Detecting threads on Reddit needing moderator attention using NLP

Kailhan Hokstam and Ruben Bijl

May 13th 2019

1 Introduction

Reddit is a website that hosts a collection of forums, “subreddits”, where users can make posts. These posts, also called “threads”, including for example links to other websites and text posts, other users can then comment on these posts, and “upvote” and “downvote” each other. These subreddits are moderated by users and therefore they use their own free time to ensure a high quality of threads on the subreddits they moderate. Certain subreddits such as /r/politics can attract unwanted, uncivil discussion and /r/AskScience answers, anecdotes that do not meet their guidelines on threads leading to them being “locked” by the moderators. Thus people can not comment anymore. Locking is often used when it is not possible to look at every single comment on a thread, moderators can also delete individuals comments or whole posts.

We would like to develop a classifier that can detect if a thread should be locked, based on the comments in the thread. This could be used in combination with PRAW, “The Python Reddit API Wrapper”, to automatically lock threads or notify the moderators that a thread likely needs extra moderation, which is thought to be the case if a thread has the characteristics of a locked thread.

2 Plan

Our idea is to use Pushshift in combination with Python to scrape Reddit and build a dataset of threads, comments and if they were locked or not. The threads could be picked such that there is a bias to newer threads to reflect current moderator policy. More specifically we would first like to find a subreddit with a high percentage of locked threads and build the dataset specifically for that subreddit. This will help the classifier learn because the different subreddits have different rules and moderators that might lock threads for different reasons, which hardens the problem. From then on we could look at how it generalizes to other subreddits and/or create a classifier trained on multiple subreddits. From the dataset, we want to create a representation per thread with a locked or unlocked label that we can feed to a classifier. In the end, we expect to have a script in Python that can, given a thread, a document, predict how likely it is that this thread eventually would need to be locked by the moderators. While creating this, we want to run experiments on this to obtain the best accuracy in finding threads needing to be locked. For evaluating our model we want to test the model's performance on data we haven't seen from Reddit before. For each of the classifiers created, we can measure recall, precision, and accuracy. Out of those results, plots of precision/recall will be made.

We want to use LSTM to transform word vectors based on the words in the threads to sentence vectors and then use a Gated Recurrent Neural Network (GRNN) to combine the sentence vectors and have a document representation on which we can use the softmax function and get an output that tells how likely the corresponding thread is to be locked as described by Tang (2015) and implement this in Python using Keras. To begin with pre-trained word vectors like the Twitter set, to capture a more informal way of speaking used on social media platforms, which can be obtained from Stanford's GloVe will be used and afterwards, GloVe can also be used to obtain vector representations based on training data gotten from Reddit. This structure would allow us to take relations between sentences into account. An idea is also to insert tags into the document that represents the "level" of a comment and capture the fact that comments on a thread are not independent and might be part of a (sub)discussion.

3 Literature

- Tang, D., Qin, B., & Liu, T. (2015). Document modeling with gated recurrent neural network for sentiment classification. In Proceedings of the 2015 conference on empirical methods in natural language processing (pp. 1422-1432).

- Manevitz, L. & Yousef, M (2001). One-Class SVMs for Document Classification (pp. 139-154).

- Del Vigna, F. & Cimino, A. & Dell'Orletta, F. & Petrocchi, M. & Tesconi, M. (2017). Hate me, hate me not: Hate speech detection on Facebook (pp. 86-95).

- Kotenko, I. & Chechulin, A. & Komashinsky, D. (2015). Evaluation of Text Classification Techniques for Inappropriate Web Content Blocking (pp. 412-417).