SAND2014-1415C

Using Data Compression to Detect Inflection Points

Travis Bauer
Sandia National Labs
tlbauer@sandia.gov

Thomas Brounstein
Sandia National Labs
trbroun@sandia.gov

ABSTRACT

The Web contains a wealth of information that captures the evolution of groups of individuals. Mining this information can help us better understand human dynamics and build better data mining algorithms. Due to the complexity and richness of this data, it is difficult to find general analysis techniques for understanding the underlying dynamics. This paper discusses how data compression algorithms can be brought to bear on this problem and introduces a new measure called Complexity Difference (CD). CD can be used to detect inflection points in complex, human generated time series data without requiring costly feature extraction. It is rooted in Kolmogorov Complexity and related to Normalized Compression Distance. This paper will review the applicability of compression techniques to data mining. It will then describe Complexity Difference and show its application to several data sets. The first is a series of edit histories from individual Wikipedia pages. The second is the 21M PubMed titles provided for the WebSci14 data challenge. In both cases, we show that data compression without feature extraction can be applied to find points in time where the group dynamics change.

Categories and Subject Descriptors

I.2.7 [**Artificial Intelligence**]: Natural Language Processing – *language parsing and understanding, text analysis.* H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing – *linguistic processing*

General Terms

Algorithms, Measurement, Human Factors, Languages, Theory

Keywords

Compression, Text Analysis, Event Detection

1. INTRODUCTION

When looking at a time series of human generated data, a natural question to ask is whether there are key points in time when there are shifts. This kind of inference is different and more complex than questions about the content of the data itself. For example, questions such as "when was the first time that some specific topic was mentioned," are more easily answered because they are about the content of the data itself. But sometimes we'd like to ask if there are unusual periods of time, and use this as a basis for further analysis. We may not know *a priori* what features would be useful in making that determination.

In this paper, we will present a new algorithm called Complexity Difference (CD) that can be applied to complex time series data to address this problem. The algorithm is related to Kolmogorov Complexity and Normalized Compression Distance (NCD). NCD is a method for comparing two items using data compression [6]. The advantage of such techniques is that they work without requiring individual feature extraction.

We will start with a brief review of Kolmogorov Complexity and Normalized Compression Distance. Then we'll define Complexity Difference and apply it to several data sets.

2. BACKGROUND

2.1 Inferring Information about Groups Using Web Data

Information on the Web is often used as a means of acquiring a deeper understanding of the people and organizations that create and consume web content. For example, a current active area of research is inferring information about populations of people from social media. Additionally, the Wikipedia edit history has been studied as a means of thinking about group dynamics [14].

Much of this analysis relies on feature extraction and prior knowledge of critical elements. While these methods can be very powerful, they are also specific—when the genre or type of question being asked changes, the technique breaks down. By avoiding user specified features, complexity analysis is a more robust operation. For any individual problem, complexity analysis may be weaker than the optimal feature based algorithm, but across the range of all possible problems, complexity will give useful results without manual tweaking and manipulation.

Additionally, traditional methods, such as machine-learning, that require these specified features make it difficult to ask general questions, such as "find an important point in time" when that point in time may come from one of many different possible feature sets. However it is possible to get access to this kind of information through the use of complexity analysis, as discussed in the next section.

2.2 Compression

2.2.1 Kolmogorov Complexity

Kolmogorov Complexity is an idealized way of evaluating the quantity of information in an individual piece of data. Intuitively, the Kolmogorov Complexity of a piece of data is the shortest program that could be used to reproduce that piece of data [5]. For example, consider the string composed of the letter *a* repeated ten billion times. Printed out, it would take up a considerable amount of space. Intuitively, such a string is actually very simple and its shear length is not an indication of the complexity of the information it contains. The same string can also be represented simply with the phrase "*a* repeated 10 billion times." With those directions, one could perfectly reproduce the string. Compare this with a truly random sequence of ten billion characters. To reproduce such a specific sequence, you would need the full sequence itself. There wouldn't be a shorter set of directions that could be used to produce the sequence. So even though the original strings are the same length (ten billion characters), the sequence of *a*'s is simpler than the random sequence. This difference is captured with the notion of Kolmogorov Complexity.

Another way to think about Kolmogorov Complexity is that it identifies and removes redundancy in the data, leaving only the parts of a data item that can't be anticipated.

2.2.2 *Compression as an Approximation of Kolmogorov Complexity*

Compression algorithms are one way to approximate Kolmogorov Complexity because they identify and isolate redundancy. Li et. al. have a detailed treatment of the relationship between Kolmogorov Complexity and data compression [6]. They also include a description of the properties that a compression algorithm must have in order to be reliably useful as an approximation. In our experiments, LZMA served as a better compressor than BZip2. Figure 1 shows the behavior of BZip2 and LZMA compression over massively repetitive data. We took a single, short Wikipedia article and produced a series of documents of varying length composed of the same article repeated multiple times. The x-axis is the length of the document. To be an approximation of Kolmogorov Complexity, we'd expect that the compressed size would remain relatively constant since the shortest string needed to represent the document would be the compressed size of the original document along with the number of times it was repeated. The figure shows that LZMA has this property and BZip2 doesn't.

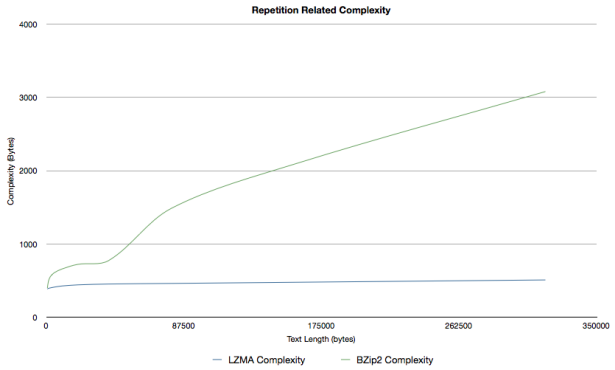

Figure 1. Compressed page length of a repeated Wikipedia page

Figure 2 shows another view of the same information over the same set of documents. This time the y axis is a log plot of the compression ratio and the x axis is size. This graph shows that as the data size gets larger, massively redundant text files are about 10 times larger using BZip2 than using LZMA. This is consistent with the size of the Wikipedia download files. The history files are compressed versions of every revision of a page where even a small change to the page would result in the full version being stored in the XML and then compressed, thus creating a massively redundant data set. Wikipedia hosts both BZip2 and LZMA versions of the files; the LZMA files are approximately one tenth the size of the corresponding BZip2 file.

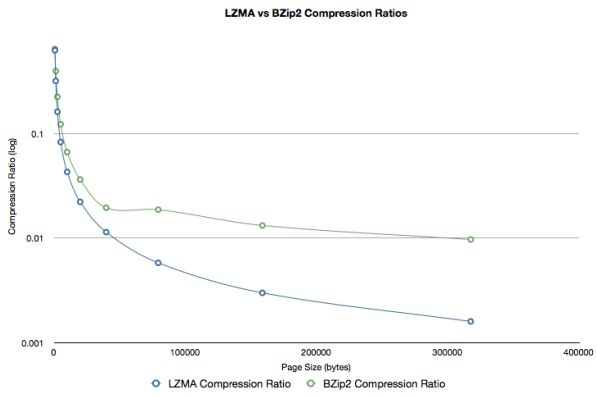

Figure 2. Log Scale view of compression ratio on repetitive Wikipedia document

All this suggests that in order to get good results, one must not choose an algorithm arbitrarily. The LZMA implementation we use for the analysis in this paper works well.

2.2.3 *Normalized Compression Distance*

Normalized Information Distance [6] is an idealized similarity metric for comparing the shared information between two items. The metric is rooted in Kolmogorov Complexity. If two items are exactly the same, then the shortest string that could generate both of them together would be the string that would generate one of them, plus a small addition to indicate that the item should be repeated.

More formally, consider two data items, *x* and *y*, both of which can be expressed in terms of a sequence of bytes. Let some function *C* take an item and return the number of bytes in a compressed version of the item. Figure 3 illustrates how the compression function can be used to compute normalized compression distance between two items. The circle on the left illustrates the number of bytes in the compressed version of *x*, computed by *C(x)*. The circle on the right illustrates the number of bytes in the compressed version of *y*, computed by *C(y)*. The entire area contained by the two circles represents the number of bytes in the concatenation of *x* and *y*, *C(x,y)*. The overlap of the two circles, represented by *v*, is what enables a comparison of two items. The area *v* is the overlap in content between the two; the size of *v* represents the content in *x* that is also in *y*.

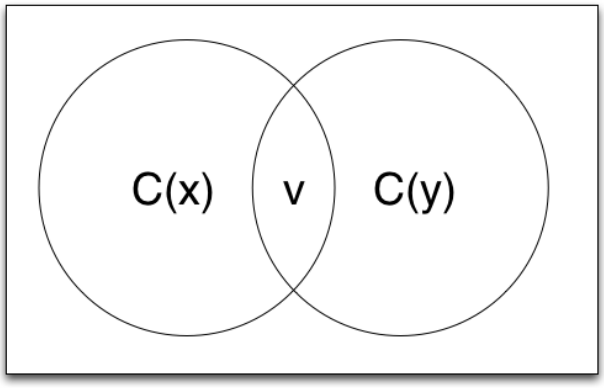

Figure 3. Normalized Compression Distance

The size of *v* is proportional to the amount of shared content. Two extreme situations are illustrated in Figure 4. If *x* and *y* are

nearly identical (the left side of the illustration), *v* is very large, almost the same size as the compressed version as either of the items individually. If *x* and *y* share no information, then *v* is empty and *C(x,y)* is simply the sum of *C(x)* and *C(y)*.

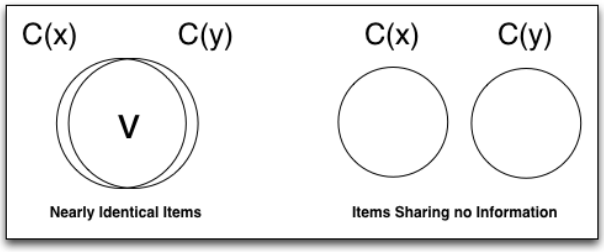

Figure 4. Two extreme compression scenarios

Normalized Compression Distance (NCD) is the ratio of the largest amount of unshared information in either *C(x)* or *C(y)* to the larger of *C(x)* and *C(y)*, as shown in Equation 1. Chen et. al. explain how this operation defines a metric that meets specific, provable criteria.

$$NCD(x,y) = \frac{C(x,y) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}}$$

Equation 1. Normalized Compression Distance

The accuracy of these computations relies on how well the compressor approximates the Kolmogorov Complexity of the items. An analysis of the features needed by a good compressor appears in the literature [1,6].

2.2.4 Applications of Normalized Compression Distance

Normalized Compression Distance has been applied to DNA sequence analysis and the relationships among different languages [6]. It has also been applied to clustering text and time series information as well as finding anomalies in time series [4]. It has even been used for clustering music [2,3] and Internet traffic [12].

However, the technique has not found broad application to Web data. One possible reason for this is that at its core, NCD is about comparing two items to each other. For text documents, it is straight forward to create a vector space and compare the items within that vector space, as is usually done, and which generates more concrete results than the NCD approach, with more user control.

3. Complexity Difference

This section describes a new algorithm called Complexity Difference. Complexity Difference starts with the assumption that the redundancy (compressibility) in a piece of data comes from multiple sources. Looking at the difference in compression ratios where some sources are held constant lets one measure changes in other sources. This is what will let us discover changes in group dynamics over time.

More formally, let us treat a piece of information as being generated by a combination of different sources, $S_1, S_2, S_3, \ldots$ Some other function, *G*, takes these sources and combines them, producing a specific piece of information. So each specific piece of information is a function of the sources, passed through *G*.

$$G(S_1, S_2, S_3, \ldots)$$

The complexity of the final product is a combination of the complexity of the individual sources. Consider the string composed entirely of the letter 'a.' It can be thought of as a function of a single source which produces 'a.' If we call that source 'A', the function would be as follows.

$$G(A)$$

The complexity of the product is just the complexity of that single source, which is small. In other words, it would have a very large compression ratio.

Now consider a sequence of truly random characters. Any sequence of random characters can be thought of as the product of some source that produces random characters. If we call that source R, the function would be as follows.

$$G(R)$$

The compression ratio of such a product would be quite large In fact, it would be approximately one because no compression would be possible. Thus, it is possible to compare the amount of information in two pieces of data simply by knowing the compression ratios. This is invariant with respect to the lengths of the strings. Two truly random strings, one of a billion and one of 100 characters, would both have the same compression ratio (of one) because the same information source was producing them.

Now let us consider a slightly more complex situation, one that mixed the sources A and R. as follows:

$$G(A,R)$$

The strings are generated as a function of the two constituent generators. One is a truly random character generator and the other generates sequences of the letter 'a.' A sample string might be as follows:

dijfjeaaaaffihenvifaaaaaakfbibeekaaaaaaaaaa

The compression ratio of this string would be somewhere between the compression ratio of A and that of R. It would be less than 1, but greater than if the whole sequence was determined. The measured complexity of the product would be a function of the two generators along with the function that combines them.

Natural language is a combination of several different "information sources." The English language, for example, has a certain complexity due to its grammar. For a given topic domain, the specialized vocabulary also provides a certain amount of complexity. Complex topics with a large vocabulary will not compress as well as topics with a smaller vocabulary. Finally, a certain amount of complexity is due to the ideas themselves. Although vocabulary selection and idea complexity certainly are inter-related, it's also the case that the ideas are expressed in the arrangement of terms and not just the vocabulary choice. The same distribution of terms could be used to express a wide variety of different ideas, some of them more complex than others. So we might represent any given natural language text as a function of three generators, language structure (S), vocabulary choice (V), and idea complexity (I). Thus, a piece of natural language is generated by the function).

$$G(S,V,I)$$

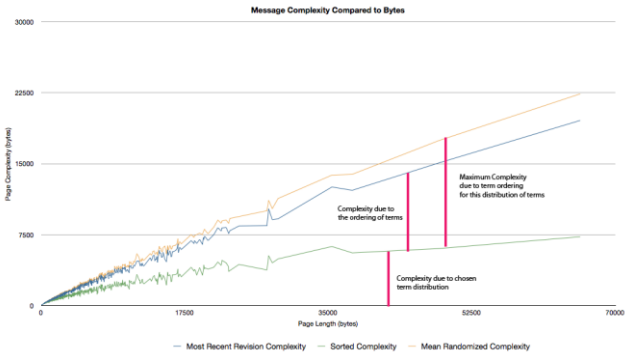

Figure 5. Compression ratios of Wikipedia pages

This is measurable in English language text. Consider the compression ratio of Wikipedia pages as shown in Figure 5. This graphs 1,110 Wikipedia pages of various sizes. The x-axis represents uncompressed article length and the three lines represent the size of different compressed versions of the page. The bottom line is the size of the compressed version of the document with every term instance put into alphabetical order. By putting all the terms in order, we've eliminated parts of the complexity due to language. The top line is the average compression size of three documents where all the terms from the document at that x-value were arranged randomly, which represents the situation where every term position was critical (unpredictable). The middle line is the compressed size of the actual document. The bottom line thus represents the information conveyed simply by the choice of terms. The top line represents the maximum potential information in the document if the arrangement were purely random. The actual compression size is somewhere between the two.

Assuming that all the documents are in a natural language, changes in the compression ratios should reflect changes in the complexity of the content, namely the vocabulary selection and the complexity of the underlying ideas. Our goal was to detect this change in complexity.

4. FINDING INFLECTION POINTS IN THE HISTORY OF A WIKIPEDIA PAGE

4.1 Data

The Wikipedia edit history provides an ideal place to test the concept that we can control for some forms of complexity and detect others. We looked at four pages, three of which have been pointed out as being targets for edit wars [8] and one which we picked simply because it had a significant number of page revisions.

Wikipedia makes its entire revision history available for download, meaning users can see how a page changes over time. This history, along with the data itself, has made Wikipedia the subject of much data mining research [8,13] and for understanding group behavior. Many pages have evolved over time with thousands of revisions. The Elvis page is a good example of this. There are 16,501 revisions, including revisions made by registered and anonymous users. As information gets added on a topic, the page can grow or shrink. If a page becomes too large or complex, the authors of a page may decide to split the page in to multiple pages, one for each subtopic. That way, each individual page covers fewer topics. This happened with the Elvis page when the users decided to split the article, creating new pages, each devoted to a different topic.

And yet, the core topic for any page and the language used stays constant. This suggests that changes in compression ratios may reveal critical changes to the page.

4.2 Approach

Assuming that the language stays constant, if the underlying vocabulary and the complexity of the ideas grow or shrink, we'd expect that change to be reflected in the compression ratios. So for each set of documents (each revision in the case of Wikipedia page edits), we compute a function over all of the page lengths of each revision to the number of compressed bytes for that revision. From that, we compute what we would expect the compressed size to be for each page. Finally, we compute the difference between the expected and actual number of bytes, which we then plot. What we find empirically is that this value varies with respect to inflection points in the group over time.

4.3 RESULTS

4.4 Power Law Distribution of Compression Lines

The Power Law description of the relationship between original and compressed bytes takes the form in the following equation.

$$y = ax^b$$

In our data a power law distribution was the best fit, especially for the smaller page sizes. However, a linear regression also fit well and provided almost the same results as the ones described in this paper. The table below shows the parameters to this equation for each article.

Table 1. Parameters to the exponential equations

Page	*a*	*b*
Elvis	**1.0653**	**0.8981**
Anarchism	**0.9234**	**0.9033**
Global Warming	**1.6266**	**0.8500**
Pumpkin	**1.6963**	**0.8473**

4.5 Elvis

Let's consider the page on Elvis. In our data set than runs through 2011, the Elvis page has 16,501 revisions.

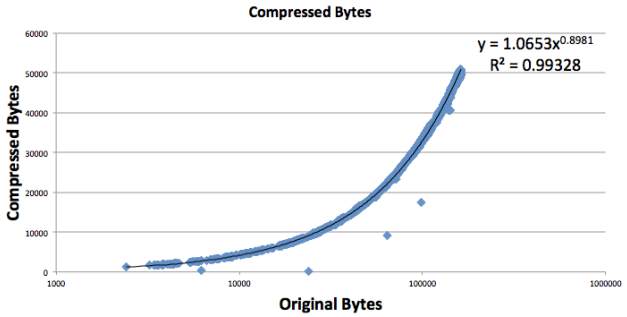

Figure 6. Elvis Original and Compressed Bytes

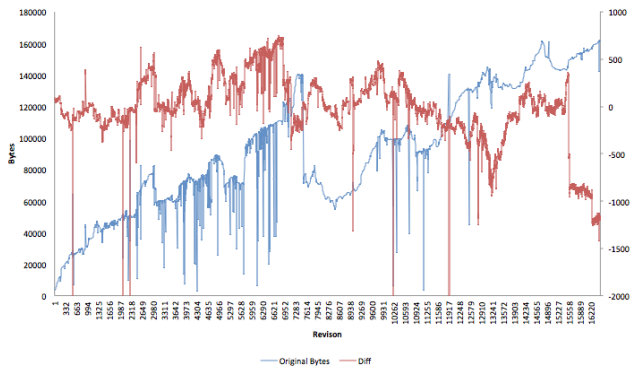

Figure 7. Elvis Page Length and Compression Difference

Figure 7 plots the length of the Elvis page over time and the difference between the expected and actual compression ratios. Revisions that were clearly vandalism (sudden deletions of content or massive increase in content, either of which were immediately fixed) have been dropped from this analysis. It's clear that the page increases in length over time with the exception of a sudden decrease. Examining the revision comments around this time shows that the editors decided that the page had grown too complex and that it should be split into different subsections, one for each major topic: one page for his movie career, one page for Graceland, etc. There is a sudden drop in the page length when subsections were cut out to form new pages. After that, however, the page starts growing again, soon being even larger than before. So did the effort to simplify the page fail? If we were to only look at the page length, we might conclude that. However, a closer examination of the page shows that this is not the case. Before the break, we see the difference between the expected and actual compression ratio increasing. In other words, the vocabulary was getting more complex. But after the split, this difference remains the same and even declines. This is reflective of the complexity of the vocabulary remaining more constant.

4.6 Anarchism

The Wikipedia page for Anarchism has 16,006 revisions, including revisions of registered and unregistered users. Like Elvis, the compression to original bytes ratio follows a power law distribution.

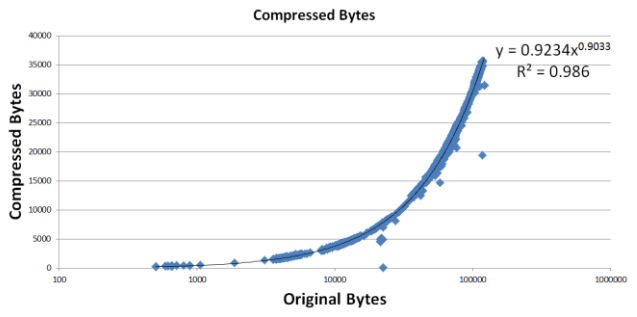

Figure 8. Anarchism Original and Compressed Bytes

Early on, the page shows an erratic, but slowly increasing compression difference as shown in Figure 7. At a certain point in the history, the compression diff starts decreasing with two small spikes.

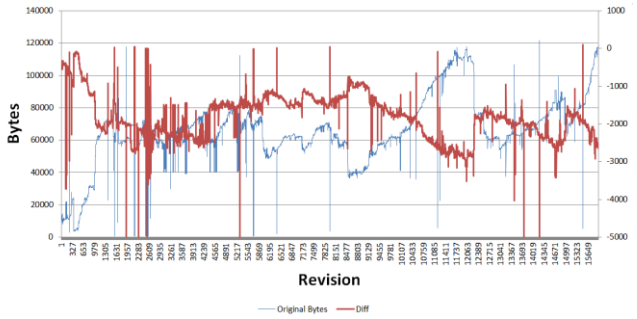

Figure 9. Anarchism Page Length and Compression Difference

4.7 Global Warming

The Wikipedia page for global warming has 18,581 revisions. The compressed and original bytes also exhibits a power law distribution, but with somewhat different parameters than Anarchism and Elvis.

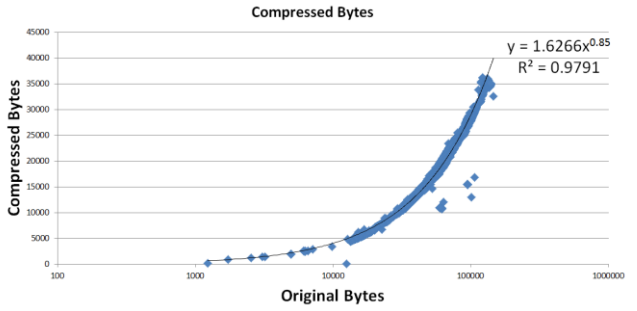

Figure 10. Global Warming Original and Compressed Bytes

The page has a clean and simple increase in compression difference from early on throughout approximately half of its life followed by a stable or slowly decreasing compression diff.

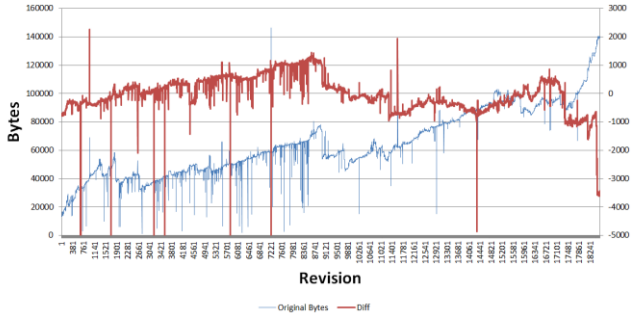

Figure 11. Global Warming Page Length and Compression Difference

4.8 Pumpkin

Finally, the page for Pumpkin also has a power law distribution relationship between compressed and original bytes. The page is also heavily edited, but considerably less than the other pages with only 1,949 revisions.

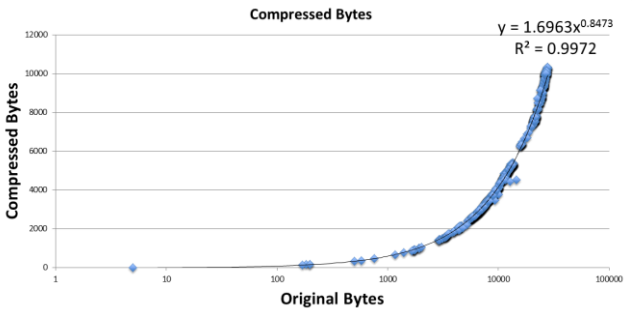

Figure 4. Pumpkin Original and Compressed Bytes

The compression difference history for this page is different than for the other three pages. It has a fairly stable Complexity Difference for much of its life. Then there is a sudden drop, followed by an increasing compression diff. This change took place in March of 2008. A look through the logs of the page shows that the Pumpkin page was vandalized and in mid-2008, shortly after this shift, the page was locked down and a large amount of new content was added. The inflection point occurs when the page was locked down and improved.

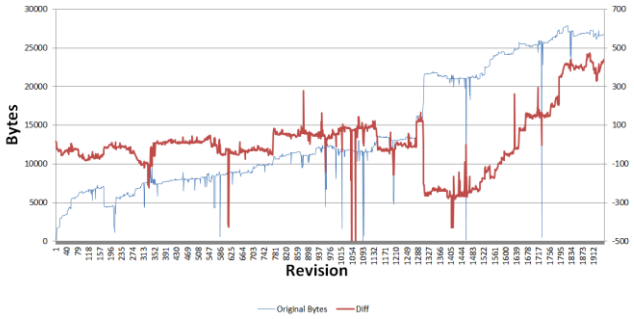

Figure 5. Pumpkin Page Length and Compression Difference

5. FINDING INFLECTION POINTS IN RESEARCH PAPER TITLES

5.1 Data

We also examined approximately 21.5 million publications from PubMed [7] that were made available for this year's WebSci14 data challenge. We split the paper titles into different files by year. (There are 1.7 million papers in the data set do not have years associated with them. We excluded these from this study.)

5.2 Results

We computed the Complexity Difference for the PubMed articles. The CD analysis shows that the complexity of the information stays relatively constant until later years even though the raw data size starts increasing earlier.

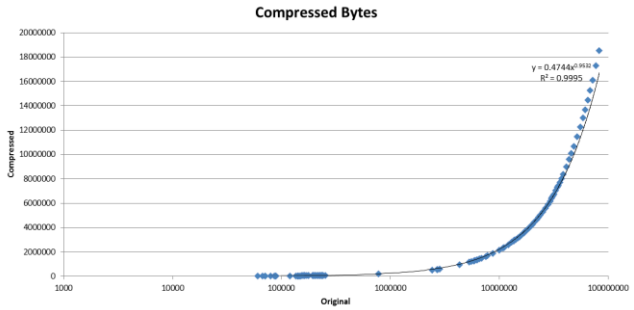

Figure 6. PubMed Original and Compressed Bytes

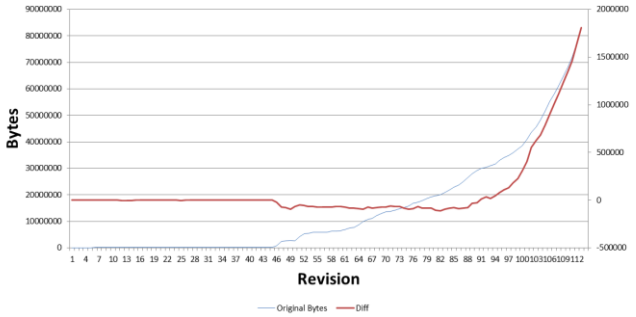

Figure 7. PubMed Complexity Difference

5.3 Other Applications of Compression

Using compression as a proxy for information content allows for other avenues of exploration. NCD, as outlined earlier, creates a powerful similarity measure. Additionally, we may wish to know the amount of new information produced by a new document. For the PubMed data set, this is the same as asking "given the article titles in some year N, how much new information was there in year N+1". Put another way, we wish to find $C(y_{n+1})$-$v_{n+1,n}$. Rearranging Equation 1, this is equivalent to finding $C(y_{n+1},y_n) - C(y_n)$. We then scale this by $C(y_{n+1})$. Intuitively, this value is the new information in year N+1, divided by the total information in year N+1; or more simply the percent of information in year N+1 which is new. This is described in Equation 2.

$$NI(y_n) = \frac{C(y_{n+1}, y_n) - C(y_n)}{C(y_{n+1})}$$

Equation 2. New, Unique Information

5.4 Results

We ran an NCD analysis of each year's paper titles against each other year, for years in the range 1900-1982. We then used these results to create a graph where each node represents a year and edge weights give the NCD between two years of papers (scaled so that larger values indicate the years are more similar; in traditional NCD the smaller a value the more similar the two documents).

Figure 8 shows the results when the edges are filtered so only the 10% strongest edges are shown. There are three distinct communities visible. The first, on the far right, contains the years from 1900-1907. The years 1908-1910 act as a bridge to the large community in the middle, which encompasses the years 1908-1926. The final community contains the years from 1926-1944. In general, two years will have a high NCD if they are close chronologically, and this is well illustrated by the community structure in the graph.

After 1945, each node is independent. The hypothesis is that there is an exponential growth of information, and as information grows it becomes more varied. Thus, the later years are dissimilar, simply because there is so much, and such different, information being discussed each year.

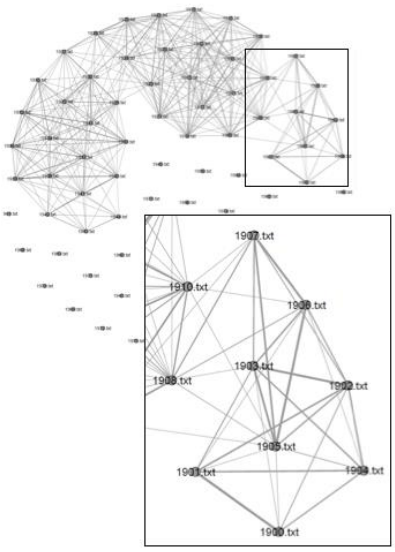

Figure 8. The 10% strongest edges in a similarity graph

This effect is robust: the same pattern is evident when all but 25% of the edges are removed (Figure 15) with 25% of the edges visible. In this case, the region from 1900-1944 is a dense graph, and only a handful of edges connect to other nodes. No year after 1960 has an edge connected to it, and 1960 only has degree 1 (it's connected to 1959). Put another way, the publications in the 1900s and 1940s are more similar to each other than publications from the 1970s to the 1980s.

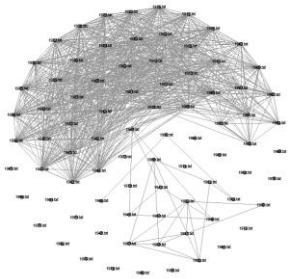

Figure 9. The 25% strongest edges in a similarity graph

On the surface this is a surprising result, but lends credence to the claim that exponential growth of knowledge leads to exponential specialization and distinction. As more information is gained, it splinters into different, distinct subfields that are dissimilar.

It's easy to check that each year more publications are written (a simple count of publications per year shows this to be true), but it's not so obvious that each of those publications contains new information. To check this, we look at two key metrics. First, we find the compressed size of the year file, using compression size as a proxy for information content. Additionally, we use the normalized new information content function given in Equation 3.

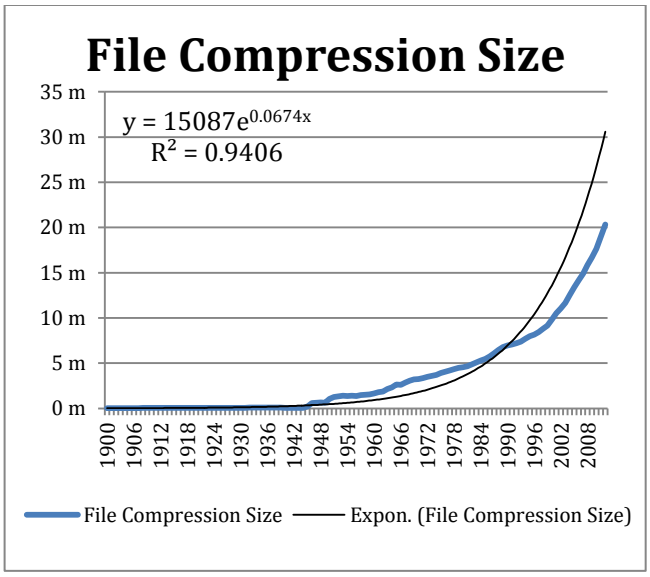

Figure 10. File Compression Size

Figure 10 shows a graph of the file compression size for each year, from 1900-2012. As expected, this graph shows an exponential growth in the amount of information contained in each year, giving evidence to the theories outlined above.

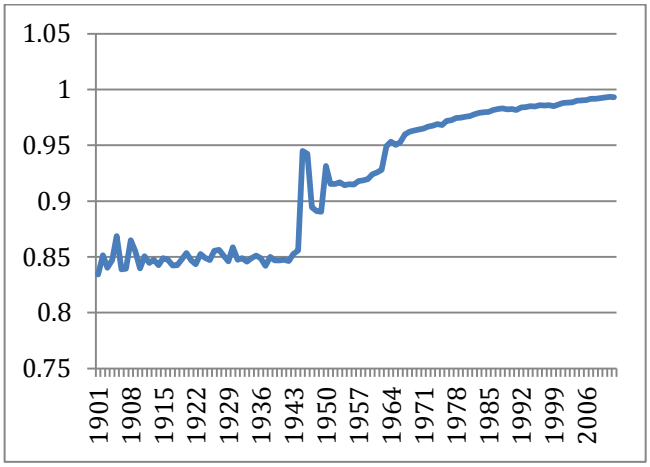

Figure 11. Normalized New Information Content

Figure 11 shows the new information each year as a percent of total information from that year (note that the chart starts at 1901 since this equation requires data from the previous year). For years before 1945, the new information produced each year is roughly 85% of the total information. Between 1945 and 1962, the new information content is around 92%, and afterwards the new information asymptotically approaches 1.

This makes clear a relation outlined earlier: as time goes on, more of the information being created each year is new information. This explains why the NCD for years later in the data set is so small. Similarly, the valley from 1945-1962 corresponds to the newly connected nodes in the second network. Later nodes weren't connected since a larger percent of the information in

each year was new information, and thus dissimilar to previous years.

The spike at 1945 is interesting, and corresponds to a change in the data set. Prior to 1945, each file was relatively small (approximately 200kb at a maximum), but afterwards the files became much larger (approximately 2500kb at a minimum), with 1945 falling roughly in the middle (808 kb). This increase in PubMed articles corresponds to an increase in the percent of new information published each year, and is related to real life events, most notably the post-war science boom.

6. DISCUSSION

The use of compression to analyze data in various ways has a number of advantages. First, it is straightforward to apply. All that is needed is to compress two items individually, and then compress the concatenation of them, and compare the results.

Second, a compression analysis does not require *a priori* knowledge of features to extract. This means both that the kinds of data that can be analyzed and the kinds of questions that can be asked are fairly broad.

There are some downsides of the application of compression. While on the one hand, it is not necessary to extract features; neither does one have much control over the features used.

Through our work in this paper, we have shown compression analysis is a useful tool for analyzing data sets where no prior knowledge of the important features is known.

7. Conclusion

This paper introduced a new measure called Complexity Difference that can be used to analyze change in group dynamics over time. Additionally, we have shown other uses for compression as a measure of information theory, notably in Normalized Compression Distance analysis and as a measure of normalized new information content. We've applied these measures to various data sets and shown that they identify specific points in time where there are inflection points in the complexity of the information in the data.

8. REFERENCES

[1] Cebrián, M., Alfonseca, M., & Ortega, A. 2005. Common Pitfalls Using the Normalized Compression Distance: What to Watch Out for in a Compressor. *Communications in Information & Systems*, *5*(4), 367–384.

[2] Cilibrasi, R., & Vitanyi, P. M. B. 2005. Clustering by Compression. *IEEE Transactions on Information Theory*, *51*(4), 1523–1545.

[3] Cilibrasi, R. L., Vitanyi, P. M. B., & De Wolf, R. (2004). Algorithmic clustering of music (pp. 110–117). *Proceedings of the Fourth International Conference on the Web Delivering Music*. doi:10.1109/WDM.2004.1358107

[4] Keogh, E., Lonardi, S., & Ratanamahatana, C. A. 2004. Towards parameter-free data mining. Presented at the KDD '04: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM. doi:10.1145/1014052.1014077

[5] Li, M., & Vitányi, P. 2008. *An Introduction to Kolmogorov Complexity and Its Applications*. *link.springer.com*. New York, NY: Springer New York. doi:10.1007/978-0-387-49820-1

[6] Li, M., Chen, X., Li, X., Ma, B., & Vitanyi, P. M. B. 2004. The similarity metric. *IEEE Transactions on Information Theory*, 50(12), 3250–3264. doi:10.1109/TIT.2004.838101

[7] Light, Robert P., Poolley, D., and Borner, K. 2013. Open Data and Open Code for Big Science of Science Studies. In *Proceedings of the International Society of Scientometrics and Infometrics Conference*. http://cns.iu.edu/docs/publications/2013-light-sdb-sci2-issi.pdf.

[8] Massa, Paolo. "Social networks of Wikipedia." *Proceedings of the 22nd ACM conference on Hypertext and hypermedia*. ACM, 2011..

[9] Medelyan, O., Milne, D., Legg, C., & Witten, I. H. 2009. Mining meaning from Wikipedia. *International Journal of Human-Computer Studies*, *67*(9).

[10] Miller, C. 2012. Interactive data-driven search and discovery of temporal behavior patterns from media streams (p. 1433). Presented at the the 20th ACM International Conference on Multimedia, New York, New York, USA: ACM Press. doi:10.1145/2393347.2396512

[11] Sumi R, Yasseri T, Rung A, Kornai A, Kertész J (2011) Edit wars in wikipedia. In: Privacy, Security, Risk and Trust (PASSAT), 2011 IEEE Third International Conference on and 2011 IEEE Third International Confernece on Social Computing (SocialCom). pp. 724–727..

[12] Wehner, Stephanie. "Analyzing worms and network traffic using compression."*Journal of Computer Security* 15.3 (2007): 303-320..

[13] Welser, Howard T., et al. "Finding social roles in Wikipedia." *Proceedings of the 2011 iConference*. ACM, 2011..

[14] Yasseri, T., Sumi, R., Rung, A., Kornai, A., & Kertész, J. 2012. Dynamics of Conflicts in Wikipedia. *PLoS ONE*, *7*(6), e38869. doi:10.1371/journal.pone.0038869.t001

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. (SAND2014-1415 C)