Common misspellings over the years using Google Ngrams dataset and Hadoop MapReduce

Maree Carroll (maree.carroll [at] gmail.com)

***Abstract*— By tracking the frequency of common spelling mistakes over the years using a Hadoop MapReduce cluster on the Google Ngrams dataset, this report found by 1800 the misspelling rate dropped to negligible amounts, which may warrant more investigation – especially if ngram datasets are to be used in automated spelling correction. Experimenting with the number of nodes available in the cluster, the expected was found with shorter execution times with more nodes up to a point, where the overhead or more nodes offset the benefits (and this point is found quickly with a dataset in the low-gigabytes size).**

I. INTRODUCTION

This report looks at the use and performance of various-size Hadoop clusters using an example of tracking common spelling mistakes over the years using the Google Ngrams dataset and Hadoop MapReduce. First, the report explains how the map reduce program has been implemented. Next, experiments undertaken on the programs are reported on with output and timings. The results are then critically analysed, in particular the scalability of the programs as more nodes are added to the cluster. Looking at the output of the mapreduce program, we discuss any findings that warrant further investigation. Before concluding, a review of the literature of other studies on this dataset is discussed.

II. IMPLEMENTATION

The following is a description of the dataset used and an explanation of how the map reduce program was implemented and run.

A. About the dataset

The dataset used is the AWS Public Data Set Google Books Ngrams, in particular the American English (eng-us-all) 1-gram dataset, available at `s3://datasets.elasticmapreduce/ngrams/books/20090715/eng-us-all/1gram/data`.

As explained by Amazon Web Services, Inc. (2016), the dataset is stored as a single object in Amazon S3 as a LZO compressed sequence file (which can be easily used by Hadoop Map Reduce with row offset as type LongWritable and raw data type TextWritable). Each entry is a tab separated string consisting of {1-gram} {year} {occurrences} {pages} {books} as explained below:

- **1-gram** - The actual 1-gram
- **year** - The year for this aggregation
- **occurrences** - The number of times this n-gram appeared in this year
- **pages** - The number of pages this n-gram appeared on in this year
- **books** - The number of books this n-gram appeared in during this year

The dataset size is 291,639,822 rows (3.0 GB). The Ngram words were extracted from the Google Books corpus, where as part of the Library Project they have collected the worlds from books in several libraries using Optical Character Recognition (OCR) technology.

The other dataset used is the list of the most common misspellings according to Oxford Dictionaries. (2016). From this web page, two lists of commonly misspelled words, their correct spellings and their misspellings, has been gathered.

B. About the programs

Each input, as described earlier, in in the form of a text delimited string of $< 1-gram >< year >< occurrence >< pages >< books >$, from which the program described here extracts only the 1-gram, year and occurrence (number of times the 1-gram appeared in year) values.

MapReduce programs, in their simplest form, consist of a *mapper* and a *reducer*.

In the common misspellings by year program described here, the mapper emits the $< key, value >$ of $< year, count >$ where count is encoded as a negative value (multiply its occurrence value by -1) if the 1-gram (word) is one of the commonly misspelled words and has *incorrect* spelling, and positive (no change applied to count) if the word is one of the commonly misspelled words that has *correct* spelling.

The 1-gram corpus is processed by the map reduce program to identify words that are in the commonly misspelled words list, encode their count (by year) as either correctly spelled or incorrectly spelled, and emit this encoded value (with the year value as key) from the mapper.

The reducer then uses the encoding to count the number of misspellings from the commonly misspelled words corpus, summing the counts of misspellings (numerator), and both misspellings and correct spelling counts as a total (denominator). The result of dividing misspellings count by total is then multiplied by 100 (and rounded) so the reducer can emit $< year, percentagemisspelled >$.

$$percent = round((float)n/(float)total * (float)100) \quad (1)$$

In the processing applied here, a combiner is not used because a combiner requires the operation performed by the reducer to be both *commutative* **and** *associative*, yet the processing to determine the percentage that commonly

misspelled words are indeed misspelled as a percentage, is not both of these things. This comes at a cost of having more data moved around at the shuffle and sort stage.

Pseudocode for the mapper and reducer are shown below in Algorithm 1.

Algorithm 1 MisspellingsPercentage

procedure MAP
    misspellings ← list of *incorrect spellings*
    correctspellings ← list of *correct spellings*
    word ← first value of *string*
    year ← second value of *string*
    count ← third value of *string*
    if *word*inlist of *incorrect spellings* **then** emit (*year*,*count* * -1)
    if *word*inlist of *correct spellings* **then** emit (*year*,*count*)
procedure REDUCE
    $n \leftarrow 0$.
    $total \leftarrow 0$.
for each count:
    truecount ← *count*.
    if $count < 0$ **then**
        $truecount \leftarrow count * -1$.
        n ←*n* + *truecount*.
    total ←*total* + *truecount*.
end:
    percent ←*n* / $total * 100$. emit (*year*,*percent*)

C. Limitations

The following limitations in the use of the Google Ngram dataset have been identified:

- Zhang, S. (2016) noted limitations in the use of n gram to study language because of OCR errors.
- Zhang also noted the over-abundance of scientific literature, so is unlikely to representative of literature more broadly speaking.
- Represents only six percent of human writing, so findings cannot be generalised across all literature.

There is also an obvious limitation with the use of the common misspellings dataset. This is a limited number of words that are being used as a corpus for gauging misspelling, and there has been no attempt to assess how well this reflects words used over the period captured by Google Ngrams. The common misspellings list was determined from a different corpus to that of the Google Ngrams, that being the Oxford English Corpus, described as ”an electronic collection of over 2 billion words of real English” which is likely a different population of words to the

III. EXPERIMENTS

A. Description of Experiments

Experiments were undertaken to try to gauge the right size cluster for this mapreduce program and dataset. Timings were taken for map reduce runs using various cluster sizes, increasing the number of task nodes made available to the run until the runtime started to curve upwards (i.e. a degradation of performance).

The degradation of performance happens when the overhead of more nodes outweighs the benefits of more workers.

Starting out with the default task nodes, the number of task nodes was increased by two each ”boost”, and the execution time of the same mapreduce program was recorded and plotted.

B. Findings

TABLE I
EFFECT OF CLUSTER SIZE

Task nodes	Run time
0	1114000
2	655000
4	412000
6	67000
8	82000
10	256000

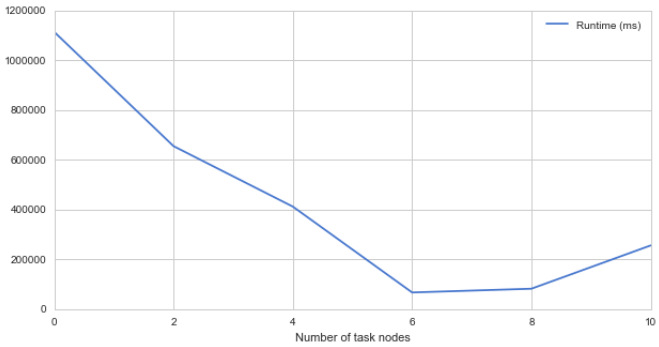

Fig. 1. Plot of run time for map reduce (milliseconds) with additional task nodes (cluster boosting)

IV. CRITICAL ANALYSIS

Discussing the results as illustrated in the figures above, it seems for this particular mapreduce program and dataset the optimum number of nodes is likely between five and eight. Time constraints got in the way of a more conclusive answer, however with further experiments a confidence interval could be determined. Simply throwing more nodes at the problem only works up to a point, and for Hadoop implementation of MapReduce (and the problem at hand) this number of nodes is reach unexpectedly early. This suggests that either Hadoop MapReduce is very inefficient on a smaller dataset (only 3 gigabytes) because of the overheads for each node, or that the problem can be solved in a better way that will harness more nodes more effectively. Certainly the lack of combiner would contribute to reduced scalability of this solution. Time constraints dictated the level of further investigation in this regard, but it is certainly something that could be a topic for further research and literature review.

V. ANALYSIS OF OUTPUTS

The results of the mapreduce program were concatenated and turned into a comma-separated form so they could be more easily plotted. The illustration of the prevalance of misspellings across the years in the Google US English 1-gram corpus is shown in Figure 2 below.

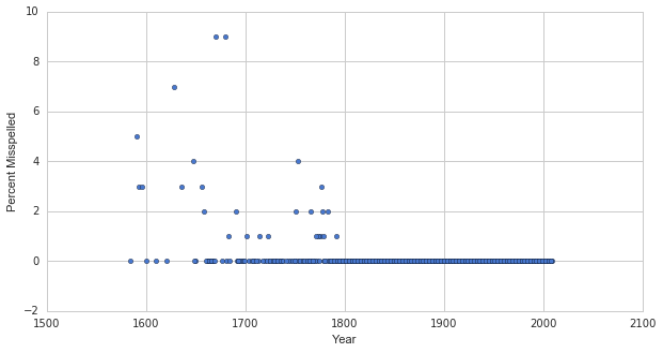

Fig. 2. Plot of the percentage of commonly misspelled words that were misspelled

Taking into the account the limitations on generalisability and OCR errors mentioned earlier, as well as the limitations of the incorrect spellings dataset, we can see prior to the 19th Century commonly misspelled words that were indeed incorrectly spelled in literature could get up as high as nine percent. Just prior to the 1800s the misspelling rate drops off to zero. Referring to American Printing History Association. (2016), there is some indication that the adoption of new printing technology was quite prevalent at this time, which may have had a positive effect on spelling correctness (and possibly the production of text that produced less confounding OCR errors).

Luey, Beth (2009). suggests that by 1800 the ”globalisation” of literature was occurring, with a rise in American authors and editors, which would have also had a positive influence on spelling correctness and help explain the drop off in misspellings at this time.

VI. LITERATURE REVIEW

Papers on Ngrams with regards to spelling errors include research into the use of Ngrams for learning rules for correction of spelling and grammar[4,9], as well as proposals for using similar (web page) ngrams datasets[6] for applications in spelling correction[3,5].

Different from this research into the prevalence of misspellings in Google Ngrams corpus, there is some research into studying how words have changed over time where words are spelled differently have evolved into new correct forms rather than being considered misspellings. This would be something to take into account for more thorough research into prevalence of misspellings.

VII. CONCLUSIONS

Hadoop MapReduce can be used to process the freely available Google Ngrams for finding interesting patterns. These findings can then be further investigated to see if the same patterns emerge across other datasets.

Hadoop MapReduce cluster size should always be tuned to both the program and the dataset size, as the overhead of nodes to the benefits of more workers is a trade off that is approached quickly in smaller datasets, and one should not make assumptions.

This research into the prevalence of misspellings in published literature is important because of the work being done to learn rules for spelling correction from ngrams.

REFERENCES

[1] Amazon Web Services, Inc. (2016). AWS Public Data Set Google Books Ngrams. [online] Available at: https://aws.amazon.com/datasets/google-books-ngrams/ [Accessed 8 Oct. 2016].

[2] American Printing History Association. (2016). History of Printing Timeline - American Printing History Association. [online] Available at: https://printinghistory.org/timeline/ [Accessed 8 Oct. 2016].

[3] Bassil, Y. and Alwani, M. (2012). Context-sensitive Spelling Correction Using Google Web 1T 5-Gram Information. Computer and Information Science, 5(3).

[4] Google Books. (2016). Patent US6618697 - Method for rule-based correction of spelling and grammar errors. [online] Available at: https://www.google.com/patents/US6618697 [Accessed 7 Oct. 2016].

[5] Islam, A. and Inkpen, D. (2009). Real-word spelling correction using Google Web IT 3-grams. Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3, [online] pp.1241-1249. Available at: http://dl.acm.org/citation.cfm?id=1699670 [Accessed 7 Oct. 2016].

[6] Lin, Y., Michel, J., Aiden, E., Orwant, J., Brockman, W. and Petrov, S. (2012). Syntactic annotations for the Google Books Ngram Corpus. Proceedings of the ACL 2012 System Demonstrations, [online] pp.169-174. Available at: http://dl.acm.org/citation.cfm?id=2390499 [Accessed 7 Oct. 2016].

[7] Luey, Beth (2009). Modernity and Print III: The United States 1890-1970 A Companion to the History of the Book, p. 369.Wiley-Blackwell, Oxford.

[8] Mihalcea, R. and Nastase, V. (2012). Word epoch disambiguation: finding how words change over time. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2, [online] pp.259-263. Available at: http://dl.acm.org/citation.cfm?id=2390727 [Accessed 7 Oct. 2016].

[9] Nazar, R. and Renau, I. (2012). Google books n -gram corpus used as a grammar checker. Proceedings of the Second Workshop on Computational Linguistics and Writing (CLW 2012): Linguistic and Cognitive Aspects of Document Creation and Document Engineering, [online] pp.27-34. Available at: http://dl.acm.org/citation.cfm?id=2388652 [Accessed 7 Oct. 2016].

[10] Oxford Dictionaries. (2016). Common misspellings — Oxford Dictionaries. [online] Available at: https://en.oxforddictionaries.com/spelling/common-misspellings [Accessed 7 Oct. 2016].

[11] University of Pennsylvania. (2016). Web 1T 5-gram [online] Available at: https://catalog.ldc.upenn.edu/ldc2006t13 [Accessed 7 Oct. 2016].

[12] Zhang, S. (2016). The Pitfalls of Using Google Ngram to Study Language. [online] WIRED. Available at: https://www.wired.com/2015/10/pitfalls-of-studying-language-with-google-ngram/ [Accessed 7 Oct. 2016].