Algorithms and Optimization for Big Data Final Exam Paper Solution

Maunil Vyas(1401007)
SEAS, Ahmedabad University

***Abstract*—This article represents an algorithm to solve the career recommender system problem using a hashing based mechanism. Here two modules have been demonstrated for the recommender system, one in which system reads user's profile and then try to correlate it with similar other users and based on that provide the career path. In the second module, it gives the whole career guidance based on user's career goal selection. The algorithm has been implemented on python 2.7 and tested using proper data set.**

***Index Terms*—Career Recommender System, Hashing, Dictionary, Content Based Filtering, Singular Value Decomposition**

I. Introduction

A. Motivation

Day by day automation is covering all sort of day to day life problem and here on such problem is been demonstrated. The problem that we are looking for is an automated recommended system for career development. Recommended systems are widely adopted by many big software giants companies like Amazon for products, Netflix for entertainment etc and still, they are evolving. They are actually business strategies which support the business back end using the technological advancement.

B. Problem Statement

The problem is been divided into two modules.

- A module that reads user's profile and suggest a career path in terms of skillset – to be acquired.
- A module in which user enters a career goal and based on this career goal and other related information the platform suggest a career path.

The flow is like II represent the mathematical formulation of a problem, III talks about the algorithm and IV will be on Result discussion and complexity analysis.

II. Mathematical Formulation of a Problem

A. Module 1

- The problem can be seen as a large (M $\times$ N) matrix, where M represents the different skills and N represents different career options.(See Fig-1)
- Here the individual column represents the individual personality, the column has a boolean values which indicates whether a person has acquired a certain skill or not.

Manuscript received April 27, 2017

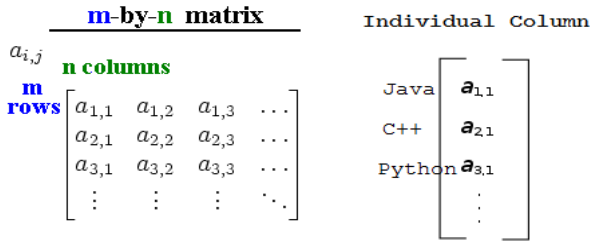

Fig. 1. Matrix form

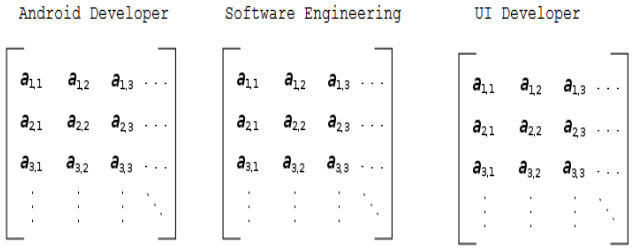

Fig. 2. Sub-Matrices

- With the prior knowledge of the data, we can then subdivide the matrix into specific sub-matrices where sub-matrix represent a specific career option.(See Fig-2)
- Now suppose a new user comes with certain acquired skills means a new column comes to us, so the interesting question that we want to answer here is which sub-matrix is best matching for the new column, In other sense based on the acquired skills which career fits the best option to that individual.

B. Module 2

- Here again, the same matrix analogy can be applied, A new column comes to us and it tells us that it want to be in a specific sub-matrix X so what should be required? In other sense, a new user comes with a set of acquired skills and he or she wants to have a specific career goal so what other things he or she required to fit into that career goal ?

III. Approach

The trivial answers are for the first module find out the correlation between new column and other columns of

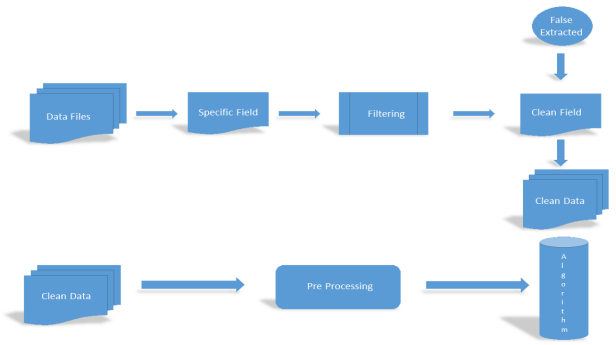

Fig. 3. Data-Cleaning

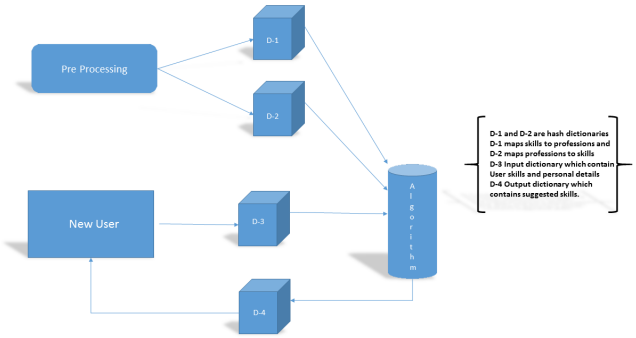

Fig. 4. Algo 1:Hashing Based Approach

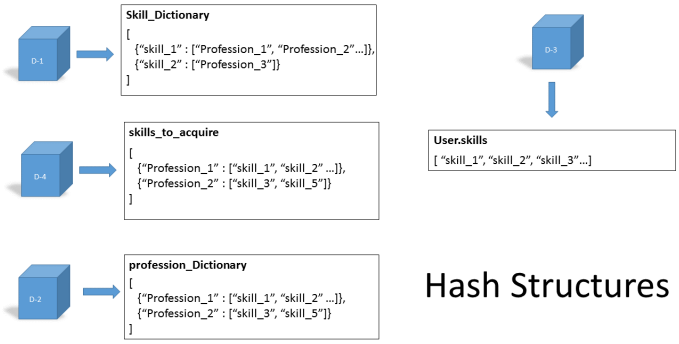

Fig. 5. Algo 1: Hashing Based Approach

different sub matrices and where the correlation is maximum that would be our answer, for the second module based on the career goal we have to fill the column for required skills.

Fig 3 depicts the data cleaning and data preprocessing task.

A. Data Cleaning

- The data has been already labeled so here sub-matrices are already been formed,but the data is not in a proper form and it has too many anomalies, Proper data cleaning is required.
- So as one can see from the figure 3, specific data field has been captured from a data set and it has been passed through a filter which consist of a English language detector and ASCII character detector, certain false extraction is also occured in this filtering process so one has to add those to make sure the proper data have been extracted out.
- Once the data has been cleaned, the preprocessing can be done onto that.

Why do we require pre-processing?
Assume that the sub matrix on which we are interested to work is too large means we have too many data for the same personality and to do the process on this large data set itself is a computationally challenging task the reason is almost all the matrix computations takes $O(n^3)$. So one way to solve the above issue is to reduce the dimensions of a matrix by applying the SVD technique onto a matrix and find out essential skills, also the proposed hash algorithms works on the most common skills which are been noticed in a particular profession so to extract out most common skills from a bunch professionals is itself a heavy computationally task, SVD again would be helpful because now our space becomes smaller.

The proposed pre-processing technique has been tested on the given data where the data matrix was 102 x 25 and after applying SVD it was 100 x 10 and still both leads to almost same top 20 common skills.

B. Algorithm for Module 1

As per the figure 4 one can clearly see that after pre-processing the data, Hash data structure has been created which is stated as D1 and D2 in figure 4.

Algorithm For Module 1

Find_Career_Path_1 (User, Skill_Dict, Prof_Dict)

1. skills_to_acquire := {}; //declaring new empty dictionary
2. list := []; //declaring the empty list
3. j := 0;
4. for_each **skill** in User.skills: //Find out Career paths
5. list[i] := Skill_Dict[**skill**];
6. j := j + 1;
7. end
8.
9. list = remove_duplicate_element(list);
10. Order the list in terms of maximum matching
11. for_each **element** in list: //Suggest additional skills
12. s:=extract skills from Prof_Dict[**element**] not in User.skills;
13. insert_into_dict(skills_to_acquire[**element**],s);
15. end
16. return **skills_to_acquire;**

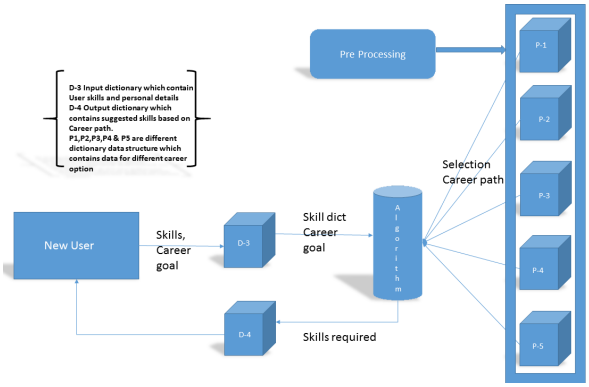

Fig. 6. Algo 2:Hashing Based Approach

C. Algorithm for Module 2

Find_Career_Path_2 (User, user_choice, Prof_dict)
1. skills_to_acquire := []; //declaring new empty list
2. Companies :=[];
3 A_Info :=[]; //Addtional Information
4. list := [];
5. j := 0;
6. s:=extract skills from Prof_Dict[**user_choice**] not in User.skills;
7. insert_into_dict(skills_to_acquire[**user_choice**],s);
   //Inserting skills to dictionary profession wise
8. Companies:=extract c from Prof_Dict[user_choice][c]
9. A_Info := extract a_i from Prof_Dict[user_choice][a_i]
10. return **skills_to_acquire**
11. return **Suggested_Companies**
12. return **A_info**

IV. Complexity Analysis

Let see one by one. Here I am not incorporating the filtering and ASCII checking cost which are associated with data cleaning.

Pre-Processing Cost
1. SVD : $O(N^3)$
2. Finding out the priorities skills : $O(ReducedDimension^1)$
3. Creating Hash : $O(1)$
Overall $O(N^3)$

Cost of Algorithm for Module 1
1. Fetching value from hash or dict: $O(1)$
Z = $(No\_of\_user's\_acquired\_skills)$
2. First Loop in Pseudo code : $O(Z)$
3. Removing Duplicates: $O(No\_of\_career\_selected * Z)$
4. Second Loop in Pseudo code: $O(No\_of\_career\_selected)$
Overall $O(No\_of\_career\_selected * Z)$

Cost of Algorithm for Module 2
1. Fetching value from hash or dict: $O(1)$
Z = $(No\_of\_user's\_acquired\_skills)$
2. Skill Extraction from Hash : $O(1)$
L = $(No\_of\_Common\_skills - Z)$
3. Skills Suggestion: $O(L)$

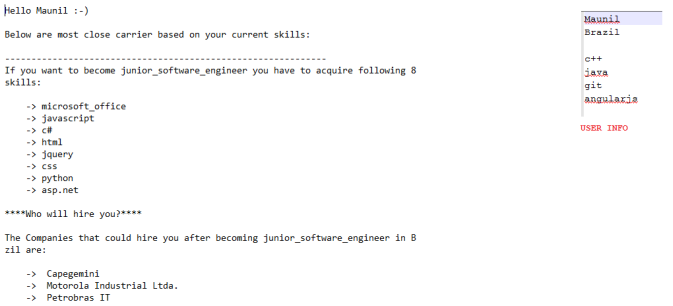

Fig. 7. Output of Module 1

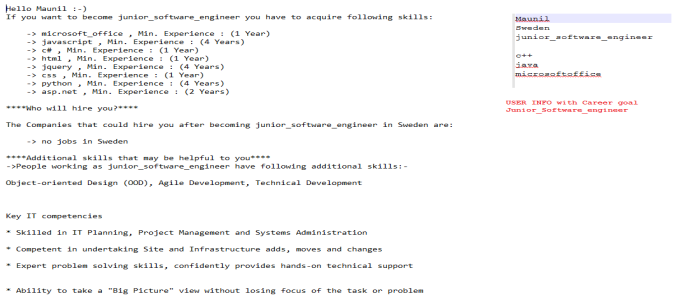

Fig. 8. Output of Module 2

Overall $O(L)$ or $O(Z)$

From the above analysis one can easily find out that pre-processing has a huge impact on the performance of the algorithm, Here almost both modules are govern by the cost of SVD because the running cost of those algorithms are way sort compare to SVD.
One way to appreciate the SVD policy is that the on line version of it, using the online version of SVD we can reduce the pre-processing cost in subsequent iterations.

V. Implemented Result Discussion

Figure 7 and Figure 8 depicts the output of a running code, right now the code has been designed for 4 different career options, for module 1 the code has been designed such a way that it will try to give top most 3 career options which are nearest based on user current expertise and provides suggestions to acquired related skills into that domain it also suggest the companies from user's country. Module 2 will provide skills with minimum years required for the experience, Additional informations, Companies based on user's country

Acknowledgment

I would like to express my special thanks of gratitude to my professor Dr. Ratnik Gandhi who gave me the golden opportunity to do this wonderful work as part of my final exam.

Github_Code:.https://github.com/Maunil/AOBD17_1401007/tree/master/Final_Exam