Analysis of the common reccommendation systems with the common frameworks: Spark and Flink

Final Report for the BigData project

Mirko Morandi
University of Trento
176043
mirko.morandi@unitn.it

Zhiheng Xu
University of Trento
174222
zhiheng.xu@unin.it

ABSTRACT

In this paper we provide an extensive analysis of the actual state of the art of recommendation systems.
Collaborative Filtering is the current buzzword in the world of recommendations, came to notoriety after the Netflix Prize challenge. In this paper we aim to analyze the current implementations of two different algorithms used for Collaborative Filtering: **ALS** and **Stochastic Gradient Descent** in combination with the common frameworks `Spark` and `Flink`.

Keywords

Flink; Spark; CF; Collaborative Filtering; ALS; SGD; Scala

1. INTRODUCTION

Recommender systems are now trending due to the overwhelming availability of data. These systems have the ability to discover hidden relationships between users and items, and use these patterns to improve the user's taste prediction. Reserachers discovered a "neighbourhood" of users with a similar taste which can be revealed by their previous actions: both implicit and explicit. `Collaborative filtering` is by far the most common approach adapted also by some of the biggest companies in the IT sector such as: **Amazon**, **Facebook** and **Netflix**. Altough it's massive presence in the market `CF` is not the only approach available for a recommender system, but it is actually the successor of `Content-Based filtering`. The latter aims to profile a user searching the correlation with the item's peculiarity. By item peculiarity we refer to its implicit and explicit characteristics, for example a song's genre, subgenre, writer, composer, year of composition, beats per second etc. The problem with this approach lays in the difficult of retrieving all the necessary information, which sometimes are not even available or discloable. Furthermore with the raise of the BigData paradigm some frameworks started to grow from the accademic world to the Apache Foundation: **Flink** and **Spark**. Those frameworks can be seen an extension of the Hadoop ecosystem, and both of them have their own pros and contros which will be briefly analyzed further in this paper.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

2. COLLABORATIVE FILTERING

The paper is structured as follow: description in more details of `Collaborative Filtering` with it's problems, what are the most commong algorithms used with `CF` and a brief introduction to both **Flink** and **Spark**.

2.1 Collaborative Filtering Approaches

Collaborative Filtering can be subsenquently defined in two different approaches:

2.1.1 Memory-based Content Filtering

In `memory-based CF` uses user ratings to compute similarity between user and items and subsequently make a recommendation. Usually this approach involves "neighboring" algorithms such as **K-Nearest Neighbours** to build relationships between users. The similarity between two users is calculated using the **cosine similarity**.
Cosine similarity is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them.[4]

$$\cos(\mathbf{t},\mathbf{e}) = \frac{\mathbf{te}}{\|\mathbf{t}\|\|\mathbf{e}\|} = \frac{\sum_{i=1}^{n} \mathbf{t}_i \mathbf{e}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{t}_i)^2}\sqrt{\sum_{i=1}^{n} (\mathbf{e}_i)^2}} \quad (1)$$

The recommendation is made by finding the top K similar users and aggregate their user-item matrices to find the appropriate recommendation. The typical problem of this approach is the difficult with scaling when the data gets bigger. Due to the `BigData` paradigm expansion this approach has been deprecated favoring the following approach.

2.1.2 Model-based Content Filtering

The most common approach to `CF` is through the factorization of a very big and sparse matrix.[3] For example during the Netflix Prize at the participans were given a matrix of 8.5 billions of ratings, of which only 100 millions were non zero values. *Model-based CF* uses machine-learning and data mining algorithms to uncover the latent factor model between users and items to predict the missing ratings.**latent factor models** are hidden relationships between users and items hardly discoverable in the original data; usually they

may for example denote the quantity of action in a movie or the complexity of the characters.These vectors are then used to create the missing values in the user-items matrix. But what about the error of the prediction? There's an error function better known as **Root Mean Square Error** which is used to calculate the difference between the real and the predicted value.

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(c_i - \bar{c}_i)^2} \quad (2)$$

Furthermore the model-based content filtering can be expanded in two distinct sections: *user and item based content filtering* depending on the priority given to the prediction.

2.2 Related problems

2.2.1 Cold start problem

Due to the nature of CF, the system needs a huge amount of data in order to produce a reliable prediction. But what happens if our system hasn't collected any or not enough information yet? This problem is called **cold start** and can be tackled with some advanced machine learning solutions called *active learning*.

2.2.2 Shilling Attacks

CF can be exploited to perturbate its prediction system with a technique called *shilling attacks*. This happens when the input system (e.g. ratings) are given in the correct way trying to alter the recommendations to the one favoured by the attacker. It has also been noticed that these kind of attacks affect more user-based CF algorithms insted of item-based.[5]

2.2.3 Sparsity

In the era of the *web 2.0* the matrices who compose the datasets are usually very sparse due to the typical proportion for which $nUsers \ll nItems$. As said previously the matrix which was given had only 100M ratings out of 8.5 billions of records. This problem is solved using matrix factorization.

3. MATRIX FACTORIZATION

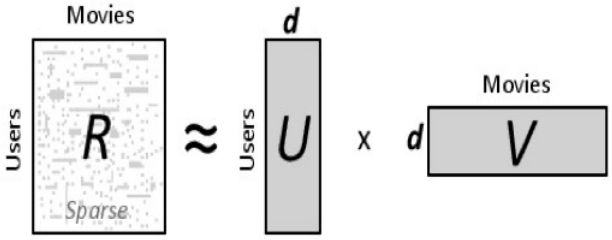
During the last decade a huge effort has been applied to solve the problem of big datasets with an incredible amount of data. Let denote $\boldsymbol{R} = r_{ij}$ denote a user-movie where each element rij represents a rating given by a user i to an item j with a value from 0 to 5; where 0 means non rated and 1 to 5 a rating ranging relatively from *very poor* to *awesome*. Let also define m the number of users and n the number of movies in the system. The problem of *recommender systems* is to predict the missing values of $\boldsymbol{R}$ using the known ratings.

The concept behind *matrix factorization* is to find two matrices $\boldsymbol{V, W}$ relatively $\boldsymbol{m}$ x $\boldsymbol{p}$,$\boldsymbol{n}$ x $\boldsymbol{p}$. which product can approximate a much bigger matrix $\boldsymbol{R}$ of dimensions $\boldsymbol{m}$ x $\boldsymbol{n}$.

$R \approx V * W$

Table 1: Example of a sparse user-item ratings matrix

items/users	U1	U2	U3	U4	U5
I1	5	3	-	1	3
I2	4	-	2	1	-
I3	2	2	-	5	-
I4	4	3	-	4	2
I5	-	5	5	4	5



The process consists in a low-rank approximation of the user-item matrix using for both users and items some feature vectors which are used to model the prediction with a inner vector product of the selected user and item.

3.1 Tables

Because tables cannot be split across pages, the best placement for them is typically the top of the page nearest their initial cite. To ensure this proper "floating" placement of tables, use the environment **table** to enclose the table's contents and the table caption. The contents of the table itself must go in the **tabular** environment, to be aligned properly in rows and columns, with the desired horizontal and vertical rules. Again, detailed instructions on **tabular** material is found in the *LaTeX User's Guide*.

Immediately following this sentence is the point at which Table 1 is included in the input file; compare the placement of the table here with the table in the printed dvi output of this document.

To set a wider table, which takes up the whole width of the page's live area, use the environment **table*** to enclose the table's contents and the table caption. As with a single-column table, this wide table will "float" to a location deemed more desirable. Immediately following this sentence is the point at which Table 2 is included in the input file; again, it is instructive to compare the placement of the table here with the table in the printed dvi output of this document.

3.2 Figures

Like tables, figures cannot be split across pages; the best placement for them is typically the top or the bottom of the page nearest their initial cite. To ensure this proper "floating" placement of figures, use the environment **figure** to enclose the figure and its caption.

This sample document contains examples of **.eps** files to be displayable with LaTeX. If you work with pdfLaTeX, use files in the **.pdf** format. Note that most modern TeX system will convert **.eps** to **.pdf** for you on the fly. More details on each of these is found in the *Author's Guide*.

As was the case with tables, you may want a figure that spans two columns. To do this, and still to ensure proper "floating" placement of tables, use the environment **figure***

Table 2: Some Typical Commands

Command	A Number	Comments
`\alignauthor`	100	Author alignment
`\numberofauthors`	200	Author enumeration
`\table`	300	For tables
`\table*`	400	For wider tables

Figure 1: A sample black and white graphic that has been resized with the `includegraphics` command.

to enclose the figure and its caption. and don't forget to end the environment with figure*, not figure!

3.3 Theorem-like Constructs

Other common constructs that may occur in your article are the forms for logical constructs like theorems, axioms, corollaries and proofs. There are two forms, one produced by the command `\newtheorem` and the other by the command `\newdef`; perhaps the clearest and easiest way to distinguish them is to compare the two in the output of this sample document:

This uses the **theorem** environment, created by the `\newtheorem` command:

THEOREM 1. *Let f be continuous on $[a, b]$. If G is an antiderivative for f on $[a, b]$, then*

$$\int_a^b f(t)dt = G(b) - G(a).$$

The other uses the **definition** environment, created by the `\newdef` command:

Definition 1. If z is irrational, then by e^z we mean the unique number which has logarithm z:

$$\log e^z = z$$

Two lists of constructs that use one of these forms is given in the *Author's Guidelines*.

There is one other similar construct environment, which is already set up for you; i.e. you must *not* use a `\newdef` command to create it: the **proof** environment. Here is a example of its use:

PROOF. Suppose on the contrary there exists a real number L such that

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = L.$$

Then

$$l = \lim_{x\to c} f(x) = \lim_{x\to c} \left[gx \cdot \frac{f(x)}{g(x)} \right] = \lim_{x\to c} g(x) \cdot \lim_{x\to c} \frac{f(x)}{g(x)} = 0 \cdot L = 0,$$

which contradicts our assumption that $l \neq 0$. □

Complete rules about using these environments and using the two different creation commands are in the *Author's Guide*; please consult it for more detailed instructions. If you need to use another construct, not listed therein, which you want to have the same formatting as the Theorem or the Definition[**?**] shown above, use the `\newtheorem` or the `\newdef` command, respectively, to create it.

A *Caveat* for the TEX Expert

Because you have just been given permission to use the `\newdef` command to create a new form, you might think you can use TEX's `\def` to create a new command: *Please refrain from doing this!* Remember that your LATEX source code is primarily intended to create camera-ready copy, but may be converted to other forms – e.g. HTML. If you inadvertently omit some or all of the `\def`s recompilation will be, to say the least, problematic.

4. CONCLUSIONS

This paragraph will end the body of this sample document. Remember that you might still have Acknowledgments or Appendices; brief samples of these follow. There is still the Bibliography to deal with; and we will make a disclaimer about that here: with the exception of the reference to the LATEX book, the citations in this paper are to articles which have nothing to do with the present subject and are used as examples only.

5. ACKNOWLEDGMENTS

This section is optional; it is a location for you to acknowledge grants, funding, editing assistance and what have you. In the present case, for example, the authors would like to thank Gerald Murray of ACM for his help in codifying this *Author's Guide* and the **.cls** and **.tex** files that it describes.

APPENDIX

A. HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are different in the appendices. In the **appendix** environment, the command **section** is used to indicate the start of each Appendix, with alphabetic order designation (i.e. the first is A, the second B, etc.) and a title (if you include one). So, if you need hierarchical structure *within* an Appendix, start with **subsection** as the highest level. Here is an outline of the body of this document in Appendix-appropriate form:

A.1 Introduction

A.2 The Body of the Paper

A.2.1 Type Changes and Special Characters

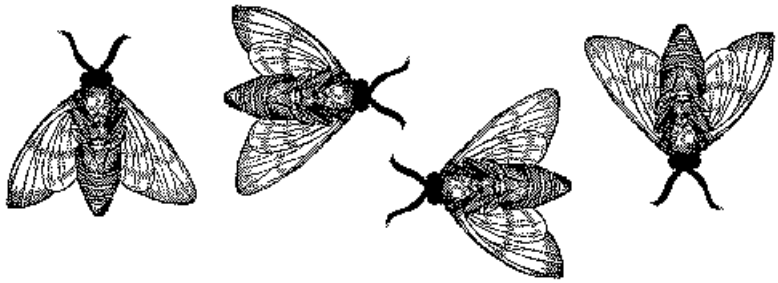


Figure 2: A sample black and white graphic that needs to span two columns of text.

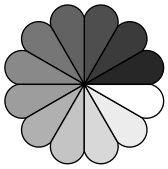


Figure 3: A sample black and white graphic that has been resized with the `includegraphics` command.

A.2.2 Math Equations

Inline (In-text) Equations.

Display Equations.

A.2.3 Citations

A.2.4 Tables

A.2.5 Figures

A.2.6 Theorem-like Constructs

A Caveat for the TEX Expert

A.3 Conclusions

A.4 Acknowledgments

A.5 Additional Authors

This section is inserted by LaTeX; you do not insert it. You just add the names and information in the `\addition-alauthors` command at the start of the document.

A.6 References

Generated by bibtex from your .bib file. Run latex, then bibtex, then latex twice (to resolve references) to create the .bbl file. Insert that .bbl file into the .tex source file and comment out the command `\thebibliography`.

B. MORE HELP FOR THE HARDY

The sig-alternate.cls file itself is chock-full of succinct and helpful comments. If you consider yourself a moderately experienced to expert user of LaTeX, you may find reading it useful but please remember not to change it.