Programming Language Popularity in Relation to Its Presence on Social Media

Justin M. Peterson, Mohanapriya Ramachandran, Ateeq Rahman

ABSTRACT

Data mining and analysis has opened many doors for researchers and product developers to improve their output and drive results based on a multitude of data sources. "Big Data" has become a coined phrase that describes many concepts, but at the root of each of them is the idea that "Big Data" must be able to produce a relevant result through analytics. Many people only analyze the phrase and think in terms of scale, while the size of the data set may not be the most important factor to consider.

As a case study, take Stack Exchange [3] for example. This service is a Q&A forum commonly associated with software development and computing topics that provides a publicly available data set containing questions, their associated answers, and a number of up votes signifying how relevant a specific answer is. This service can cover any development topic under the sun, but analyzing every question and response will not lead to any conclusive or meaningful results. Instead, pieces of this data set combined with many analysis techniques and other publicly available archived data can validate hypothesis or predict trends in other data sets.

This piece of research aims to determine the most popular set of programming languages through analysis of publicly available social discussions involving these languages. There are several criterion which can define the popularity of a language through techniques such as sentiment analysis of comments, or calculating the frequency of discussion on web forums regarding specific languages.

1. OVERVIEW

There are often emotionally fueled debates between members of enterprise organizations on which language to pick for a new project. While the sentiments of other developers may not be an all inclusive way to pick which language would be best for a given application, going into the decision process with an overview of popularity of a language is a nice starting point to the discussion. The success or failure of a project may not completely depend on the selection of a programming language, but the more information a developer can know up front about their project and the language they decide to use can make all the difference.

There are many challenges that must be tackled to allow the analysis of these sources to be deemed as relevant. The data application developed would need to utilize tools like sentiment analysis and determine a way to include outliers in the popularity rating of a language. For example, regardless of all of the positive appeal given to a language there will always be some developers that provide strong negative reviews because of a specific component of the language. There must be a way to define the specific subtopics of a discussion that are identified as being either positive or negative. There also may be an associated credibility with each source of data. For instance, StackExchange can provide a rating of popularity for a specific user profile that posts a question or answer, but how can the credibility of tweets or Facebook posts be determined other than by sheer volume? Our team will spend a fair amount of time developing and testing strategies to analyze these issues.

A few example data sets that our team has decided to use for this analysis include a crawl of trending twitter topics, tweets, and users from the WWW conference [2], the collection of publicly available StackExchange data, projects on SourceForge, and data scrapes from Facebook [1]. Our team could also look into more broadly ranging data sets such as google search trends, but this research has not currently been done. These chunks of information come from independent sources and will be able to flex our teamâĂŹs knowledge in distributed processing and analysis of large data sets. An example of this process would be to create a dictionary of programming language names and associated strings and parse the massive collection of tweets available to find text strings and associated metadata. These relevant tweets would be stored in our DBMS layer along with information collected from other sources. Sentiment analysis would then be run against these strings and relevant words associated with the defined dictionary to determine an overall positive or negative sentiment for the statement. Statistical analysis that we learn in the course could then be run across this set of result data.

In accordance to the project guidelines, our group intends to divide the project up into a data mining and storage section as well as a data processing and analytics section. The mining and storage section will be done by parsing the existing data sets in their current format and storing them in an aggregate SQL or Key/Value data store. We have not yet determined which store we will use but we will make that

clear in the near future for validation. The analytics section will be done by constructing models in rattle and R to first determine an appropriate measurement of probability, and then an ordering of language popularity based on application context.

2. DESIGN CONSIDERATIONS

TBD

3. ARCHITECTURE

TBD

4. IMPLEMENTATION

TBD

5. LESSONS LEARNED

TBD

6. CURRENT STATUS & FUTURE WORK

TBD

6.1 Tables, Figures, and Citations/References

TBD

7. REFERENCES

[1] 2005 facebook crawl. `https://archive.org/details/oxford-2005-facebook-matrix`.

[2] H. Kwak, C. Lee, H. Park, and S. Moon. What is Twitter, a social network or a news media? In *WWW '10: Proceedings of the 19th international conference on World wide web*, pages 591–600, New York, NY, USA, 2010. ACM.

[3] D. Posnett, E. Warburg, P. Devanbu, and V. Filkov. Mining stack exchange: Expertise is evident from initial contributions. In *Social Informatics (SocialInformatics), 2012 International Conference on*, pages 199–204, Dec 2012.