Learning Sentence Vector Representations to Summarize Yelp Reviews

Neal Khosla
Stanford University
nealk@cs.stanford.edu

Vignesh Venkataraman
Stanford University
viggy@stanford.edu

June 9, 2015

Abstract

Summarization is a key task in natural language processing that has many practical use cases in the real world. One such use case is with regard to product or restaurant reviews, which often contain repetitions of the same or similar opinions ad nauseum. It would be ideal to parse the set of reviews for a particular entity and generate a summarization that encompasses all the key points contained in the full set. This paper details a three-pronged approach to tackling this issue as it pertains to the Yelp Dataset of reviews, using naive statistical NLP, the Word2Vec model, and a newer paragraph vector model to try to learn vector representations of sentences; these learned representations are then used to cluster and extract relevant sentences from the superset using k-means clustering. The paragraph vector model, in particular, achieves good performance on a ROUGE-based evaluation metric that measures the overlap between the key sentences for a place of business as hand-labeled by a human and the key sentences returned by the algorithm.

1 Introduction

The challenge we have taken up is that of summarizing Yelp reviews for different businesses. Namely, we seek to take the set of reviews for a given business and be able to output some sort of summary or set of relevant opinions that a user might want to discern about the business if they were to read through all the reviews themselves. This problem is interesting given that many users do exactly this using Yelp on a daily basis. Since many Yelp users visit the site in order to form opinions on a business, they often read many reviews for a given business to form a more educated and accurate opinion. Our goal was to approximate this process and demonstrate the possibility of a system to derive a sort of "consensus" about each business that would enable users to (if this were in production) skip out on reading through all the reviews for a given business, thus saving users time and effort whilst making the Yelp product more compelling.

Given the relative open-endedness of this problem as well as its difficulty, we set out to find a way to capture the information contained in the reviews in a manner that would at least extract some of the relevant information without necessarily giving us a perfect summary of all the reviews. In technical terms, we hope to give users high "precision" with less promises about the "recall" - that is, the returned results should accurately capture at least some of the most common sentiments expressed in the overall set of reviews for a business without guaranteeing that all relevant viewpoints are captured.

At a high level, our approach was to find a way to best represent the different review types of a business in vector space. Much like word vectors and sentence vectors, we operated under the hypothesis that we could learn representations of the information contained in reviews and then extract the most relevant sets of such information. To do this, we essentially threw the kitchen sink at the problem, generating vector representations for review sentences in a variety of different ways, and then used various forms of clustering to extract key phrases (or sentences) from these vector representations.

2 Background and Related Work

While there hasn't been a lot of work we found in applying Deep Learning directly to summarization problems, there has been a lot of work on understanding meaning and representing natural language in vector form. The first model we examined was the word2vec model developed by Mikolov et al. [4]. In particular, we looked at the skipgram model that tries to predict the surrounding words given a current word. Another model we examined was the GloVe model proposed by Socher et al. [5]. This model is similar in many ways to the skipgram model. Both Word2Vec and Glove have been shown to learn word vector representations quite well, and they accurately capture the relationships between words (the canonical example used for this is Word2Vec('king') - Word2Vec('man') + Word2Vec('woman') = Word2Vec('queen'), a remarkable result). However, GloVe ends up being much more of a memory hog, with common implementations in Python requiring memory quadratic in the size of the vocabulary. As a result of this, and the fact that GloVe word vectors have been shown to be just about as good as Word2Vec word vectors, we decided to use Word2Vec as one of our models. In order to to extend the individual word vectors generated by Word2Vec to the more complex sentence vectors, we chose to simply add the word vectors for each word in a given sentence; we also considered concatenation and pointwise multiplication.

While research into word vector representations has yielded excellent results, there isn't nearly as much literature on learning representations for things like sentences or paragraphs; obviously, research into this is ongoing. Our research revealed a small number of papers that have sought to learn vector encodings for entities larger than individual words. Formeost among these is the 'Paragraph Vector' model developed by Tomas Mikolov and Quoc V. Le [2]. This (very recent) model learns fixed-length vector encodings for variable length inputs (like sentences in a review, for example) through deep learning, specifically by averaging or concatenating both learned word vectors and a special context-specific 'Paragraph Vector' (here applied to a sentence) to predict the next word given a context. Learning these sentence and word vectors is accomplished via standard neural network feed-forward and backpropagation steps.

The only other paper we could find that directly tried to represent sentence-level structures was the Dynamic Convolutional Neural Network algorithm proposed by Nal Kalchbrenner et al. [1], which uses a combination of convolutions and dynamic k-max pooling to try to represent sentences and learn sentence structure. The issue with this model is that the learned representations are essentially word vectors again, and representing a sentence requires either combining word vectors or tacking word vectors together in the matrix form that is fed into the convolutional neural network. This, combined with the large memory and processing power requirements for efficiently running convolutional neural networks at this scale, ruled this model out for our purposes.

We also considered the methodology proposed by Socher, Et. al in "Parsing Natural Scenes and Natural Language with Recursive Neural Networks" [6]. This involved training a recursive neural network with some type of labeling. However, there were a number of situational problems that came up as we attempted this strategy. The first issue was that not every review had a label associated with it, whether this was a star rating or a usefulness rating as determined by other users voting on the review. The usefulness rating, specifically, was very sparse across the entire dataset, and the lack of normalization of the metric (it is a counted, rather than averaged, metric, it is optional for users to vote on usefulness, and thus reviews that are more viewed are always viewed as more useful) made it impossible for us to use in a logical manner. Using the reviewer's own rating reeked of confirmation bias, and thus we avoided this strategy.

3 Approach

Fundamentally speaking, our technical approach revolved around two different steps: encoding review information in vectors, and using these vectors to extract key phrases that capture the essence of the reviews for a given place of business. As reviews are generally composed of sentence-level concepts, we focused our vectorization and extraction efforts on individual sentences. We now describe our efforts to complete both of these steps respectively.

3.1 Learning Sentence Vector Representations

Our baseline approach for learning sentence vectors was formulated using a simple bag-of-words model to extract frequency information from the reviews for a single business; this corresponded to a frequentist statistical approach that is very naive and abandons all semantics and meanings in favor of merely counting occurrences. Thus, the size of the vector is roughly the size of the vocabulary set (collapsed down to a lower fixed dimension based on the most common words) and the value at each vector index i represents the count of word i in the sentence. We expected this to perform moderately well but not outstandingly, as frequency and word counts are poor-to-mediocre approximators of true meaning and linguistic nuances are beyond the scope of a basic bag-of-words model.

The word-vector-based model we used was Word2Vec, proposed by Mikolov et al. from Google. This unsupervised model learns vector representations for words using provided corpora. For this model to be useful for sentences, rather than discreet words, we employed the naive strategy of summing all learned word vectors for a sentence together and using this additive result as a sentence vector; we did this for simplicity of dimensioning, as the sum of two n-dimensional vectors is still an n-dimensional vector, allowing variable length sentences to be collapsed into fixed-length vector representations. However, we also considered point-wise multiplication and concatenation as other conversion strategies for converting word vectors to sentence vectors.

We also learned representations of the different sentences used in reviews following the methods described in the paper "Distributed Representations of Sentences and Documents" [2]. This recent publication is in effect an extension of the Word2Vec algorithm; it learns not only word vectors, but also representational vectors for a specific 'context' structure of arbitrary length and composition. For our purposes, this 'context' will obviously be a sentence, as learning sentence vectors will allow us to represent sentence meanings with a series of numbers.

It is also worth noting that there are two distinct options for training all of these models: training the models on the entirety of our dataset, or training a different model for each place of business and the reviews pertaining to it. There are a number of pros and cons to both approaches. Training on the entire dataset will allow our learned word vector representations, in all models, to be more accurate from a universal perspective. However, training on a large dataset is decidedly more computationally intensive. Additionally, the meaning of a word in the context of a specific place of business might be subtly different than the 'universal' meaning of a word, and per-review subtleties will be overwhelmed by the sheer weight of the entire dataset. On the flipside, training a model specific to each place of business will be much quicker (by many many orders of magnitude) than training on the entire dataset, and a per-business model might more accurately capture the meanings of the words and sentences as they pertain to specific reviews for this specific business. However, there is also a lot less data on which to train these models, and thus the models are also susceptible to outliers and vaguely trained or untrained words.

In light of these considerations, we chose to train all our models on a per-business basis, thus allowing for more localized vector expressiveness and a more pertinent-to-business representation. We also culled out stop words for the Bag of Words models, but not for the neural network based models. These results will be summarized in the Experiment section below.

3.2 Extracting Key Phrases

Our approach for extracting the key sentences for a business from these learned vectors was based on k-means clustering. Namely, we took the review sentence vectors we generated and clustered them into k clusters. After this, we took the most central sentence from each cluster as "characteristic" representation of the cluster. We experimented with a variety of hyperparameters in this instance, such as the number of cluster centers, the distance metric used (i.e. Euclidean/Cosine, L1, Chebyshev). We also considered experimented with simply taking sentence vectors that are "far apart," distance wise, and returning them; logically, this would give us sentence vectors that represent unique 'opinions.' However, after some basic experimentation, we realized that this did not take into account the weight of popular opinion. A contrived example of this is as follows: if 6 people said that a restaurant was fantastic and a single sentence said a restaurant wasn't, unfortunately that outlier sentence will be included in the summary since it is "far apart" from the other sentences. k-means clustering accomplishes essentially the same task as picking "far apart" sentence vectors, with a lot more robustness, and as such we chose to use it exclusively.

4 Experiment

4.1 Dataset

For our project, we used the Yelp challenge dataset [7], a publicly released dataset curated by Yelp that includes business and review data collected on `www.yelp.com` from over 10 cities and 4 different countries. This data is very large in scale as it contains 1.6M reviews by 366k users for 61k businesses as well as 481k business attributes (hours, parking availability, etc.). This data has been publicly released by Yelp for use in academic research and projects.

Obviously, with so much data, memory and resource management becomes a huge concern. The overall dataset weighs in at 1.43 GB of raw JSON, and to even read parse all the data into business-sized chunks was a lengthy and resource-intensive operation. In order to test the viability of our approach, which is inherently unlabeled and unsupervised, we elected to use a randomly chosen subset of the data, containing exactly 1000 places of business and the reviews pertaining to them. We also threw out any businesses that had under 5 reviews, since our goal was to output anywhere from 3-6 "key sentences" from each business' reviews.

4.2 Evaluation

A major difficulty with performing a summary task is evaluating its correctness and efficacy for real world use. Unfortunately, the Yelp dataset does not come prelabelled with anything other than review score and (occasionally) usefulness ratings; review score rarely, if ever, has any bearing on a specific sentence's importance, and as we mentioned before, the usefulness ratings are unnormalized and cannot be used to distinguish individual sentences that are relevant for a summary. As such, we were forced to look to manual methods to evaluate our methods' success.

The canonical metric used to evaluate automatic summarization is ROUGE, which stands for Recall-Oriented Understudy for Gisting Evaluation. The software package that comes with the official version of ROUGE compares a computer-generated summary against a set of human-generated references, with varieties based on n-gram co-occurrence, longest continuous subsequence, and others [3]. Given time and financial constraints, we chose to create our own ROUGE-like evaluation metric named YELP (acronym to be determined). YELP is quite simple in theory: a human goes through and picks out any and all key sentences that he or she would like to see included in a summary of the reviews for a business. Then, the sentences spit out by the summarizer are compared to these "key" sentences, and the accuracy score is the number of truly "key" sentences, as picked by the human, divided by the total number of key sentences returned by the algorithm. Optimizing for this metric roughly corresponds with the concept of "precision" detailed earlier, as it reveals the percentage of returned sentences that are relevant to the overall summary. The concept of "recall," while another important metric, is less relevant to our topic given the way we have (painstakingly) labeled the data; since we are including all sentences that we would accept in a good summary in our ground-truth reference, our recall will necessarily suffer. Thus, we evaluate on precision exclusively, leaving recall optimization as a dataset-labeling exercise for the future. In total, we were able to hand-label 100 places of business, totalling about 15,000 sentences worth of text.

Additionally, in order to try to account for the randomness of our experiments and clustering, we set the **numpy** random seed to be 1234 wherever appropriate and possible.

4.3 Trials and Results

In order to optimize our results, we employed grid-search based hyperparameter tuning. Simply, we started with a coarse grained search to find optimal regions for our hyperparameters and then narrowed our search in these regions. The first table below demonstrates our results for the Naive Bag of Words and Additive Word2Vec models, which consistently performed worse than our Paragraph Vector model. The second table shows the results for our Paragraph Vector model, which earned our maximum precision score of about 58%. We also provide a three dimensional plot that shows precision as a function of dimensionality and training epochs for our Paragraph Vector models.

Trial	Model	Dimension	k	Epochs	Precision
I	Naive BoW	100	3	N/A	52.00%
II	Naive BoW	100	5	N/A	48.80%
III	Naive BoW	200	3	N/A	50.67%
IV	Naive BoW	200	5	N/A	50.40%
V	Naive BoW	500	3	N/A	48.00%
VI	Naive BoW	500	5	N/A	48.00%
VII	Naive BoW	1000	3	N/A	48.00%
VIII	Naive BoW	1000	5	N/A	48.00%
IX	Naive BoW	50	3	N/A	53.33%
X	Naive BoW	50	5	N/A	52.80%
XI	Naive BoW	50	3	N/A	51.67%
XII	Naive BoW	50	5	N/A	50.40%
XIII	Word2Vec	100	3	10	37.33%
XIV	Word2Vec	100	5	10	44.80%
XV	Word2Vec	100	3	50	53.33%
XVI	Word2Vec	100	5	50	47.20%
XVII	Word2Vec	100	3	100	44.00%
XVIII	Word2Vec	100	5	100	43.20%
XIX	Word2Vec	100	3	75	45.33%
XX	Word2Vec	100	5	75	46.40%
XXI	Word2Vec	200	3	50	53.33%
XXII	Word2Vec	200	5	50	48.00%
XXIII	Word2Vec	400	3	50	49.33%
XXIV	Word2Vec	400	5	50	46.40%
XXV	Word2Vec	800	3	50	54.67%
XXVI	Word2Vec	800	5	50	50.40%
XXVII	Word2Vec	1600	3	50	45.33%
XXVIII	Word2Vec	1600	5	50	51.2%

Table 1: Results of Coarse-Grained Trials for Naive Bag of Words and Word2Vec Models

Trial	Dimension	k	Epochs	Precision
I	25	3	10	56.00%
II	25	3	50	56.00%
III	25	3	100	57.33%
IV	50	3	10	53.33%
V	50	3	50	53.33%
VI	50	3	100	50.67%
VII	100	3	10	48.00%
VIII	100	3	50	48.00%
IX	100	3	100	49.33%
X	300	3	10	53.33%
XI	300	3	50	53.33%
XII	300	3	100	44.00%
XIII	600	3	10	52.00%
XIV	600	3	50	52.00%
XV	600	3	100	49.33%
XVI	1000	3	10	54.67%
XVII	1000	3	50	54.67%
XVIII	1000	3	100	52.00%

Table 2: Results of Coarse-Grained Trials for Paragraph Vector Model

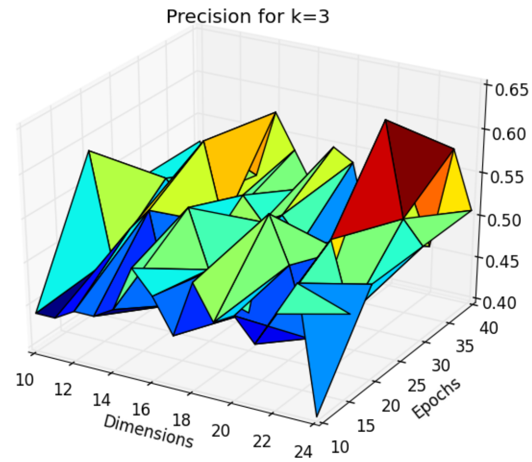

Figure 1: Precision vs. Hyperparameters for $k = 3$

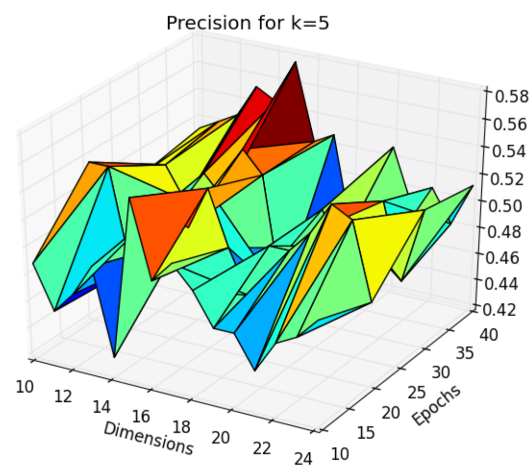

Figure 2: Precision vs. Hyperparameters for $k = 5$

4.4 Analysis

All of our models experienced varying levels of success on the summarization task. As shown in the tables above, the Naive Bag of Words model was remarkably performant for its simplicity, achieving precision scores quite similar to the Additive Word2Vec model across different hyperparameters and clustering regimes. The best BoW model, averaged across the $k = 3$ and $k = 5$ precision scores, was the 200-dimensional version. It is interesting to note that, in general, lower-dimensioned BoW models performed better than higher dimensioned ones; since the lower dimensioned models only accounted for the most frequent (non-stop) words, this result makes some sense, as forcibly including less common or less relevant words in a statistical approach will just add noise to the result, as these less common words don't have any meaning to them from an algorithmic perspective.

The Additive Word2Vec model did end up outperforming the Naive BoW model, but only barely. It was much more sensitive to hyperparameter tuning, and searching for optimal hyperparameters was a challenging exercise. The best Word2Vec model had 800-dimensional word vectors and was trained for 50 epochs on each business' review data, and it achieved an average precision of about 52.54% on the combined $k = 3$ and $k = 5$ tasks. The Word2Vec model appears to benefit from training for more epochs and higher word vector dimensions, especially for the more challenging $k = 5$ task. However, the limitations of the Word2Vec model as it applies to a full sentence worth of words are readily apparent - Word2Vec might be outstanding at representing individual word semantics, but it struggles when a large array of word vectors are summed in order to represent a sentence.

The Paragrah Vector model performed the best of all our models, and was able to learn remarkably compact and effective (as small as 10 to 50-dimensional) representations of sentences. It was also the slowest to train, as one would expect for a model of its complexity. It achieved a maximum $k = 3$ performance of 58.66% and a maximum $k = 5$ performance of 57.33%. Numerically, these top-scoring models outpaced all other models by over 5%.

4.5 Examples and Analysis of Results

We also visually inspected the returned results of the Paragraph Vector model, in order to subjectively determine whether they would be good fits for a Yelp summarization. Some characteristic examples are shown and commented on below.

Positive Examples

Example 1: St. Mary's Basilica

> This was our third trip to the Phoenix/Scottsdale area this year, and this was the third church that we attended in the area.
> The masses are beautiful and the people are friendly and welcoming!
> The wedding was a full Mass and made it such a special day for my sister, her husband, myself and everyone who witnessed the ceremony.
> During the holidays, I make sure I attend mass at St. Mary's Basilica.
> It reminds me of being back East and the time I spent traveling abroad .

We can see in this positive example, which is the summary for a Catholic church, that the model does a good job of extracting relevant thoughts on the church. Namely, the model extracts relevant information about the quality of the mass at the church as well as reviews that emphasize how they make sure to attend the church and that the people are great. Compared to the unfiltered review text, which contains tons of anecdotal filler that isn't directly pertinent to the quality or characteristics of the church, this summary is definitely superior.

Example 2: Chuy's Restaurant

> Seems like all of their locations are pretty similar.
> Chuy's was pretty good.
> I'm pretty forgiving but I can't forgive the junk they cooked here.
> I've been to Chuys in Tucson for dinner, which is always have had a good experience with it being that it is a total hole in the wall.
> Chuys is always going to be a good choice for mesquite when you need a quick bite!

The extracted sentences for Chuy's Restaurant also show promising results. In particular, the sentences extracted comment on the quality of the food at the restaurant and what types of settings you might want to go to it for. Other key points emphasize the strength of the experience and the similarity of the locations of the chain restaurant. While the results are not perfect, they do demonstrate the diversity of opinions that any one business could have, with one person claiming that the food was "junk". The strength of this summary is, again, that impertinent anecdotes and filler sentences are stripped out, leaving only key opinions behind.

Example 3: Harbor Lake Therapeutic Massage

He can also work on your TMJ problems.
I have arthritis in all my joints and he has been able to keep me moving and active.
My therapist is Larry and he is terrific.
They are that good.
Larry can get the kinks out of my neck and shoulders like no one can.

The final example we examine that demonstrates the strength of our model is the summary for Harbor Lake Therapeutic Massage in Las Vegas, NV. All 5 of these sentences develop a consensus that this is an incredibly high quality massage therapy parlor that can help you work through issues you may be having. The sentences show a positive sentiment towards the business in a variety of different uses.

Negative Examples
Example 1: United Artist's Theatre

They have a special seating area in the theater but I understand they coat $5.00 more.
I consider a ”new school” theater to be one with stadium seating.
My fiance's work gave him two sets of Regal movie tickets.
Another plus is that it is about a mile from my house.
I used to go out of my way to come here, simply to avoid the crowds.

This review is for United Artist's Theatre in Scottsdale, AZ. The sentences picked out here have the general problem of having irrelevant information. The first sentence talks about a "special seating area" but gives very little information on whether it is something that would be interesting. The next talks about the author's views on what a "new school" theatre is, completely irrelevant to information about the quality of the business. The last few sentences talk about very personal things that have nothing to do with the quality of the business.

Example 2: Eagle Crest Golf Course

what is this mexico?
This review is for the practice facility only, I have never played the course.
WOW, how about not serve it if it is full of black things?
So we took our time and really enjoyed ourselves.
We went on a friday morning and there were very few other golfers.

The summary for Eagle Crest Golf Course in Las Vegas, NV, also has some questionable results. Most of the key sentences picked out here have nothing to do with the quality of the experience at the golf course. The first sentence picked here is a complete non-sequitur. Again, this result demonstrates the difficulty of picking out strictly relevant sentences to summarize the review and how this was not always a given with our model.

Example 3: Tri-Color Locksmith

Many thanks to Dave and Tri-Color!
The tech Joey was very friendly and had the job done in no time.
I won't call another locksmith again.
Over and over again the folks at the store made me feel like a jerk that they had to rescue from utter incompetence.
I called Tri-color expecting to have them come out in the next day or so.

While the results in this example, for Tri-Color Locksmith, are unlike the other negative examples in that they all contain relevant information, we wanted to highlight this example as it is characteristic of part of the problem. While some of the sentences here demonstrate really positive experiences with the business, and in particular the service of the business, others have the exact opposite sentiment. This is part of the problem of this summarization problem - it can be impossible to summarize opinions or form any consensus when the opinions are so varied.

5 Conclusion

Our models demonstrated that there is good potential for accurately summarizing keypoints of Yelp reviews. We recognized the potential for our models to extract mostly relevant and interesting sentences, even though some of these might not aptly be described as a summary of the reviews. However, it is important for us to reiterate the challenge of completing this task given the difficulty of the task even for humans. In particular, things that are relevant to one human may be completely irrelevant to another human. We also struggled to discern a clear and obvious consensus for particular businesses. In many cases, there were a number of reviews with completely contradictory messages. For example, the first review we hand-labeled was for a hair salon in which the first 3 reviews said it was an amazing hair salon and the next 3 said it was the worst place ever and they'd never return; in this situation, what was the proper summary to take away? Given the difficulty for humans such as ourselves to perform this task, we wonder if we have defined the problem in a manner that makes it difficult to really properly measure how we did. In repsonse to this, we have considered redefining the problem to include elements of sentiment analysis. In particular, we consider the possibility of performing this task with either aspect specific sentiment analysis or just summarization of positive and negative sentiments, which could also be aspect specific if need be. Given the results we do have, it could prove very useful to merely print out the k cluster centers as determined by our algorithm and also return the number of other sentences that map to these centers, as this would prove slightly more informative for a user and give more weight to frequently expressed opinions. Alternatively, we are considering a proposed solution that revolves around a different strategy for sentence extraction than clustering. We propose the possibility of using Mechanical Turk to label sentences as key points or not key points and training on these labels in order to learn what an "important" sentence looks like. This would allow us to then identify key sentences dynamically and would also solve the issue of having different numbers of key points for different businesses.

In addition, we have considered the possibility of attempting this with other methodologies. We would like to try using models such as a Tree-LSTM or a ConvNet that might be able to capture different representations of the sentences in a more succesful manner.

References

[1] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. "A convolutional neural network for modelling sentences". In: *arXiv preprint arXiv:1404.2188* (2014).

[2] Quoc V Le and Tomas Mikolov. "Distributed representations of sentences and documents". In: *arXiv preprint arXiv:1405.4053* (2014).

[3] Chin-Yew Lin. "Rouge: A package for automatic evaluation of summaries". In: *Text summarization branches out: Proceedings of the ACL-04 workshop*. Vol. 8. 2004.

[4] Tomas Mikolov et al. "Efficient Estimation of Word Representations in Vector Space". In: *arXiv preprint arXiv:1301.3781* (2013).

[5] Jeffrey Pennington, Richard Socher, and Christopher D Manning. "Glove: Global vectors for word representation". In: *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)* 12 (2014).

[6] Richard Socher et al. "Parsing Natural Scenes and Natural Language with Recursive Neural Networks". In: *Proceedings of the 26th International Conference on Machine Learning (ICML)*. 2011.

[7] Yelp. *Yelp Dataset Challenge*. URL: `http://www.yelp.com/dataset_challenge`.