TECHNISCHE UNIVERSITÄT BERLIN

AIM-3: SCALABLE DATA ANALYSIS & DATA MINING

Trend and Topic Detection on Hacker News

USING LDA AND A COMBINATION OF WORD2VEC AND K-MEANS

Authors:
Bram LEENDERS & Marc ROMEYN[1]

July 14, 2015

[1] {bcleenders, marc.romeyn}@gmail.com

1 Introduction

1.1 What is HackerNews

Hacker News is a social news site: it aggregates news by allowing users to submit stories. Interesting submissions can be upvoted by other users and all submissions are ranked by popularity.
The content on Hacker News is mostly related to science, in particular computer science. The guidelines for what content can be posted are very broad; the guidelines specify on topic as "*anything that gratifies one's intellectual curiosity*"[2].
Hacker News started in February 2007, eight years ago at the time of writing, and has experienced rapid growth, resulting in a daily 2.6 million pageviews and 3.5 unique visitors per month[3]. One of the reasons suggested for this popularity is Hacker News' similarity to how Reddit used to be[4]: user-submitted content with a very minimalistic, terminal-like interface.

1.2 Research Question

A lot happened and changed during the last eight years, especially in the field of computer science. This research is ment to demonstrate how these real-life events are correlate with the activity on Hacker News.
This lead to the following research question:

Research Question: what correlation can we find between real-world events and the popularity of corresponding topics on Hacker News?

This question depends on two other questions, since we have not yet specified what we mean by popular nor what topics we mean exactly. These subquestions are:

SubQuestion 1: what topics does the Hacker News content consist of?
SubQuestion 2: how does one quantify the popularity of a topic?

1.3 Outline

We first address the second subquestion in section 3 by ranking posts and users over the entire Hacker News history. This will help characterize the dataset and show how various ways of measurent often give similar results.
Dividing elements into clusters is a common machine learning task. We have used two methods to divide the articles into categories: latent Dirichlet allocation and a combination of Word2Vec and k-means. Both methods are explained in section 4.
The results of the classification and the trends are shown in section 5.

[2] `https://news.ycombinator.com/newsguidelines.html`
[3] `https://news.ycombinator.com/item?id=9219581`
[4] `http://techcrunch.com/2013/05/18/the-evolution-of-hacker-news/`

2 DATASET

As explained in the previous section, our analysis is on the set of all stories posted on Hacker News. It is a large set of news articles, blog posts, essays, tutorials and other types of mostly textual media. Nearly all of it is English, although there are a few other languages used as well. Let us first describe what exactly we refer to when we use the word story. To do this, we will describe the four catagories of content on Hacker News:

- Stories: the majority of submissions are stories. A story can be either a link to another webpage or a relatively short text by the submittor.
- Jobs: companies sponsored by YCombinator (the seed investor behind Hacker News) can post job offers on Hacker News. The percentage of jobs is very small: under one percent of the total volume.
- Polls: users can submit multiple choice questions for other users to answer.
- Comments: the three types mentioned above can receive comments by other users.

Since this research focuses only on the stories, we have left out the other three types. The dataset used in our research contains all stories between February 19th 2007 (the date Hacker News was launched[5]) and June 10th 2015 (the day we ran our crawler). This is a time span of 3033 days, during which a total of over 1.5 million stories were submitted.

2.1 DATA PROCESSING STEPS

The following steps were taken in the data processing:

1. **Data retrieval**: crawl all data and write to json files. Done on a single machine, but perfectly parallelizable.
2. **Simple analysis**
   - Store in Postgres
   - Load into Tableau
3. **Train models**: use unsupervised LDA and Word2Vec&k-means to create models of the topics. Done on the PowerLinux cluster at TUB.
4. **Apply models**: given the trained models, assign each story to a topic. Output tuples with the topic and article id.
5. **Trend detection**: using the article information and article/topic tuples, we can calculate the popularity of a topic in a given month. By checking this manually, we could spot trends. We did this with two different techniques:
   - **Zeppeling**: an Apache project for data analysis. Works very well but not always very intuitive/accessible.

[5] `https://news.ycombinator.com/hackernews.html`

- **Postgres**: the database of all stories and topics is several gigabytes in size. Although that's pretty large, a single database can still handle it. This allowed for some easier querying of the data since Postgres supports more SQL functions than Zeppelin.

2.2 Data Retrieval

To crawl all stories, we used the official Hacker News API[6]. This API returns some basic metadata about the story, such as the submitter, title, points (upvotes), a (possibly empty) story text and a (possibly empty) url. Stories generally either have a story text or a url, most only have a url.
For the stories with a non-empty story text field, fetching the story from the API is enough. For other stories, we also fetched the content linked to by the url. An important remark here is that some urls (especially the old ones) have become invalid over the course of time. Whenever the content was no longer available, we set the text to an empty string (the metadata can still be processed for user statistics etc.).
To crawl all stories (including the linked content), we used a homemade crawler that fetches the content, strips out meaningless text and html and saves the result in chunks of 1 day's worth of content. The code for this crawler is publicly available on GitHub[7].
The algorithm used for the extraction of meaningfull content is GoOse[8]. It uses heuristics to rank the importance and relevance of html elements on a webpage. For example: if it detects a `<div>...</div>` block with a lot of words inside, then that is likely to be important. If, on the other hand, it finds an html block `<button>Login</button>`, then it will remove the block for it is probably not a relevant part of the text of the page.
The resulting dataset is about 4.4 GB in size.

3 Exploratory Research

We have now established some semantics of the data gathered for our research. Before diving into the data analysis, we will first quantify the data by providing the reader with some statistics.

3.1 Overall Activity

Statistics	
Stories	1,544,261
Submitters	165,126
Upvotes	16,668,848
Comments	7,383,865

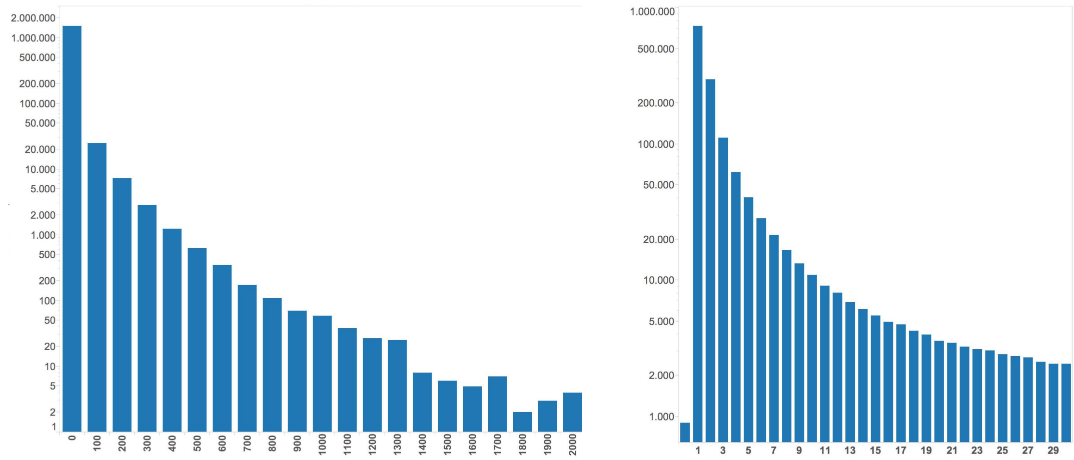
[6] https://hn.algolia.com/api
[7] https://github.com/bcleenders/AIM/tree/master/crawler
[8] https://github.com/advancedlogic/GoOse

An interesting remark, is that January 6th 2014 did not have any submissions. The reason for this was a long downtime of the Hacker News website [9].

3.2 Upvote Distribution

If users like a story, they can express their approval by upvoting it. These upvotes are then used to rank the stories by popularity and calculate the (currently) most popular submissions. Since the formula for determining a posts "score" strongly favours newly submitted stories, even stories with few upvotes can spike to the frontpage.

To enter the frontpage (the top 30), a story often needs at least ten upvotes, stories that stay on the frontpage for a longer period of time will often have over a hundred upvotes. The distribution of upvotes per story can be seen in figure 3.1. Stories placed in buckets of size 100 based on the number of upvotes they received. The graph plots the size of each bucket Note that these graphs are scaled logarithmically: in the graph showing all posts (figure 3.1a), 97.5% of the stories are in the first bucket, i.e. only 2.5% of the stories get over 100 upvotes. The second graph (3.1b) is a zoomed in version with a bucket size of 1, and shows the distribution for points with 30 upvotes or less, a set of over 1.4 million stories.



(a) Distribution over all stories
Bucket size 100

(b) Distribution over low-ranked stories
Bucket size 1

Figure 3.1: Distributions of upvotes over posts (log scale)

There are only six stories with over 2100 upvotes. For completeness' sake, these are the stories whose popularity is quite literally "off the charts":

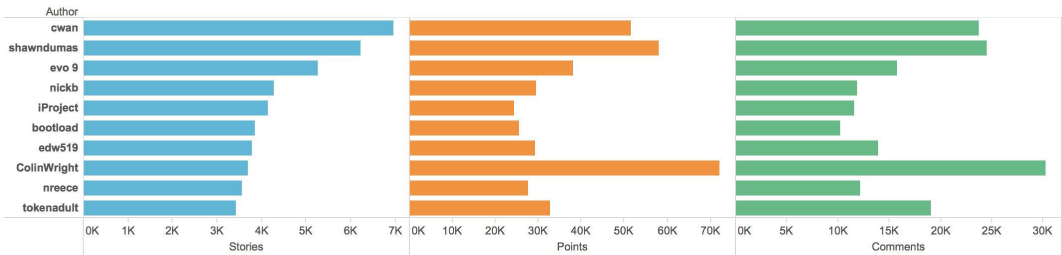
[9] `https://twitter.com/hackernewsonion/status/420068968464789505` - "Hacker News is DOWN, but your chances of getting into YC if you know how to scale a plain text website are UP."

Most upvoted stories	
Title	Points
Steve Jobs has passed away	4271
Tim Cook Speaks Up	3086
2048	2732
Don't Fly During Ramadan	2617
Hyperloop	2549
Microsoft takes NET open source and cross platform	2376

3.3 Top Users

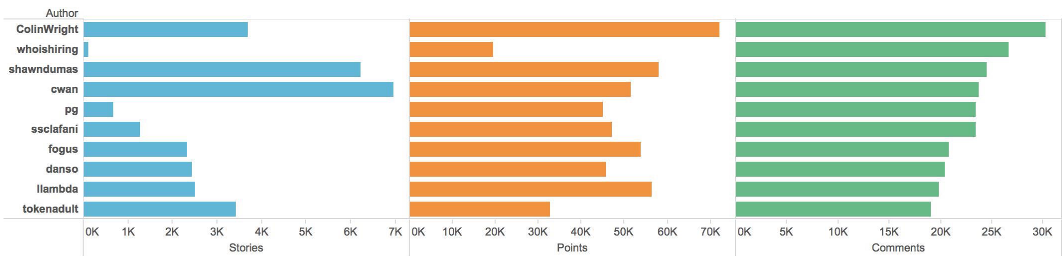
In figure 3.2, we show the statistics for some of the most active submitters. These statistics illustrate that Hacker News has a very active group of users. Together, this top 10 submitted 45,190 stories, which equals an average of 1.5 stories per user per day over the last 7 years.

Figure 3.2: Top 10 submitters by submitted stories



If we take the top 10 not by the number of stories a user has submitted but by the number of comments his or her stories have received, we get the rankings in figure 3.3.

Figure 3.3: Top 10 submitters by received comments



The interesting second place in this top 10 is user "_whoishiring". This user has almost no posts (a mere 114) but received 26,649 comments. Experienced Hacker News readers may have already expected this, for this user starts a monthly thread where companies can post job offers and others can respond to these. As such, the user only posts one story per month but its stories are discussed very actively.

3.4 Top Domains

As already stated in the description of the dataset, many of the stories are links to external sites. To provide some insights into which sites attract a lot of attention from the Hacker News community, we made three top ten lists that rank the sites by the same criteria as we did in the previous section: by number of stories, by points and by number of comments.

Most popular domains		
By Number of Stories	By Points	By Comments
techcrunch.com (27.711)	github.com (400.373)	techcrunch.com (172.854)
github.com (26.596)	techcrunch.com (365.207)	nytimes.com (153.471)
youtube.com (21.977)	nytimes.com (287.234)	github.com (127.812)
nytimes.com (18.125)	arstechnica.com (176.633)	arstechnica.com (80.666)
medium.com (14.172)	wired.com (161.698)	wired.com (75.406)
arstechnica.com (12.657)	medium.com (132.468)	washingtonpost.com (56.257)
wired.com (10.867)	bbc.co.uk (98.031)	medium.com (53.924)
bbc.co.uk (8.118)	washingtonpost.com (97.537)	bbc.co.uk (51.828)
en.wikipedia.org (7.058)	youtube.com (96.339)	theatlantic.com (41.530)
businessinsider.com (6.877)	theatlantic.com (77.628)	online.wsj.com (36.729)

These rankings provide some insights into what types on news are popular. The big geeky news sites (Techcrunch, Ars Technica and Wired) are present and are about as popular as the big newspapers (NY Times, BBC, Washington Post).
The high ranking of github.com (a code hosting site, *not* a news site) can be explained by the large number of open source projects hosted on GitHub that submit links to new versions and press releases on Hacker News. Some examples of these projects are Facebook's React framework, Twitter's Bootstrap, SQLite and io.js.
Notably, GitHub's competitors (e.g. BitBucket, GitLab and Beanstalk) do not show up in these rankings. This demonstrates GitHub's overpowering popularity in the code hosting market, at least in the open source community.
In the top lists given above, we have provided the number of articles, points and comments on groups of articles. The strong similarity between the different methods of ranking indicates a strong correlation between how well a story scores on various features. This is in line with what one might expect: people post, like and comment most of things they consider most interesting.

4 Clustering Articles Into Topics

In order to cluster the articles into topics we chose two different unsupervised approaches.

1. LDA: clusters based on the content of the articles. This algorithm represents articles as a mixture of different topics which contain words with a certain probability.

2. A combination of Word2Vec with K-Means: clusters the articles based on the titles. With a trained Word2Vec-model every word can be transformed into a vector. Transforming a title into a vector is simply adding all the vectors of the words together. These title vectors can then be fed into K-Means in order to do the unsupervised clustering.

4.1 Latent Dirichlet Allocation

Let's start by formally define the terms used in Latent Dirichlet Allocation (LDA):

- ***Word:*** the basic unit of discrete data, defined to be an item from a vocabulary indexed by $\{1,\ldots,V\}$. We represent words using unit-basis vectors that have a single component equal to one and all other components equal to zero. Thus, using superscripts to denote components, the vth word in the vocabulary is represented by a V -vector w such that $w^v = 1$ and $w^u = 0$ for $u \neq v$.

- ***Document:*** a sequence of N words denoted by $\mathbf{w} = (w_1, w_2, \ldots, w_N)$, where w_n is the nth word in the sequence.

- ***Corpus:*** collection of M documents denoted by $\mathbf{D} = \{\mathbf{w_1}, \mathbf{w_2}, \ldots, \mathbf{w_M}\}$.

LDA is a generative probabilistic model of a corpus. The basic idea is that documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words.[1]

4.2 Word2Vec

4.3 Measuring popularity of a topic

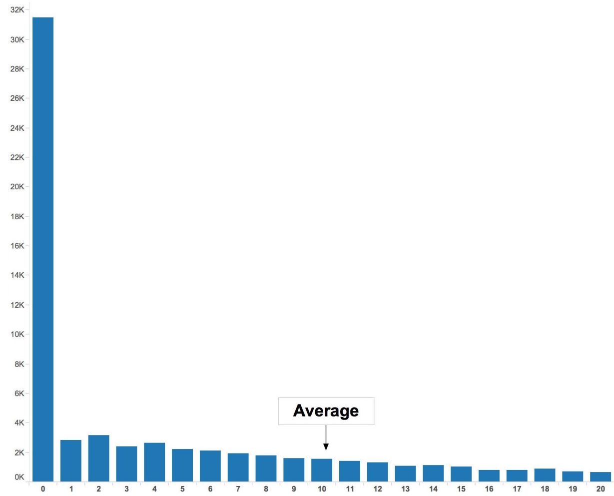
First and foremost, our popularity must account for different lengths of months and a smaller userbase in the early years of Hacker News. To do so, we must use relative scores per month, rather than providing absolute number.
We have decided to base our popularity measure on all three features we used for the rankings above: number of articles, number of upvotes and number of comments. If a topic receives 2% of the arcticles, 2% of the upvotes and 5% of the comments, the popularity score is 3% (the average of the three).
To give a formal definition: let S_m be the set of all stories in month m and T_i all stories in the topic number i. With these two definitions, $S_m \cup T_i$ is the set of all stories on a topic i in month m. Furthermore, let s_u (resp. s_c) be the number of upvotes (resp. comments) story s has received. Then, given a topic id i and a month m, the score for that topic in that month is:

$$score(i,m) = \frac{1}{3}\frac{|S_m \cup T_i|}{|S_m|} + \frac{1}{3}\frac{\sum_{s \in S_m \cup \in T_i} s_u}{\sum_{s \in S_m} s_u} + \frac{1}{3}\frac{\sum_{s \in S_m \cup \in T_i} s_c}{\sum_{s \in S_m} s_c}$$

5 Results

In this section, we will show and discuss some of the results of the research described in the earlier sections. More specifically: we show how exactly the popularity of various topics changed over time. Since we have a thousand topics, we cannot show charts for every individual topic. Instead, we picked a few topics we consider to be interesting and show these. Note that, since there are 1,000 topics, the average popularity is 10 basis points (10‱). The popularity is positively skewed, however: roughly a third of the months/topic combinations have a popularity below 1‱. To stress this point, we include figure 5.1.

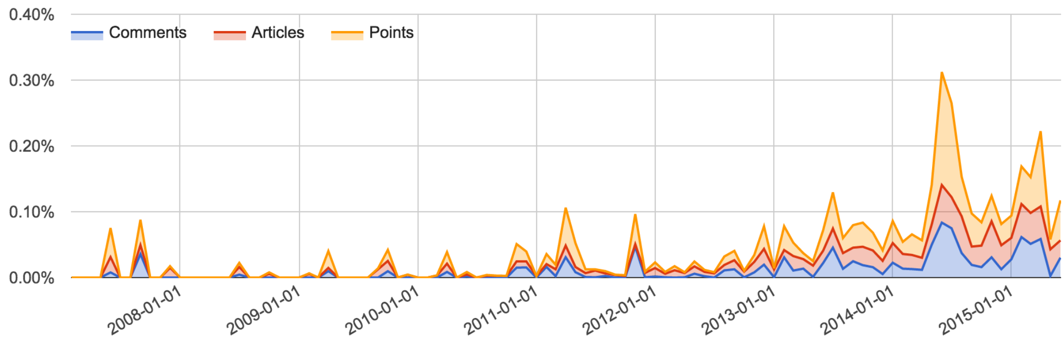
Figure 5.1: Popularity distribution - popularity (in bp) vs. number of topic/months



5.1 Popularity plots

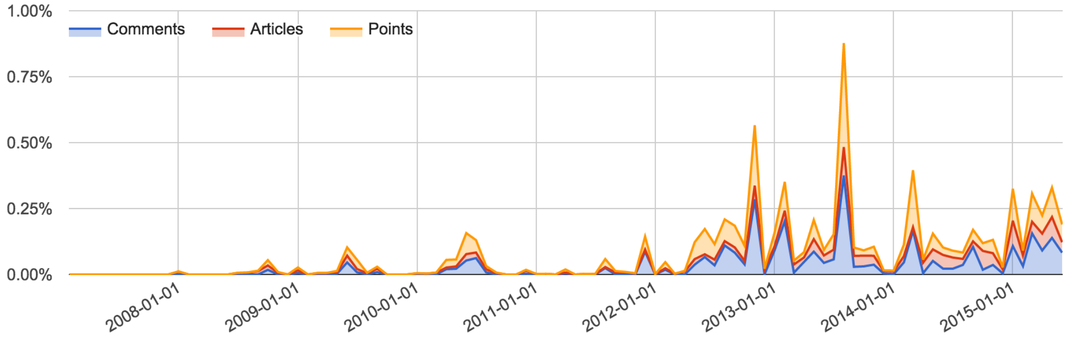
In this section, we will show some awesome plots of how the popularity of various topics changed over time.

Figure 5.2: Popularity of Docker, CoreOS, etcd, containers, OpenStack



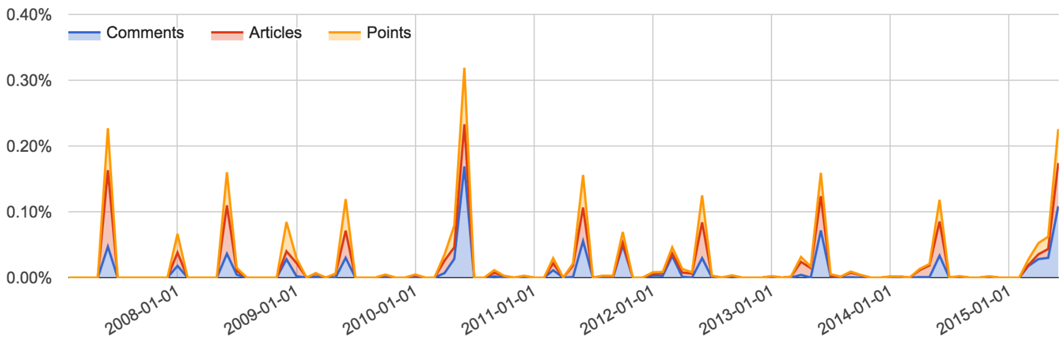
We start the overview with a trend plot (figure 5.2) of a new and upcoming technique: containerization. Docker and CoreOS were released in 2014 and before that, containerization was already used as a term for seperating Linux processes (which explains earlier peaks). The trends show how the release of Docker 1.0 (June 2014) created a lot of buzz, and the sustained popularity afterwards.

Figure 5.3: Popularity of Elon Musk, SpaceX, Tesla, Hyperloop



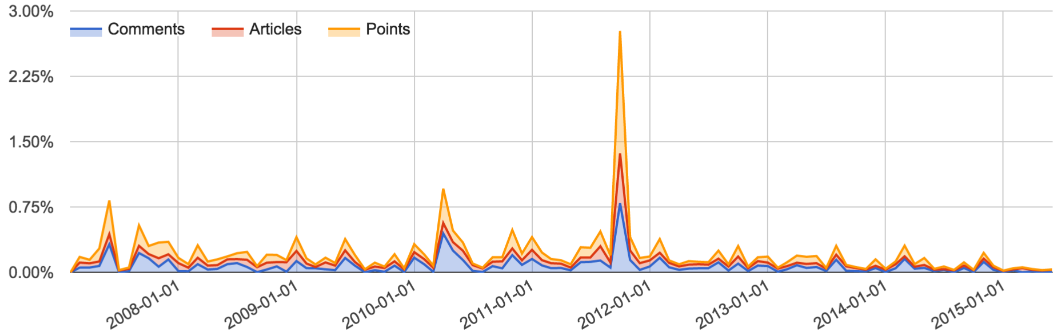
Elon Musk is described by Wired.com as a ”maverick entrepreneur" due to his big and risky projects. We see this reflected in the chart of his popularity (figure 5.3). Halfway 2010, Elon was out of cash (despite a 200 million dollar buyout from PayPal), which sparked some interest. The big spike late 2012 marks his expressed intent to have SpaceX (his spacecraft company) fly to Mars. The biggest spike in the chart, halfway 2013, marks the introduction of Hyperloop, a high-speed transportation system.

Figure 5.4: Popularity of keynote, Apple, @scale, developers conference



In the popularity charts of keynote-related topics (figure 5.4), one can see yearly patterns of spiking popularity. Each June has a spike of over 0.1%, which corresponds to Apple’s annual WWDC (Worldwide Developers Conference). The spikes smaller than 0.1% are just noise.

Figure 5.5: Popularity of Steve Jobs, Wozniak, imagineers



On the topic of Apple: not only it's products spark the interest of the Hacker News community; it's former CEO also had a big impact in the community. We show figure 5.5 to indicate the enormous impact the death of Steve Jobs had. He shares this topic group with his co-founder Steve Wozniak, but the news of his death (October 5th, 2011) dwarfs all other events in Hacker News history.

Figure 5.6: Popularity of Raspberry Pi, Kindleberry

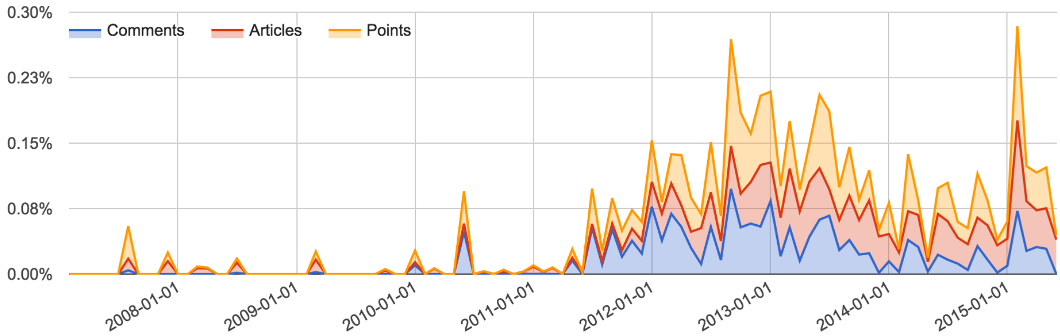
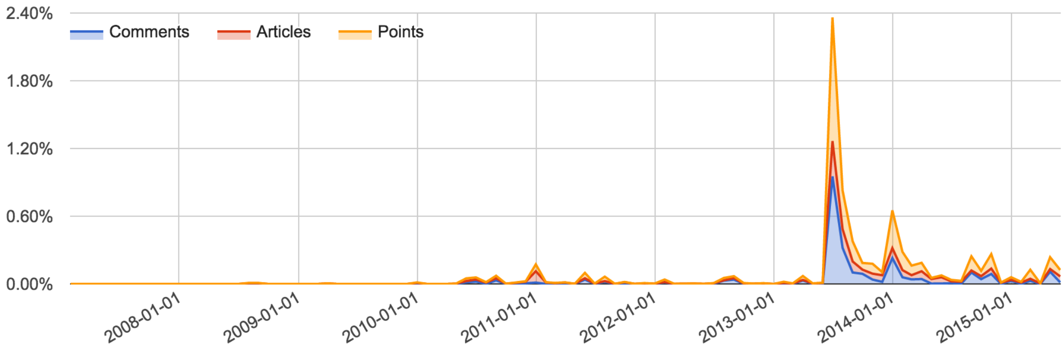


Figure 5.7: Popularity of Edward Snowden, Wistleblower, leaks, Reddit



Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

6 Conclusion

The goal of this research was to find out how the popularity of topics on Hacker News evolves over time. To answer this question, we defined a popularity measure based on the percentage of articles, upvotes and comments in a topic.
We then continued by dividing the dataset 1.5 million articles into topics using two unsupervised clustering algorithms: Word2Vec in combination with k-means and Latent Dirichlet allocation.
The resulting distribution of stories over these topics allowed us to determine the popularity of topics. Looking at the popularity plotted over time, we have seen three types of “trends":

- Single events, e.g. deaths and product releases (Steve Jobs, Hyperloop). These cause big spikes during the month of the event, but don’t have long-lasting effects.
- Recurring evens, e.g. annual conferences (WWDC). These clearly show up in the results, although they are generally not as extreme compared to single-time events.
- Long-lasting trends, e.g. concepts or technologies (Raspberry Pi, Docker/CoreOS). These topics attract attention over a longer span of time, but even within these time spans the months with big news (product releases) generally have a much higher (by a factor of two or three) popularity. Only really big topics have such a long-lasting trend.

References

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.