Report for Assignment 1

Data-Intensive Computing SS2024

Lukas Mahler (11908553) Julian Flür (11807481)

Contents

Introduction

In this exercise we are tasked with performing apreprocessing step for a text classification task. We are going to calculate the chi-square value for words in the text corpus. The data we are working on are reviews from various Amazon aritcles, divided in disjoint categories.

Problem Overview

Data set

We are working on a data set of Amazon reviews. While there are 10 attributes, such as **helpful** or a **reviewTime**, we are only interested in two of them. Namely:

- **category**: the category that the product belongs to
- **reviewText**: the content of the review; this is the text to be processed

Each review is in exactly one category and requires no preprocessing.

The review text on the other hand has to be split into unigrams where each token is one word. We split on whitespaces as well as a list of special characters and digits. Also all the text is cast to lower case and certain stopwords are ignored for the analysis.

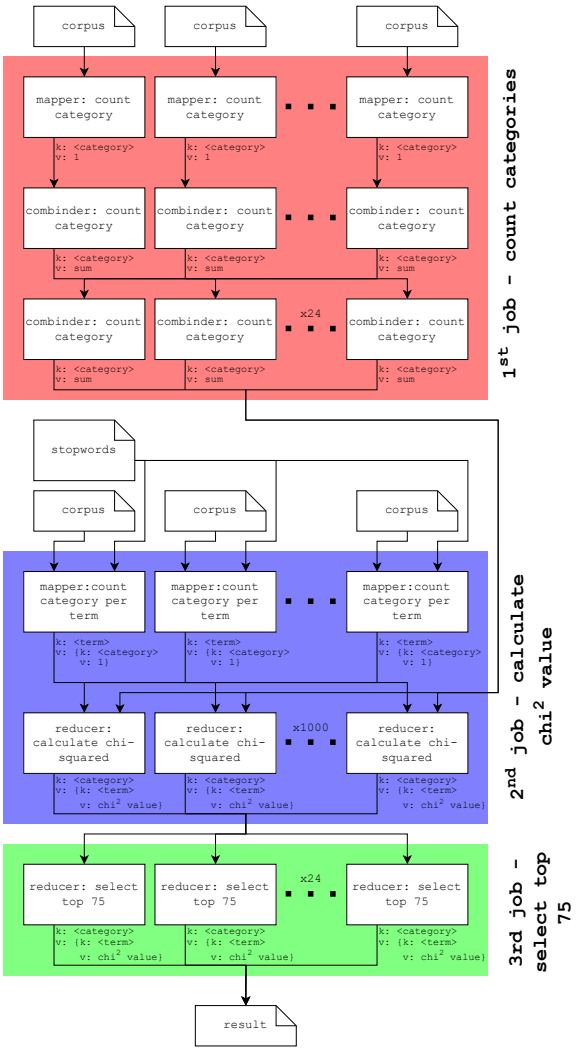
Chi-square value

The chi-square value is a metric, expressing the dependence between a token and a category. Essentially the more often a term is used in a category and the less it is used in reviews from other categories the more important it is.

$$\chi^2_{\text{tc}} = \frac{N(AD - BC)^2}{(A + B)(A + C)(B + D)(C + D)}$$

Methodology and Approach

Essentially we split the problem into two map reduce jobs. In a first step the amount of reviews for each category is calculated, while the second job counts the amount of reviews a term occurs in per category. With the result of the first jobs we are able to calculate the chi-square value for each term and category. As a final step we can select the top 75 terms for each category and return the result.



First MapReduce Job

This map reduce job is very straight forward.

mapper: For each review check the category and append a key-value pair of the form {k: <category>, v: 1} to the result.

combiner: On each worker we can sum up the amount of reviews per category

and return {k: <category>, v: # of reviews}.

reducer: In this step we sum up the amount of reviews over all the workers and the result is {k: <category>, v: # of reviews}.

We specifie 24 reducers since there are only a limited amount of categories and spawning more reducers will cause no benefit.

Second MapReduce Job

The second job is similar to the first, but since a toke can occure in multiple categories we work with dictionaries as values. Adding a combinder for this job did not result in a shorter runtime, thus it is not implemented.

mapper: For each review iterate over the tokens and return {k: <term>, v: {k: <category>, v: 1}}.

reducer: During initialization we read the result of the first map reduce job. Then we sum up the amount of reviews per term and category. With this information we can calculate the chi-square value and return {k: <category>, v: {k: <term>, v: χ^2 value}}

For this job we spawn 1000 reducers. A few tests have shown, that a larger amount of reducers increases the runtime significantly. We did not check for diminishing returns since 1000 reducers gave enough performance.

Third MapReduce Job

Finally we can order the collection of terms for each category and select the top 75 terms. This is implemented as a third map reduce job for a better structure but only a reducer step is implemented.

reducer: For each key (category) we order the results and select the top 75 terms.

Conclusions

The total runtime on the cluster is about 12 minutes, which is below the threshold. We decided on the structure of three map reduce jobs for a simple structure of the interim results. This way the keys are always of the same type and a straight forward pipeline can be used.

Result

Since it is not possible to manually check our results we heuristically check them. For example the top 5 terms for the category “Book” are: author, reading, characters, written and reader. All of these terms are obviously related to books and expected to show up in this list. It is worth noting that the term characters

also shows up in other categories such as “Kindle Store”. Again this does not come as a surprise since the categories are related, and characters being a topic of a review is also not surprising.

Runtime

In the table below the runtime is given.

step	runtime
count review per category	57 sec
calculate chi-square value	8 min 42 sec
select top 75 terms	1 min 58 sec
total	**11 min 37 sec**