Career Recommendation System From Similarity of LinkedIn Users' Skills

Nand Parikh
Roll No: 1401023

School of Engineering and Applied Science Ahmedabad University

***Abstract*—Job seeking is a tricky, tedious and time consuming process as people looking for jobs have to collect information from many different sources.In order to automate and simplify this task, Job recommendation systems have been proposed. LinkedIn - the world's largest professional network uses collaborative filtering known as *browsemap* for job suggestions and other related aspects. In this report, an algorithm is proposed for career path recommendation in terms of skill-set by making use of data from LinkedIn user's profiles. This algorithm uses the concepts of transformation matrix and clustering to build the recommendation system. Given a training data of users having a set of skills and job positions, it can suggest a preferable career choice for some artibrary given skill-set.**

Keywords: Transformation Matrix, Clustering, Skill Space, Euclidean Norm

I. Introduction

In this paper, a career recommendation system is presented which is designed for professional platforms like LinkedIn. The basic idea is to discover similarity between different skills and then to find out the corresponding job position with those skills. Clustering is used to partition data into previously unknown group of similar skills. The proposed algorithm is an online version that can continuously learn and adapt to new data. Also, it can be noted that this algorithm doesn't require parameters to be tuned, which makes this approach easy to use and scalable in real life scenarios. The rest of the paper is organized as follows. Section II states the assumptions. Section III describes the detailed approach to the problem. Section IV discusses the algorithm and its efficiency in terms of complexity. In Section V, results are showcased. Finally, Section VI concludes the paper.

II. Assumptions

Following assumptions are made considering the challenge and the depth of the problem:

i.) The training data and the test data is cleaned before given as input to the system.

ii.) Every user in the training set has exactly one job. However, same user can have multiple skills.

iii.) Two users with same skill-set can have different job position.

III. METHOD DESCRIPTION

The very first problem for this kind of dataset is the representation of data. If we consider a set U = u_1,u_2,... of user profiles, each of them describing the skills and job position of that user as strings of data. Then each user profile has a certain no of skills associated with them.Note that the same skill may be associated with more than one profile. If there are total N_s unique skills, then each of this skill can be mapped to a positive integer between 1 to N_s. Similarly, if there are N_j unique jobs, the same mapping can be applied to range of 1 to N_j. However, merely mapping these strings to unique integers doesn't make that much of difference. Hence, we try a different representation.

Every skill can be represented as a vector in 3-dimensional space. Thus, we can form a matrix having the collection of all such skill vectors. Let's call this collection as *"Skill Space"*. We can represent it as the set of all distinct skills denoted as S = s1,s2,.... Similarly, each job can be represented as a vector in 3-dimensional space and we can call the collection of such vectors as *"Job Space"* denoted by J. Now, it's instinctive that 2 skills which are very similar will be located near each other in the respective space compared to 2 skills which are not that similar to each other. For example, skill like "Animation" located at $[10\ 23\ 5]^T$ will have "Graphic Design" located near by at $[10\ 22\ 6]^T$, however "Biology" will be far away from it at $[1\ 2\ 3]^T$ due to less correlation. In general, set of related skills will be much closer to each other forming a cluster in the said space and same will be the case for job space.

Now, we define transformation matrix A(3×3) that maps the vectors in "Skill Space" to possible job vectors in "Job Space".

$$A_{(3\times3)} \times S_{i(3\times1)} = J_{i(3\times1)}$$

Here, A is initialized randomly and all the skill vectors are also randomly initialized. Even though the skill vectors are randomized, after training the skill vector and job vector will re-align to the desired location. Let the no of skills of a particular user be K_i, then those skill vectors are related and must move towards a weighted mean. After the updating of S_i, the J_i is realigned using this weighted mean making sure that the related jobs are clustered together.

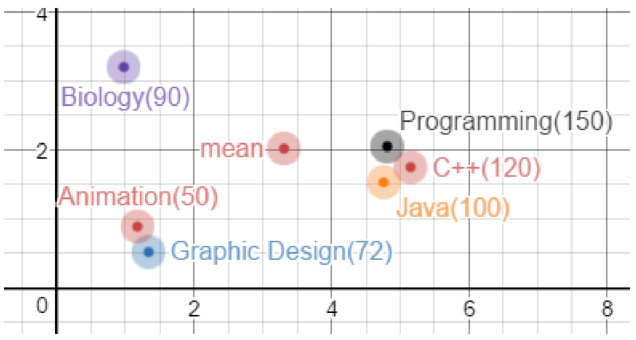


Fig. 1: Skill Space

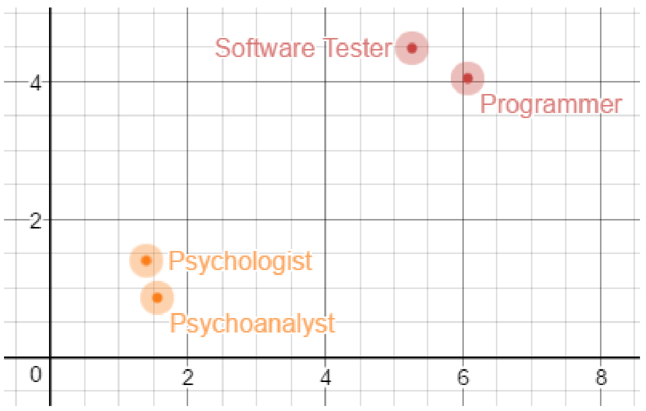


Fig. 2: Job Space

IV. Algorithm

A. Training Pseudo Code

Following is the pseudo code of online version for training and aligning skill vector and job vectors

Algorithm 1 Align skill vectors and job vectors

```
1:  Randomly initialize S
2:  Initialize Scount to 0
3:  Initialize J to 0
4:  Initialize Jcount to 0
5:  Randomly initialize A such that it is invertible
6:  for every user $i$ in training set do
7:      read profile [Ui, UJi]
8:      increment Scount
9:      update all skill vectors
10:     build Si using Ui
11:     Ji = A*Si
12:     if job is new to the program then
13:         J(:,Uji) = mean(Ji);
14:     else
15:         find vector in Ji nearest to J(:,Uji)
16:         update the job vector according to weighted mean
17:     increment Jcount
```

B. Module 1

Following is the pseudo code for module 1

Algorithm 2 Recommend skills based on partial knowledge

```
1: Read Ui
2: for every skill vector Si in Ui do
3:     display two nearest skills in skill space
```

C. Module 2

Following is the pseudo code for module 2

Algorithm 3 Recommend skills based on career goal

```
1: Read Uji
2: xHat = inv(A)*J(:,Uji)
3: find nearest skill x to xHat
4: threshold = 3*norm(x-xHat)
5: for all skills i in skill space do
6:     if norm(S(:,i)-x) is less than threshold then
7:         display S(:,i)
```

D. Efficinecy

The proposed algorithm is online and continuously updates the skill vectors and job vectors with new training data. The time complexity for algorithm 1 that trains data is O(n_j) for each candidate where n_j is the total number of jobs in job space. The primary cost of algorithm 1 lies in finding the nearest job vector from the approximated job vector.
Algorithm 2 reads candidate profile and recommends the related skills for the candidate to pursue. The time complexity of this algorithm is O($k_i n_s$) where k_i is the number of candidate's current skill and n_s ($n_s >> k_i$) is the total number of skills in the skill space. The primary cost of this algorithm lies in finding the nearest skills in skill space.
Algorithm 3 suggests skills according to the candidate's goal. The time complexity of this algorithm is O(n_s) where n_k is the total number of skills in the skill space. The primary cost of this algorithm lies in finding the nearest skills in skill space.

V. Results

Proposed algorithm is implemented in MATLAB and is tested on cleaned LinkedIn's users' profile dataset.

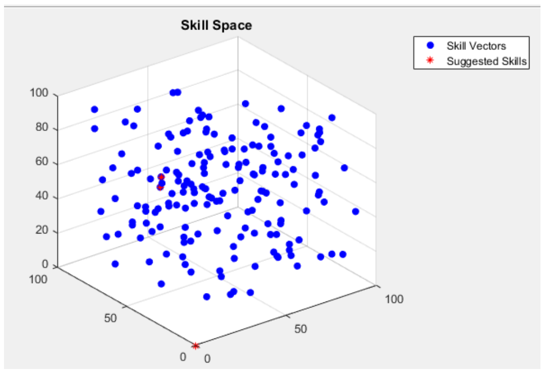
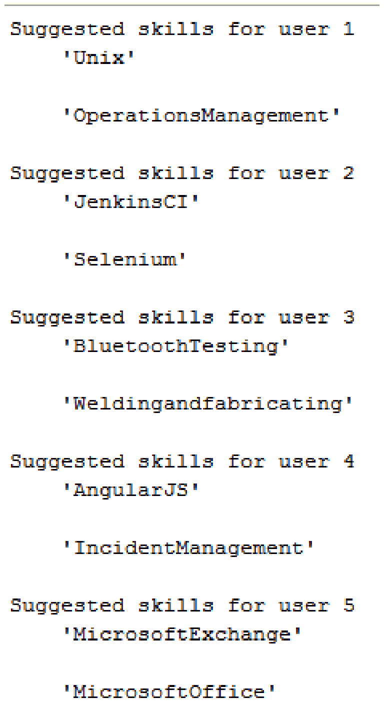
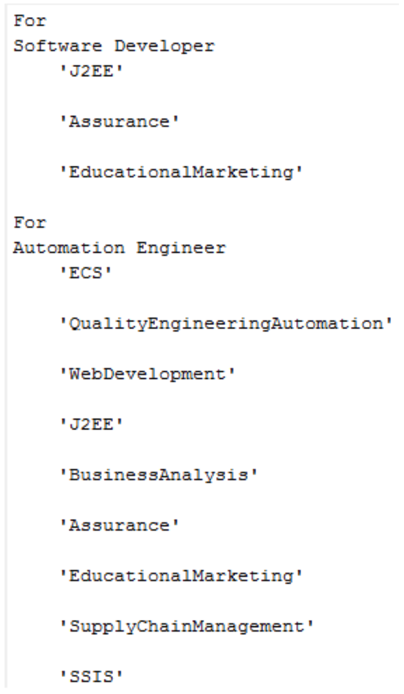


Fig. 3: Scatter plot for Module 1 and 2



```
Suggested skills for user 1
    'Unix'

    'OperationsManagement'

Suggested skills for user 2
    'JenkinsCI'

    'Selenium'

Suggested skills for user 3
    'BluetoothTesting'

    'Weldingandfabricating'

Suggested skills for user 4
    'AngularJS'

    'IncidentManagement'

Suggested skills for user 5
    'MicrosoftExchange'

    'MicrosoftOffice'
```

Fig. 4: Module 1 Output



```
For
Software Developer
    'J2EE'

    'Assurance'

    'EducationalMarketing'

For
Automation Engineer
    'ECS'

    'QualityEngineeringAutomation'

    'WebDevelopment'

    'J2EE'

    'BusinessAnalysis'

    'Assurance'

    'EducationalMarketing'

    'SupplyChainManagement'

    'SSIS'
```

Fig. 5: Module 2 Output

VI. Conclusion

The proposed algorithm finds out the best suitable career path in terms of skill-set based on current user's skills and also based on the user's career goal. Although, the algorithm works fine, some challenges are faced. They are the following:

i.) In algorithm 3, getting the right threshold value for skill recommendation is difficult due to some corner test cases.

ii.) This algorithm does not take into account user's location and his experience in previous jobs and other such parameters.

iii.) Due to the random values chosen, sometimes the algorithm may take more time/training data to converge properly and to give desired output.

References

[1] https://www.researchgate.net/publication/298211329

[2] https://en.wikipedia.org/wiki/Collaborative filtering

[3] https://en.wikipedia.org/wiki/Transformation matrix

[4] https://www.desmos.com/calculator