Meme Tracking in Scale with MapReduce

Project Proposal

Hohyon Ryu
School of Information
University of Texas at Austin
hohyon@utexas.edu

Jae Hyeon Bae
Computer Science
University of Texas at Austin
metacret@gmail.com

Nicholas Woodward
School of Information
University of Texas at Austin
woodward.nicholas@gmail.com

ABSTRACT

This article is to present our project proposal for Fall 2011 Data-Intensive Computing for Text Analysis class. Using MapReduce-based text processing for keyword extraction, possible applications are discussed. First, using implicit links, sparse links between blog articles that hampers applying information retrieval techniques can be overcome. Second, with Latent Dirichlet Allocation, we may generate a topic model based on extracted keywords. Finally, with the extracted keywords we can visualize blogs and keywords that may help perceive how the blogosphere appears in the big picture.

Categories and Subject Descriptors

H.4.m [**Information Systems Applications**]: Miscellaneous; H.5.4 [**Information Interfaces and Presentation**]: Hypertext/Hypermedia

General Terms

MapReduce, Blogosphere

Keywords

Keyword Extraction, Data-intensive Processing, Implicit links in Blogosphere, Meme tracking

1. INTRODUCTION

Web logs, also known as blogs, are a very important and interesting source of diverse information. They provide insight into the culture of the society in which they exist. People search blogs to find out how to cook, what to do, what to see, and to discover what others think. Because blogs are typically written in plain language by individuals enthusiastic for their interest, they may be more easily understood than longer, more-detailed news articles and cover a wider personal range of topics. However, unlike web information retrieval, blog information retrieval is still very limited and faces many challenges. Some well-established Web information retrieval techniques do not perform well on blogs because of sparse links, multimedia contents and the short length of the contents [?]. In particular, PageRank [?], which made a revolutionary improvement by taking into account the link structure of HTML Web documents, does not perform well for blog posts as they lack strong link structure. Another problem for blog information retrieval is the length of the blog posts. They vary from one line to several pages and require smoothing or normalization to achieve effective information retrieval. In this study, a novel method for augmenting implicit citation links to the explicit HTML links and document expansion for blog posts will be introduced. Extraction of implicit citation links from blog posts will address the sparse link problem that hampers utilizing PageRank and other link-based retrieval algorithms for blog information retrieval. The methodology of extracting implicit citation will utilize Blog content extraction using Decruft[1] and noun chunk extraction based on Python NLTK[2]. By augmenting each document's implicit links, the proposed method tries to improve blog information retrieval. We will also explore other applications that could potentially improve information presentation for the Blogosphere.

2. MEME EXTRACTION AND ANALYSIS

2.1 Test Collection

BLOGS08[3] is a TREC[4] test collection that crawled Web Logs from Jan/14/2008 to Feb/10/2009. It consists of 3 components: feeds, permalink documents, and homepage documents. We use only permalink documents that contain 28,488,766 documents and whose uncompressed size is 1445GB. Because the collection contains many duplicates, non-English blog posts, advertisement, and Blog website frameworks, we will use NLTK and Decruft to extract English blog contents without boilerplate text.

2.2 Preprocessing Blogs08

Preprocessing Blogs08 Collection is done using Python with NLTK and Decruft packages. Decruft extracts only meaningful content from a blog page filtering out navigation links, advertisements, sidebar contents, and link to other contents and sites. Using a language identification module,

[1]http://www.minvolai.com/blog/decruft-arc90s-readability-in-python/
[2]Natural Language Toolkit, http://www.nltk.org/
[3]http://ir.dcs.gla.ac.uk/test_collections/blogs08info.html
[4]http://trec.nist.gov/

NLTK checks if the extracted content is written in English. Blogs08 also contains a tremendous amount of duplicate documents and documents without significant amount of text. We set a lower-bound of 5 words and remove duplicates, also using Python. After all these preprocess are completed, every extracted content of a blog post is written into a single line per document for the MapReduce meme extraction process.

2.3 Meme Extraction

Preprocessed Blogs08 data is comprised of key-value pair where the key is document number(docno) and the value is the preprocessed HTML content of the document. Using Hadoop Streaming, Python MapReduce code extracts candidate phrases and group documents by the common phrases. The mapper first processes input HTML content using BeautifulSoap[5] and strips all HTML tags and JavaScript code. Using NLTK, the mapper tags all the words' part of speech using HUNPOS-tagger[6] and extracts noun phrases, verb phrases, prepositional phrases, and combinations of all these types. The mapper outputs the text key-value pairs where key is a phrase, and the value is docno. The reducer collects all the docnos that share the same phrase and output the phrase and docnos pair. If the number of documents is less than 5, the reducer ignores the phrase.

2.4 Meme Clustering and Topic Analysis

To track mutation and propagation of similar memes, similar memes must be grouped together. For the grouping, we use MapReduce-based clustering. Clustering is the most basic technique to retrieve meaningful information from the corpus. In this project, we will use K-means clustering as the starting point. Because we do not have any prior information of meme distribution, we will apply Canopy clustering [?] to choose initial seeds of clustering instead of random seeds selection of K-means clustering. There are two threshold parameters $T_1, T_2, T_1 > T_2$ at Canopy clustering. Because this algorithm is pretty fast, we can experiment with various combinations of threshold parameters to get reasonable number of clusters. After K-means clustering with centroids generated by Canopy algorithm, we can start analyzing meme distribution in each cluster and find which memes are releated to each other. Cosine distance will be used as the distance measure between two memes. Suppose that document P, Q are represented as two vectors $(P_0, P_1, ..., P_n)$ and $(Q_0, Q_1, ..., Q_n)$, cosine distance between two documents can be expressed mathematically as the following:

$$D(P,Q) = 1 - \frac{\sum(P_i \times Q_i)}{\sqrt{\sum(P_i)^2 \times \sum(Q_i)^2}}$$

For Canopy generation and K-means clustering, Apache Mahout[7] implementation will be used. Latent Dirichlet Allocation (LDA)[?] is a generative probabilistic model that can be utilized to detect topics in a document collection. LDA models the document as the random mixture over latent topics, and topic can be represented by a distribution over words. Using LDA, we can create meme mixtures for each topic and topic assignments for memes in each document. We can then group memes with similar topics and analyze several aspects of the data, including topic distrubition over documents and relationshp among topic space. Yahoo! LDA[8] implementation will be used. LDA algorithm requires several parameters, of which the most critical is the number of topics that should be specified. This number can be the same as K set with the aforementioned K-means clustering. In order to find the optimal the number of topics we will calculate the perplexity of test collection with the various number of topics. Formally, for a test set of M documents, the perplexity is

$$perplexity(D_{test}) = exp\{-\frac{\sum_{d=1}^{M} log\ p(\mathbf{w}_d)}{\sum_{d=1}^{M} N_d}\}$$

3. APPLICATION FOR EXTRACTED MEMES

3.1 Meme Tracking

Previous research has focused on the interaction between blogs and Internet news sites [?], tracking memes as they pass from one class of websites to the other and potentially back again. Our proposal is to focus on blog to blog interactions, using the metdata provided in Blogs08 to associate blogs with certain memes over time. We will be able to track memes from their approximate origin as they disseminate out to other blogs, and from this data we can cluster similar blogs into groups based upon their common usage of certain memes. Potential similarity measures include the quantity and frequency of memes that travel between blogs and the lag or time it takes for a meme on one blog to appear on another. We will also examine the blog clusters themselves to observe how their composition changes over time as blogs constantly reference different memes.

3.2 Information Retrieval Performance Improvement

The idea of using link structure for information retrieval has been recognized as the key to successful retrieval algorithms since the emergence of Google and its core algorithm, PageRank [?]. Another effective approach to utilizing links in information retrieval called HITS was proposed around the same time [?]. PageRank and HITS are now the minimum standard of the web information retrieval to assure the quality of retrieved web pages. Unfortunately, these powerful algorithms cannot be fully exploited for blog information retrieval due to lack of strong link structure in the blogosphere.

With extracted keyphrases using MapReduce and NLTK, we will provide more prior information for a MapReduce-based search engine Ivory[9] to improve blog search performance.

3.3 Visualization for Users

There have been many attempts to visualize the blogosphere [?, ?]. However, they were insufficient in helping users with better navigation. Using the visualization techniques integrated into the navigation system, we would be able to benefit actual users by improving the process of finding blog articles, thus allowthing them to more easily navigate between related articles.

[5]http://www.crummy.com/software/BeautifulSoup/
[6]http://code.google.com/p/hunpos/
[7]http://mahout.apache.org

[8]https://github.com/shravanmn/Yahoo_LDA
[9]http://www.umiacs.umd.edu/ jimmylin/ivory/docs/index.html

4. PROJECT PLAN

4.1 Division of Work

- Hohyon Ryu: Meme Extraction, Meme Tracking
- Jae Hyeon Bae: Clustering Memes, Scalability Issues
- Nicholas Woodward: Theoretical Background, Meme Tracking, Practical Application

4.2 Timeline

- Oct/6: Preprocessing
- Oct/13: Meme Extraction
- Oct/20 & 27: Meme Clustering
- Nov/3 & 10: Meme Tracking and Application
- Nov/10: Presentation Webpage Buildup
- Nov/17: Finalizing Paper
- Dec/1: Presentation