Suggesting Subreddit For The User's Reddit Submission To Maximize Score

Varsha Venkatesh
Stony Brook University
Dept of Computer Science,NY, USA
110310657

Vijay Teja Gottipati
Stony Brook University
Dept of Computer Science,NY, USA
110622613

Sai Santhosh Vaidyam Anandan
Stony Brook University
Dept of Computer Science,NY, USA
109819365

ABSTRACT

Placing social media content in the relevant community of users is a challenging task. The popularity of the post depends on factors like the time it is posted and the group it targets, apart from the quality of the content as such. Reddit is a highly trafficked social media website which has more than 203 million users and 7,700 active, user-created communities. Finding the right thread to post the specific content is crucial to gain the attention of the users. The proposed model helps the user better place the post by suggesting the subreddit that's more likely to give better votes. This is achieved by proposing a two-step model. The first step outputs a set of likely subreddits that the post can belong to and in the second step we use a suite of subreddit-specific classifiers to score the post in the context of that subreddit. We then pick the subreddit which maximizes the potential score and produce that as the output.

1. INTRODUCTION

Reddit.com is an entertainment, social networking and news site consisting of millions of users creating, commenting and voting on posts. Users can cast either positive or negative vote and the ultimate score of the post/comment is the difference between the positive(upvote) and negative votes(downvote). Based on this feedback the posts which has higher scores will move to the top page thus becoming more visible to the users. The site is gaining popularity over the years with users casting more than 21 million votes a day. The community to which the user posts plays a crucial role in determining the score of the post due to factors like number of subscribers to the community and the tone of the subscribers.

2. PRIOR WORK

Predicting the popularity of a social content is an interesting problem. To our knowledge only few prior work has worked with Reddit data. There is a significant amount of work done in predicting the popularity of a tweet based on its content by using language models. The features of the twitter data is very different from Reddit submissions, as the twitter posts are usually small messages. There have been prior work on subreddit recommendations [3]. But for our proposed work, these systems would be not be useful for finding out similar subreddits. This is because recommendation engines usually work on suggesting correlations [4]. For example, the recommendation engine would probably suggest NBA subreddit if you follow NFL subreddit. But the content of the posts in the 2 subreddits are more different than alike.

Work that analyse the popularity of reddit post by taking into consideration resubmissions [6] is another motivating prior work. The authors Himabindu Lakkaraju, Julian McAuley, Jure Leskovec have studied the impact of title, time and the user community it is posted to of various reddit submissions that resubmitted several times in different communities at different time with different title. However our work is unique as no prior work to our knowledge has taken the post content and suggested the best subreddit that the user should target in order to receive high popularity.

3. DATA COLLECTION

Reddit corpus of all user submissions is available from January 2006 to August 31 2015. The corpus consists of 200 million user submissions along with metadata about the submissions such as score, title, author, upvotes, downvotes, self text, created utc time, subreddit it belongs to and many more. The post corpus can be combined with its corresponding comments using the id of the posts that the comment refers.

The reddit data grows at the rate of 10 million submissions per month and over 50 million comments, thereby arising the problem of large data analysis. Both the post and comments dataset are available in Google Bigquery platform [1]. We used bigquery API to run SQL queries and select the posts for May 2015. The resulting dataset was 3.5GB and was too large to be exported directly from Bigquery. We then exported the table to multiple CSV files within Google Cloud Storage Bucket and then downloaded these files to disk. We then recombined all the split CSV files to get back the original data.

4. DATA CLEANING

The total size of the data was around 3,153,963 rows. From these we pruned records where either the body was null or made of links. Since the analysis was primarily based on the contents of the posts, we had to clean some rows due to the unicode encoding. Using NLTK libraries we removed punctuation, stopwords and words which are less than 3 letters long.

5. EXPLORATORY DATA ANALYSIS

The May 2015 dataset chosen contains data from more than 50,000 subreddits. For the purpose of exploratory analysis, we choose a few select subreddits.

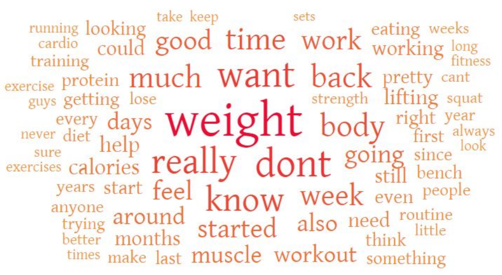
The main purpose of the EDA is to better understand the content of the post data and explore what are the features we can use for further analysis. We identify three potential features

- Frequency of words used in the post text
- Karma of user, both link and comment karma
- Time at which the posts are submitted

First, we take a look into the frequency of words used in the post self text for the subreddits to find if they are words that are specific to subreddits.

In fitness, most of the post discussions seem to revolve around weight, body, time, cardio, muscle - words that are generally associated with fitness.

Figure 1: Word Cloud - Fitness Subreddit



Moving to a different domain, we look at the nba subreddit. It is surprising how much a single player, Lebron has dominated the discussion in the NBA subreddit for an entire month. We also notice few swear words make it to our top list, something clearly absent from the Fitness subreddit.

A look at the Politics subreddit shows a large number of discussion about "people", "vote", "bill", "money" etc. It is also interesting to note that Bernie Sanders is the most discussed among the politicians while discussions regarding republicans and democrats are nearly even.

The take away from these reports is that each subreddits have their own set of frequent words. Based on this observation, we decided to use the words in posts as our main feature set for selecting the relevant subreddits.

Figure 2: Word Cloud - NBA Subreddit

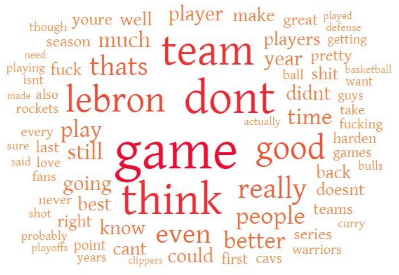
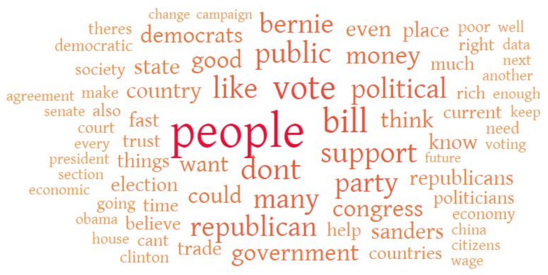


Figure 3: Word Cloud - Politics Subreddit

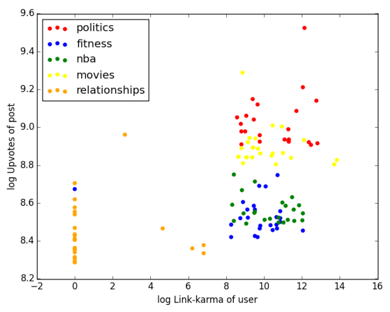


Next, we try to understand if or how karma affects the votes a post gets. Karma "reflects how much good the user has done for the reddit community." [2] Every Reddit user has two types of Karma - "Link Karma", a score calculated on the up/down votes for links he has submitted as posts and "Comment Karma", a score calculated on the up/down votes for the comments he has made. There is no explicit use of having karma, it is simply a mechanism to keep the user engaged.

To study how karma affect the posts, we use Reddit API to extract the top 25 posts at that moment from the five subreddits chosen randomly - "politics", "fitness", "nba", "movies", "relationships". We find the author of these posts and using the API get their link-karma and comment-karma.

First, we check for any discernible patterns when the votes of a post is plotted against the Link-karma of the user.

Figure 4: Link Karma Vs Upvotes of Top post

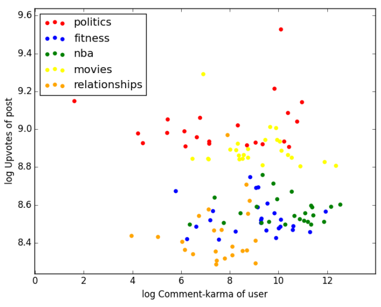


The "relationship" subreddit plot seems to suggest the users have very low link-karma. This might be because, being a largely text based subreddit, the large user base get no karma out of the posts here. For the other subreddits, we see that higher karma points does not directly relate to higher karma points within themselves.

The Pearson Correlation Coefficient (which ranges between -1 for negative correlation to +1 for positive correlation) between votes and link-karma is 0.19. This further suggests that Link-karma does not affect the votes an user gets for a post.

Next, we do a similar analysis for user votes against comment-karma. Again we see that there is no direct co-relation.

Figure 5: Comment Karma Vs Upvotes of Top post

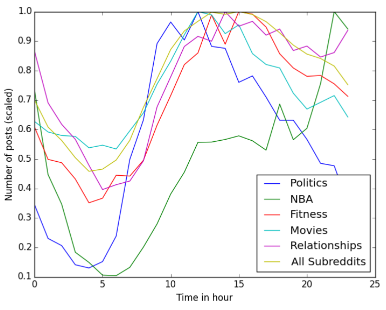


The Pearson Correlation Coefficient between votes and comment-karma is -0.07. Hence we decide that neither link-karma nor comment-karma play an important role in deciding the votes a post gets.

For these reasons, we do not consider karma of the users for further analysis.

In order to see how time influences posts, we plot the number of posts submitted every hour for each subreddit and one for all posts. We can see that the selected subreddits, along with the one for all posts more or less follow the same peaks and falls suggesting that they all tend to be active around the same time.

Figure 6: Time Vs Number of Posts



Ideally, if we had a time series of votes data for individual posts, we could have analyzed when the most votes are obtained. Since we only have the time at which posts are submitted, the only deduction we can make is when a subreddit has the most traffic. We can use this result to suggest a suitable time for the user to actually submit his post based on the subreddit's peak traffic time to get the most attention.

6. PROPOSED MODEL

To suggest the best subreddit for a given post content, we propose a 2-stage pipeline made up of the following 2 components:

1. The first component, candidate subreddit recommender, is a model that, given a post content, suggests a list of candidate subreddits that the post would best be suitable for.
2. The 2nd component, subreddit score classifier, is a classifier trained for individual subreddits that, given a post content, predicts the likely class of score that the post will receive in that particular subreddit.

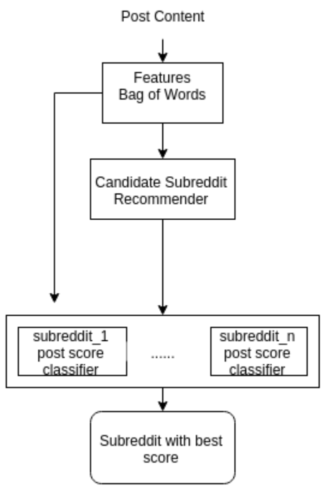


Figure 7: Flowchart representing the workflow

Pipeline: The post content is fed as input to the candidate subreddit recommender, to get the list of candidate subreddits. The post content is then fed into each the suggested subreddit classifiers to get the predicted score class. We then pick the subreddit that predicts the max score class for the post and produce that as the output.

6.1 Candidate Subreddit Recommender

The first approach we had planned to use was to find the cosine similarity between the normalized bag-of-words vectors of the post content and the normalized word distributions of the subreddits, and use this score to suggest the list of top k subreddits that the post can belong to. For this approach we will have to decide on some value of k that is common for the whole model. This can cause problems since one single value of k might be too relaxed (some posts might be similar to just few - less than k - subreddits) for some posts and too restrictive (some posts might be similar to many - more than k - subreddits) for others.

To mitigate this problem we decided on using clustering algorithms to cluster the various subreddits based on the similarity between them. Clustering algorithms are mature and implicitly choose the best partitions between the clusters, thus avoiding the need for choosing a global k. Once we have clusters, we can map the post to the corresponding cluster.

For calculating the similarity measure between the subreddits, we used the most 1000 frequent words across the selected subreddits as the bag-of-words features. We compute the words counts of these frequents words for each subreddit and use this count vector as the feature vector of that subreddit. Similarity score now becomes the Euclidean dot product of these vectors (cosine similarity), $a.b = |a||b|cos\theta$. The resultant similarity plot for the chosen 17 subreddits using the distance matrix is shown below in Figure 8. Using these vectors we use k-means clustering to cluster the selected subreddits.

Once we have clusters, we need to map the post to the cluster. There are 2 ways in which we can map a post to the corresponding cluster:

1. In the first method, we can train a model that maps a given post to its corresponding subreddit. From the clusters we have generated before, we find the cluster that the predicted subreddit and map the post to that cluster.
2. In the alternative method, we can train a model that maps a given post directly to a cluster, skipping the intermediate steps.

Using the above methods, we train a model to learn the cluster a post might belong to. The output of this component is the list of subreddits in that cluster.

6.2 Subreddit specific post score classifier

For the 2nd phase, we need a suite of subreddit specific classifers that predict the likely score class of a given post. We use a Naive Bayes model with bag-of-word features for this component. From all the post text in all the selected subreddits, we compute the 1000 most frequent words and use these words as the bag-of-word features. For every post, we then create a word vector with 1000 most frequent words and their corresponding word counts. These become the training features. For generating the target labels, we define 3 score classes as follows:

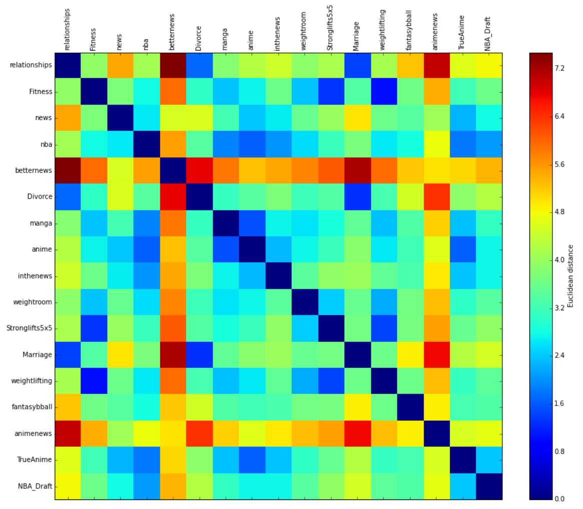
Low	Score <10
Medium	Score between 10 to 100
High	Score >100

: Score Classification

7. TRAINING

For training and evaluation, we picked 17 subreddits that belongs to 5 categories news, relationships, anime, nba and fitness. All these posts were then split into training and testing data in the ratio 2:1.

Figure 8: Similarity Matrix of Subreddits



news	46984
betternews	24684
relationships	13285
nba	12952
Fitness	11756
anime	5999
manga	2021
inthenews	1780
weightlifting	556
weightroom	356
Divorce	352
Marriage	190
Stronglifts5x5	127
NBA$_{D}raft$	89
TrueAnime	63
animenews	21
fantasybball	20

: Subreddits with number of user submissions

We have 3 models to train:

1. Post -> Subreddit classifer
2. Post -> Cluster classifer
3. Subreddit specific classifiers for predicting Post -> Score class

The first model is used for trying the 1st method in section 6.1. This model is also the baseline model for our whole system of mapping a post to the best subreddit.

We tried training all the models individually, using the Python Scikit library and found that the RAM requirement for training went past 60GB and the program crashed. To overcome this we trained the model using online learning. Instead of making the model learning the entire data in one shot, we split the training data into manageable 1000 chunks and partially fit each chunk to the model.

8. EVALUATION

The baseline model for evaluation of our system would be a subreddit classifier i.e a classifier that takes the post as input and outputs the subreddit that the post belongs to.

There are two methods in which we can evaluate our model - implicit and explicit.
Implicit Metric In this method, we filter the post that has received high positive ratings and pass each of these posts to our model. We expect the prediction for these posts to match the subreddit that the post is already in. Hence, one evaluation metric we are considering is the percentage of the high scoring post that get predicted to the same subreddit that they were posted in.

Explicit Metric The second way of evaluation is to find duplicate submissions from the Reddit corpus where the same post is submitted to different subreddits and received different scores. The 2nd evaluation metric is the percentage of posts where we correctly predict the top scoring subreddit for that post.

From the data, we could not find more than 5 posts which had the same content, but was posted across different subreddits. As a result, explicit evaluation was not feasible.
For implicit evaluation, we chose all the posts in the training data which were of the score class 2. We expect these posts to be matched to the same subreddits that they were originally posted in. The results of implicit evaluation are as follows:

Method	Accuracy
Baseline	80.88
Our model 1	46.51
Our model 2	46.59

The model 1 uses the baseline model to predict the likely subreddit and then use that to find the corresponding cluster. Model 2 directly trains on cluster label to map a post to corresponding cluster. On its own, the cluster classifier in model 2 correctly predicts the cluster of the best subreddit with a 92.49 percent accuracy. This more than the 80.88 percent we get from the subreddit classifier. As a result, we see an increase in performance of model 2 over model 1.

9. INTERPRETATION

We see that our model performs very poorly compared to the baseline model. On performing error analysis we see that the subreddit specific score classifiers perform very poorly on class 1 and class 2 predictions. We evaluated the subreddit models using F-1 metric to observe both the precision and recall of each class. The averaged results over all subreddits are as follows:

Class	Precision	Recall	F1-score
1	0.92	0.98	0.95
2	0.16	0.06	0.09
3	0.3	0.06	0.1
Avg	0.86	0.9	0.87

As we can see the F1 score of class 1 and 2 is 0.09 and 0.1 respectively. Even though we are using the baseline model as a component in our model, we are still getting worse results due to the subreddit score classifiers.

10. REFERENCES

[1] ”Reddit data hosted on Google BigQuery, https://bigquery.cloud.google.com/dataset/fh-bigquery:reddit”

[2] ”Reddit data hosted on Google BigQuery, https://www.reddit.com/wiki/faq”

[3] ”Recommending Subreddits by Computing User Similarity”, “http://aakashjapi.com/recommending-subreddits-by-computing-user-similarity-an-introduction-to-machine-learning/”

[4] ”Building a Recommendation Engine for Reddit”, “http://blog.manugarri.com/building-a-recommendation-engine-for-reddit-part-1//”

[5] AboutReddit, ”https://www.reddit.com/about”

[6] ”Whats in a name? Understanding the Interplay between Titles, Content, and Communities in Social Media”

[7] Clustering Reddits comments http://www.arimorcos.com/blog/Clustering