Popularity and Quality in Social News Aggregators: A Study of Reddit and Hacker News

[Extended Abstract]

Greg Stoddard
Northwestern University

Categories and Subject Descriptors

H.1.2 [**Models and Principles**]: User/Machine Systems

Keywords

Voting; Reddit; Position Bias; Social Influence

1. INTRODUCTION

One of the many narratives surrounding the growth of social media is that our systems for liking, retweeting, voting, and sharing are giving rise to a digital democracy of content. As the narrative goes, virality enabled "Gangnam Style" to dominate international audiences, helped the Ice Bucket challenge raise millions of dollars for ALS research, and we now interpret trending topics on Twitter as a signal of societal importance [6]. There's a considerable amount of academic work that interrogates this narrative by delving deeply into understanding the properties of virality. For example, scholars have studied the propagation and correction of rumors [4], the role of influential users in spreading information [1], or whether information actually diffuses in a viral way at all [7].

Although many papers hint at it, few papers directly address a basic question: do these systems promote the best content? Does this "digital democracy" actually work? As a thought experiment, imagine polling a large population of people and asking them to rate every music video uploaded to Youtube in 2012. Would "Gangnam Style", the most watched video on Youtube, still come out on top? Evidence from the MusicLab experiment of [12] suggests that it might not. In this experiment, the authors set up a website where users could listen to and download songs from unknown artists. When visiting the site, participants were randomly assigned to a "world" and presented a list of songs that were ranked by the number of downloads the song had in that world. This design let the authors observe the evolution of popularity of the same song across different worlds. They also included one world in which songs were ranked randomly. The number of song downloads in this control world served as a measure of intrinsic song quality.

They found that the popularity of a song could vary wildly across worlds; songs with the largest share of downloads in one world went relatively ignored in another one. Higher quality songs were more popular on average, but there was a large variance in popularity for all but the best and worst songs. This variance was caused by a rich-get-richer effect. Songs with more downloads were ranked higher in the list and were more likely to be sampled by future listeners. Furthermore, participants were able to see the number of current downloads each song had, and were more likely to sample songs with a higher number of downloads. In the presence of such effects, the authors conclude, popularity is a noisy and distorted measure of quality.

Present Work What do these results imply about the relationship between quality and popularity on real world socio-technical systems? Facebook, Twitter, etc all have a rich-get-richer phenomenon because posts with more likes, retweets, and views are more visible, on average, than their less popular counterparts. Does this imply that there's a distorted relationship between quality and popularity on these platforms? Unfortunately we do not have the ability to run randomized experiments on these platforms, so the main challenge of answering this question is developing a metric of quality that can be estimated from observational popularity data.

In this paper we show that social news aggregators are a good setting to study the quality-popularity relationship relative to other social media sites. We conduct our study on two aggregators, Reddit and Hacker News. Reddit is a popular site where users submit links to content from around the web, and other users vote and comment on those links. Hacker News is an aggregator dedicated to programming and technology-related issues but is otherwise similar in structure. Reddit received approximately 450 million page views in December 2014, while Hacker News received approximately 3.25 million.

These aggregators have several properties that facilitate disentangling observed popularity from inherent quality. The first property is that content visibility is easier to measure on Reddit and Hacker News. The interface of each site is a simple non-personalized list of links, so the observed article ranking is (approximately) the same for all users. Due to the similarities in UI, estimating visibility on Reddit or Hacker News is very similar to estimating position bias in search results and search ad rankings. We exploit this similarity in our techniques. The second property is that both sites only use votes to rank articles, rather than more complex measures like impressions or social-tie strength, and these votes are publicly observable. Furthermore, each site publishes their algorithm for converting votes into a ranking.

Finally, recent empirical work shows that popularity on Reddit exhibits signs of a distorted relationship between quality and popularity [5]. Gilbert finds that over half of popular image submissions on Reddit are actually reposts of previous submissions. The same picture may receive no upvotes on it's first submission but its second or third submission may gain thousand of upvotes.

WWW 2015 Companion, May 18–22, 2015, Florence, Italy.
ACM 978-1-4503-3473-0/15/05.
http://dx.doi.org/10.1145/2740908.2742470.

Our Contributions[1]
The main contributions of this paper are developing a metric for article quality and a method to estimate it from observed voting data. We define quality by the score (number of upvotes or upvotes minus downvotes, depending on the site) that an article would have received if articles were ranked randomly and no social signals were displayed about the articles. This is only a hypothetical process but we show that we can estimate this counterfactual score from observed popularity data.

The key to our analysis is the use of time-series observations of voting behavior for each article. Observing the same article at different points in it's life allows us to disentangle the influence of different factors on voting. We develop a simple poisson regression model for learning parameters from observed data. Our model includes factors for article and position effects, as well as time decay and social influence. Since we lack the ability to evaluate against ground truth data from Reddit or Hacker News, we evaluate this model on data from the MusicLab experiment. We find this method is effective at recovering ground truth quality parameters, and further show that it provides a good fit for Reddit and Hacker News data.

We then examine the relationship between article quality and popularity using the developed quality estimates. We find a surprisingly strong relationship between popularity and quality but with an important caveat. Many articles submitted to Reddit and Hacker News did not just generate enough observations to be included in our analysis, and its likely that there are many high quality articles that never received even a small amount of attention. However among the set of articles with a reasonable amount of attention, we conclude that popularity is a good indication of relative quality.

2. DATA

The design of Reddit and Hacker News are quite similar. The interface of each site is an ordered list of articles, with 25 or 30 articles appearing on each page. Logged-in users of each site can upvote or downvote each article, and these votes are used to rank articles.

Reddit Reddit is composed of many different sub-communities called "subreddits". For example r/news[2] is the subreddit for discussing news and current events. Within a subreddit, articles are ranked in decreasing order of their "hot score", which is defined by[3]:

$$\log(u_i - d_i) - \frac{1}{750}\text{age}_i$$

Where u_i, d_i is the number of upvotes and downvotes received by article i and age_i is the number of minutes between the current time and the time the article was submitted [4].

Hacker News Hacker News allows people to upvote stories but not to downvote them. Second, there are only two different article rankings: the "new" ranking which is a chronological list of articles, and the "top ranking". In the "top ranking", articles are ranked by[5]:

$$\frac{(u_i - 1)^{.8}}{(age_i + 2)^{1.8}}$$

Data Collection We collected data at 10 minute intervals over a two week period from 5/26/14 to 6/6/14 from each site. For each site, we record the number of votes (upvotes and downvotes) and position of each article. We can compute the number of votes an article received in the 10 minutes between scrapes using this data. For our purposes, each observation is a tuple (t, i, j, v_i^t), meaning that article i at time t was observed in position j, and received v_i^t upvotes in the time period t to $t+1$. For Reddit, each observation is a tuple $(t, i, j, v_i^t, s_i^t u_i^t, d_i^t)$ where u_i^t and d_i^t are the number of upvotes and downvotes, $v_i^t = u_i^t + d_i^t$ is the total number of votes and $s_i^t = u_i^t - d_i^t$ is the change in score. We collect all articles that appear in the top ranking of Hacker News (which is at most 90), and the top 500 ranked articles in five different subreddits. We then filter the data in a number of ways, such as limiting observations to be during 8am to 8pm EST on weekdays, removing articles for which we only observe few data points, etc. Summary statistics for the filtered datasets are shown in table 1.

Dataset	Observations	Articles	Score
Hacker News	29K	750	66 (39)
r/todayilearned	40K	1187	125 (16)
r/videos	45K	2249	42 (2)
r/worldnews	40K	1417	39 (6)
r/news	33K	1132	38 (6)
r/pics	57K	1883	53 (5)

Table 1: Summary statistics for the data used. The last column shows the mean (and median) score for articles in the dataset.

Terminology: In this work, we'll refer to *score* as the number of upvotes in the case of Hacker News, or the difference of upvotes and downvotes in the case of Reddit. We'll also use that term to refer to an article's score at a specific point in its life, i.e. score at time t. We use the term *popularity* to refer to its final score, i.e. the score it has at the end of it's lifetime.

3. MODEL

The measure of quality that we wish to capture is the score an article would have received if voting were free from biases. Specifically, we define quality to be the total score an article would have received if articles were ranked randomly and no social signals were displayed about the articles. We use a model that separates observed voting data into confounding factors, such as position and social influence bias, and article-specific factors. After fitting this model, we use the fitted parameters to estimate article quality.

The largest issue is that we do not observe the number of users who may have viewed an article but decided not to vote on it. The observed Reddit data allows us to directly estimate the probability that an article will receive an upvote conditioned on it receiving a vote by taking the ratio of upvotes to total votes. However we cannot directly estimate the probability of receiving a vote, for both Reddit and Hacker News. This problem is exacerbated by the presence of a potential strong *position bias*, i.e. that users are more likely to look at highly ranked articles than articles that are buried down in lower pages. Fortunately this is a common problem encountered in estimating the click-through-rates of search results and ads ([3],[2]), so we can use techniques from this literature. One model used in this literature is the *examination hypothesis*, proposed by [11]. If the users examines position j, they click on that article with probability q_i. Thus the probability that a user clicks on article i in slot j is $q_i \cdot p_j$. The p and q parameters can then be estimated from observed clicking behavior in search logs.

The analogy from estimating the probability of an article receiving a click to an article receiving a vote is straightforward, but direct application of this model isn't possible because the granularity of

[1]A detailed, full version of this paper is available on author's website.

[2]by convention, "r/" is prefixed to the name of a subreddit

[3]`github.com/reddit/reddit`

[4]There's additional logic to handle the case where $d_i \geq u_i$ but most of our observations have $u_i > d_i$

[5]`news.ycombinator.com/item?id=1781013`

our data is votes cast over a 10 minute interval rather than individual voting data. We must instead estimate the rate that an article receives votes. A natural model for modeling rates is a poisson process, and recent work [2] shows that the binomial model of the examination hypothesis can effectively be replaced with the following poisson model:

$$v_i^t \sim \text{Poisson}(\exp(p_i^t + q_i))$$

Where v_i^t is the votes received by article i at time t and p_i^t is the position it appeared in. This model accounts for position bias but there are other factors that may affect voting behavior. We first add an age factor to allow for the interestingness of an article to decay over time. Next we add a factor to account for a potential social influence bias. Both sites display the current score of articles, and thus signal something about how other users evaluated these articles. Prior work shows that displaying current popularity can cause a significant social influence on user behavior [8], [10],[9], [12]. We add a term for score effects but first apply a log transformation to account for the large disparities in scores on Reddit and Hacker News. Our full model is as follows:

$$v_i^t \sim \text{Poisson}(\exp\{p_i^t + q_i + \beta_{age} \cdot age_i^t + \beta_{score} \cdot \log(S_i^t)\}) \quad (1)$$

In summary, the full model estimates an article quality effect q_i for each article, a position bias effect p_j for each position, a time decay effect β_{age}, and a score effect β_{score}. We emphasize that the position variables are treated as categorical variables, meaning that a position bias is estimated for each position j and there's no necessary relationship between p_j and $p_{j'}$ for all j, j'. We learn parameters via maximum likelihood estimation, that is we find the value of parameters that maximize the probability of the observed data in the poisson model. This is exactly equivalent to a standard poisson regression. We use the StatsModels python module[6] to implement the poisson regression.

4. EVALUATION

One challenge in evaluating this model is that we do not have ground truth to compare against. Instead we validate this model by applying it to data from the MusicLab experiments [12] and comparing against the ground truth estimates from that experiment. We find the model performs quite well at recovering the ground truth from that experiment. We then show the poisson model is a good fit for the Reddit and Hacker News data, even when evaluated on out-of-sample data during cross-validation.

MusicLab

Participants in the MusicLab experiment [12] were shown a list of unknown songs that they could listen to and download. When participants entered the website, they were assigned to 1 of 9 different worlds. In the first 8 worlds, songs were ordered by the number of downloads the song received within that world (download counts were displayed to users). In the 9th world, songs were shown in a random order to each user and the current download count was not displayed. We use data from the first 8 worlds, the ones which were ranked by popularity and subject to social influence, to train the poisson regression model and predict the number of downloads of each song in the random world. We find this method is quite accurate; the linear relationship between predicted downloads and observed downloads (the ground truth) is strong. It underestimates true downloads by approximately 10%, but does so consistently for each article and explains 80% of the total variance in popularity.

Reddit and Hacker News

[6] http://statsmodels.sourceforge.net

	In Sample Fit			Out of Sample Predictions		
	R^2	MAE	MSE	R^2	MAE	MSE
r/pics	0.76	1.09	7.30	0.62 (0.01)	1.14 (0.01)	8.51 (0.40)
r/videos	0.79	1.15	9.62	0.65 (0.03)	1.22 (0.01)	13.64 (2.59)
r/todayilearned	0.71	1.75	22.66	0.61 (0.03)	1.85 (0.02)	32.24 (3.74)
r/news	0.56	1.11	3.63	0.57 (0.01)	1.14 (0.01)	3.87 (0.18)
r/worldnews	0.57	1.27	9.10	0.52 (0.01)	1.32 (0.01)	10.65 (1.17)
Hacker News	0.69	0.70	1.82	0.65 (0.01)	0.74 (0.01)	2.08 (0.11)

Table 2: Accuracy metrics for the full Poisson model. In-sample values show the fit of the model to the dataset when all data is used. Out-of-sample predictions are fit on a training set and predicted for a test over 5 fold cross-validation (standard errors shown in parentheses).

Given that our model effectively recovers ground truth data from the MusicLab experiment, we now turn to evaluating it on the Reddit and Hacker News data. For each observation v_i^t, the predicted number of votes $\hat{v}_i^t$ is equal to the conditional mean of the poisson distribution, e.g.

$$\hat{v}_i^t = \exp\{q_i + p_i^t + \beta_{age} \cdot age_i^t + \beta_{score} \cdot \log(S_i^t)\}$$

For Reddit this only predicts the number of votes on an article, not the change in score. To predict the rate of upvoting, we multiply expected votes by the probability of receiving an upvote conditioned on receiving a vote. We can similarly predict the rate of downvoting and then take the difference to predict the change in score. Let r_i^{up} be the observed ratio of upvotes to total vote for article i and r_i^{down} be the ratio of downvotes. The predicted growth in score for article i at time t is:

$$\hat{s}_i^t = \hat{v}_i^t \cdot (r_i^{up} - r_i^{down})$$

We evaluate the accuracy of predictions using the coefficient of determination (R^2 value), mean absolute error, and mean squared error. We compare predicted votes to observed votes for Hacker News ($\hat{v}_i^t$ vs v_i^t) and predicted change in score to observed change in score for Reddit ($\hat{s}_i^t$ vs s_i^t). In addition to reporting the accuracy on the in-sample data, we run a 5-fold cross-validation and report prediction accuracy on the out-of-sample data points. Train-test splits are constructed by randomly dividing observations into five partitions. Every partition will contain at least a few observations of all articles, and many observations of all positions. Results are shown in table 2. The model performs well for both in-sample and out-of-sample prediction, capturing between 50% and 80% of the variance in the voting data.

5. ANALYSIS

We now use the parameter estimates from the poisson model to examine the relationship between estimated quality and observed popularity. We treat data from different subreddits as completely different datasets, so there's no relationship between parameters across different subreddits. Recall that our definition of an article's quality is the expected score of an article in a hypothetical voting process where the ordering is randomized and the current score is not displayed. Quality, denoted Q_i, can be estimated with the fitted q_i parameters by the following:

$$Q_i = \lambda \cdot e^{q_i}$$

Where λ is some constant related to age and position effects. The λ term is the same for all articles within a dataset because the ordering is randomized in this hypothetical voting process. Similarly, the score term is dropped because from the above expression because score would not be shown.

$$Q_i = \lambda \cdot e^{q_i} \cdot (r_i^{up} - r_i^{down})$$

	Score	Views
Hacker News	.80	0.49
r/todayilearned	.75	0.81
r/videos	.63	0.70
r/worldnews	.54	0.70
r/news	.59	0.75
r/pics	.63	0.77
MusicLab	.57	0.35

Table 3: Spearman correlation between estimated quality and observed score, and quality and estimated views.

We measure the relationship between quality and observed popularity using the spearman correlation coefficient. The first column of table 3 shows the spearman correlation coefficients between quality and popularity. Hacker News has the strongest relationship with a correlation of .8 and r/worldnews has the weakest with a correlation of .54. Qualitatively we observe that popularity is generally increasing but there articles of similar quality can experience large differences in popularity. There are only a few instances of a mediocre quality article becoming one of the most popular articles in a subreddit, and few instances of high quality articles ending up with low scores. These results are generally consistent with the MusicLab experiment but we do find that the quality and popularity have a significantly stronger relationship in Reddit and Hacker News than in the MusicLab experiment.

We had initially expected the quality-popularity relationship to be weaker on Hacker News than Reddit because of the lack of downvoting. Our theory was that a low quality article that made it to the front page of Hacker News would remain for a long time and become popular because there was no ability to downvote it off of the front page. This theory is partially true; the second column in table 3 shows the relationship between quality and total views. We estimate total views by $\sum_t \exp\{p_i^t\}$, i.e. the sum of position biases for the positions that article i appeared in during its lifetime. The relationship between total views on Hacker News is much weaker than on Reddit, indicating that lower quality articles are being seen comparatively more often on Hacker News. However this did not translate to a weakened quality-popularity relationship as we had expected.

Discussion There is one important caveat to these results. Many articles submitted to Reddit and Hacker News fail to gain any votes and quickly disappear. For example, there were 5000 articles submitted to Hacker News over the period of observation but only 1500 of ever appeared in the top ranking. On r/pics, only 25% of pictures made it into the top 100 ranking at some point. And even though we filtered the data so that we had a reasonable number of observations for each article, the median article on Reddit still had a score of 5 or 6.

One interpretation of these results, combined with the results of [5], is that there's a sort of "two-stage process" of popularity on Reddit and Hacker News. Many articles fail to receive any reasonable amount of attention after being submitted and fade into obscurity, regardless of quality. However, amongst the set of articles that receive a reasonable amount of attention, relative popularity is a strong indication of relative quality.

6. CONCLUSION AND FUTURE WORK

This paper tries to understand the relationship between intrinsic article quality and popularity in social news aggregators. The heart of the problem is estimating parameters from data that allow us to reason counter-factually about the popularity of an article if the voting process were not subject to the large biases that exist in reality. We found that the most popular content on Reddit and Hacker News were, for the most part, higher quality articles than less popular content, which is surprising given the number of confounds on Reddit and Hacker News.

The poisson regression model presented in this paper is an initial approach to quality estimation, and can be improved in many ways. The most immediate is expanding the model to include a richer set of temporal features and social influence related features, such as commenting data. Although the role of social networks is relatively minimized on social news aggregators, we suspect that we could improve prediction accuracy on voting data from an article's early lifetime by incorporating such features. There are a number of limitations to this study. The main limitation is that our method cannot estimate the quality of a large set of articles because they do not remain in the rankings of Reddit or Hacker News long enough to generate many observations. This highlights the interesting property that early voting has a large influence on eventual popularity. Quantifying the influence of early voters on popularity and its implications is an interesting direction for future research.

7. REFERENCES

[1] E. Bakshy, I. Rosenn, C. Marlow, and L. Adamic. The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web*, pages 519–528. ACM, 2012.

[2] Y. Chen and T. W. Yan. Position-normalized click prediction in search advertising. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 795–803. ACM, 2012.

[3] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 331–338. ACM, 2008.

[4] A. Friggeri, L. A. Adamic, D. Eckles, and J. Cheng. Rumor cascades. 2014.

[5] E. Gilbert. Widespread underprovision on reddit. In *Proceedings of the 2013 conference on Computer supported cooperative work*, pages 803–808. ACM, 2013.

[6] T. Gillespie. Our misplaced faith in twitter trends. *Salon*, October 2011.

[7] S. Goel, D. J. Watts, and D. G. Goldstein. The structure of online diffusion networks. In *Proceedings of the 13th ACM conference on electronic commerce*, pages 623–638. ACM, 2012.

[8] T. Hogg and K. Lerman. Effects of social influence in peer online recommendation. *arXiv preprint arXiv:1410.6744*, 2014.

[9] C. Krumme, M. Cebrian, G. Pickard, and S. Pentland. Quantifying social influence in an online cultural market. *PloS one*, 7(5):e33785, 2012.

[10] L. Muchnik, S. Aral, and S. J. Taylor. Social influence bias: A randomized experiment. *Science*, 341(6146):647–651, 2013.

[11] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *Proceedings of the 16th international conference on World Wide Web*, pages 521–530. ACM, 2007.

[12] M. J. Salganik, P. S. Dodds, and D. J. Watts. Experimental study of inequality and unpredictability in an artificial cultural market. *science*, 311(5762):854–856, 2006.