# Introduction

# Literature Review

## Neural Networks

### Recursive Neural Networks

Recursive Neural Networks (RvNN) approach the computation of word vectors by applying a particular compositionality function over a binary tree recursively. It is usually done bottom-up, so each child vector is calculated, which in turn calculates the parent vector. Socher et al. (2013) propose an enhanced version of this network, known as a Recursive Neural Tensor Network, where they use the same tensor-based composition function for all nodes in the binary tree. While Recursive Neural Networks are a powerful tool in natural language processing, as Mikolov et al. (2013) points out, they are often subject to high computational complexity due to their nonlinear hidden layers.

### Convolutional Neural Networks

Although Convolutional Neural Networks (CNN) are often used for tasks such as image classification, they’re also extremely useful for sentiment analysis. Kim, Y. (2014) shows that a CNN trained with one convolution layer achieves excellent results across multiple benchmarks.

### Recurrent Convolutional Neural Networks

Lai, S., Xu, L., Liu, K., & Zhao, J. (2015) propose a new type of neural network that takes advantage of recurrent neural network’s O(n) time complexity. While RNNs are a biased model, putting emphasis on later words in a corpus rather than earlier words, RCNNs use an unbiased max-pooling layer that gives equal emphasis across the corpus. RCNNs also take advantage of a bidirectional recurrent structure, which produces considerably less noise than a traditional window-based neural network.

## Types of Distributed Representations of Words and Documents (Word Embeddings)

### Word2vec (Shallow Neural Network Implementation)

Word2Vec is a powerful model developed by researchers at Google led by Tomas Mikolov that produces rich word embeddings. It improves on existing work in this space by proposing two new models for learning distributed representations of words. The goal of these new models was to minimise the computational complexity of the original models by removing the non-linear hidden layer from existing feedforward and recurrent neural net language models (NNLM).

Mikolov et al. (2013) propose a continuous bag of words model (CBOW) that is similar to the feedforward NNLM model where the non-linear hidden layer is removed. This reduces the computational complexity, allowing a much larger vocabulary to be used during the training step. It is trained by trying to classify a target word from a window of source words.

They also propose a continuous skip-gram model (SG), which is similar to CBOW, but instead performs the inverse training step by trying to classify source words based on a target word in the same sentence.

Mikolov (2013) suggested during a discussion of the various use-cases for these models that the CBOW model works better for a dataset with short sentences but a high number of samples (i.e. tweets, short reviews), while the SG model works better for a dataset with long sentences but a low number of samples (i.e. small quantity of large documents).

### Doc2Vec (Shallow Neural Network Implementation)

Following the research led by Mikolov et al. (2013) that produced word2vec, researchers tried to extend the word2vec model to produce phrase-level or sentence-level representations. At first a simple approach was used, which simply created a weighted average for all words in a corpus, thus giving a weighted model for a complete sentence.

This has weaknesses, as Mikolov (2013) states: “The first approach, weighted averaging of word vectors, loses the word order in the same way as the standard bag-of-words models do”.

The extended work done by Mikolov et al. (2013) borrows the same fundamental concepts that were used to create the word2vec models. The only difference being that instead of applying the SG and CBOW models to words in sentences, it is applied to sentences within a corpus.

Doc2Vec will be used for my own implementation of a sentiment analysis deliverable. I will train and test my model using the Yelp dataset.

## Frameworks and Libraries

Existing frameworks and libraries can be used to save time implementing the neural network models previously mentioned. Thanks to their popularity, models such as word2vec and doc2vec have been implemented in various libraries. Gensim (Rehurek, R., 2014) is one such library that has implementations of the work done by Mikolov et al. (2013) on word2vec and doc2vec.

## Datasets

Yelp (a popular online review website) offers a comprehensive dataset for academic use. It contains approximately 4.7 million text-based reviews.

## Summary of Research

The research surrounding sentiment analysis highlights its often complex nature. Representing language in a model that can be consumed by a natural language processing tool while maintaining its context can prove difficult. There are multiple approaches to this problem, but it is clear from the research undergone that a neural network backed solution will perform the best. Various types of neural networks each offer their own unique advantages, and can be tuned to increase performance.

For training the model, the CBOW architecture will be used, as it should perform better than the SG model due to its high-volume of samples that are short in length (see Yelp dataset).

Although there are existing libraries for these (doc2vec, word2vec) models, it was still important to know how the algorithms are implemented at a naive level. Tensorflow’s tutorial for implementing the word2vec algorithm using Python and Tensorflow was followed to gain a basic understanding of the implementation.

In order to train the models in a neural network, it is useful to have a large text corpus to train with. Thankfully, Yelp (a popular online review website) have provided a dataset for academic use. It contains approximately 4.7 million text-based reviews that can be used for training and testing the models.

The purpose of this research and project is to produce a sentiment analysis application tailored towards businesses with an online presence. Businesses often only get a very uninformative high-level insight into their reviews (i.e. star ratings, number of 1-5\* reviews etc).

These businesses would benefit from an application that uses sentiment analysis to analyse their reviews and output meaningful data that they can use to improve their services.

At a high-level, this application will provide:

* Categorisation of reviews into good and bad categories.
* Filtering of reviews not only by their sentiment but also by their context. For example, a restaurant could see all of the good reviews that mentions the quality their food.

# Development

In this chapter, we explore the development process that lead to the production of the sentiment analysis application. It will focus on relevant ideas learned while studying existing literature and applying them to the domain of sentiment analysis.

## Experimental Development

Now that appropriate neural network models, frameworks, libraries and datasets had been identified, experimental development was undergone to find the most appropriate platforms and libraries for sentiment analysis.

### Tensorflow MNIST

Tensorflow is an option for developing neural networks. The main benefit of using Tensorflow is that it abstracts a lot of the complexity around developing a platform to create a neural network.

Things such as types of neural networks, regression models, training models and loss models are made easy to implements thanks to many of Tensorflow’s high-level libraries.

In order to test the viability of Tensorflow, a basic MNIST program was created (MNIST is essentially the “Hello World” program of the machine-learning space). It was very easy to implement the softmax regression and the loss function.

### Word2Vec implemented in Python and Tensorflow

Once a basic understanding of the Tensorflow library was met, the viability of Tensorflow for implementing the work of Mikolov et al. (2013) on the word2vec model was to be tested.

## Developing and Training the Model

Use a graph to show the process

### Doc2Vec

### Finding appropriate datasets

#### Yelp

### Classification

#### Modelling sentiment analysis

#### Finding a suitable classifier (Logistic Regression)

#### Classifying sentiment of unseen Yelp reviews

# Evaluation

Confusion matrix

# Conclusion

# Bibliography

Socher, R., Perelygin, A., & Wu, J. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. Proceedings of the …, 1631–1642. Retrieved from <http://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf>

Santos, C. N. dos, & Gatti, M. (2014). Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. Coling-2014, 69–78. Retrieved from <http://www.aclweb.org/anthology/C14-1008>

Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. <https://doi.org/10.3115/v1/D14-1181>

Lai, S., Xu, L., Liu, K., & Zhao, J. (2015). Recurrent Convolutional Neural Networks for Text Classification. Retrieved from <https://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9745/9552>

Mikolov, T., Corrado, G., Chen, K., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. Proceedings of the International Conference on Learning Representations (ICLR 2013), 1–12.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, & K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 26 (pp. 3111–3119). Curran Associates, Inc. Retrieved from <http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf>

Tomas Mikolov. (2013). de-obfuscated Python + question - Google Groups. Retrieved November 25, 2017, from [https://groups.google.com/forum/#!msg/word2vec-toolkit/NLvYXU99cAM/E5ld8LcDxlAJ](https://groups.google.com/forum/" \l "!msg/word2vec-toolkit/NLvYXU99cAM/E5ld8LcDxlAJ)

RaRe Technologies. (2016). Doc2Vec Tutorial on the Lee Dataset. Retrieved from <https://github.com/RaRe-Technologies/gensim/blob/develop/docs/notebooks/doc2vec-lee.ipynb>

RaRe Technologies. (2015). Gensim Doc2vec Tutorial on the IMDB Sentiment Dataset. Retrieved from <https://github.com/RaRe-Technologies/gensim/blob/82c394a9085d583e8a75c2bb32ecd37cf61236f0/docs/notebooks/doc2vec-IMDB.ipynb>

Le, Q. V., & Mikolov, T. (2014). Distributed Representations of Sentences and Documents, 32. Retrieved from <http://arxiv.org/abs/1405.4053>

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., … Zheng, X. (2016). TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. Retrieved from <https://arxiv.org/abs/1603.04467>

Rehurek, R. (2014). gensim: Topic modelling for humans. Retrieved from <http://radimrehurek.com/gensim/>

Tensorflow. (n.d.). Vector Representations of Words | TensorFlow. Retrieved November 26, 2017, from <https://www.tensorflow.org/tutorials/word2vec>

Yelp Inc. (n.d.). Yelp Dataset. Retrieved November 25, 2017, from <https://www.yelp.com/dataset>