Frameworks: Final Report

Final Project Report for the 'Frameworks' Project for FDAC 2018

Jerry Duncan
UTK FDAC 2018 Student
The University of Tennessee
Knoxville, USA
jdunca51@vols.utk.edu

Trish Goedecke
UTK FDAC 2018 Student
University of Tennessee Health Science Center
Memphis, USA
tgoedecke@uthsc.edu

Paul Preston Provins IV
UTK FDAC 2018 Student
The University of Tennessee
Knoxville, USA
pprovins@vols.utk.edu

I. Problem

Many JavaScript frameworks quickly rise and fall in popularity as an endless stream of new ones emerge yearly. This makes the process of selecting a framework to use for new projects a hard problem for developers seeking stability and long term support.

II. Motivation

We believe that there exists a predictive ability of topic models at various levels of K (number of topics) that will help us solve this problem. We think that the correlations between topic prevalence and download rates will enable us to develop a popularity prediction model of JavaScript frameworks. As a long-range goal, we are also seeking a mathematical method of determining the most effective dimensionality, K, of an LDA topic model.

III. Approach

A. Data

Before we could start trying to use topic modeling to predict what framework would be the most popular and the most stable, we needed to determine what data we thought would be the most relevant to each framework and its success.

1) We determined that the data that is most relevant to a framework's future popularity is its download history from NPM (Node Package Manager) as downloads per day is essentially its current popularity. In order to collect that data, we leveraged NPM's download count API that allowed us to get NPM's up-to-date download data by day for each of the six frameworks. One caveat though is that what they count as a download is somewhat vague and sometimes their data for a particular day might be zero due to errors in their database. We determined there is not another source of data that speaks directly to a framework's popularity like its current download count so we started looking for data sources that would potentially speak more towards the current opinion of each framework.
2) The first data source we thought would apply was Stack Overflow questions and comments. While Stack Overflow questions and comments are supposed to be unbiased, due to them being another form of social media, often times they would be riddled with opinions skewed one way or the other (e.g. Don't use that framework, instead use xyz as it is a much better solution to your problem regardless of validity). Unfortunately their data is not easy to get so we had to download a 160 GB data dump, parse it ourselves, and then store it in a MongoDB database. After that, we ran a custom cleaning function that would remove all code snippets, links, stop words, and other extraneous data so that we could later feed it to our topic model.
3) The second data source we thought would apply was GitHub issues and comments. Like Stack Overflow questions and comments, both GitHub issues and comments are another form of social media that we thought would have some insight into the current opinion of their respective frameworks. We leveraged GitHub's api to gather every issue for each framework and then one by one we had to query each issue's comment url to get the list of comments for each issue. Once all of that data was collected, we put it through a similar cleaning function to the one used for the Stack Overflow data that would remove code snippets, links, stop words, emoticons, and other extraneous data and packaged each issue with its comments into a document for use in the topic model.

B. Modeling

Our primary modeling technique is topic modeling with latent Dirichlet allocation (LDA). Our proposal indicated that we would run topic models of the GitHub issues and comments data, as well as the Stack Overflow Q&A, with models of a variety of topic dimensions (K). We ran models of 10-100 topics (by tens), to compare the predictive effectiveness of the various dimensions of the models.

Using the topic frequencies per framework-month as features, we planned to predict growth in popularity of frameworks in a long short-term memory recurrent neural net time series model. The effectiveness of growth prediction within our given data set is the measure of the effectiveness of the topic model for a given dimensionality K.

C. *Website*

Originally, we had planned to put our prediction model online to be accessible by all and have it run on-the-fly. Unfortunately we did not get that far. We picked the most popular framework at the time of writing (React) and made a website that took all of the data we had collected (NPM downloads, Stack Overflow questions and comments, and GitHub issues and comments) and displayed it in an easy to understand format that could be manipulated so that anyone who visits it will be able to make their own prediction of what framework would be the most popular based off of what data they thought was the most relevant.

IV. Project Management

Jerry was the team lead and was responsible for planning the major deadlines for collection, cleaning, and modeling our data as well as keeping our GitHub repository organized. He collected the download counts of each of the six frameworks from the npm api. He also collected all of the issues and comments from each of the six frameworks' GitHub repositories. For both sets of data he cleaned the data and minified it for use in Trish's topic modeling and the website. Lastly, he created a website that displayed interactive graphs for people to use and make their own predictions.

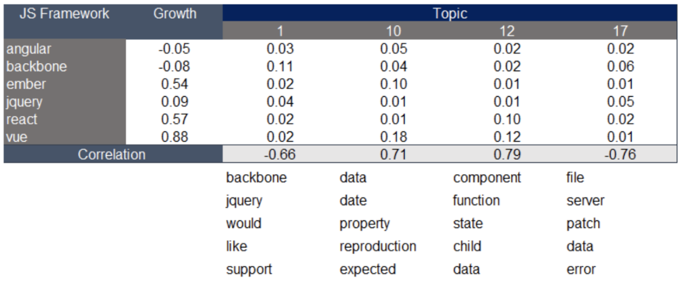
Preston played a large role in collecting and cleaning the Stack Overflow data. He set up a Google Cloud instance to download Stack Overflow's entire 160 GB data dump and then set up another Google Cloud instance to parse the data and put it into a queryable MongoDB database. Although there existed queryable instances of the data already (Big SQL, StackOverflow Meta), these instances did not provide the results adequate for the team's topic modelling as the SQL instances were limited in what could be returned. Then he cleaned it and properly formatted it for use with Jerry's website and Trish's topic modeling. He made sure that all project deadlines were met and that the papers were properly formatted, edited, and submitted on time.

Trish's primary role was to perform topic modelling on the data the Stack Overflow data that Preston collected and the GitHub and NPM data that Jerry collected. She made sure that we properly cleaned the data and didn't remove relevant data before she used it. She was also the primary writer for the proposal and final paper since she was the most familiar with the subject matter and relevant work in the field. Trish plans to continue this project in the coming semester.

V. Outcome and Results

Thus far, we ran a single 20-topic model on a sample of our GitHub issues and comments data. We used 300 issues with their respective comments from each Framework (i.e., N=1800); we did not separate these issues by month for time series analysis yet. We found several topics correlating either positively or negatively with 1-year proportional increase in framework downloads, measured from September 2017 to September 2018, as shown in Figure 1.

Because we were not able to complete all of our goals, we gathered some statistics from the data we collected and put them on the web. We decided to utilize the React framework to accomplish this goal. Because we were unable to provide a predictive model, we expect that users can extrapolate their own predictions of frameworks that interest them based off of the data that we have collected and presented to them. The groundwork laid by the website secures a strong footing for any future development and could serve as a hub for any new data that is collected in the future.



JS Framework	Growth	Topic			
		1	10	12	17
angular	-0.05	0.03	0.05	0.02	0.02
backbone	-0.08	0.11	0.04	0.02	0.06
ember	0.54	0.02	0.10	0.01	0.01
jquery	0.09	0.04	0.01	0.01	0.05
react	0.57	0.02	0.01	0.10	0.02
vue	0.88	0.02	0.18	0.12	0.01
Correlation		-0.66	0.71	0.79	-0.76
		backbone	data	component	file
		jquery	date	function	server
		would	property	state	patch
		like	reproduction	child	data
		support	expected	data	error

Figure 1. JS Frameworks: Growth Rates and Topics. Growth rates in downloads of JavaScript frameworks September 2017-September 2018. With 4 most highly correlated topics, from 20-topic LDA topic model, Top 5 terms of each topic shown.

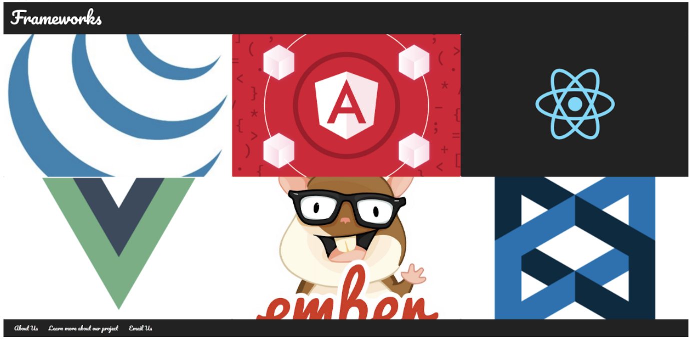


Figure 2. The website homepage with all six frameworks we analyzed

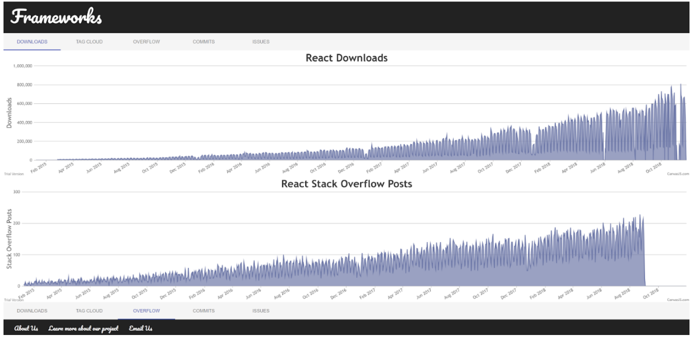


Figure 3. Example of a framework's page with all of the data we collected.

VI. Limitations

We were unable thus far to model our full GitHub issues and comments data (N = 40,333), due to the R software and the desktop OS X computer running the analysis both freezing, and have not added the Stack Overflow data to the model for the same reason. We also have yet to model a variety of topic dimensionalities, and to break the data into training and testing sets for predictive modeling. We have not yet applied long short-term memory recurrent neural net time series models using topic proportions as features to predict growth in framework downloads.

VII. Related Works

A. *Dimensionality of K*

Topic modeling using latent Dirichlet analysis (LDA) is a dimensionality reduction technique for large corpuses of text, clustering vocabulary terms into topics found to be distributed probabilistically among documents. Dimensionality, the number of topics (K) to model, is as yet qualitatively determined, though a number of methods have been suggested to quantitatively determine K. Methods proposed have included measures of perplexity and its difference, goodness of fit, and pointwise mutual information.

Thus far, dimensionality has often been determined in an ad hoc or post hoc manner. A common technique to find the most appropriate K is to fit an LDA model over a broad set of K values, comparing results at each level. We see this for example in Heinrich et al. (2005), comparing correlation levels at different scopes, where scope refers to levels of K as well as to numbers of documents, term types, and document types. Some studies that have addressed the issue are described below. As we will see, the definition of a best-fitting topic model remains to be determined.

1) Perplexity (P): Blei et al. (2003) suggested a method for measuring the fit of a topic model. They used perplexity, which had already been in common usage with language modeling methods such as latent semantic analysis (LSA). The meaning of perplexity is in its relationship to likelihood. As Blei et al. define it, perplexity is algebraically equivalent to the inverse of the geometric mean per-word likelihood. This means that the greater the likelihood of the model, on an aggregated per-word level, the lower the perplexity score will be. Lower perplexity scores thus indicate greater optimization of the topic model.

Perplexity is facilitated as a measure, in that it is an output of most LDA computational packages - an intrinsic measure of the model. A weakness of the perplexity optimization method, particularly for text data, is that it may not correlate well with human judgment regarding topics (Chang et al., 2009). Topic models selected by perplexity scores tend to rate lower in human coherence than other models. Perplexity score selections are also described as lacking in stability and efficiency (Zhao et al., 2015).

A challenge that remains, with perplexity and most other measures of optimization, is in interpreting the meaning of the measures. Perplexity falls quickly with increasing K but eventually starts increasing, perhaps due to overfitting. Smaller values of K have particular value in some cases. In text data, ten or twenty topics can generally be understood meaningfully by humans. At larger values, the intuitive meaning of topics becomes lost. Not all topic modeling is of text data, however, and intuitive interpretation is not always the goal. Change in Perplexity (dP)

Zhao et al. (2015) argue that the change in perplexity (dP) is a stable and efficient means for finding an appropriate number of topics, from a machine learning standpoint. Zhao et al. (2015) use the absolute value of change in perplexity. Their proposed method is to select the K with the first change in direction of the dP measure as the appropriate model dimensionality. Zhao et al. (2015) measure the stability and effectiveness of the rate of perplexity change using the following methods.

To evaluate stability, dP is compared with the perplexity-based method. In repeated sampling, change in perplexity provided more stable results than did perplexity. To evaluate efficiency, Zhao et al. (2015) used cluster analysis on model output. Again, the dP method out-performed the perplexity-based method for selecting the best number of topics to use.

2) Goodness Of Fit (GOF): Goodness Of Fit is a method which compares the assignment of topics in documents with a uniform distribution which would occur if they were randomly assigned. Bowman (2016) applies this method and describe its theoretical underpinnings. The goodness of fit method applies a hypothesis testing approach. The null hypothesis is that for any document in a corpus, 1/Kth of the document might be expected to consist of each of the K topics, by random uniform distribution.

The application of this method involves measuring what proportion of documents, for any level of K, significantly differ from the uniform distribution. A strength of this method is that it is a purely mathematical approach, requiring little subjectivity or interpretive skill. A weakness is that, as it tends to favor larger values of K, it may not lead to topics which have intuitive interpretability. As noted above, smaller numbers of topics modeled tend to lead toward greater interpretability by humans.

3) Pointwise Mutual Information (PMI): Pointwise Mutual Information (PMI) is a statistical measure of the discrepancy between observed coincidence of random variables X and Y given their joint probability, with the coincidence which would be expected assuming independence. Meaningfully, PMI is used to gauge the degree of topic coherence in a model. The dominant words within each topic, those with highest probability of occurring within the topic, are rated pairwise for PMI with one another. The scoring is standardized to adjust for the dimensionality of a model. Higher PMI scores are interpreted as an indication of greater topic coherence.

The Pointwise Mutual Information approach to optimizing K requires a sophisticated analytical tool extrinsic to the topic model. As applied by Niraula, Banjade, Stefanescu, and Rus (2013), this involved forming:

> ”... all possible pairs with the top 10 or 20 words in each topic... [The corpus] contained 1,284,156,826 tokens and 5,693,208 word types (i.e. unique words) counted after removing digits and punctuation and changing to lowercase. After re-

moving the stop words, the number of tokens was 672,542,579."

The major strength of the PMI approach for finding K is that for text documents, the resulting model correlates well with human judgment. A challenge of this approach is that it requires the development of an extensive PMI comparison database, for text or any chosen topic medium.

4) Topic Coherence: A troubling concern regarding topic modeling as a dimension reduction technique for data is that topics modeled may make little sense to human interpreters; or may appear flawed. A number of researchers have sought to evaluate or improve on the coherence of topics modeled.

Newman et al. (2010) tested automated techniques for evaluating topic coherence, including pointwise mutual information (PMI) and other lexical relatedness measures based on WordNet, Wikipedia and Google's search engine. They found several Wikipedia-based measures to show strong results, including one using PMI from Wikipedia source data. PMI measures the relatedness between any two words as the discrepancy between the relative frequency of their co-occurrence p(x,y) and the relative frequency predicted by their joint probabilities when independence is assumed p(x)p(y).

Mimno et al. (2011) addressed the concern that topics derived from the automated LDA method frequently baffle human interpreters. They identified several specific types of topic flaws, presented a method for evaluating topic coherence automatedly, and provided a new statistical topic model which they call a Generalized Pólya Urn model. In their discussion, Mimno et al. (2011) express the concerns that going forward, improving semantic cohesion will be of primary concern to text modeling researchers, along with scaling to ever larger data sets.

B. WORD2VEC

Bengio et al. (2001) introduced word embedding as a dimension reduction of vocabularies from one dimension per term to a vector representation per term, in generally several hundred dimensions. Mikolov et al. (2013a) introduce word2vec in two simultaneously presented methods: Continuous-Bag-of-Words (CBOW) and Skip-Gram. These word embeddings derive predictions and hence vector weightings from the words immediately surrounding them in consecutive text. Mikolov et al. set themselves the challenge of increasing the size of text corpora to billions of words which can be incorporated in a neural net model. They improve upon earlier neural net models with an approximational method to reduce computational complexity.

Mikolov et al. first compare their method against a Feed-forward Neural Net Language Model (NNLM; Bengio et al., 2003), explaining that the NNLM computational complexity Q could be defined as:

$$Q = N * D + N * D * H + H * V$$

where N is the number of preceding terms included in the model (typically 10), $N * D = P$ is the size of the projection layer, which might be $500 - 2000$, H is the size of the hidden layer, often $500 - 1000$, and V is the size of the vocabulary. This complexity is dominated by the $H * V$ term, because of the typically large vocabulary size. Mikolov et al. explain that they reduce computational complexity by using hierarchical softmax, and representing the vocabulary with a Huffman binary tree.

Mikolov et al. then make comparison with Recurrent Neural Net Language Models (RNNLM; Bengio & LeCun, 2007; Mikolov et al., 2010). They explain that the RNNLM has no projection layer, but rather the hidden layer connects with itself using a time delay. This results in a computational complexity for RNNLM of:

$$Q = H * H + H * V$$

Here, word representations (D) have the same dimensionality as hidden layer H, and $H * V$ can be reduced to $H * log_2(V)$ using hierarchical softmax.

Mikolov et al.'s (2013a) new architecture employs log-linear models and relies on earlier work, particularly Mikolov's 2007 master's thesis work. In this two-step method, continuous word vectors are first learned using a simple model; then an N-gram NNLM is trained on top of these. For the CBOW model, the current word is predicted from a window of N preceding and subsequent words; this research team found their best results with $N = 4$. Computational complexity Q for CBOW is found as:

$$Q = N * D + D * log_2(V)$$

For the Skip-Gram model, the predictive order is reversed; the current word is used to predict words within its window. In both cases, it is not the prediction that is the desired output; rather, the weightings used to obtain predictions are the output of interest.

VIII. Future Work

Based on our original proposal, there were three major goals that we had intended to complete but were unable to due to unforeseen complications and a lack of time to dedicate to the project. In order from least complicated to most: The first goal was to create a prediction model that would use a coupling of the NPM download data with our topic model try to find correlations between the current opinion about a framework (via Stack Overflow questions and comments and GitHub issues and comments) and how many people are downloading that framework. The second goal was to simplify that model so that it could be run in a short period of time in someone's browser or on a server so that when people visited our website they would be able to see a live prediction based off of restraints they applied to our up-to-date data set.

The last goal was to objectively evaluate the effectiveness of topic models at varying levels of K. We plan to continue our proposal as planned during the spring semester, as a project for COSC 594 Evidence Engineering. To overcome the R limitations, we plan to use python for topic modeling. This will include dividing data into training and testing sets so that predictive effectiveness may be measured. It will also include using the topic model output of topic proportions as features in LSTM neural net time series models, with growth values measured per month rather than per year.

Acknowledgment

Yuxing Ma of the University of Tennessee also contributed to the idea of this project and had a limited role.

References

[1] Blei, D.M., Ng, A.Y. & Jordan, M.I., 2003. Latent dirichlet allocation. Journal of Machine Learning Research, 3 (Jan), 993-1022.

[2] Bowman D. (2016). Selecting the number of topics in a latent Dirichlet allocation model. Statistical Learning and Data Science Section, Alexandria, VA, 1849-1857. American Statistical Association.

[3] Heinrich, G., Kindermann, J., Lauth, C., Paa, G., & Sanchez-Monzon, J. (2005). Investigating word correlation at different scopes–a latent concept approach. In Workshop Lexical Ontology Learning at Int. Conf. Mach. Learning.

[4] Mimno, D., Wallach, H. M., Talley, E., Leenders, M., & McCallum, A. (2011, July). Optimizing semantic coherence in topic models. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (pp. 262-272). Association for Computational Linguistics.

[5] Newman, D., Lau, J. H., Grieser, K., & Baldwin, T. (2010). Automatic evaluation of topic coherence. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, 100-108. Association for Computational Linguistics.

[6] Niraula, N., Banjade, R., tefnescu, D., & Rus, V. (2013). Experiments with semantic similarity measures based on lda and lsa. In Statistical Language and Speech Processing (pp. 188-199). Springer Berlin Heidelberg.

[7] Stevens, K., Kegelmeyer, P., Andrzejewski, D., & Buttler, D. (2012). Exploring topic coherence over many models and many topics. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, 952-961. Association for Computational Linguistics.

[8] Wallach, H. M., Murray, I., Salakhutdinov, R., & Mimno, D. (2009). Evaluation methods for topic models. In Proceedings of the 26th Annual International Conference on Machine Learning, 1105-1112. ACM.

[9] Zhao, W., Chen, J. J., Perkins, R., Liu, Z., Ge, W., Ding, Y., & Zou, W. (2015). A heuristic approach to determine an appropriate number of topics in topic modeling. BMC Bioinformatics, 16(13), S8.