Recovering subreddit structure from comments

James Martin

December 9, 2015

1 Introduction

Unstructured data in the form of text, produced by new social media such as Twitter, Facebook, and others are of great interest to companies seeking to better understand their customer's needs and opinions. Also, user experience can be lengthened and improved by using data to suggest related links to visit. Mining and processing this data is no trivial task, due to its complexity and size. In this paper, I focus on exploring data generated by the popular website reddit.com, dubbed "the front page of the Internet." The website is stylized in all lowercase, "reddit.com" but the entity is a proper noun, "Reddit."

Reddit is essentially a bulletin board, users post links to interesting articles, pictures, videos and "comment" on posts by other users. "Subreddits" are bulletin boards specific to a topic. Subreddits are prefixed with "r/", stemming from their URL. For example, the subreddit for discussions on historical events, "r/History", has the URL www.reddit.com/r/History. Posts on Reddit are sorted by how many "upvotes" they receive, posts with more upvotes appear higher on the board. In this project, I did not make use of this feature but discuss its potential in the section 4.

By this hierarchy, there are only two levels of structure to Reddit: the global landing page that contains the top posts from all subreddits, and the individual subreddits themselves. For example, if I were to visit reddit.com, I may not see any posts from r/History because they may not be popular enough to appear on the front page. However, when I visit r/History, I will see all the posts to that board and I can click on a specific post to read the submitted comments.

Comments are quite different than posts. Posts, as mentioned, are links to content. Comments are a user's thoughts on a given post and vary greatly in length and substance. Furthermore, comments are not often moderated, so off topic discussions can occur with no consequence. Some subreddits foster discussions in the comments section, while many do not. Reddit has its own culture, rife with inside jokes, "memes," and other general noise. This noise is compounded by the fact that misspellings and use of slang are frequent. It is fairly easy to see that deriving structure from such noisy, unstructured text is no easy task.

My goal was to determine whether it is possible to recover subreddit structure from clustering vector representations of the comments. If this is achieved, a subreddit recommendation engine could be created to enhance user experience with the following pipeline:

1. A comment from a user is submitted to the pipeline
2. The comment is converted to a single vector representation
3. The comment is classified into a subreddit by a nearest-neighbor approach
4. The nearest subreddits to the subreddit found in step (3) are reported to the user

This project attempts to serve as a proof of concept for the above pipeline. In what follows, I describe the data I used, the model I trained, why I selected the model, possible improvements to the model, my results of creating "prototypical comments," recommending similar subreddits, and my conclusions.

2 Data and setup

The data I used were a subset of the 1.7 billion Reddit comments released to the public. These comments date from 2007 to May 2015. Kaggle.com hosted a subset of these comments from May 2015, which I downloaded and used for this project. This data set contained comments from 2,611,449 distinct users across 50,138 unique subreddits and totaled 30GB.

To analyze these data, I downloaded the database from Kaggle onto a personal department machine, and ran an instance of Jupyter Notebook as a server. More information about this tool can be found on Jupyter's documentation website [2]. This allowed great flexibility in pivoting during the project and gave me access to a wealth of tools and machine learning libraries available for Python, without the limitations of storage or speed of my personal laptop. Though tools are not the focus of this project, they are worth a mention because they helped me think more deeply about the models I was training through the ability to rapidly prototype.

To further narrow the data, I selected twelve subreddits that fit most or all of the following criteria. Each subreddit I selected:

1. Was (in theory) a good topic to spark substantive discussions
2. Contained (in theory) a central topic or theme
3. Contained at least 10,000 comments
4. Contributed to breadth of topics across the subreddits chosen
5. Contained popular culture references to add complexity

The first set of subreddits I selected and trained models on yielded some unexpected and fairly undesirable results. Upon checking what the most similar words to "man" were, I found misogynistic and sexist terms. I believe this was the result of including r/gaming, the subreddit dedicated to video games. I decided to scrap the models I had trained on that set and select the following subreddits instead:

- r/Mathematics
- r/ComputerScience
- r/History
- r/Philosophy
- r/ExplainLikeImFive
- r/AskAnthropology
- r/Homebrewing
- r/Bicycling
- r/Food
- r/Science
- r/Movies
- r/Books

This set yielded me 931,315 unique comments of varying length. This size struck a fine balance between training time and accuracy and suited my needs to develop a proof of concept. I believe including more subreddits into this data set would increase model accuracy and robustness to noise. This is further discussed in section 4.

3 Model selection

To represent text as vectors, I trained a skip-gram model using a Python implementation of Google's word2vec tool. Though I trained multiple models using varying hyperparameters, I was guided by Google's documentation to ultimately select the model using a context (window size) of 10 words, a minimum word count of 10, and a downsampling parameter of 1e-5 [5].

The context of 10 words is recommended for accurately computing hierarchical softmax. The minimum word count of 10 means that words that occurred less than 10 times across the corpus were discarded from the vocabulary. I chose to use a skip-gram model over continuous bag of words in hopes that the skip-gram model might better encode the vernacular of Reddit and allow for predictions of what I call "prototypical comments." Since skip-gram models are useful for predicting surrounding words in a sentence or document, I hoped to recover words that were commonly found in comments originating from a specific subreddit.

3.1 Original approach to model selection

My original approach was to determine which model worked best for classifying a comment to its respective subreddit. My plan was to choose the model which yielded the smallest misclassification rate, so I divided comments from each subreddit into a training and testing set, trained word2vec models with different hyperparameters, then classified each comment in the training set to a subreddit with the following procedure:

1. Transform each comment into a list of words

2. Convert each word in the list to its vector representation

   For each **topic** in set of subreddits:

3. Calculate cosine distance of each word to **topic**

4. Average distance of all words in the list

5. If the distance is the smallest seen, label that comment with **topic**

This proved to be an expensive, limited, and naive procedure of labeling comments. I was hoping that the words in the comments would be fairly related to the word best representing the topic of the subreddit. For example, "history" was the chosen topic of the r/History subreddit, "explanation" was the chosen topic of r/ExplainLikeImFive. Classification success rates were observed to be no greater than 12%. Also, computing distance to a chosen topic was subjective. Further limitations of assigning a subreddit a topic is discussed more in the section 5.

Other methods of model evaluation such as the analogical reasoning task [1] introduced by Mikolov et al. [4] were not applicable, as the core task in this project was to leverage the vernacular of Reddit to derive structure of subreddits.

3.2 Results of model selection

Using my model, I then transformed each comment into a single 1x300 vector representation by averaging each word's vector representation. The average word representation of a comment was the sum along each of the 300 dimensions of each word in the comment, then a division by the number of words. Words not in the vocabulary were discarded and not included in the representation. After running a principle component analysis to find the top three components that explain the variance in this representation, I observed the following structure seen in Figure 1.

The model I selected yielded a vocabulary of size 43,723. Each feature of the vector representation was a scalar value between $[-1, 1]$.

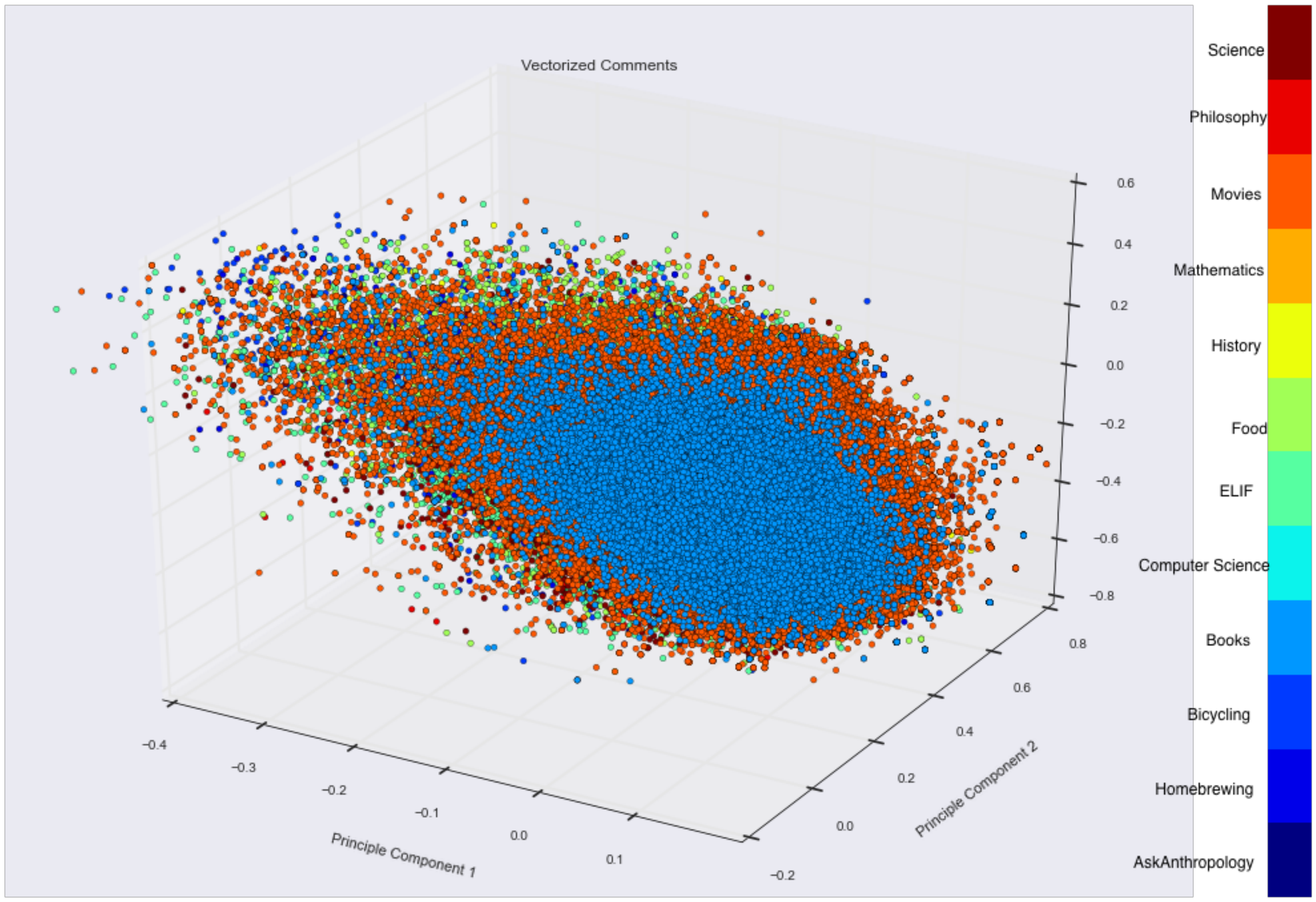

Figure 1: Each color represents the ground truth subreddit to which the comment belongs. By the plot, posts in r/Books and r/Movies produced the clear clusters. These results are discussed further in the Results section.

4 Possible model improvements

My model can be improved in four ways. The first would be to include more data. The accuracy of word2vec models improve with larger amounts of training data [5]. Especially considering the amount of noise in Reddit comments, selecting comments from all 50,138 subreddits would enlarge the vocabulary and possibly become more robust to non-descriptive comments.

The second improvement could be to explore different ways of representing comments based on the representations of the words in the comment. I chose to represent comments as an average of the representations of the individual words. A weighted average may yield a better representation of the comments.

A third improvement to the model would be to incorporate the "score" of the comment. Reddit comments can be "upvoted" or "downvoted," providing a "controversiality score." This could have been a useful feature for the model, as a prototypical comment may be one that has a controversiality score within a certain range.

A final improvement could be made by selecting a different model entirely. Representing each comment as a paragraph vector may be a much better choice instead of representing comments as an average of each word vector [3]. A model like GloVe may also outperform skip-gram models because of its ability to leverage repetition across the entire corpus [6]. Since Reddit vernacular is likely to appear across all subreddits of the website, GloVe likely would better capture context of comments.

5 Results

Once each comment was transformed to a 1x300 vector representation, I ran a K-means clustering algorithm with k=12 to determine if subreddit structure could be recovered. This method could be expanded for a larger dataset by using k={# of subreddits in the data set}.

My hypothesis was that running K-means on the comments would yield centroids that were near word representations related to the topics of the subreddits I had chosen. This was not entirely the case. Instead of seeing coverage of all topics, I saw significant coverage of movies and books. To determine what each centroid represented, I found the top 15 words with representations closest to each centroid. Below are notable results and possible explanations.

One cluster seemed to capture popular Reddit jargon and shorthand:

Cluster 1	'thanks','thank','sorry','fanboy','misread','joking','eragon','lol', 'here','btw','haha','rlm','sarcastic','reread','wow'

Several clusters captured words about movies. Cluster 7 appeared to capture words about more light-hearted movies:

Cluster 7	'pulpy','macgruber','rango','superbad','cujo','gunplay','pearce','romcom', 'heston','worldbuilding','caddyshack','mortdecai','elizabethtown','blizzcon','baz'

Another was a bit more dark:

Cluster 11	'eragon','blizzcon','gyllenhal','pearce','rewatchable','talia','tonally','voiceover', 'typecast','apocalypto','furiousa','storyteller','slasher','tasm','baddie'

I noticed a fair amount of overlap between the clusters, which is not desirable. However, one cluster in particular stood out:

Cluster 6	'didion','daemon','mullet','springmeier','exupery','piranha','fingolfin','shardblade', 'alexandre','wraith','gorman','kaladin','elie','burghley','cronin'

This cluster appears to capture the structure of r/Books. Didion is likely the last name of Joan Didion, novelist and journalist. Springmeier is likely the last name of Fritz Springmeier, who has written a number of books on conspiracy theories. Alexandre might be a nod to Dumas, and Exupery a nod to Antoine de Saint-Exupery, author of *The Little Prince.* Cronin could refer to A.J. Cronin, a Scottish novelist and Elie might refer to the famous Elie Wiesel, survivor of the Holocaust and author of *Night.* Other words in this cluster are references to elements in fantasy novels or series, such as daemon, wraith, and fingolfin. This is an exciting result because the structure of a subreddit was found with no supervision from just vector representations of comments.

For the context of this paper, a "prototypical comment" of a subreddit is one that consists words most similar to the center of a cluster that covers many comments. I took two approaches to analyzing what a prototypical comment for a subreddit would look like.

The first approach was to use words from the clusters generated by K-means. Based on this approach, a prototypical comment for r/Books would contain words from Cluster 6. This would imply that comments mentioning fantasy novels or novels by the mentioned authors are popular. A prototypical comment for r/Movies could include any of the words mentioned from Cluster 7 or Cluster 11. Though the clustering did not yield great results, I was impressed that a cluster was found around authors and books. This leads me to believe that with a robust enough model, it may be possible to recover structure of subreddits and create prototypical comments simply by representing comments as vectors and applying a clustering method.

The second approach was supervised, I found the most similar words to a one-word description, or topic, of each subreddit. This yielded much more reasonable results than finding the most similar words to a cluster's centroid. For example, the 5 most similar words to the label "philosophy" are:

'metaphysics','freud','empiricism','epistemology','sociology'

However, this method of finding words in prototypical comments of a subreddit could be confounded by cross-pollination of words. For example, the 5 most similar words to the label "history" include:

'thrillers','fascinating','dystopian','lore','parallels'

These terms seem to capture elements of book and movies, which is not unreasonable due to the existence of discussions surrounding anything's "history." This means that a one-word description of the subreddit may not be fully representative of its comments.

5.1 Nearby clusters

Since these results of clustering were not useful, it is not plausible to create a subreddit recommendation engine using this specific model. However, given the results of the clustering found a cluster about books and several clusters about movies, it is not unrealistic to believe that this recommendation engine is still possible. Once the centroids have been found and determined to accurately represent subreddits, suggesting nearby centroids is a matter of computing Euclidean distances.

6 Conclusions

From my exploration, I can draw two conclusions. The first is that noise in Reddit comments drowns out deeper structure. My hypothesis was that conversations in the comments on the bicycling subreddit, for example, were mostly about bicycling. Since comments can be about any topic, a conversation about Freud could appear on the subreddit dedicated to riding bicycles. These off-topic conversations, along with the volume of random or non-descriptive comments such as "wow! looks great!" likely skew the structure of subreddits when represented as vectors. However, it is not unreasonable to think that most subreddits have a "prototypical comment" that contains words frequently used and meaningful to the topic the subreddit is meant to discuss. This was observed in my results, with the prototypical comments found for r/Books and r/Movies.

Models such as paragraph vectors and GloVe may do a better job of sorting out the important, substantial comments from the noise. With these better models, the vector representations of the comments may cluster much better than simply averaging word vector representations.

The second conclusion I can draw is that a subreddit recommendation engine based entirely on comments is possible. With a better model trained on more data from more subreddits, there is a chance that comments will cluster together in accordance with their subreddit of origin. My results do not rule out this possibility.

7 Future work

The next step to progressing this work would be to incorporate more subreddits into the dataset. The accuracy of these models depends on the amount of training data, to a certain point. Coming just shy of 1 million comments yielded a vocabulary of size 43,723. With more comments, a better model could be trained. This would not be difficult, but would require more time to train and evaluate.

I am also curious about the results of using a generative probabilistic model of the comments. Using LDA, I would like to see if the set of topics generated somehow resemble the subreddits chosen to get the set of comments. This work would require a much deeper understanding of these generative models.

References

[1] Analogical reasoning task. `https://code.google.com/p/word2vec/source/browse/trunk/questions-words.txt`. Accessed: 2015-12-09.

[2] Jupyter notebook server. `http://jupyter-notebook.readthedocs.org/en/latest/public_server.html`. Accessed: 2015-12-09.

[3] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In Tony Jebara and Eric P. Xing, editors, Proceedings of the 31st International Conference on Machine Learning (ICML-14), pages 1188–1196. JMLR Workshop and Conference Proceedings, 2014.

[4] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. CoRR, abs/1301.3781, 2013.

[5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, Advances in Neural Information Processing Systems 26, pages 3111–3119. Curran Associates, Inc., 2013.

[6] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), pages 1532–1543, 2014.