December 2013

Data Mining Revision Controlled Document History Metadata for Automatic Classification

Dustin Maass
University of Wisconsin-Milwaukee

DATA MINING REVISION CONTROLLED DOCUMENT HISTORY METADATA FOR AUTOMATIC CLASSIFICATION

by

Dustin Maass

A Thesis Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Master of Science

in Computer Science

at

The University of Wisconsin-Milwaukee

December 2013

ABSTRACT

DATA MINING REVISION CONTROLLED DOCUMENT HISTORY METADATA FOR AUTOMATIC CLASSIFICATION

by

Dustin Maass

The University of Wisconsin-Milwaukee, 2013
Under the Supervision of Professor Munson

Version controlled documents provide a complete history of the changes to the document, including everything from what was changed to who made the change and much more. Through the use of cluster analysis and several sets of manipulated data, this research examines the revision history of Wikipedia in an attempt to find language-independent patterns that could assist in automatic page classification software. Utilizing two sample data sets and applying the aforementioned cluster analysis, no conclusive evidence was found that would indicate that such patterns exist. Our work on the software, however, does provide a foundation for more possible types of data manipulation and refined clustering algorithms to be used for further research into finding such patterns.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction and Motivation

In the nearly 13 years since it was launched, Wikipedia has grown to include over 30 million articles in 287 different languages. In order to assist people looking for certain topics of information, Wikipedia was designed with a categorization system in place. Despite this system, many pages are still either not categorized or poorly categorized. Given the size of the knowledge base, manual categorization and validation of the existing categorizations for all Wikipedia's pages are not feasible.

This thesis describes research that attempts to identify patterns within the revision histories of Wikipedia articles. The goal of the research is to demonstrate that patterns in the version history metadata (frequency of changes, number of active authors, size of changes, etc.) could be used as part of a larger system for automatic categorization of Wikipedia pages. The research limits the classification task to the version history metadata and ignores the specific content of the changes because such an approach holds the promise of providing language-independent information for automatic classification.

1.1 Thesis Layout

The remainder of this paper is organized as follows: Section two presents previous work relating to either automated categorization methods or to data mining techniques applied to Wikipedia. Section three describes the data mining techniques used and techniques for assessing their effectiveness. Section four deals with setting up the data set selection, as well as our custom software we used for the data acquisition and manipulation. Section five discusses how the revision data was processed to produce the values used in data mining. Section six explains the results and findings of the data

mining analysis. Section seven describes the conclusions drawn from the results of the analysis. Finally, section eight describes some possible avenues for further research and possible software enhancements.

2 Related Work

Wikipedia has already been the basis for considerable research. Many of the works focus on utilizing the content of Wikipedia to aid researchers in performing research regarding other media. For example, Ayyasamy et al. used the existing classification system in Wikipedia to classify weblogs [2]. They sought to use articles in Wikipedia to build a mapping of terms to concepts to topics. They then used the mapping to better categorize the weblogs based on key terminology used in it.

Of more relevance to this research are several papers that used the revision history of Wikipedia as the primary focal point. Max and Wisniewski [10] utilized the revision history to build a resource corpus that can be used for identification of linguistic phenomena. Their research aimed to provide a resource for building improved language-based applications, such as more advanced spell-checkers and paraphrasing utilities. They attempted to accomplish this by examining and categorizing the revisions themselves. Their work expanded on research by Nelken and Yamangil [13], whose work had the same premise but on a more limited scope. Nelkin and Yamangil sought to identify “eggcorns”, or pairs of words that are correctly spelled and phonetically similar.

In addition to the Wikipedia aspects of this research, we also investigated the use of clustering algorithms. In this direction, we pulled heavily from the techniques described by Berkhin [3] and Pang-Ning, Steinbach, and Kumar [14]. Pang-Ning et al. wrote a thorough book on data mining, including a good reference on several different

analysis algorithms. Berkhin provides a more in-depth survey of clustering algorithms including *k*-means clustering, which is the primary type of analysis used in our research. Nazeer and Sebastian [5] delve into ways to improve the accuracy and efficiency of clustering algorithms, specifically the *k*-means algorithm. They revised the *k*-means algorithm slightly, creating a more efficient and accurate algorithm. Goutte and Gaussier [6] and Powers [15] have performed research into measuring the precision, recall, and F-Score of various algorithms. Goutte and Gaussier studied the confidence levels of precision, recall, and F-Score. Powers investigated various replacement measurements that aimed to be a more comprehensive means of measuring accuracy.

3 Data Mining and Clustering

In order to locate patterns within Wikipedia's revision history metadata, we turned to the sub-discipline of data mining. Data mining is the analysis of large quantities of data in an effort to identify interesting patterns. It includes several different types of analysis that can be used to this end. For instance, association rule analysis seeks rules to govern relationships between different variables, while anomaly detection aims to identify records that could represent erroneous or otherwise interesting data. In our research, we focused solely upon cluster analysis for pattern identification.

3.1 Clustering Algorithms

Cluster analysis is performed by grouping objects into classes through the proximity to one another using one or more characteristics of the objects. Given our desire to identify groupings among pages, cluster analysis was the logical option for our research. There are two basic types of clustering algorithms available for use.

In hierarchical clustering, the entire data set is plotted in a high dimensional space and the two closest points are “clustered” together. The central point of the new cluster is calculated and considered as a point. Then the next two closest points (or clusters) are combined to form a new cluster. This process is repeated until all the points and clusters are merged into a single cluster. The resulting pairings can then be drawn out to create a hierarchical tree showing the breakdown of related (closely-positioned) pairs.

The other basic type of clustering algorithm is *k*-means clustering. In a *k*-means clustering algorithm, all the data points are plotted in a high dimensional space, based on the values of interest. Next, centroid points are arbitrarily chosen for each of the *k*-clusters, with an emphasis on spacing them as far apart as feasibly possible. Then, all points are assigned to their nearest *k*-cluster centroid.

After all the points have been assigned to a cluster, the centroid point of each *k*-cluster is recalculated based on the potentially new set of points. Then all of the points are reassigned to their nearest centroid. The centroids are then recalculated again. This process of reassignment and recalculation repeats until none of the centroids move when they are recalculated, meaning that no points were reassigned to a different cluster after the previous recalculation. In order to prevent any situations where a point bounces perpetually between two clusters, the algorithm is usually limited to a predetermined number of iterations.

For our research, since the goal is a simple detection of groupings, we chose to use the *k*-means clustering algorithm. After we processed our data sets using the *k*-means clustering algorithm, we then attempted to generate a formula for determining which grouping/category a random piece of data, or possibly a new piece of data, belongs to.

3.2 Rapid Miner Data Mining Tool

When it came to executing the cluster algorithm analysis, it was decided to use a pre-existing commercial piece of software. This allowed more time to be invested in the deciding upon which manipulations of data should be considered as well as on the analysis of the results. The software selected for performing our cluster analysis is Rapid Miner, created by the Rapid-I company. Rapid Miner offers an open-source data mining solution that provides a graphical interface for building processes that can be executed upon multiple sets of data. In addition, Rapid Miner provides a set of direct APIs that allow it to be hooked directly into custom software. It also provides a great deal of flexibility in terms of data sources and data output formats, providing for much leeway in the other software components that are needed.

3.3 Measuring Effectiveness

The effectiveness of a cluster analysis is typically assessed by three different metrics: Precision, Recall, and F-score. These metrics are closely related and are based on a categorization of each data point's cluster assignment into one of four result categories: true positives, false positives, false negatives, and true negatives. We calculate each of the four values, for each cluster, as follows:

- true positive (tp) – the number of objects in this cluster which belong here
- false positive (fp) – the number of objects in this cluster which should not have been placed here
- true negative (tn) – the number of objects not placed in this cluster which should not have placed here

- false negative (fn) – the number of objects not placed in this cluster which belong here

The determination of which predefined group belongs to each calculated cluster was achieved by taking the group with the highest number of objects in each cluster as the group that is associated with that cluster.

Each of the different styles of data manipulation was run through the same clustering algorithm. Each one was then measured in terms of all three effectiveness measures.

3.3.1 Precision

The precision, or confidence, of a particular cluster is a measure of how successful the clustering algorithm is at including only objects of the cluster's pre-grouping into that cluster. This is calculated by taking the number of true positives and dividing them by the number of true positives plus the number of false positives, i.e. the total number of objects in the cluster.

$$Precision = \frac{tp}{tp + fp}$$

Figure 1: Precision Formula

Precision was measured for each of the k-clusters that were generated. The individual cluster precisions were averaged together to obtain the precision for the entire analysis.

3.3.2 Recall

The recall, or sensitivity, of a particular cluster is a measure of how successful the clustering algorithm is at obtaining all of the objects expected to be contained within the cluster, i.e. the objects from the grouping that belongs to a particular cluster. This is calculated by taking the number of true positives and dividing by the number of true positives plus the number of false negatives.

$$Recall = \frac{tp}{tp + fn}$$

Figure 2: Recall Formula

Recall was measured for each of the *k*-clusters that are generated. The individual cluster recalls were then averaged together to obtain the recall for the entire analysis.

3.3.3 F-Score

When researchers want to give a single score to a cluster analysis they typically use the F-Score, which is the weighted harmonic mean of both precision and recall. The weighting of the F-Score is used to focus on either the accuracy of correctness (i.e. precision) or the completeness (i.e. recall) of the clustering. In the F-Score formula, the value of β is the weighting of the precision over recall. It is calculated by dividing the product of the precision and recall scores by the sum of the recall and weighted precision. This is then multiplied by one plus the weight squared.

$$F_{\beta} = (1 + \beta^{\beta}) \times \frac{Precision \times Recall}{(\beta^{2} \square Precision) + Recall}$$

Figure 3: F-Score Formula

Since, in our research, we wanted the algorithm to be both as correct as possible

and as complete as possible, we used the balance-weighted F-Score (referred to as an F1-Score). This uses a β of 1, which results in:

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Figure 4: F1-Score Formula

The final value is a number between zero and one, with a value of one representing a perfect clustering algorithm. Unlike the recall and precision measurements, the F1-Score was calculated using the averaged recall and precision scores and was only calculated once for the entire clustering algorithm.

4 Data Set Selection

We chose two data sets to execute our analysis upon. Data set #1 of our research contained small groups of articles assigned to very specific categories. Each grouping was comprised of 20 different pages chosen haphazardly, each closely coupled to the others by a very specific type of topic, e.g. famous historical battles, mammals. A total of 14 groups were selected. This data set provided 1,030,661 revision entries for use in the analysis.

Birds	Comedy Television Shows
Historical Battles	Historical Leaders
Megacities	Movie Stars
Rabbis	Scientists
Theorems	World Music Award Winners
Fortune 500 Companies	Mammals
Northern Europe Countries	Software People

Table 1: Data Set #1 Categories

With concerns that the small number of articles from the first data set would prevent more general trends from possibly appearing, we decided that a second data set would be useful. Data Set #2 was intended to provide a larger article count which would allow us to look for general trends of three different, broader categories. Each category of articles was comprised of between 1685 and 1707 pages, with each group representing a broad type of topics. A total of 3 groups were used, providing 859,831 revision entries for use in the analysis.

Category	# of Articles
Greek Mythology	1,707
Quantum Science	1,685
US Olympic Gold Medalists	1,690

Table 2: Data Set #2 Categories

4.1 Metriki Data Extraction and Manipulation Software

In order to obtain the revision history data from Wikipedia, we made use of the Metriki software, originally designed by Peine [15]. His software accesses the revision history of Wikipedia by sending the request to Wikipedia's server via a URL. Wikipedia then returns a XML file that contains all the requested information. It also provided a starting point for manipulating the data for further analysis.

Metriki was also expanded upon by Shah [17]. That expansion provided a more developed database structure as well as a further expansion upon the data manipulation. Shah also executed some preliminary data analysis upon a small data set.

Above and beyond what Peine and Shah had developed, further modifications to the database layout were needed. One additional field was necessary in the page

information table to track which group the page belonged to for post-analysis reference. Also, the entirety of the user information table was discarded as everything we needed to know about the users was accounted for in the revisions table. This latter modification was made in order to improve the performance of the database.

New modules were also added to the Metriki software. The first one was used to manipulate the revision information into the various formats and measurements we used in the cluster analysis. This converted the raw MySQL data into an easier-to-import CSV format. Another module that was added was a module to take the output from Rapid Miner and compute measurements of effectiveness on the data output.

In addition to these upgrades, it was noted during the execution of the Metriki software that excessive time was required to parse the downloaded Wikipedia data and place it into the MySQL database. It was discovered that the existing design of the database used proper foreign key setups which, in conjunction with a defensively-coded MySQL custom Insertion function, caused many redundant SELECT SQL statements. As a result, the duration of the basic INSERT statements grew into human-perceivable durations for relatively small batches of information (in the realm of a few hundred revisions). To remedy this problem, the custom Insertion function and the foreign key constraints were removed from all the tables. In general database practices the foreign key constraints would be left in place. However, given that the data would only be inserted by a single user and that the dependent data was ensured to be present by the Metriki software, the decision was made to remove the foreign key constraints in order to reduce the download and processing time.

5 Data Preparation

An important goal we hoped to achieve in creating and choosing measurements was to identify metrics that showed higher variability among the articles. This was important because metrics that show little variation can't be used to distinguish the articles. We identified four distinct measurements and one combination of those measurements to analyze.

5.1 Edits Over Fixed Time

The first data measurement that we looked into was the distribution of edits over a fixed amount of time from the page's creation. This could identify basic trends such as articles being edited repeatedly right after their creation or articles being edited routinely over a longer time span. This could also identify if a category of pages was updated several times and then never updated again.

To measure this information, all the edits for each article were broken into buckets based on when the edit occurred in relation to the creation date of the article. Edits outside of the predefined number of time periods were ignored for the purposes of this analysis. Each bucket represented a single period of time (e.g. one week or one month). Once all of the edits were assigned to their respective buckets, or discarded if necessary, then the percentage of edits in each bucket relative to the total number of edits in all the buckets, and not the total edit count, was calculated for each bucket. These percentages became the data that was provided to the Rapid Miner process for analysis.

For example, in the 90-Day measurement, 90 buckets were created. The first one was for all edits performed the day that the article was created; The 90^{th} bucket was for all the edits performed on the 90^{th} day after the article was created. All edits after the 90

days would be ignored for this measurement. The percentages would be calculated based on the number of edits that occurred within the first 90 days.

Here are the time periods and number of periods that were measured:

Time per Period	Quantity of Periods
Days	90, 180, 360
Weeks	13, 26, 52
Months	6, 12

Table 3: Edits Over Fixed Time

5.2 Edits Over Lifetime

The second measurement that was analyzed was the number of edits over the lifetime of the article. Very similar to the first measurement, it used the entire set of edits performed on an article. This allowed us to look for trends similar to those found by the Edits Over Fixed Time measurement, but allowed for adjustment based on the length of time the article has been around. The articles' lifetimes needed to be taken into account since newer articles may not have had the same number of edits performed to them as older articles. This could potentially mask certain articles from being clustered correctly now, or prevent certain articles from being clustered correctly later, since this data is perpetually dynamic.

To measure this information, the same technique that was used in the Edits Over Fixed Time was used, with one alteration. The edits were assigned to each bucket based on what percentile of the article's lifetime the edit occurred in (e.g. first 1%, the 75^{th} 1%). Each bucket represented a predefined number of percentiles. This bucket sizes were one of the following sizes:

Bucket Size	Number of Buckets
1%	100
2%	50
4%	25
10%	10

Table 4: Edits Over Lifetime

5.3 Authors Over Fixed Time

The next measurement that was investigated was the number of authors over a fixed time. This allowed us to see if all the edits were performed by a small select group, or if the group of individuals performing the edits was larger or changed/grew/shrank over time.

To measure the author count, all the edits for each article were placed again into one of a number of buckets based on when the edit occurred in relation to the creation date of the article. Similar to the Edits Over Fixed Time, each bucket represented a single period of time. Once all of the articles were assigned to their respective buckets (or discarded), the number of unique authors in each bucket was counted. These raw counts became the data that was provided to the Rapid Miner process for analysis.

For the Authors Over Fixed Time, the only bucket size we looked at was a one-month sized bucket. We analyzed this data for 12, 24, and 36-month durations.

5.4 Average Edit Size Over Time

Another potentially useful measurement was the average edit size over time. This measurement was also done over different fixed time frames. This would identify differences between articles that have large chunks of data that were added early on with

only minor edits later, versus articles that grew in small amounts over the entire time frame, versus articles that had large chunks of data continuously added/removed.

To measure the edit size, all the edits for each article were placed again into one of a number of buckets based on when the edit occurred in relation to the creation date of the article. Similar to the Edits Over Fixed Time, each bucket will represent a single period of time. Once all of the revisions were assigned to their respective buckets (or discarded), the average size of all the edits in each bucket was calculated. These averages became the data that was provided to the Rapid Miner process for analysis.

Here are the time periods and number of periods that were measured:

Time per Period	Quantity of Periods
Days	90, 180, 360
Weeks	13, 26, 52
Months	3, 6, 12

Table 5: Average Edit Size Over Time

5.5 Combined Measures

In addition to the four previously noted trends, we also examined a few combinations of multiple trends. Each of the individual trends involved in the combination measurements were calculated independently. Then the data was concatenated into a single set of values. Those concatenated sets were provided to Rapid Miner. The combinations that we explored were:

Trend 1	Period	# of Periods	Trend 2	Period	# of Periods
Edits Over Lifetime	10%	10	Authors Over Fixed Time	Month	12
Edits Over	10%	10	Authors Over	Month	24

Lifetime			Fixed Time		
Edits Over Lifetime	10%	10	Authors Over Fixed Time	Month	36

Table 6: Combined Measures

6 Results – Data Set #1

The results of the processing on the first Data Set were modest. The resulting F1-Scores ranged from 0.0095 up to 0.3557, with an average of .1882. Given the resultant F-scores range from 0.0 to 1.0, these results were far from conclusive. The result information is provided in Appendix A.

We took note of two particular findings in our results. The first was that both the 3-Month Edits Over Fixed Time and the 3-Month Average Size Over Fixed Time caused the Rapid Miner software to crash. The cause of these crashes remains unknown.

The second, and more interesting, observation was that two particular measurements provided all-around higher results than all of the others. All of the Edits Over Lifetime measurements and all of the Authors Over Fixed Time measurements scored between .2035 and .3557 with an average of .2973, while other data points ranged from .0095 to .2558 with an average of .1481. It was this observation that prompted the further examination of a combined measurement of the Edits Over Lifetime with the Authors Over Fixed Time.

7 Results – Data Set #2

The second data set provided much better F1-Scores on the average. They ranged from .1676 to .4357 with an average of .3096. Again, given the possible F1-Score range, this did not appear to be very conclusive. The full set of results is listed in Appendix B.

There were again two notable observations regarding these results. First, similarly to the first data set, the 3-Month Edits Over Fixed Time and the 6-Month Edits Over Fixed Time both caused the Rapid Miner software to crash during the analysis. The cause is again unknown.

The second, and again more interesting, observation was that seven of the measurements resulted in an unusual anomaly where Rapid Miner placed all of the articles into the same cluster. All seven data measurements were re-executed in the Rapid Miner software two additional times, with Rapid Miner returning the same anomaly each time for all seven measurements. This is being treated as an error on the Rapid Miner software. Under this presumption, when the results for this data set were re-examined, the F1-Scores ranged from .2821 to .4357 with an average of .3727.

8 Discussion

Given that the F1-Scores in our findings never exceeded .5000, there is no way to claim that we have any conclusive patterns within the data. There are several issues that could be interfering with the possibility of better results.

Firstly, there are automated scripts, or bots, that continuously parse Wikipedia's articles performing automatic changes to pages. Some of these bots look for potential vandalism to flag and/or correct. Others look for and correct spelling and grammar errors. Still more will parse a pages content looking for words/phrases that could link to other pages making those links. Because of this uncontrolled autonomous behavior, the edits from the various bots could be skewing the data and hiding any true patterns from the revision history metadata.

Secondly, our research has intentionally disregarded all language-specific context.

It may be that there exists trends within that context that were overlooked. A combination of some revision history metadata measurements with some limited language-specific context could lead to trends.

Lastly, while our research performed analysis on several different data measurements, there are many more possible data measurements that could be calculated and analyzed. Since all revision history entries also contain a flag indicating if the author of the edit is working anonymously, it is possible to analyze anonymous versus logged-in users. It would also be possible to analyze minor versus major edits since they are also tracked via a flag in the metadata. In place of looking at the percentages of Edits Over Fixed Time, it could be worthwhile to look at the raw edit counts over those fixed times. In addition to these and other data measurements, there are numerous possible combinations of measurements that could reveal trends.

9 Conclusion

Though we were not able to make any definitive conclusions, we were able to observe several other points that are worth noting from the results of this research.

Firstly, the use of more general categorizations appears to yield significantly better results. Though far from conclusive, there may be justification to perform further investigation into looking at even more broad categories, such as the basic person, place, thing, and idea groupings. This could provide a more hierarchical approach to determining more specific page categorizations by allowing different algorithms to be used upon different subsets of data.

The second conclusion that can be drawn from these results is that it may be necessary to perform the cluster analysis upon a wider array of data measurements,

including different combinations of two or more of the basic measurements being used.

10 Future Work

There are three main routes that can be explored via future research at this juncture. There are also two additional software optimizations to the Metriki software that could be used to expedite data processing.

The first main route of exploration could be to obtain different types of data for processing. This could be done in two different ways. It could be achieved by using a greater number of articles from additional broad categories. Making the categories even broader still could also be used to accomplish this task.

The second main route could be to add additional data measurements. Both new types of measurements and different periods/period counts of existing measurements would suffice for this. It could also be achieved through additional combinations of the existing and new data measurements.

Another potential route of exploration could be focusing on removing potentially noisy data. This would include things such as vandalism and vandalism corrections. These edits could be causing a skew in the data that may be obfuscating more interesting, and subsequently useful, patterns. It is also possible that the vandalism and their corrections could be, in and of itself, a interesting pattern. Though this avenue of research is not a trivial task, it could definitely aid in the search for patterns in the revision history of articles.

The first software optimization for Metriki would be to obtain bot status with Wikipedia. This would permit Metriki to download the revision history data in larger chunks, permitting a significantly faster download and processing time. The second

optimization that could be done to Metriki would be to incorporate the Rapid Miner module directly into Metriki itself. This would allow the cluster analysis to be controlled from Metriki, thus improving the overall performance of the analysis software.

REFERENCES

[1] Anderka, Maik, Benno Stein, and Nedim Lipka. "Towards automatic quality assurance in Wikipedia." *Proceedings of the 20th international conference companion on World wide web*. ACM, 2011.

[2] Ayyasamy, Ramesh Kumar, et al. "Mining Wikipedia knowledge to improve Document indexing and classification." *Information Sciences Signal Processing and their Applications (ISSPA), 2010 10th International Conference on*. IEEE, 2010.

[3] Berkhin, Pavel. "A survey of clustering data mining techniques." *Grouping multidimensional data*. Springer Berlin Heidelberg, 2006. 25-71.

[4] Conrad, Jack G., et al. "Effective document clustering for large heterogeneous law firm collections." *Proceedings of the 10th international conference on Artificial intelligence and law*. ACM, 2005.

[5] Nazeer, KA Abdul, and M. P. Sebastian. "Improving the Accuracy and Efficiency of the k-means Clustering Algorithm." *Proceedings of the World Congress on Engineering*. Vol. 1. 2009.

[6] Goutte, Cyril, and Eric Gaussier. "A probabilistic interpretation of precision, recall and F-score, with implication for evaluation." *Advances in Information Retrieval*. Springer Berlin Heidelberg, 2005. 345-359.

[7] Hegland, Markus. "Data mining techniques." *Acta Numerica 2001* 10 (2001): 313-355.

[8] Makhoul, John, et al. "Performance measures for information extraction."*Proceedings of DARPA Broadcast News Workshop*. 1999.

[9] Marxer, Ricard, Hendrik Purwins, and A. Hazan. "An f-measure for evaluation of unsupervised clustering with non-determined number of clusters." *Report of the EmCAP project (European Commission FP6-IST, contract 013123), http://mtg. upf. edu/files/publications/unsuperf. Pdf* (2008): 1-3.

[10] Max, Aurélien, and Guillaume Wisniewski. "Mining Naturally-occurring Corrections and Paraphrases from Wikipedia's Revision History." *LREC*. 2010.

[11] Milne, David, and Ian H. Witten. "An open-source toolkit for mining Wikipedia."*Artificial Intelligence* (2012).

[12] Müller, Emmanuel, et al. "Evaluating clustering in subspace projections of high dimensional data." *Proceedings of the VLDB Endowment* 2.1 (2009): 1270-1281.

[13] Nelken, Rani, and Elif Yamangil. "Mining Wikipedia's article revision history for training computational linguistics algorithms." *Proceedings of the AAAI Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy*. 2008.

[14] Pang-Ning, Tan, Michael Steinbach, and Vipin Kumar. "Introduction to data mining." *Library of Congress*. 2006.

[15] Peine, Zachary. "Metriki". Master's Project Report, University of Wisconsin-Milwaukee, 2012

[16] Powers, D. M. W. "Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation." *Journal of Machine Learning Technologies* 2.1 (2011): 37-63.

[17] Shah, Nairuti. "Capstone Project Report". Master's Project Report, University of Wisconsin-Milwaukee, 2012

[18] Smets, Koen, Bart Goethals, and Brigitte Verdonk. "Automatic vandalism detection in Wikipedia: Towards a machine learning approach." *AAAI Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy*. 2008.

[19] Sorg, Philipp, and Philipp Cimiano. "Enriching the crosslingual link structure of wikipedia-a classification-based approach." *Proceedings of the AAAI 2008 Workshop on Wikipedia and Artifical Intelligence*. 2008.

[20] Strube, Michael, and Simone Paolo Ponzetto. "WikiRelate! Computing semantic relatedness using Wikipedia." *AAAI*. Vol. 6. 2006.

[21] Voss, Jakob. "Measuring wikipedia." (2005).

[22] Yamangil, Elif, and Rani Nelken. "Mining Wikipedia revision histories for improving sentence compression." *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*. Association for Computational Linguistics, 2008.

Appendix A: Data Set #1 Results

Measurement	Period Length	Period Count	Precision	Recall	F1-Score
Edits Over Fixed Time	Month	3	N/A	N/A	N/A
		6	.1131	.1607	.1327
		12	.1087	.1607	.1297
	Week	13	.0445	.1107	.0635
		26	.1337	.1857	.1555
		52	.1383	.1750	.1545
	Day	90	.1450	.1035	.1208
		180	.1853	.1321	.1543
		360	.2095	.1464	.1723
Edits Over Lifetime	1%	100	.1938	.2142	.2035
	2%	50	.5595	.2607	.3557
	4%	25	.5472	.2607	.3531
	10%	10	.3923	.2750	.3233
Authors Over Fixed Time	Month	12	.5501	.1999	.2933
		24	.4693	.1964	.2769
		36	.4839	.1928	.2758
Average Size Over Fixed Time	Month	3	N/A	N/A	N/A
		6	.0051	.0714	.0095
		12	.0051	.0714	.0095
	Week	13	.0051	.0714	.0095
		26	.2161	.2178	.2170
		52	.2043	.2071	.2057
	Day	90	.2846	.1964	.2324
		180	.2539	.1892	.2168
		360	.2300	.1928	.2098
Edits Over Lifeime / Authors Over Fixed Time	10% / Month	10 / 12	.1315	.1892	.1552
		10 / 24	.2118	.2071	.2094
		10 / 36	.2748	.2392	.2558

Appendix B: Data Set #2 Results

Measurement	Period Length	Period Count	Precision	Recall	F1-Score
Edits Over Fixed Time	Month	3	N/A	N/A	N/A
		6	N/A	N/A	N/A
		12	.1119	.3333	.1676
	Week	13	.1119	.3333	.1676
		26	.1119	.3333	.1676
		52	.1119	.3333	.1676
	Day	90	.1119	.3333	.1676
		180	.1119	.3333	.1676
		360	.1119	.3333	.1676
Edits Over Lifetime	1%	100	.4116	.3941	.4027
	2%	50	.2698	.3897	.3189
	4%	25	.2667	.3901	.3168
	10%	10	.3931	.3933	.3932
Authors Over Fixed Time	Month	12	.1119	.3333	.1676
		24	.4109	.3777	.3936
		36	.2389	.3441	.2821
Average Size Over Fixed Time	Month	3	.2730	.3995	.3243
		6	.3746	.3767	.3757
		12	.4262	.4199	.4230
	Week	13	.3953	.3893	.3923
		26	.4101	.4122	.4124
		52	.3726	.3686	.3706
	Day	90	.4270	.3458	.3821
		180	.2711	.3437	.3031
		360	.4176	.3644	.3892
Edits Over Lifeime / Authors Over Fixed Time	10% / Month	10 / 12	.4667	.4087	.4357
		10 / 24	.4388	.3940	.4152
		10 / 36	.3959	.3625	.3785