this new method is the decreased number of gates. Represented by:

$$h_t = h_{GRU}(h_{t-1}, x_t) = (1 - z_t) \times h_{t-1} + z_t \times \tilde{h}_t$$

$$z_t = \sigma(x_t W^{xz} + h_{t-1} W^{hz})$$

$$r_t = \sigma(x_t W^{xr} + h_{t-1} W^{hr})$$

$$\tilde{h}_t = \tanh(x_t W^{zs} + (h_{t-1} \times r_t) W^{hz})$$

$$y_t = O_{GRU}(h_t) = h_y$$

(2)

Each gate has a simple meaning: compute the next hidden state ($h_t$) given the previous hidden state ($h_{t-1}$) and current input ($x_t$) using two gates. Update gate ($z_t$): How much information from past state is kept and how much new information is added. Determined based on an interpolation of the previous state and the proposals. Reset gate ($r_t$): How much the past state contributes to the new candidate state ($\tilde{h}_t$). Used to control the access to the previous state and compute a proposed update. In this work, I will use Bi-directional GRUs to concatenate the output of processing a sequence forward and backwards.

To complete this section about neural networks, I will explain the second proposed network, Convolutional Neural Network (CNN). It is important to note that the following features are part of the neural network framework proposed in (Kim, 2014). In this work I will use region of word embeddings (n-grams) to represent one or more words in vectors. The CNN utilizes these vectors in a different way as they will pass some layers that applied convolutional filters to local features. The author Kim (2014) chose to train a simple CNN with one layer of convolution on top of word vectors obtained from an unsupervised neural language model.

As we can see in Figure 2, the CNN architecture starts with the the $k$-dimensional word vector corresponding to the $i$-th word in the sentence. Each vector has a different length, according to the number of words that we want to represent. The convolutional layer, is an operation that involves a filter $w$, which is applied to a window of $h$ words in order to produce a new feature. This new feature represents the initial word vector although in a small feature map (size vector). The model uses multiple filters (with varying window sizes) to obtain numerous features. These features, that form the max-pooling layer, are passed to a fully connected layer which gives the final result, the output vector. This project will follow the approach of Johnson and Zhang (2017), which has two convolutional layers with stride two (meaning the new feature map is represented in a half vector), and a max-polling layer as to form a pyramid network.
Finally, to achieve better results I will use two classical regression metrics that have the capability to quantify the obtainable number of errors when comparing the achieved prediction with the desired prediction Chai and Draxler (2014). The Root Mean Squared Error (RMSE) measures the average magnitude of error and gives a quadratic scoring rule. RMSE will give a higher weight to large errors which will be later reflected in greater penalization in the final outcome. The Mean Absolute Error (MAE), also measures the average magnitude of the error in a set of predictions, without considering their direction. It is the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight. The classical regression metrics are given by:

\[
RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \quad (3)
\]

\[
MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \quad (4)
\]

where \(y_i\) is the actual or desired value of the output, and \(\hat{y}_i\) the predicted value. The final result produced by this method represents the average of absolute errors.

As I mention before, this project is based on two solutions. The purpose of this first paper is to introduce a new concept for text classification, deep pyramid Convolutional Neural Networks (DPCNN), in this case with 15 weight layers that benchmark six data sets. With this architecture, we can minimize the complexity of the networks. A Convolutional Neural Network (CNN) is a feed-forward network (with convolution layers to convert amounts of data (e.g. text or image) in a vector, interleaved with pooling layers. Johnson and Zhang (2017) studied this neural network and found an architecture that can achieve the best accuracy, increasing the depth. They have managed to do so by creating a pyramid architecture that goes through convolution blocks and a down-sampling layer over and over. This is followed by the convolution of blocks with two convolution layers and a shortcut interleaved with pooling layers with a maximum of stride two for down-sampling. Such occurs with the number of feature maps fixed, which make the computation time for each convolution layer is halved). Finally, the last pooling layer converts internal data to one vector. When we say that we perform a max-pooling with size three and stride two, this means that each pooling layer produces a representative vector of a document by taking a maximum of three internal vectors. The process of conversion of a word into a vector (word embedding), Johnson and Zhang (2017) compute a function \(Wx + b\) for each pretending word of a document. \(x\) represents a \(k\)-word region, \(W\) represents the weights and finally \(b\) represents the bias. The last two variables are
trained with the parameters of other layers. In the experimental part, they used some databases such as "Yahoo! Answers", Yelp and Amazon reviews, with 10K documents from the training data to use as a validation set on each data-set. The best results are from a DPCNN unsupervised tv-embedding ("tv" stands for two views). In conclusion, this paper shows how to solve the problem of high performance deep word-level CNNs with large training data sets.

For the RNN, Yang et al. (2016) separated their model into two parts: the first one has a hierarchical structure of documents; and the second one has two levels of word and sentence-level. In the first insight, they have a hierarchical structure starting with words from sentences and sentences from a document. The document representation started building representations of sentences and then aggregating those sentences in a document representation. After this process, they observed that each word can have a different informative weight, therefore, they started creating the meaning of importance of a word and sentence depending on the context that is inserted.

The gating mechanism used is the GRU-based sequence encoder to track the state of sequences without using separate memory cells. Basically, this mechanism controls how the information is updated to a certain state, at a certain time, and computes the new state based on the previous. For the experiments, they utilized six document classification data-sets (Yelp, IMDB review, Yahoo Answer and Amazon review), where 80% of the data was used for training, 10% for validation and the remaining 10 % for the test. By analyzing the results and comparing with other methods, we can conclude that their proposed hierarchical attention model has the best performance across all data sets.

3 The Deep Neural Model for Classifying Restaurants

In this section, I explain the development process that I was doing for this M.Sc. Project. Additionally, it is provided with all the essential information to understand the proposed architectures analyzing each review of the Yelp data set by extracting every word and sentence from them. The proposed Convolutional Neural Network (CNN) architecture is illustrated on Figure 3 and it follows the general approach proposed by Johnson and Zhang (2017). The first component of the model takes as input reviews from the Yelp data set, which are separated in $k$-regions to produce a region embedding.

The region embedding layer computes a representation for each word (or region of words) of a review, generating a vector that will be the next input vector (for the next layer). This approach is used to balance the vector in order to have words from different reviews. Here, the objective is to represent regions of one or more words. These vectors go through a convolutional layer (working as a filter), which creates a second vector to represent each set of words that are together. This filter helps on the differentiation of each vector. To apply the filter, each vector is multiplied by a trained weighted matrix to achieve a final representation vector of one review. In the following step, the output vectors pass through a max-pooling layer, which selects the maximum value of each row in the matrix of vectors. Regarding the concatenation of all the maximum output vectors (internal vectors), these pass through a fully connection layer in order to aggregate each review (text region) in one vector.

The Recurrent Neural Network (RNN) model has an input that can be seen as having a hierarchical structure. The proposed neural network explores a combination of different mechanisms to generate intermediate representations for the textual contents, such as word embeddings, a hierarchical process of recurrent units, and neural attention. This structure is divided into three levels: one that takes into account the words of a sentence, another considering the sentences from a review, and finally all the reviews for a restaurant.

Starting with Figure 4, which receives word sequences and produces a word embedding vector for each word. This vector passes through Bidirectional-GRU’s, to achieve the desired output vector. The vector that represents a sentence, is the input of Figure 5 where the process is similar to the last one, but this time represents review vector. That vector is the input of Figure 6, and goes through the process that I will explain in the text below. The representation of words, I used some pre-trained approaches as word2vec or Doc2VecC, following the approaches from Chen (2017) and Mikolov et al. (2013).
The attention mechanism extracts the most important words for the meaning of the sentence and aggregates the representation vector to form a sentence vector. To compute this output (sentence vector), I need to feed the concatenation of the two states $h_{it}$ into a feed-forward node to get $u_{it}$. Consequently, it is necessary to measure the importance of the word to get a normalized importance weight vector $\alpha_{it}$. Finally, I compute the sentence vector $s_i$, which corresponds to a weighted sum of the bidirectional GRU outputs.

$$u_{it} = \tanh(W_w h_{it} + b_w) \quad (5)$$

$$\alpha_{it} = \frac{\exp(u_{it}^T u_w)}{\sum_t \exp(u_{it}^T u_w)} \quad (6)$$

$$s_i = \sum_t \alpha_{it} h_{it} \quad (7)$$

In the sentence-level, represented in Figure 5, I already have the sentence vectors. As such, it is only necessary to concatenate in the same way as with the words and then get the annotation of a review. The attention mechanism extracts the most important sentences for the meaning of the review, taking the sentence vector as an input and using a context vector to measure the importance of the sentences. Lastly,
Figure 5: Proposed Recurrent Neural Network, Sentence Encoder.

Figure 6: Proposed Recurrent Neural Network, Sequences of Reviews Encoder.

I compute a review vector that represents the most important information sentences in a review, leveraging another bidirectional GRU output together with an attention mechanism. This output vector is the input vector for the bidirectional GRUs of reviews hierarchical level, represented in Figure 6, which will execute equally to the other levels and give the result of my recurrent neural network.

4 Experimental evaluation

This section presents the experimental evaluation of the proposed methods, which involved tests with a Yelp data set. The first stage of this work consisted in the ”preparation” of the data set. Yelp provided a set of zip files (each zip file corresponding to a different city) containing csv files. These csv files separated the information into two parts, one file with the corresponding health inspection scores, and another file with all the information relative to the coordinates of the establishments (address, latitude, longitude and phone number). I was able to combine them by their ”business id”. At this stage, the reviews corresponding to the scores were missing, bringing us to the next action. In order to obtain all the reviews, Yelp provided an API with a business search tool. The Yelp data set is divided into three parts. One file with the basic information about the establishments (e.g. name, city, zip code, latitude, longitude and phone number), another with all the reviews associated to the venues between the years 2012 to 2018, and finally, the third part with the health inspection scores.

The available data was split into two subsets, with 80% (2345/3181 establishments) for model training and 20% (636/3181 establishments) for validation. The word vocabulary that is considered by the model was generated using the instances of the training subset. The word embedding layer in the first level of the model considered a dimensionality of 175, and the output of the GRUs had a dimensionality of 175 as

1https://www.yelp.com/developers
Model training was made in batches of 32 instances, using the Adam optimization algorithm with default parameters (Kingma and Ba, 2014).

To evaluate the results I use two classical regression metrics that have the capability to quantify the obtainable number of errors when comparing the achieved prediction to the desired prediction. First, the Root Mean Squared Error (RMSE) measures the average magnitude of error and gives a quadratic scoring rule. RMSE will give a higher weight to large errors, later reflecting in a greater penalization in the final outcome. The RMSE punishes variance, as it gives errors, with larger absolute values, more weight than errors with smaller absolute values. Second, the Mean Absolute Error (MAE) also measures the average magnitude of the error in a set of predictions, without considering their direction. It is the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight. RMSE has the benefit of penalizing large errors more so can be more appropriate in some cases.

This section presents the obtained results of the case study, and compare them to other similar approaches. I start this section by explaining the baseline strategies considered in the experiments and justifying why are the most valuable for this work. Finally, I interpret the obtained results by visualizing the attention weights of the recurrent neural network, and at the same time comparing with previous works.

To help in the analysis of the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE), I added two metrics, namely one that represents the average health inspection score of each state/city of the data set, and then tested with different sizes of the data set, especially with the LA part since being the most complete. The baseline strategies and evaluation models are the following: an Recurrent Neural Network with a hierarchical model with three levels of GRUs and with the attention mechanisms at each level, combined with word embeddings mechanism, based on the proposal from Yang et al. (2016); an Convolutional Neural Network with a pyramid structure model, combined with region embeddings mechanism, based on the proposal from Johnson and Zhang (2017); a baseline strategy that only uses the average health inspection score of each state/city of the data set; a small data set to see the behavior of the networks. I will choose the Los Angeles data set that is the most complete state.

Table 2, shows that the Recurrent Neural Network achieved better results than the Convolutional

<table>
<thead>
<tr>
<th>Number of establishments</th>
<th>Los Angeles</th>
<th>Complete Data Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of reviews</td>
<td>1089</td>
<td>3181</td>
</tr>
<tr>
<td>Average number of reviews per establishment</td>
<td>18,72</td>
<td>18,32</td>
</tr>
<tr>
<td>Number of sentences</td>
<td>176555</td>
<td>501235</td>
</tr>
<tr>
<td>Average number of sentences per review</td>
<td>8,66</td>
<td>8,59</td>
</tr>
<tr>
<td>Number of words</td>
<td>2259147</td>
<td>6215064</td>
</tr>
<tr>
<td>Average number of words per sentence</td>
<td>12,79</td>
<td>12,39</td>
</tr>
<tr>
<td>Number of reviews in the training part</td>
<td>15626</td>
<td>44666</td>
</tr>
<tr>
<td>Number of sentences in the training part</td>
<td>99255</td>
<td>286246</td>
</tr>
<tr>
<td>Number of words in the training part</td>
<td>1778426</td>
<td>4848366</td>
</tr>
<tr>
<td>Number of reviews in the validation part</td>
<td>3810</td>
<td>11052</td>
</tr>
<tr>
<td>Number of sentences in the validation part</td>
<td>23197</td>
<td>69913</td>
</tr>
<tr>
<td>Number of words in the validation part</td>
<td>429249</td>
<td>1186606</td>
</tr>
<tr>
<td>Average scores in the training part</td>
<td>94,94</td>
<td>80,70</td>
</tr>
<tr>
<td>Average scores in the validation part</td>
<td>94,8</td>
<td>79,90</td>
</tr>
</tbody>
</table>

Table 1: Statistics of the data set.
<table>
<thead>
<tr>
<th>Model</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recurrent Neural Network (total data set)</td>
<td>3.59</td>
<td>1.93</td>
</tr>
<tr>
<td>Recurrent Neural Network (Los Angeles)</td>
<td>4.20</td>
<td>2.65</td>
</tr>
<tr>
<td>Convolutional Neural Network (total data set)</td>
<td>4.72</td>
<td>3.31</td>
</tr>
<tr>
<td>Convolutional Neural Network (Los Angeles)</td>
<td>5.21</td>
<td>3.57</td>
</tr>
<tr>
<td>Recurrent Neural Network (average data set score in training part)</td>
<td>3.86</td>
<td>2.43</td>
</tr>
<tr>
<td>Convolutional Neural Network (average data set score in training part)</td>
<td>4.79</td>
<td>2.91</td>
</tr>
</tbody>
</table>

Table 2: Performance metrics for different variants of the neural models.

Neural Networks. To compare the two networks I tested with three different data sets, as I said before on the topics above. Firstly, each network receives the data set completed with all the states combined. Secondly, I try with a small data set, Los Angeles data set that is the most complete state, and the results worsened since the size of the data set are smaller, so the networks do not have so many parameters to compare. Lastly, I split the data set in states, and the score of each restaurant is the average score of the state, which combines in a smaller variation, and consequently, the results are better. To conclude this subsection, the data set was not balanced, this means for example the training part with just good reviews and the validation part with bad reviews, increasing the complexity of the task. Generally, the model corresponded to the variations expected.

To end this section, I am going to analyze the weights for the attention layer in the Recurrent Neural Network. The results from Table 2, have already shown that the neural attention mechanisms can led to an increased performance. More interesting, neural attention can also offer model interpretability, by showing the weights associated to each part of the input (review, sentence or word). The attention mechanisms allow analysts to understand which words and fields are more meaningful in each prediction.

As Figure 7 shows, each word on a review has your own weight. For example, the first words service and non-existent have a higher weight. This makes sense because the word service means a lot when you go to some restaurant, and the word that describes how the service was is non-existent, which result on a bad review of the restaurant. In order to validate that our model is able to select informative sentences and words in a document, we visualize five reviews and their respective weights in the hierarchical attention layers.

5 Conclusions and Future Work

In this paper, I presented two types of deep learning techniques for coding the text reviews from Yelp data set with the purpose of giving a health inspection score based on that text. This technique can help and improve the quality of the restaurants. I reported the implementation to reach the final desired output.
This section overviews the main contributions, and highlights possible directions for future work.

One of the most difficult parts of this work is finding a good data set, with all the information available. If the models received a bigger input, with more negative reviews per example, 80% of the reviews in this data set have a health inspections score between 80-100, which made it difficult for me to achieve better results. Results show that deep learning techniques can be used to classify the restaurants in terms of sanitary problems, and the text classification tasks are very reliable. Of course, it is difficult to combat the fake reviews. The reviews from Yelp are very useful for this type of studies concerning with the public health of the restaurants in the United States of America. The attention mechanism implemented in the neural network allows the proposed model to attribute different attention weights at three different levels (i.e., at the review level, sentence level, and finally the word level). The biggest benefit of theses attention weights is the capability of the model to give more or less attention to individual words or a set of words, and at the same time offering opportunity to interpret the classification results by the visualization of the different weights assigned to the input.

The model seems to be a valid approach to use with all types of text classification problems, not only with online reviews. The capability to adapt the model to other daily situations, it is simple and very useful in other topics.

A possible direction for future research is the enrichment of the proposed approach, for example with a more complete data set or with different types of input. For instance, a different approach to handle out-of-vocabulary words by considering alternative mechanisms for exploring context in the generation of the word embeddings, and not just n-grams method. The neural networks can be improved as well. In this project, I used Bi-directional GRU, but other possible approach is to change the gated unit of the recurrent nodes to improve the model effectiveness. Besides different types of recurrent nodes, many other options can also be considered for improving the neural architecture.

References


