Market-based Higher Education Program Recommendation

Ana Isabel Neves Alves de Sousa

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Professora Doutora Ana Cristina Costa Aguiar
Second Supervisor: Professora Doutora Luísa Marina Moya Praça de Araújo Lima

July 16, 2016
Market-based Higher Education Program
Recommendation

Ana Isabel Neves Alves de Sousa

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Professora Doutora Ana Paula Cunha da Rocha
External Examiner: Professor Doutor Bruno Emanuel da Graça Martins
Supervisor: Professora Doutora Ana Cristina Costa Aguiar

July 16, 2016
Abstract

The choice of a higher education program is determinant in the future career. It needs to take into account several aspects, such as: future employability; whether the program fits the skills required by the desired job; and whether the program’s scope of depth and breadth fits the needs of the student. It is clear that this is an important decision and that it is worth to invest the time to gather information for improving the final outcome; however, to the best of our knowledge, there are no dedicated tools for this purpose. Since alumni represent the most trusted source of information about the career paths a certain program confers access to, we propose a recommendation system based on alumni data to address this problem.

In this thesis, we aim to find out how to use information about alumni and job offers in order to recommend the most appropriate higher education programs to access a certain job. The recommender system has as a input the user’s desired career and as a output a ranking of higher education programs to take in order to pursue that career. This is achieved by performing a match between the skills required for the career and the skills that an education program confers to attending students.

Since this goal fits into the recommender systems category, we present an overview on the state of the art of this research area, with special focus on collaborative filtering methods, in particular, on top-N recommendation tasks.

We provide a survey of information sources — social networks with business purposes — aiming at identifying the most appropriate data sources for gathering information about alumni and job offers. We then retrieve the information available from the sources selected, analyze and clean it, building a data set with information about alumni, jobs and skills associated to both.

The skills gathered allow the comparison between alumni and jobs through similarity measures. We explore possible measures for assessing the proximity of alumni profiles and the desired careers. Since the measures explored only contemplate binary features, we propose one for weighted features.

We design and implement an user based recommendation prototype system with Computer Science as the target discipline, where we apply decision processes to establish the most suitable system parameters depending on the data set. We then propose an approach to trim the recommendation list based on the similarities between the desired career and its possible recommended programs.

In summary, we achieve our goal by devising a methodology for recommending higher education programs for a specific job. We also provide an evaluation of our system with a group of test users.
Resumo

A escolha de um curso de ensino superior é determinante na carreira futura de uma pessoa. Esta escolha tem de ter em conta vários aspectos, tais como: a futura empregabilidade do curso; se este fornece as competências fundamentais para o trabalho desejado; e se o âmbito do curso se adapta às necessidades do aluno. Assim sendo, torna-se claro que esta decisão é importante e que vale a pena investir tempo na recolha de informações que possam melhorar o resultado final; no entanto, de acordo com o nosso conhecimento, não existem ferramentas com esta finalidade. Tendo em conta que os antigos alunos de um curso representam a fonte mais confiável de informação sobre as saídas profissionais a que ele confere acesso, propomos um sistema de recomendação que tem como base informação acerca de antigos alunos.

Nesta dissertação pretendemos descobrir como usar informação sobre antigos alunos e ofertas de emprego por forma a recomendar os cursos de ensino superior mais adequadas para aceder a um determinado trabalho. O sistema de recomendação tem como dados de entrada a carreira desejada do utilizador e, como dados de saída um ranking de cursos de ensino superior apropriados para essa carreira. Este ranking é realizado através do estabelecimento de uma correspondência entre as competências necessárias para a carreira e as competências que os cursos oferecem aos seus alunos.

Uma vez que este objetivo se encaixa na categoria de sistemas de recomendação, apresentamos uma visão geral sobre o estado da arte desta área de investigação, com especial foco sobre os métodos de filtragem colaborativa, em particular, nas tarefas de recomendação dos N itens de topo.

Fornecemos um levantamento sobre as fontes de informação - redes sociais com fins profissionais - com o objetivo de identificar quais as mais adequadas para extrair informação sobre antigos alunos e ofertas de emprego. Em seguida, recolhemos a informação disponível nas fontes selecionadas, analisamo-la e limpamo-la, construindo assim uma base de dados com informações sobre antigos alunos, empregos e competências associadas a ambos.

As competências recolhidas permitem a comparação entre antigos alunos e empregos através do uso de medidas de similaridade. Assim, exploramos possíveis medidas para avaliar a proximidade entre os alunos e as carreiras desejados. Dado que as medidas exploradas apenas contemplam características binárias, propomos uma medida que contempla características ponderadas.

Projetamos e implementamos um protótipo de sistema de recomendação com Ciência da Computação como disciplina base. Neste protótipo, aplicamos processos de decisão para estabelecer os parâmetros mais adequados para o sistema, que são dependentes da base de dados. Por fim, propomos uma abordagem para podar a lista de recomendações, que tem como base as semelhanças entre a carreira desejada e os diferentes cursos.

Em síntese, alcançamos o nosso objetivo através da elaboração de uma metodologia para a recomendação de cursos de ensino superior para alcançar um trabalho em específico. Adicionalmente, fornecemos uma avaliação do nosso sistema feita por um grupo de utilizadores de teste.
Acknowledgements

A dissertation project might be an individual work, but it surely needs the right backup to achieve successful results. Therefore, I feel that I should recognise everyone that helped me throughout this journey.

Firstly, I would like to thank Instituto de Telecomunicações for providing me the best working conditions to develop this work. Without their support, the challenges faced had not been possible to overcome in the same successful way.

I would also like to thank my supervisors, Ana Aguiar and Luísa Lima, for all the guidance and input throughout the development of this dissertation. Their knowledge and experience were crucial to accomplish the goals established. Thank you for always pushing me to do more and better.

Lastly, a special recognition to my family and close friends for their love, understanding and support during the last months.

Ana Sousa
Contents

1 Introduction .......................................................... 1
  1.1 Motivation ....................................................... 1
  1.2 Goals .......................................................... 2
  1.3 Contributions .................................................. 2
  1.4 Dissertation Structure ...................................... 3

2 Recommender Systems ............................................. 5
  2.1 Content-based Methods ...................................... 6
  2.2 Collaborative Methods ...................................... 7
    2.2.1 Memory-based ........................................... 7
    2.2.2 Model-based ............................................. 9
    2.2.3 Challenges of Collaborative Filtering ............... 10
  2.3 Hybrid Approaches .......................................... 12
  2.4 Evaluation of Recommendation Tasks ...................... 12
    2.4.1 Predictive accuracy metrics ........................... 12
    2.4.2 Classification accuracy metrics ....................... 13
    2.4.3 Rank accuracy metrics .................................. 13
  2.5 Conclusions .................................................. 13

3 Higher Education Program Recommendation .................. 15
  3.1 Methodology .................................................. 17
  3.2 Technologies ................................................ 17

4 Building a Data Set .............................................. 19
  4.1 Social Networks .............................................. 19
    4.1.1 Landing.jobs ............................................ 20
    4.1.2 Stack Overflow Careers ................................ 20
    4.1.3 Upwork .................................................. 20
    4.1.4 Other Platforms ........................................ 20
    4.1.5 Analysis of Available Information .................. 21
  4.2 Data Model ................................................... 21
  4.3 Data Gathering ............................................... 22
  4.4 Data Analysis ............................................... 23
  4.5 Data Cleaning ................................................ 27
    4.5.1 Jobs ...................................................... 28
    4.5.2 Skills .................................................... 28
    4.5.3 Educations ............................................... 29
  4.6 Conclusions .................................................. 30
## CONTENTS

5  **Designing the Solution**  
5.1 User-item Matrix  
5.2 Similarity measure  
5.2.1 Incorporating weights into the similarity measure  
5.3 Choosing the most suitable $k$  
5.3.1 Plot measure values  
5.3.2 Consecutive distances rule  
5.3.3 Moving average window  
5.3.4 Use a minimum value for $k$  
5.3.5 Birthday problem  
5.3.6 Binary search  
5.4 Conclusions  

6  **Proof of Concept System**  
6.1 Analysis  
6.2 Choosing the most suitable $N$  
6.3 Generation of a trimmed recommendation list  
6.3.1 Similarity Computation  
6.4 Conclusions on the value of $k$ and $N$  
6.5 Web based system  
6.5.1 Survey  
6.6 Conclusions  

7  **Conclusions and Future work**  
7.1 Conclusions  
7.2 Contributions  
7.3 Future Work  

References  
A  **Job categories and subcategories**  
B  **Education degrees cleaning details**  
C  **Education locations cleaning details**  
D  **Higher Education Program Recommendation**  
E  **Survey**
List of Figures

3.1 Logic architecture. Full lines represent information that we have. Dashed lines represent relationships we need to establish. A job offer has a job position and a set of skills associated to it. Each alumni has a set of skills and has taken one or more higher education programs. ........................................ 16
3.2 Solution. The input is the user’s desired job. That job has a set of skills required, than can be acquired in a several programs. The output is the ranking of programs recommended to take in order to be closer to achieve that job. ................. 17

4.1 Data model with jobs, alumni, educations, alumni jobs, skills, job categories and subcategories. ................................................................. 22
4.2 Flow diagram presenting the steps taken in order to gather information about jobs and alumni. ................................................................. 23
4.3 Histogram representing the number of skills per job. .......................... 24
4.4 Histogram representing the number of jobs per category. ...................... 25
4.5 Alumni distribution by country. ....................................................... 26
4.6 Histogram of the number of alumni that share a certain education before cleaning data. ................................................................. 26
4.7 Histogram representing the frequency of skills in jobs. ........................ 27
4.8 Flow diagram presenting the steps taken in order to categorize jobs. ......... 28
4.9 Flow diagram presenting the steps taken to clean education degrees. ........ 29
4.10 Flow diagram presenting the steps taken to clean educations locations. ... 30
4.11 Flow diagram presenting the actions performed in step 7 of the locations cleaning. 31
4.12 Histogram of the number of alumni that share a certain education after the data cleaning process. ......................................................... 32
4.13 Histogram of the number of skills per education after the data cleaning process. 33
4.14 Histogram of the number of skills per alumni after the data cleaning process. 33

5.1 Variation of different similarity measures values (color) depending on the number of skills that a job has (x-axis), a alumni has (y-axis) and the number of skills in common between the two (z-axis). ........................................ 39
5.2 Distribution of skill weights in all subcategories. ............................... 40
5.3 Variation of Weighted-Jaccard similarity values (color) depending on the number of skills that a job has (x-axis), a alumni has (y-axis) and the number of skills in common between the two (z-axis). .......................... 40
5.4 Decision process on the value of k. Diamond shape boxes represent decisions and squared boxes represent actions. The blue squared box represents the final decision reached with our data set. ............................ 41
LIST OF FIGURES

5.5 Examples of the variation of Weighted-Jaccard measure for four different subcategories, considering that the values are in descending order.  

5.6 Histogram of the number of alumni with a certain education frequencies depending on \( k \). The frequencies are in a log scale to facilitate the visualization.  

5.7 Box plot of education frequencies depending on \( k \).  

6.1 Number of possible recommendations by subcategory, depending on the value of \( k \).  

6.2 Density of education frequencies depending on the value of \( k \).  

6.3 Distribution of top-5 recommendations with \( k = 5000 \) \( k = 2500 \).  

6.4 Distribution of top-10 recommendations with \( k = 5000 \) and \( k = 2500 \).  

6.5 Distribution of the number of possible recommendations by subcategory, depending on the value of \( N \).  

6.6 Number of possible recommendations by subcategory with \( k = 2500 \) and \( N_{min} = 5 \). Red bars represent numbers greater than 10.  

6.7 Web based system.  

6.8 Answers to the survey question "Do you think this system could help students choosing a higher education program?"  

6.9 Answers to the survey question "How adequate did you consider the areas of expertise used?", where 1 is not adequate at all and 4 very adequate.  

6.10 Answers to the survey question "How useful did you consider the recommendations?", where 1 is not useful at all and 4 very useful.  

6.11 Answers to the survey question "Do you have any suggestions for possible improvements?". From top to bottom, the answers are "Filter recommendations by country or by university.", "Filter recommendations by degree.", "Disable online courses recommendations.", "Only see recommended online courses with more detail.", "Possibility to see profiles of alumni that took each of the recommended programs." and "Other".  

E.1 Survey first page with questions related to the user.  

E.2 Part one of the survey second page with questions related to the web based system.  

E.3 Part two of the survey second page with questions related to the web based system.
List of Tables

2.1 An example of an user-item matrix. ................................. 6

4.1 Data sources summary. Crosses represent data that is present in each API. 21
4.2 Number of database entries before and after the data cleaning. ............... 31

5.1 Extract from the job matrix. ........................................... 36
5.2 Extract from the alumni matrix. ....................................... 37
5.3 Contingency table ......................................................... 37

6.1 Example of the number of alumni that share the programs possible to be recom-
mended in a certain subcategory. ........................................ 51

A.1 Frequency of jobs in each subcategory and association between categories and
subcategories ................................................................. 66
Chapter 1

Introduction

Choosing a higher education program is not an easy decision for multiple reasons. There is a wide variety of higher education programs available, with important differences both in depth and breadth of lectured topics. The future employability of these programs varies; this fact is especially relevant concerning the current lack of detailed employability statistics. When a student enrolls in a degree, he usually does not know the full employability potential of a program.

1.1 Motivation

For some students, it is straightforward to think about a dream job; however, the higher education program to take in order to achieve that goal is not always clear. Furthermore, the skills that an educational program claims to confer its students are not always the ones that the alumni feel to have acquired during the course of the program.

As a consequence, having so many options can represent a real challenge. Knowing that this decision has a large impact in the future career makes it even more important to invest time to make a better-informed decision that takes into account the alumni experience. Since they have attended the program and felt the direct impact of their decision in their professional careers, it is reasonable to consider them as the most trusted source of relevant and real information about some of the possible paths that the program confers access to.

Finally, the talent acquisition process for companies is not a simple process due to several factors. An important practical consequence of the work presented in this thesis could be to present indicators to companies of the best programs to find candidates, as well as the best programs to invest in for internal employee training and formation.
Introduction

1.2 Goals

The main goal of this dissertation is to discover how to use alumni and job offers information to recommend higher education programs that are likely to provide the skills that will allow students to pursue a certain career.

In order to accomplish our goal, we design a recommendation algorithm for higher education programs. Finally, we build a proof of concept system with Computer Science as the target discipline, in order to reduce the scope of recommendations and also to enable manual validation of the proposed system.

The input of our recommendation system is the user’s desired career, and its output is a ranking of possible programs that he could take in order to be closer to his goal. This ranking is computed taking into account skills acquired by a set of alumni, the capabilities that every program provides and the ones required for that specific career.

In order to build the recommender system, the sources of relevant information – social networks designed for business purposes – need to be identified and explored. Through these social networks, we are able to extract the required information about the alumni, such as their skills and education, as well as job offers and skills associated to them. The data collected from the information sources then needs to be analyzed and cleaned.

Different recommendation algorithms and approaches need to be researched and experimented, in order to identify the most suitable approach for the problem in hands. Taking this into consideration, we analyze the influence of the different system parameters on the resulting recommendations, and identify the best option having in consideration our data set and the impact of recommendations on the users of the system.

Finally, due to the subjective nature of a recommendation, we suggest to validate the proposed solution manually by distributing a questionnaire among 59 people.

1.3 Contributions

With this thesis, we aim at contributing with:

- A data set with alumni, jobs and skills associated to them;
- A higher education program recommendation algorithm;
- A metric to evaluate the similarity between items characterized by non-binary features;
- A sensitive analysis on the recommender system parameters;
- A prototype system to evaluate and validate the proposed solution.
1.4 Dissertation Structure

In addition to this introduction, this dissertation contains six other chapters. In Chapter 2, we describe the state of the art on recommender systems and present related work. In Chapter 3, we provide a more detailed description of the system and its goals. In Chapter 4, we describe our data set, namely the sources of information and how data was collected, analyzed and cleaned. In Chapter 5, we provide the details of the proposed algorithm, as well as present the full decision process taken to derive this methodology. In Chapter 6, we elaborate our proof of concept with the support of a web based system that allows user evaluation. Finally, in Chapter 7, we present the conclusions and future work.
Introduction
Chapter 2

Recommender Systems

Nowadays – even though we may not always realize it – recommendations are ubiquitous. In real life, when someone wants to watch a movie, that person will probably ask one of his friends his opinion about it; when making a decision on purchasing a product, the most common method is to search for online reviews. Automated recommender systems are quite similar to these real-life recommendations, with the considerable advantage that we can have access to a much wider range of users who have the same preferences.

Recommender systems have become an important and interesting area both in academia and in industry, with a wide range of work being done in this rapidly expanding area [AT05]. Some of the most well-known recommender systems in industry are Amazon\(^1\) that recommends books, CDs and other products, MovieLens\(^2\) that provides movies recommendations, and Netflix\(^3\) that besides movies also recommends series.

A typical recommender system is characterized by:

- A set of users \(C\);
- An active user \(c\) – the one to whom the recommendation will be provided;
- A set \(S\) that comprises all items that can be recommended.

The main goal of these systems is to maximize the usefulness/utility of a particular item \(s \in S\) to the user \(c\), which is represented by the utility function \(u(c, s)\). This utility is often represented as a rating that expresses the opinion of the user \(c\) towards an item \(s \in S\). Ratings can be either explicit – expressed as a value on a scale, or implicit – expressed as a purchase or as a click.

Ratings given by users can be converted into a user-item matrix. In the user-item matrix in Table 2.1, Maria is the active user, who we want to provide a computed recommendation to. Andrea, John and Maria are users belonging to \(C\), and X-Men, Cinderela, Batman and Sherlock

\(^1\)Available at amazon.com
\(^2\)Available at movielens.org
\(^3\)Available at netflix.com
Recommender Systems

<table>
<thead>
<tr>
<th></th>
<th>X-Men</th>
<th>Cinderela</th>
<th>Batman</th>
<th>Sherlock Holmes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Andrea</td>
<td>Like</td>
<td>Dislike</td>
<td></td>
<td></td>
</tr>
<tr>
<td>John</td>
<td>Like</td>
<td></td>
<td>Like</td>
<td></td>
</tr>
<tr>
<td>Lucas</td>
<td>Like</td>
<td>Like</td>
<td>Dislike</td>
<td></td>
</tr>
<tr>
<td>Maria</td>
<td>Like</td>
<td>Like</td>
<td>?</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.1: An example of an user-item matrix.

Holmes are items in the space of items $S$. For instance, the rating given by Andrea to the item X-Men was a Like, while her rating to the item Cinderela was a Dislike.

In our problem, the active user is the one that requests the recommendation, while items to be recommended are higher education programs. We infer user ratings by using the skills that are common between the job that the active user likes and the ones that each user has.

The main categories for recommender systems nowadays are [AT05]:

- **Content-based**: Recommendations given to the user are based on items that he has liked in the past.
- **Collaborative**: Recommendations given to the user are based on items liked by people who have similar preferences and tastes.
- **Hybrid approaches**: Combination of content-based and collaborative recommender systems.

Since we are interested in providing recommendations based on the users that are similar to the job that the active user likes, our problem belongs to the collaborative category. For the sake of completeness, we also provide a brief overview of content-based and hybrid approaches for recommendation.

### 2.1 Content-based Methods

In content-based methods, recommendations are made by identifying which items are similar to the ones preferred by the active user in the past. More formally, it can be defined as the estimation of the utility $u(c, s)$ of an item $s \in S$ for user $c$, which is based on the utilities $u(c, s_i)$ that $c$ has given to items $s_i \in S$ that are similar to the current item $s$ [AT05].

A similarity measure is a distance measure between two users or items and is used as a weight. The more similar two users $c$ and $c'$ are, the more weight $c'$ ratings will have on making recommendations for user $c$ [AT05]. The same applies for items.

Content-based methods make recommendations by analyzing information and finding patterns within it. To make predictions, they rely on the user and items features that are extracted from textual information [SK09].

Some of the limitations of these methods are: limited content analysis (the features associated with the objects recommended have to be in a format that can be automatically parsed by a computer such as text, otherwise they have to be assigned manually), overspecialization (user $c$ will
Recommender Systems

only be recommended items similar to those he liked in the past) and the new user problem (a user
has to rate a significant number of items before the system understands his preferences). More
details can be found in [AT05].

2.2 Collaborative Methods

Collaborative recommender systems, also known as collaborative filtering systems, make recom-
mendations to a new user based on the known preferences of a group of users similar to him. The
fundamental assumption of such systems is that if two users $c_i$ and $c_j$ rate $n$ items similarly, it is
reasonable to assume that they will also act similarly towards other items [SK09]. More formally,
the utility $u(c, s)$ of item $s \in S$ for the active user $c$ is estimated based on the utilities $u(c_j, s)$
assigned to item $s$ by those users $c_j \in C$ who have similar preferences to $c$ [AT05].

Collaborative methods have the ability to filter any type of content (such as text, artwork or
music) because the recommendation process is based on data from other users, whereas content-
based methods do not have that ability since they rely solely on the history of the user for each type
of content. The main advantage of collaborative methods over content-based methods is perhaps
the fact that they do not depend on error-prone machine analysis of content [HKR00].

In the problem we address, the active user chooses his preferred job, which has an associ-
ated set of skills. That set of skills can be acquired by frequenting certain educational programs.
Thereby, the recommender system finds users that have similar preferences to the active user,
which are expressed in the form of skills. After that, only the most common education programs
among those users are recommended.

According to [SK09], algorithms that perform collaborative filtering fit into one of three main
classes: memory-based, model-based or hybrid, which are described next.

2.2.1 Memory-based

Memory-based algorithms make their predictions based on the entire collection of items already
rated by users. Every user is considered to be part of a group of people with similar interests and
preferences, who are often referred to as the nearest-neighbors of the active user.

A prevalent memory-based collaborative filtering algorithm is neighborhood-based algorithm,
which comprises the following steps [SK09]:

1. Calculate the similarity (or weight) between two users or items;

2. If we want to recommend a ranking of items rather than just a single item, then it is a top-N
   recommendation task and we need to identify the $k$ most similar users or items;

3. Produce a prediction for the active user.

The algorithm steps are described in detail in the next section.
2.2.1.1 Similarity Computation

The similarity computation is a critical step in memory-based collaborative filtering algorithms and can be performed between items or users. In the case of item-based algorithms, only users who have rated both items $s_i \in S$ and $s_j \in S$ are considered. Thereafter, the similarity computation is performed to determine the similarity $w_{i,j}$ between the two items [SK09]. For user-based algorithms, the similarity $w_{u,v}$ is calculated between users $c_u \in C$ and $c_v \in C$ who have both rated the same items [SK09].

There are several methods to compute the similarity between users or items. The application of the correct similarity measures results in a more accurate data analysis. There is not a measure that works well for all implementations, so various have to be experimented in order to identify the one that fits best to the specific problem and context.

[SSSHT10] presents 76 binary similarity and distance measures that were collected and analyzed. Hierarchical clustering was performed to estimate the similarity between measures.

Two of the most used similarity measures are described next; other measures can be found in [SSSHT10].

**Correlation-Based Similarity** – The similarity is measured by computing the Pearson correlation or other correlation-based similarities. Pearson correlation measures the extent to which two variables linearly correlate with each other [RIS+94].

**Vector Cosine-Based Similarity** – The similarity between two items is calculated by treating each item as a vector of ratings and then computing the cosine angle formed by the vectors [SM86].

Since similarity computation is a critical step, there has been active research on improving these measures. [JSB11] presents a probabilistic definition of item similarity. A co-occurrence between two items $s_i \in S$ and $s_j \in S$ is defined as the number of users that have liked both items. Item similarity is defined as the ratio between the actual number of co-occurrences and the number of co-occurrences that would happen if user choices were random. It then applies the user’s usage history of the system to the item similarity matrix. It presents experiences with real-world usage data from different data sets to access the algorithm quality. Its quality was measured against several well-known algorithms and it tied for first place. In [CAL] the accuracy of the Dice similarity measure is improved by processing existing correlations between the characterizing item features.

2.2.1.2 Top-N Recommendations

The purpose of the top-N recommendation algorithm is to recommend a set of N top-ranked items that will be interesting for the active user. It tries to discover relations between different users or items and uses those relationships to compute the recommendations.

Top-N recommendations can also be achieved with model-based collaborative filtering approaches such as association rule mining based models [SK09]. [DK04] proposes a item and model-based algorithm that first, computes the similarity between the items, and then combines these similarities in order to compute the similarity between a set of items and a possible item to recommended.
Recommender Systems

**User-based Top-N Recommendation Algorithms** – Typically, these algorithms comprise the following steps [SK09]:

1. Identify the $k$ users that are the most similar to the active user using a similarity measure;
2. Identify the set of items $U \subset S$ liked by those users and their frequency;
3. Recommend the top-$N$ most frequent items in $U$ that the active user has not rated yet.

User-based top-N recommendation algorithms have some limitations regarding scalability and real-time performance mostly because of the similarity computation. As the number of users grows, the number of comparisons grows too and that can affect the performance of the recommender system.

**Item-based Top-N Recommendation Algorithm** – Item-based algorithms address the scalability problem of the user-based ones. Usually they comprise the following steps [SK09]:

1. Compute the $k$ most similar items for each item in the set of items $S$ using a similarity measure;
2. Identify the set of items $V$ by taking the union of the $k$ most similar items and removing the set of items $U$ that the user already liked;
3. Compute the similarities between items in $V$ and $U$;
4. Sort $V$ by decreasing order of similarity to obtain the list of Top-$N$ recommended items.

We are interested in providing the active user with top-$N$ recommendations rather than a single recommendation, allowing him to choose the one that is most suitable to achieve his desired job.

**2.2.1.3 Prediction and Recommendation Computation**

In this step, the predictions or recommendations are obtained. In the neighborhood-based collaborative filtering algorithm, a subset of the nearest neighbors of the active user is chosen based on the similarities with the user. After that, a weighted aggregate of the neighbors ratings is generated in order to make predictions for the active user [cKBR99].

**2.2.2 Model-based**

Model-based approaches have been developed to overcome some of the shortcomings of memory-based approaches and to achieve better performance. These approaches use the collection of already rated items to learn a model, which can be a data mining or machine learning algorithm that is used to predict item’s rating. Among the most well-known techniques, we can highlight Bayesian belief net models, clustering models and latent semantic models [SK09].
2.2.2.1 Bayesian Belief Net Collaborative Filtering Algorithms

Each item is modelled as a node having states corresponding to the ratings given by users. Then, a network is built on those nodes in such way that each node has a set of parent nodes that are the best predictors of the child’s rating [DK04].

2.2.2.2 Clustering Collaborative Filtering Algorithms

The goal of clustering is to discover the natural grouping of a set of items [Jai10]. Items in the same cluster have a high level of similarity among each other and a low level of similarity with other cluster items.

Clustering methods can be classified into two groups: partitional or hierarchical. Partitional methods produce only one partition of the items, while hierarchical ones produce a nested series of partitions [JMF99].

Since data tends to be sparse because users do not rate every item in $S$, much more accurate predictions can be made by grouping users into clusters and also grouping items into clusters [UF98]. Consequently, clustering is usually used as an intermediate step and its results are used for analysis or classification tasks [SK09]. Clustering Collaborative Filtering can be applied in different ways. For example, [UF98] presents a formal statistical model of collaborative filtering and then compares it against different algorithms for estimating the model parameters. Users and items are divided into clusters; then, link probabilities between users and items in different clusters are established. [CH01] uses clustering, followed by a memory-based collaborative filtering algorithm to make predictions within each created cluster.

In addition, the high complexity computation of the nearest-neighbors can be reduced by performing clustering before and then compute nearest-neighbors search only among the users that belong to the nearest cluster, or to use the cluster centroids [DK04].

2.2.2.3 Latent Semantic Collaborative Filtering Models

Latent Semantic Collaborative Filtering uses a statistical modelling technique that introduces latent class variables in order to discover user communities and interest profiles. Latent variables are variables inferred (using a mathematical model) from variables that are measured. The main advantage of this technique in comparison with memory-based methods is higher accuracy and scalability [SK09].

2.2.3 Challenges of Collaborative Filtering

Like content-based systems, collaborative filter systems also have their own limitations and challenges. The most relevant problems include: new user, new item, grey sheep (users that do not have a consistent opinion towards linking items), shilling attacks (users that give many good ratings to their own materials and bad ratings to others) and scalability (when the number of users and items grows tremendously, computational resources will go beyond acceptable levels). These problems
Recommender Systems

do not need to be addressed in the problem described here due to the static nature of the data. For more details about these problems and how they can be overcome see [AT05] and [SK09].

On the other hand, the issues that can emerge in our recommender system are explained next.

2.2.3.1 Data Sparsity

Frequently, the size of the previously given ratings is very small in comparison with the size of the ratings that need to be predicted. An effective recommender system needs to make good recommendations even when a small number of users or ratings is available.

Several ways to overcome this problem have been researched. For example, [Paz99] uses demographic user data such as gender, age and education, when calculating user similarity. Another approach is to use Singular Value Decomposition [SKKR00], which is a dimensionality reduction technique for sparse rating matrices.

Some model-based collaborative filtering algorithms address the sparsity problem by providing more accurate predictions in such situations. Some of the techniques that address this problem are: association retrieval technique [HCZ04]; maximum margin matrix factorizations [RS05]; and multiple imputation-based collaborative filtering approaches [NKH04].

2.2.3.2 Synonymy

According to [SK09], "synonymy refers to the tendency of a number of the same or very similar items to have different names or entries". This represents a challenge because most recommender systems are unable to identify this relation between different items.

One of the methods developed to solve this problem was Latent Semantic Indexing, which is a Singular Value Decomposition technique that takes a matrix of term-document association data and constructs a semantic space with terms and documents. The distance terms represents its closeness and whether it is likely for them to have the same meaning. However, this technique only solves part of the problem and the other part – words having more than one meaning, remains unsolved [SK09].

2.2.3.3 Explainability

Explaining the recommendations is viewed as an important aspect from the users perspective. Users are more likely to accept a recommendation that has an explanation than one that does not provide it, because the explanation provides more transparency and exposes the reasoning and behind a certain recommendation [HKR00]. Another advantage of providing explanations is that they help identify the likelihood of errors in the recommendation, such as model/process and data errors.

[HKR00] describes experiments performed with the intent of discovering, among other issues, what models and techniques are effective for supporting explanation in a collaborative filtering system, and the conclusion is that histograms of the neighbor’s ratings, past performance and
similarity to other items in the active user’s profile are the most compelling methods to explain the reasoning behind a certain recommendation.

2.3 Hybrid Approaches

A hybrid recommender system can incorporate the advantages of both content-based and collaborative approaches, and, at the same time, prevent some of their disadvantages.

There are four key ways to do this [AT05]:

1. Implement collaborative and content-based methods independently and combine their predictions;
2. Add some content-based characteristics into a collaborative system;
3. Add some collaborative characteristics into a content-based system;
4. Build a model that incorporates both approaches, content-based and collaborative.

In [BS97] it is possible to find an distributed implementation of such a system that recommends items based on the active user preferences and the common preferences. It also explains in detail how disadvantages from both approaches are mitigated through their combination.

2.4 Evaluation of Recommendation Tasks

The quality of a recommender system can be assessed through the use of evaluation metrics. The type of metrics used is critical and dependent on the collaborative filtering application.

[GS09] highlights the importance of the metric used when evaluating different algorithms. The choice of an evaluation metric needs to be made accordingly to the problem domain and the task of interest. Since each metric might favor different algorithms, they define the most appropriate metrics for specific tasks: recommending good items, optimizing utility and predicting ratings. Finally, they empirically show how different metrics can rank two algorithms differently, proving the importance of the metric choice.

[HKTR04] classifies evaluation metrics into the following categories:

- Predictive accuracy metrics;
- Classification accuracy metrics;
- Rank accuracy metrics.

2.4.1 Predictive accuracy metrics

Predictive accuracy metrics measure how close the recommendations are to the true ratings given by users [HKTR04].

Two important predictive accuracy metrics are:
Recommender Systems

- **Mean Absolute Error** – computes the average of the absolute difference between the predictions and true ratings. The lower it is, the better the prediction was.

- **Root Mean Absolute Error** – amplifies the contributions of the absolute errors between true ratings and the predictions made.

It is crucial to always have in mind that the most accurate recommendations might not be the most useful ones having in consideration the task in hands.

### 2.4.2 Classification accuracy metrics

Classification accuracy metrics measure how frequently a recommender makes correct decisions regarding whether an item is a good or bad recommendation [HKTR04].

Two important classification accuracy metrics are:

- **Precision** – represents the probability that a selected item is relevant.
- **Recall** – represents the probability that a relevant item will be selected.

The definition of *relevant* has been a cause of discussion, since from a recommender system point of view it is subjective and entirely dependent on the user [HKTR04].

Other classification metrics such as ROC Curves and Swets’ A Measure can be found with detail in [HKTR04].

### 2.4.3 Rank accuracy metrics

Rank accuracy metrics measure the ability of the recommender system to order items in the same way as if the user had ordered them. These metrics are appropriate for domains where the user’s preferences are non-binary and a ranked recommendation list will be presented to the user [HKTR04].

One important rank accuracy metric is the **Half-life Utility Metric**, which evaluates the utility of a ranked list, having in consideration that the probability of the user viewing an item decreases as its position on the list increases.

Other ranking metrics such as The NDPM Measure can be found in [HKTR04].

### 2.5 Conclusions

After reviewing the literature related to recommender systems, it is possible to identify where the higher education program recommendation system fits: it is a collaborative filtering recommender system, since the recommendations are based on the alumni preferences, which are expressed as their education. Within collaborative filtering algorithms, we pick a memory-based collaborative filtering algorithm, since we are interested in making predictions based on the entire collection of higher education programs. In the final system, the user will be given a list of possible higher
Recommender Systems

education programs to take in order to achieve a certain position, so it will be a top-N recommendation task. Finally, the top-N recommendation will be a user-based one, since we are interested in identifying similar users to the active one based on the job that the active user intends to have, and then recommend the set of programs that are most frequent among those users.
Chapter 3

Higher Education Program Recommendation

The main system goal is to make recommendations of higher education programs. The starting point is the job position/area of expertise desired and the result is a ranking of recommended higher education programs for the user to become a strong candidate for the job. This relationship is represented by the blue dashed line in Figure 3.1, and through job offers and alumni skills, we can establish a match between job positions and higher education programs. To achieve that, it is necessary to infer two other relationships, that are represented by grey dashed lines in Figure 3.1:

1. **Job Offers – Skills**: job offers have a job position and skills required for it, making it possible to have job positions and skills for these jobs. We consider *job offers* as being the job openings available on different social networks and a *job* as the job position contained in those offers.

2. **Higher Education Programs – Skills**: alumni have taken one or more higher education programs and have a set of skills they claim to have acquired, making possible to predict higher education programs having as a foundation the alumni skills.

The initial plan was to have alumni feeding both the higher education programs and job positions using *LinkedIn*\(^1\) data. The user would choose the job position he was interested in, then we would search for people who had that job through *LinkedIn* search and perform clustering by their skills to obtain the groups of people that were the most similar among each other. After that, we would choose the cluster that was most similar to the job position desired using a similarity measure and the recommendations would be based on the users that belong to the cluster chosen.

As shown in *Chapter 4*, that option is not feasible. Consequently, we choose to gather data from different social networks and combine it into our data set:

\(^1\)https://www.linkedin.com/
Figure 3.1: Logic architecture. Full lines represent information that we have. Dashed lines represent relationships we need to establish. A job offer has a job position and a set of skills associated to it. Each alumni has a set of skills and has taken one or more higher education programs.

- through job offers, we collect a set of job positions and skills associated to those jobs;
- through alumni profiles we collect a set of higher education programs and the skills they provide to students.

Through skills, we can establish match job positions and alumni. By doing this, it is possible to infer the real skills that a certain higher education program provides to its students, instead of just skills it claims to provide. Furthermore, using this strategy, we are also able to collect the required skills for a wide range of jobs in Computer Science.

*Figure 3.2* presents a general overview of the solution to the problem of how to use alumni and job offers information to recommend higher education programs that are appropriate to achieve a certain job. It has the following steps:

1. The user chooses a Computer Science related job/career from a closed set of options;
2. The desired career has a set of required skills – we identify these via job offers;
3. The required skills can be obtained by taking one or more higher education programs – alumni information allows us to do this match;
4. We then recommend a set of higher education programs that can provide those skills to the user.
Higher Education Program Recommendation

Figure 3.2: Solution. The input is the user’s desired job. That job has a set of skills required, than can be acquired in a several programs. The output is the ranking of programs recommended to take in order to be closer to achieve that job.

3.1 Methodology

We adopt an iterative methodology by experimenting and observing the results. We pick a strategy and observe the results for that strategy. Then, we make a change or addition, and evaluate whether the results improve. We perform these steps until we are satisfied with the obtained results.

This approach assumes special importance in the context of our recommendation system: since there is no solution that works for every problem, we iteratively search and experiment several approaches in order to determine which is the most suitable methodology for our problem.

3.2 Technologies

We use Ruby\(^2\) language to gather data from the data sources and to implement the recommender system.

For simpler and initial data analysis we use Excel\(^3\). To perform more complex graphical analysis, we move on to R\(^4\) and Python\(^5\). We take advantage of the Plotly\(^6\) and matplotlib\(^7\) Python libraries to draw complex graphs.

\(^2\)https://www.ruby-lang.org/en/
\(^3\)https://products.office.com/excel
\(^4\)https://www.r-project.org/
\(^5\)https://www.python.org/
\(^6\)https://plot.ly/python/
\(^7\)http://matplotlib.org/
To store the data, we choose an relational database, namely *PostgreSQL*\(^8\).  

\(^8\)http://www.postgresql.org/
Chapter 4

Building a Data Set

The data set is one of the most crucial aspects of a recommender system, since it is the source of users and items for generating recommendations. In order to build a data set with jobs, educational programs and the corresponding skills that are suitable as a base for our recommendation system, it is necessary to search for professional social networks that can provide this data.

4.1 Social Networks

A social network can be defined as "a network of social interactions and personal relationships" and "a dedicated website or other application which enables users to communicate with each other by posting information, comments, messages and images" [oxf16].

Social networking can be done for social, business purposes or both. Our focus is on social networks for business purposes. This type of networks allows their users to establish and document networks of people they know and trust professionally. Usually, a person’s profile on one of these networks can be seen as an online version of his curriculum vitae, because it has information regarding his education, work experience, projects he participated in, competences acquired, language skills, awards and publications. In addition, companies also have profiles with detailed information about them and job openings available at the moment. After an extensive search, the following networks were selected for further exploration: ITJobs\(^1\), Jobvite\(^2\), Landing.jobs\(^3\), LinkedIn, Stack Overflow Careers\(^4\), Upwork\(^5\) and Xing\(^6\).

We are interested in gathering information about alumni – their education, job experience and skills, and about job offers and the skills needed for it. In order to extract these data from the social networks, the Application Program Interface (API) that that each social network provides

\(^1\)https://www.itjobs.pt/
\(^2\)http://www.jobvite.com/
\(^3\)https://landing.jobs/
\(^4\)http://careers.stackoverflow.com/
\(^5\)https://www.upwork.com/
\(^6\)https://www.xing.com/
Building a Data Set

is explored to assess which jobs, education and the corresponding skills it is possible to gather. In
the following sections, we describe the explored APIs in detail.

4.1.1 Landing.jobs

*Landing.jobs* defines itself as "a candidate-driven tech jobs marketplace" and has a more corporate
employment focus. It allows the matchmaking between candidates and jobs. Companies can post
job offers and candidates can apply to those offers.

Through the *Landing.jobs* API, it is not possible to search or list candidates, since it only
allows access to the current user data, which does not include his education. However, it offers an
endpoint for job listing where each job listed has title and a set of *tags*, which are core competences
needed for that job.

4.1.2 Stack Overflow Careers

*Stack Overflow Careers* presents itself as a solution for companies to search and find developers
that are suitable for the job offers they have. Candidates are able to prove their knowledge through
the successful resolution of other people’s questions in Stack Overflow\(^7\).

It provides a paid candidate search, where it is possible to filter candidates and access their
personal information such as education, education skills, job experience, job skills and skills ac-
quired. In addition, it also offers a job search functionality using keywords and a location. Each
resulting job has a title and skills required for it.

4.1.3 Upwork

*Upwork* defines itself as "an online workplace for the world – connecting clients with top freelance
professionals" and is focused on freelancer workers. It offers an endpoint to search for freelancers
and another to search for jobs.

Freelancers search was discarded because it did not provide their education and employment
history, even though these details are present in the web view of an user’s profile. On the other
hand, each job returned on the job search has a title, category and subcategory, as well as a set
of skills required or desired for that job. Its category and subcategory represent competence or
expertise areas.

4.1.4 Other Platforms

*ITJobs* is an online platform where companies can post jobs and candidates can apply and get hired
for those jobs. Despite offering an endpoint for job search, each job does not have associated skills.

*Jobvite* has the same purpose as *ITJobs* and it offers an endpoint for candidate search and
one for job listing. The candidate search was discarded because it was not possible to access the

\(^7\)http://stackoverflow.com/
Building a Data Set

<table>
<thead>
<tr>
<th>Alumni and Skills</th>
<th>Jobs and Skills</th>
</tr>
</thead>
<tbody>
<tr>
<td>Landing.jobs</td>
<td>x</td>
</tr>
<tr>
<td>Stack Overflow Careers</td>
<td>x</td>
</tr>
<tr>
<td>Upwork</td>
<td>x</td>
</tr>
</tbody>
</table>

Table 4.1: Data sources summary. Crosses represent data that is present in each API.

candidate’s education or his skills. The job listing was discarded because each job did not have a set of skills associated.

On LinkedIn, people can post professional information about themselves such as, for example, education taken, skills and current and past job positions. LinkedIn only provides access to its people search API via the LinkedIn Partners Program; several applications were made to the partner program describing the project and all of them were rejected.

The use of ITJobs, Jobvite and LinkedIn was discarded since their APIs did not provide the information we are interested in.

Xing is "the largest business network in German-speaking countries". It offers an endpoint to search for people using various filters. Each person found has a set of core competences and an educational background. After experimenting the endpoint and analyzing the data returned, Xing API was discarded because it was only possible to access the education and skills of 15 people, possibly due to the users privacy settings.

4.1.5 Analysis of Available Information

Table 4.1 summarizes the information possible to extract from the business networks selected after exploring and experimenting their APIs. As it is possible to conclude, most of the business networks explored do not provide the information we are interested in, due to privacy settings, limited access or APIs limitations. Nevertheless, we are able to gather information on alumni education and their skills through Stack Overflow Careers and, on job offers and skills required for it through Landing.jobs, Stack Overflow Careers and Upwork.

By collecting skills for both alumni and jobs, we are able to compare them through the set of skills they are associated with. Therefore, in our data model, skills establish the connection and the possibility of matchmaking between alumni and jobs.

4.2 Data Model

For our problem, there are two main perspectives on the data set represented by the model in Figure 4.1: jobs and alumni.

Each job entry represents a job offer and has a title, a category, a subcategory and a set of skills associated. The category and subcategory can be viewed as an ontology, where a category represents a broader area of expertise and its subcategories represent more specific areas within it.
Building a Data Set

This categorization allows the grouping of jobs with similar areas of expertise and, since subcategories are more specific than categories, jobs that belong to the same subcategory can be grouped together.

Each alumni entry represents a result of the Stack Overflow Careers candidate search. It has a location, a profile identifier, a set of skills, a set of jobs and a set of educations. Each alumni education has a degree, a location, as well as a set of skills that represent the competences that the degree provides. Each alumni job has a title and a set of skills that he considers to have acquired in it.

4.3 Data Gathering

The data was gathered from the social networks described in the previous section by following the steps of the flow diagram in Figure 4.3. In step 2, we only add a category and a subcategory to already collected jobs if the job title is a match between Landing.jobs and Upwork. The job search in step 3 is done using titles of already collected jobs, as well as categories names, subcategories names and skills names. The complement of categories and subcategories in step 5, is only performed if the Stack Overflow Careers job title is contained in one of the Upwork search results.

In our system, the active user needs to choose a specific job or career; however, choosing that job from a list with thousands of jobs would not be convenient for the user. Consequently, we group jobs by expertise areas. This grouping could be achieved by performing clustering by
Figure 4.2: Flow diagram presenting the steps taken in order to gather information about jobs and skills associated to each job. However, clustering by skills would probably be difficult because each job has a low number of skills associated with it, as shown in Figure 4.3, making it harder to distinguish different jobs and separating them into clear expertise areas. Since Upwork provides categories and subcategories for its jobs, we decided to take advantage of it. The jobs that are left uncategorized at the end of the data gathering can be classified using the categories already created, by applying rules derived from the data analysis.

4.4 Data Analysis

The immediate step after gathering the data is to perform an exploratory data analysis over the data set to become familiarized with it, understand the data distribution and identify possible rules to apply to clean it.

The performed analysis has four main focus:

- Jobs;
- Alumni;
- Educations;
- Job, education and alumni skills.

To understand the distribution of skills per job, we compute the frequency of skills. It is possible to observe in the histogram in Figure 4.3 that the majority of the jobs has between 1 and 10 skills, the most common number of skills associated to a job is 5 and only a small portion has more than 10 skills. This likely means that companies only specify the core skills needed for the job in each job offer.
In order to identify the available categories and how jobs are distributed among them, we compute the number of jobs in each category, as it is possible to see in Figure 4.4.

The available categories are:

- accounting & consulting;
- administration support;
- customer service;
- data science & analytics;
- design & creative;
- engineering & architecture;
- it & networking;
- legal;
- sales & marketing;
- translation;
- web, mobile & software development;
- writing.
Appendix A presents the associations between categories and subcategories, and the job distribution within subcategories.

Almost 25% of jobs belong to the web, mobile & software development category, which can be viewed as a logical consequence of the target discipline being Computer Science and of the popularity of the web field nowadays. More than half of the jobs do not have a category and subcategory associated, which implies that in the data cleaning phase we focus our effort into categorizing them.

In Figure 4.5 it is possible to see the alumni distribution by country. Most alumni are currently located in North America, India, Canada, Russia or United Kingdom. Having this in consideration, it is expected that the educations locations of our recommendations are focused on these countries.

Regarding educations, we start by analyzing the frequency of each education, which is represented by a degree and a location. The frequency of an education corresponds to the number of alumni that has that educational program associated. There was no education that had a frequency higher than 50 in more than 150000 educations entries available as shown in Figure 4.6, which could hamper the recommendation of educational programs to users.

Since the data set has educations with the same meaning but with different names, we encounter the synonymy problem that was pointed out in Section 2.2.3.2. This is a consequence of the fields for inserting an education degree and location being free text and not an option selected from a closed set, which means that the alumni have freedom to use their preferred nomenclature.

Figure 4.4: Histogram representing the number of jobs per category.
Figure 4.5: Alumni distribution by country.

Figure 4.6: Histogram of the number of alumni that share a certain education before cleaning data.
as well as abbreviations.

By identifying possibilities to standardize education entries with different names but with the same meaning, we could overcome the problem and increase the frequency of educations. In consequence, increasing the frequency of educations would increase the success and appropriateness of the recommendations made. It was established that the standardization of educations names had to be one of the main focus during the data cleaning phase.

With respect to skills, the data analysis was performed in four stages: considering only job skills, only education skills, only alumni skills and considering job and alumni skills.

We compute job skills frequency among jobs in Figure 4.7 and conclude that most job skills only appear associated to a small set of jobs. Job skills come from different social networks and each network has its own closed set of skills or lets users write them freely. As a consequence, we observe the same problem as with educations names, mostly due to different ways of separating words. To solve this problem, we analyze job skills names and identify possible opportunities for names standardization.

Regarding education and candidate skills, we perform the same steps that we did for the analysis of job skills, reaching the same conclusions for both.

4.5 Data Cleaning

The presence of outliers in the data set that feeds the recommender system can lead to poor recommendations. Therefore, we standardize names to avoid dispersion caused by the possibility of free inputs. We identify outliers by manually analyzing data content, or by creating rules that describe the data usual behavior, and removing them from the data set.
4.5.1 Jobs

Considering that the users of the system choose one subcategory as their desired career, the main focus was to make sure that each job had a category and a subcategory associated.

First, we identify and remove categories and subcategories that are not related to our target discipline – Computer Science, such as accounting, customer service, writing, marketing and translation. In parallel with this work, we identify and remove outlier jobs in each valid category by manually analyzing titles that appear to be outside of the context of Computer Science and their associated skills.

Finally, we assign a category and subcategory to the 2102 uncategorized jobs by following the steps of the flow diagram in Figure 4.8. In step 2, we also remove uncategorized jobs that are not related to Computer Science by analyzing their titles and associated skills. The subcategories created in step 7 to categorize jobs that do not fit into any of the existing subcategories are: business intelligence, cloud, devops, infrastructure engineering, security, software architecture and technical management. Apart from finding specific words in job titles and the sequential categorization, all the steps are performed manually because we fell the need to have human control over the categorization, which was only possible due to the number of jobs available.

Regarding jobs and the skills associated to them, we consider that jobs with more than 11 skills do not focus on the core skills for the job position and decided to remove those jobs from the data set. This step removed 71 jobs.

4.5.2 Skills

We cleaned job skills by separating skills that were aggregated. We replace ‘and’, ‘+’ and ‘/’ for the individual skills – for example a skill named ‘html+css’ would be transformed into two separated skills: ‘html’ and ‘css’.

Since some skills had their words separated by white spaces and others by hyphens, we replace every white space occurrence with a hyphen in all skills. We verify the existence of plurals and transform the singular into the plural version. We also verify if there were equal skills that only
differ in the use of hyphens. To conclude the analysis, we perform a manual verification to identify other possibilities to standardize names.

Finally, we compute the frequency of each job skill and remove skills with a frequency equal to or less than one, since those skills would not have a significant impact on the recommendations.

4.5.3 Educations

Education degrees were standardized by following the steps of the flow diagram in Figure 4.9. The process followed can be summarized as: analyze the degrees names, identify possible rules to standardize them, apply those rules and repeat the last steps iteratively until no more rules can be identified.

The specification of an education degree was an open answer to users and they adopted its preferred nomenclature. As a result, apart from step 1, all the steps shown in the flow diagram in Figure 4.9 are performed with the goal of grouping degrees with the same meaning but with different writings. In step 1 we identify degrees that do not correspond to a higher education program, because many users made references to high school programs, conferences and workshops in their educations. This was done by identifying key words in education degrees and then transforming the degree into a simplified version. For instance, if a degree includes ‘high school’ we simplify the degree to just ‘high school’.

Regarding education locations, the line of thought was similar to the one used in educations degrees cleaning and followed the steps in the data flow diagram in Figure 4.10.

Some users specified locations as a faculty and a university, while others only referred the university. In addition, some used commas or spaces to separate the information within the location,
while others did not. Since our main goal is to increase educations frequency, we assume that if the university is equal among two locations, then we can consider the education location as being the same. Additionally, if we find two locations that have the same meaning but only differ on commas or spaces, we keep the one with commas or no spaces since it is more readable in that way. Step 7 of the flow diagram in Figure 4.10 expresses this goal and it is achieved by performing the steps of the flow diagram in Figure 4.11.

### 4.6 Conclusions

The data gathering was a challenging task due to the limitations of the APIs provided by different business networks, either by lacking information or requiring payment. We attempted at mitigating this problem by combining data from different social networks and paying a subscription of Stack Overflow Careers candidate search.

In our opinion, it is important to reflect on the impact of collecting information from different data sources. There was the possibility that the skills that characterized jobs and candidates were written differently. After analyzing and cleaning skills, we analyzed the overlap between job and alumni skills to verify if the skills present in jobs were also present in alumni. If they overlapped significantly it would allow the matching between job and alumni skills. From the 1569 job skills available, 1116 are also present in alumni skills. Therefore, it is reasonable to assume that we are not going to have problems matching job and education skills in the recommender system.
In addition, with the rise of business networks, people have multiple profiles and tend to focus their attention on a few of them. The information available in the other business networks might not be the most recent and accurate one. However, we do not have a way to verify this so we had to assume that the information in the profiles is the most recent and accurate one.

One advantage of this approach is that different sources of information provide data that is not dependable on the source; as such, we achieve more independence by combining information from multiple data sources.

We consider that we were able to overcome the resulting challenges of collecting data from different sources, where users had the liberty to fill fields using their preferred nomenclature, by performing an exhaustive data cleaning on the data. Table 4.2 summarizes the database content before and after the data cleaning, and clearly expresses the decrease of entries in the database after the cleaning process, particularly in the number of degrees and locations.

Finally, we should highlight the final data set distributions. We can compare the number of alumni associated to a certain education before and after cleaning data in Figures 4.6 and 4.12.

<table>
<thead>
<tr>
<th></th>
<th>Before</th>
<th>After</th>
<th>% of leftover data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jobs</td>
<td>4038</td>
<td>3134</td>
<td>78%</td>
</tr>
<tr>
<td>Job skills</td>
<td>2369</td>
<td>1569</td>
<td>66%</td>
</tr>
<tr>
<td>Degrees</td>
<td>57345</td>
<td>38795</td>
<td>68%</td>
</tr>
<tr>
<td>Locations</td>
<td>53411</td>
<td>32691</td>
<td>61%</td>
</tr>
<tr>
<td>Education Skills</td>
<td>15338</td>
<td>13928</td>
<td>90%</td>
</tr>
<tr>
<td>Candidate Skills</td>
<td>19931</td>
<td>18804</td>
<td>94%</td>
</tr>
</tbody>
</table>

Table 4.2: Number of database entries before and after the data cleaning.
Figure 4.12: Histogram of the number of alumni that share a certain education after the data cleaning process.

respectively. It is evident that, by cleaning data, we are able to increase these numbers and the recommender system will benefit from this, as it is going to be possible to make better recommendations.

It is also important to highlight the final distribution of skills per educations and alumni in Figures 4.13 and 4.14, respectively. Most educations have up to 7 skills associated, while most alumni have 5. In addition, the number of entries with more skills decreases as the number of skills increases, which most likely means that only the core skills are specified in most cases.
Building a Data Set

Figure 4.13: Histogram of the number of skills per education after the data cleaning process.

Figure 4.14: Histogram of the number of skills per alumni after the data cleaning process.
Building a Data Set
Chapter 5

Designing the Solution

To design the solution for the problem of how to use alumni and job posts information to recommend higher education programs, we used a top-N recommendation approach as described in Section 2.2.1.2; in particular, within this category of recommendation algorithms, we picked a user-based approach. The overall algorithm is represented in Algorithm 1.

Algorithm 1 User-based top-N algorithm for recommending higher education programs.
1: skill_set ← job_skills ∩ alumni_skills
2: job_matrix ← build_job_matrix(skill_set)
3: alumni_matrix ← build_alumni_matrix(skill_set)
4: while continue do
5: Input: subcategory
6: job_desired ← get_element(job_matrix, subcategory)
7: top_k_users ← find_top_k_most_similar(job_desired, alumni_matrix)
8: top_n_educations ← find_top_n_educations(top_k_users)
9: Output: top_n_educations

The features that characterize jobs and alumni are their skills, creating a categorical feature space. We start by identifying, in line 1, the set of skills that are present both in jobs and alumni. This skill set will be the base for the recommender system, since it will allow the comparison between jobs and alumni.

In lines 2 and 3 we build the user-item matrices for jobs and alumni, named job matrix and alumni matrix respectively, using the skill set created in line 1. More details on the purpose of these matrices and how they were built are provided in Chapter 2 and Section 5.1, respectively.

The user then chooses a subcategory from the ones available and in line 6, the corresponding job row is selected from the job matrix and used in line 7 to find the $k$ most similar users by applying a similarity measure. More details on how $k$ is chosen and what similarity measure was adopted can be found in Sections 5.2 and 5.3, respectively. Finally, in line 8, we take the $k$ most
5.1 User-item Matrix

Regarding jobs, the user-item matrix represents the relationship between each job posting and its associated skills. In that way, we can establish a match between the user-item and the job matrices: users correspond to jobs and items to skills. Each job is represented by a row in the matrix and each skill by a column. If the skill is present in a particular job, then its correspondent column value is 1, otherwise it is 0.

Since we are not interested in particular jobs, but in their broader areas of expertise that are represented by their subcategories, we transform the job matrix into a subcategory matrix. We compute the weight of each skill per subcategory by dividing the frequency of each skill in the jobs that belong to it by the number of jobs in it, as shown in (5.1). This results in a weighted subcategory matrix, where each skill has a value ranging from 0 up to 1, rather than just 0 or 1. If a skill appears in almost all the jobs of a certain subcategory, it will have a weight close to 1. Otherwise, if it only appears in a small set, its weight will be close to 0.

\[
\text{Skill weight in a subcategory} = \frac{\text{frequency of skill in the subcategory jobs}}{\text{number of jobs in the subcategory}} \quad (5.1)
\]

Table 5.1 shows an extract of the resulting subcategory matrix. The first column is the subcategory identifier, the following columns are skills and its weight in each subcategory. For example, subcategory 1 gives a weight of 0.331 to the skill css and 0.297 to the skill html5.

Concerning the alumni matrix, we can match it with the user-item matrix by considering alumni to be users and skills to be items. We only consider alumni that have a set of associated skills and, for each one, we identify his skills. If one skill is associated to a particular alumni, then its correspondent column value is 1, otherwise it is 0.

Table 5.2 shows an extract of the resulting alumni matrix, where the first column is the alumni identifier and skills weights are binary, which means they have either the value 0 or 1.

5.2 Similarity measure

We utilize a similarity measure to assess who are the alumni with the most similar skills to one particular subcategory, by comparing the skills of the subcategory chosen to every alumni. The
main point that similarity measures consider for categorical features and weighted categorical features like the ones that we have in our problem is the count of matches or mismatches between the features of two items. The count of matches and mismatches is represented in Table 5.3.

According to Table 5.3, item 1 is the subcategory and item 2 is one of the alumni. The variable \( a \) corresponds to the number of skills that are present both in the subcategory and in the user, \( b \) to the number of skills only present in the subcategory, \( c \) to the number of skills only present in the alumni and \( d \) the number of skills absent on both.

The similarity measure choice has a large impact on the recommendations made by a recommender system. Because of this, it is important to choose the one that seems to be the most suitable for the problem at hand. Taking our problem into consideration, we establish that the similarity measure should comply with the following requirements:

- give more weight to \( a \) since we are interested in highlighting the skills that are present in both items;
- do not take into account \( d \) because we are more interested in positive matches than in negative ones;
- do not include \( N \), which is the number of skills, because the measure values would increase rapidly.

After analyzing the binary similarity measures presented in [SSSHT10], we came up with a list of the more interesting ones accordingly to our requirements, namely highlighting \( a \) and not taking into account \( d \) or \( N \):

\[
S_{DICE} = \frac{2a}{2a + b + c} \tag{5.2}
\]

\[
S_{SW \text{-} JACCARD} = \frac{3a}{3a + b + c} \tag{5.3}
\]

| alumni_id | css  | html5 | javascript | git  | redis | tdd  | ios-sdk | ...
|-----------|------|-------|------------|------|-------|------|---------|-------
| 1         | 1    | 1     | 1          | 0    | 0     | 0    | 0       | 0     |
| 2         | 0    | 0     | 1          | 1    | 0     | 0    | 1       |       |
| ...       |      |       |            |      |       |      |         |       |

Table 5.2: Extract from the alumni matrix.

<table>
<thead>
<tr>
<th>Item 2</th>
<th>1 (presence)</th>
<th>0 (absence)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item 1</td>
<td>a</td>
<td>b</td>
</tr>
</tbody>
</table>

Table 5.3: Contingency table
Designing the Solution

\[ S_{MCCONNAGHEY} = \frac{a^2 - bc}{(a + b)(a + c)} \]  
(5.4)

\[ S_{SORGENFREI} = \frac{a^2}{(a + b)(a + c)} \]  
(5.5)

To decide which similarity measure is most suitable to the problem, we draw 4d graphs with the following components:

- **x-axis**: number of skills that a subcategory can have;
- **y-axis**: number of skills that an alumni can have;
- **z-axis**: number of skills in common between the last two, it corresponds to \( a \);
- **colour**: similarity value, it varies from 0 to 1 or from \(-1\) to 1 depending on the measure used.

*Figure 5.1* presents the resulting graphs. We analyze the graphs focusing on the color variation, since all other variables are constant between graphs. We were interested in having colors that represent high values of similarity when the number of skills in common is close to the number of skills of the subcategory and the alumni. Apart from this, we want to have a good spectrum of colors, which means that the measure has high variability, adapts well to each specific case and, consequently, is good at discriminating subcategories.

The Sorgenfrei (5.5) measure is not suitable for our case since the color almost does not vary between different points in the graph *Figure 5.1d*. The Dice (5.2), 3w-Jaccard (5.3) and McConnaughey (5.4) measures seem to accomplish our goals regarding color variation and high similarity values. We needed to choose one of these three measures and opted for the 3w-Jaccard (5.3) measure, since it is the one that appears to have a wider range of values while allowing, at the same time, a clear distinction between different points in the graph in *Figure 5.1b.*

**5.2.1 Incorporating weights into the similarity measure**

The similarity measures analyzed in Section 5.2 are used to compute the similarity between items with binary features. In the case of jobs, our values range from 0 to 1, and as such, our features are not binary. Since 3w-Jaccard measure only handles binary features, we had to adapt it in order to incorporate weights, which represent how much a skill is valuable to a certain subcategory.

Instead of (5.3), we propose:

\[ S_{WEIGHTED-JACCARD} = \frac{2a + a'}{3a + b + c} \]  
(5.6)
Designing the Solution

Figure 5.1: Variation of different similarity measures values (color) depending on the number of skills that a job has (x-axis), a alumni has (y-axis) and the number of skills in common between the two (z-axis).

Where \( a' \) is the sum of the weights of subcategory skills that are present both the subcategory and an alumni.

To see how \( a' \) influences the similarity measure, we compute the distribution of weights in each subcategory. Figure 5.2 presents the weight distribution for all subcategories. It is clear that the majority of weights are lower than 0.4, leading to \( a' \) assuming low values in comparison to \( a \).

By using Weighted-Jaccard (5.6) as our similarity measure we are able to emphasize \( a \) and incorporate the weights \( a' \), allowing the skill weights to increase or decrease the similarity value depending on if they are close to 1 or 0. As it is possible to observe in Figure 5.3, with Weighted-Jaccard similarity measure we are able to maintain our goals for the measures regarding color variation and high similarity values.

5.3 Choosing the most suitable \( k \)

In line 7 of Algorithm 1, a value for \( k \) has to be established. It is a threshold for the number of most similar users that will be considered for the next step. The decision tree in Figure 5.4 illustrates the evolution of the process followed to choose the most suitable value for \( k \). The decision reached is dependent on the data set used, namely the number of users and the frequency of each education possible to be recommended. In consequence, the decision process for determining the value of \( k \) should be performed every time the data set is updated.
Designing the Solution

Figure 5.2: Distribution of skill weights in all subcategories.

Figure 5.3: Variation of Weighted-Jaccard similarity values (color) depending on the number of skills that a job has (x-axis), a alumni has (y-axis) and the number of skills in common between the two (z-axis).
Designing the Solution

Figure 5.4: Decision process on the value of k. Diamond shape boxes represent decisions and squared boxes represent actions. The blue squared box represents the final decision reached with our data set.
Designing the Solution

5.3.1 Plot measure values

To get a sense of the similarity values variation depending on \( k \), we plot the values of \( \text{Weighted-Jaccard} \) (5.6) measure by descending order for each of the 40 subcategories.

All subcategories present similar behavior to the one presented in Figure 5.5. It is possible to observe that the derivative of these functions decreases, reaches its minimum and starts increasing after that. We establish that the value of \( k \) should correspond to the first minimum of the derivative in each subcategory, which is the point where the similarity value starts stabilizing.

5.3.2 Consecutive distances rule

The first minimum for each subcategory should be possible to obtain by the following rule:

\[
\text{distance}_{u_2 u_3} > \text{distance}_{u_1 u_2} \quad \text{(5.7)}
\]
Designing the Solution

Where $u_1$, $u_2$ and $u_3$ are three consecutive alumni whose similarities, using the Weighted-Jaccard similarity measure, with a certain subcategory are:

\[\text{similarity}_{u_1 \text{ to subcategory}} \geq \text{similarity}_{u_2 \text{ to subcategory}} \geq \text{similarity}_{u_3 \text{ to subcategory}}\]  \hspace{1cm} (5.8)

We experimented rule in (5.7) but, since we have thousands of users and some have the same value of similarity to a certain subcategory, the value of $k$ was not higher than 10. Since we are looking for the educations that appear more frequently in the $k$ selected alumni, we cannot expect to have be able to give useful recommendations with such a low $k$ value. This has even more importance within our data set, because the number of alumni that have an education in common is low compared with the number of existing educations.

5.3.3 Moving average window

After realizing that it was not viable to use the consecutive distances rule in (5.7), we decided to use a moving average window with size $w$, which considered the average distance between two consecutive elements in the last $w$ elements, transforming the rule in (5.7) into:

\[\text{distance}_{w,w+1} > \text{average} \_ \text{distance} \_ \text{last} \_ w \space \text{elements}\]  \hspace{1cm} (5.9)

When using the rule in (5.9) with a $w$ equal to 10, 20 and 50, we reached the same problem as with rule (5.7). This is due to the fact that educations in our data set are associated to a low number of alumni. In fact, only 843 out of the 106895 educations available have more than 10 alumni associated to them. Consequently, it is difficult to find the same education more than one time in a set with just a few alumni.

5.3.4 Use a minimum value for $k$

The next option was to use the moving average window in (5.9), in addition with a minimum value of users $k_{\text{min}}$ and see how the recommendations were affected by it. We used a $k_{\text{min}}$ equal to 100, 250, 300, 350 and 400 and a window with 10, 20 and 50 elements. We were able to get results that could be considered as valid – education recommendations that appeared in more than one alumni – only with a $k_{\text{min}}$ equal or higher than 250. It was clear that the higher $k_{\text{min}}$ is, more recommendations are available to the user. We also observed that the window had no influence in the resulting recommendations.

5.3.5 Birthday problem

Although we obtained results that could be seen as valid using the previous approach, we decided that, to find the optimal $k_{\text{min}}$, we needed to compute the probability that in a set of $k$ randomly chosen alumni out of a total of $N$ that have a set of skills associated and at least one education, $x$ alumni would have at least one education in common. This an instance of the birthday problem.
Designing the Solution

that concerns the probability that, in a set of n people, two of them have the same birthday\(^1\). This probability applied to our problem would be:

\[
p(k) = 1 - \frac{k! \cdot \binom{93965}{k}}{93965^k}
\]  
\[(5.10)\]

Where '!' is the factorial operator and \(\binom{93965}{k}\) is the binomial coefficient.

The probability in (5.10) would have to be computed for each one of the 106895 educations available and that would not be possible due to computational constraints.

5.3.6 Binary search

Having in consideration that we could not apply solution to the birthday problem, we discovered the most suitable \(k_{\text{min}}\) using a binary search approach. We began with \(k = 40000\) and progressively divided it by two until we reached \(k = 750\). In each iteration, we computed the frequency of each education among the \(k\) chosen alumni and how many frequencies were higher than 1.

We registered, for each \(k\), the number of resulting educations that had a frequency greater than 20, 10 and 5. The resulting graphs are plotted in Figure 5.6, using a log scale for the frequency to allow the display of a wide range of values without compressing the small values. It is possible to conclude that, as \(k\) increases, the educations frequencies decrease almost proportionally. In the box plots in Figure 5.7 it is clear that, regardless the value of \(k\), the number of alumni with common educations registered is mostly equal to 2 or 3. Naturally, with the decrease of \(k\), the appearance of numbers outside of that range of values also decreases.

5.4 Conclusions

Having in consideration all the points discussed above, we decide that we should analyze and explore our proof of concept using \(k = 5000\) and \(k = 2500\), since these two \(k\) values lead to recommendations that have a satisfactory number of educations with frequencies greater than 10 and 5, as shown in Figure 5.6.

We choose these values for \(k\), instead of higher or smaller values due to the distribution of alumni per education, as shown in Figure 4.12. With higher \(k\) values, we would be likely to select alumni that had one of the few educations shared between most alumni. This would cause the recommendations to not be as well adjusted to each subcategory. Since each subcategory has different skill weights characterizing it, the overlap of recommended educations in each subcategory would be high. On the other hand, by using smaller values for \(k\), we would not be able to provide

\(^1\)http://mathworld.wolfram.com/BirthdayProblem.html
Figure 5.6: Histogram of the number of alumni with a certain education frequencies depending on $k$. The frequencies are in a log scale to facilitate the visualization.

(a) All frequencies.

(b) Only considering educations frequencies lower than five.

Figure 5.7: Box plot of education frequencies depending on $k$. 

45
Designing the Solution

useful recommendations, since we would not be likely to select alumni that shared more than one education between them, as it was shown in Section 5.3 when we used $k$ values up to 400.

By choosing the intermediate values for $k$, we are able to balance the trade-off between having a high overlap of recommendations and not having recommendations at all. In the next chapter, we analyze the influence of the two values chosen, $k = 5000$ and $k = 2500$, in the quality of the educational programs recommendations.
Chapter 6

Proof of Concept System

The recommendations of higher education programs using our algorithm are influenced by four parameters: the data set, the similarity measure used, the value of $k$ (the number of most similar users to the subcategory we are going to select) and the value of $N$ (the number of recommendations we are going to provide).

To analyze the influence of the system parameters in the resulting recommendations, we keep the data set and the similarity measure constants, vary the value of $k$ and decide on $N$ accordingly to $k$.

In order to decide what is the most suitable value for $k$ having in consideration our problem and data set, we perform a sensitivity analysis with $k = 2500$ and $k = 5000$ and register the resulting recommendations.

6.1 Analysis

We compute the resulting number of educations shared between at least two alumni for each $k$ and for each subcategory. This value represents how many possible recommendations for each subcategory we are able to provide. The results are shown in Figure 6.1 and it is possible to conclude that, for each subcategory, it is possible to recommend more than 200 different educations with $k = 5000$ and almost 100 with $k = 2500$, which can be considered as satisfactory having in consideration the sparse distribution of educations by alumni.

In Figure 6.2, we present the number of alumni that share one of the educations possible to be recommended in Figure 6.1 and, regardless of $k$, this number is mostly between 3 an 5. Again, the sparse distribution of educations by alumni, makes our goal of having a higher value of alumni sharing one recommended education harder.

Since both $k$ values provide a good number of possible recommendations, we analyze the overlap between the top-$N$ resulting educations, namely top-10 and top-5, to see the influence of
Figure 6.1: Number of possible recommendations by subcategory, depending on the value of \( k \).

(a) All frequencies.
(b) Only considering frequencies greater than two.

Figure 6.2: Density of education frequencies depending on the value of \( k \).
Proof of Concept System

$k$ in the range of programs recommended. We use $N = 10$ and $N = 5$ because they correspond to the maximum and minimum number of recommendations that we aim at providing users.

In Figures 6.3 and 6.4 we present the number of times that each higher education program appears in a recommendation for all subcategories, considering the top-5 and top-10 recommendations, respectively. It is possible to conclude that, using $k = 5000$ the overlap between subcategories recommendations is higher than with $k = 2500$, since we have more different recommended programs with the last one. The reason behind this is that, with higher $k$ values, the more likely we are to select users that have one of the higher education programs with the higher overall frequency among alumni, as it was discussed in Section 5.4.

By using a smaller value for $k$, $k = 2500$, we are able to provide a wider range of higher education programs recommendations. Having a wider range of recommendations implies that these are probably more specific to each subcategory. At the same time, with $k = 2500$ we have at least three alumni, from the 2500 selected, that share one of the programs recommended, as we can observe in Figure 6.5a.
6.2 Choosing the most suitable $N$

In Section 2.4.3 it was pointed out that the probability of the user viewing an item decreases as its position on the list increases.

As we briefly referred in the previous section, we decided that, whenever it was possible, the recommender system should give the user a number of recommendations ranging from 5 up to 10.

The decision on the value of $N$ and the minimum frequency, among the $k$ alumni chosen, for an education to be considered as a valid recommendation is made according to the recommendations obtained.

Using $k = 2500$ we guarantee an overall frequency of each education, among the $k$ alumni chosen, higher than 2 as we can observe in Figure 6.5a, both for $N = 5$ and $N = 10$.

With $N = 5$, we are able to obtain higher values of frequencies for the recommended educations than with $N = 10$, guaranteeing an education frequency among the chosen alumni that is equal to 3 at the worst case and, in average, equal to 3 or 4.

In consequence, we established that the minimum value for $N$ should be $N_{min} = 5$. Since the frequencies of each possible recommendation have a close set of values as we can see in Figure 6.5b, we will have probably more than five recommendations with equal frequencies. To contemplate those cases we decided to establish a maximum of $N_{max} = 10$.

6.3 Generation of a trimmed recommendation list

We established $N_{min} = 5$ and $N_{max} = 10$. However, we encountered several cases similar to the one in Table 6.1. In the case of Table 6.1 we are not able to trim the recommendation list because, if we exclude the recommendations with a frequency smaller than 4, we would only have
Proof of Concept System

<table>
<thead>
<tr>
<th>Education</th>
<th>Number of alumni</th>
</tr>
</thead>
<tbody>
<tr>
<td>Education 1</td>
<td>5</td>
</tr>
<tr>
<td>Education 2</td>
<td>4</td>
</tr>
<tr>
<td>Education 3</td>
<td>4</td>
</tr>
<tr>
<td>Education 4</td>
<td>3</td>
</tr>
<tr>
<td>Education 5</td>
<td>3</td>
</tr>
<tr>
<td>Education 6</td>
<td>3</td>
</tr>
<tr>
<td>Education 7</td>
<td>3</td>
</tr>
<tr>
<td>Education 8</td>
<td>3</td>
</tr>
<tr>
<td>Education 9</td>
<td>3</td>
</tr>
<tr>
<td>Education 10</td>
<td>3</td>
</tr>
<tr>
<td>Education 11</td>
<td>3</td>
</tr>
<tr>
<td>Education 12</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 6.1: Example of the number of alumni that share the programs possible to be recommended in a certain subcategory.

3 recommendations. However, by including frequencies up to 3, we would have more than 10 recommendations, which is our maximum value for $N$.

*Figure 6.6* shows the minimum number of recommendations for each subcategory obtained with $N_{\text{min}} = 5$ and without trimming the recommendation list. It is clear that in more than half of the subcategories we exceed our maximum value for $N$.

In order to trim the recommendation list of subcategories where the list size is greater than 10, we followed the approach in *Algorithm 2* for those subcategories. Our approach takes advantage of the item similarity between skills of subcategories and alumni educations.

*Algorithm 2* Trim the recommendation list of a subcategory with size greater than 10.

1. **Input:** subcategory
2. $original\text{ }\_\text{list} \leftarrow get\text{ }recommendations(subcategory)$
3. $\text{elements}_{\text{to}\text{ }order} \leftarrow select\text{ }_{\text{minimum}_{\text{frequency}}(original\text{ }\_\text{list})}$
4. $\text{most}\text{ }\_\text{similar}_{\text{ordered}} \leftarrow []$
5. **foreach** element **in** elements$_{to\text{ }order}$ **do:**
6. similarity $\leftarrow compute\text{ }similarity(subcategory,\text{ }education\text{ }name)$
7. most$_{\text{similar}_{\text{ordered}}} \leftarrow add\text{ }\_\text{element}(element,\text{ }similarity)$
8. $\text{elements}_{\text{to}\text{ }maintain} \leftarrow original\text{ }\_\text{list} - \text{elements}_{\text{to}\text{ }order}$
9. trimmed$_{\text{recommendation}\_\text{list}} \leftarrow elements\_\text{to}\_\text{maintain} \cup \text{most}\_\text{similar}_{\text{ordered}}$
10. **Output:** trimmed$_{recommendation\_\text{list}}$

In *line 2* we take the subcategory recommended educations that have minimum frequency, which are the ones that make the list size greater than 10. For example, if the recommendations were the ones in *Table 6.1*, we would select the elements from Education 4 until Education 12. Then, for each education selected in *line 2*, we compute its similarity to the current subcategory in *line 6* and store that value in an order list from the most to less similar in *line 7*. Finally, the trimmed recommendation list is equal to the educations not selected in *line 2*, plus the most similar among the selected ones until a maximum of 10 recommendations.
6.3.1 Similarity Computation

In line 6 of Algorithm 2, we compute the similarity between a subcategory and education by performing the steps in Algorithm 3.

**Algorithm 3** Similarity computation between a subcategory and an education.

1: **Input:** subcategory, education_name
2: skill_set ← job_skills ⋂ education_skills
3: job_matrix ← build_job_matrix(skill_set)
4: education_matrix ← build_education_matrix(skill_set)
5: job_desired ← get_element(job_matrix, subcategory)
6: education ← get_element(education_matrix, education_name)
7: similarity ← compute_3w_jaccard_similarity_with_weights(job_desired, education)
8: **Output:** similarity

We start by identifying the set of skills that are present both in jobs and educations in line 2. In lines 3 and 4 we build the user-item matrices for both jobs and educations. More details on how these matrices are built are provided in Section 6.3.1.1. Then, the subcategory row is selected from the job matrix and the education row selected from the education matrix in lines 6 and 7, respectively. Finally, we compute the similarity between the subcategory and education rows using the Weighted-Jaccard measure, with $a'$ being equal to the sum of the product of skills weights present in both.

### 6.3.1.1 User-item matrix

The user-item matrix for jobs is the one explained in Section 5.1.

Regarding the education matrix, we can make the correspondence with the user-item matrix: a user is an education identifier and items are skills. We start by identifying the skills associated
to each education identifier. If one skill is associated to an education identifier, then its value on the matrix is 1, otherwise is 0.

Then, we move on to building a weighted education matrix. Considering educations that have the same degree and same location, we compute the average weight of each skill in that specific education name. We divide the frequency of each skill in the education by the number of educations identifiers that correspond to educations with that name.

Just like the job matrix in Table 5.1, the education matrix is not a binary matrix because the weights of each skill vary from 0 up to 1.

6.4 Conclusions on the value of \( k \) and \( N \)

Having in consideration all the factors analyzed in the previous sections, we decided to use \( k = 2500 \) and have \( 5 \leq N \leq 10 \) as parameters for the higher education programs recommendation. The number of recommendations presented is processed by selecting a minimum of five higher education programs and, in case of equal frequencies, extend the number of recommendations up to ten. If it is not possible to have less than ten recommendations, we trim the recommendation list using an approach based on the similarity between the items. In addition, we only consider educational programs taken by at least 3 alumni from the selected \( k \) as valid recommendations.

It is clear that the frequencies of each recommended higher education program can be considered as low but, having in consideration the sparsity of higher education programs and their overall low frequency as we pointed out in Section 4.3, we can consider the results obtained satisfactory.

6.5 Web based system

Since recommendations are subjective and its appropriateness depends on the users, we built a web based system to allow the evaluation and validation of the higher education program recommendations, as shown in Figure 6.7. The program recommendations provided on the web based system are shown in Appendix D, which were obtained using \( k = 2500 \) and \( 5 \leq N \leq 10 \).

Through the web based system, users can select an area of expertise from the 40 available and see the recommended programs. In the case of Figure 6.7, the area Animation is selected and the recommended programs appear on the right-hand side.

We decided to maintain our focus on recommendations of higher education programs and, at the same time, to allow the recommendation of online courses. The reason for this is that in some subcategories, according to the obtained results, one of the most suitable options might not be a higher education program but rather an online course, which can be viewed as a complementary education program.

6.5.1 Survey

For validation and evaluation purposes, we distributed the survey in Appendix E. Our target audience was mainly students of the last year of the Master in Computer Engineering of our school and
members of some information technology companies. We did this to make sure that the people who answered the survey and evaluated the recommendations had some knowledge in the area because, in that way, they could use their expertise in Computer Science to judge the recommendations and the system. In this survey we provided a link to our web based system, asked users to experiment it and then answer some questions regarding the system to assess their opinion.

6.5.1.1 Population

The survey was answered by a total of 63 people. The population that participated is characterized as:

- 16% women and 84% men;
- 59% are between 18 and 23 years old, 31% are between 24 and 10% are more than 30 years;
- 25% are studying a Computer Science related degree and working at the same time, 50% are only studying and 24% are working in the area.

6.5.1.2 Results

To assess if users consider that programs recommendation can help students choose a higher education program, we asked "Do you think this system could help students choosing a higher education program?". 88.9% answered positively, as shown in Figure 6.8.

In order to figure out if the users felt that the subcategories described different jobs in Computer Science appropriately, we asked "How adequate did you consider the areas of expertise used?". As it is possible to see in Figure 6.9, 87.3% gave a positive answer and, from those, 41.3% classified the areas of expertise as very useful.
Figure 6.8: Answers to the survey question "Do you think this system could help students choosing a higher education program?".

Figure 6.9: Answers to the survey question "How adequate did you consider the areas of expertise used?", where 1 is not adequate at all and 4 very adequate.
To evaluate the users’ opinion towards the suitability of each recommended program, we asked “How useful did you consider the recommendations?”. 52.4% of the users considered the recommendations as being useful and 20.6% as very useful, as it is possible to see in Figure 6.10.

Finally, users could suggest possible improvements either by choosing one of the options provided or by giving their own suggestion. The two most popular answers were: “Filter recommendations by country or by university.” and “Possibility to see profiles of alumni that took each of the recommended programs.”, as shown in Figure 6.11.

6.6 Conclusions

From our results, we consider that the usefulness of this type of systems for helping students choose a higher education program with a specific career in mind is undeniable.

On one hand, we validated the areas of expertise used since most users felt that the areas had the ability to reflect a wide range of possible careers within Computer Science. On the other hand, we evaluated the recommendations given by assessing how useful they were viewed from the users’ point of view.

It is certain that the majority of the answers were positive but, at the same time, 22.2% of users found out the recommendations as being useless. In correlation to the last question of the survey, we can speculate that, by using filters in recommendations and providing some kind of explanation behind them, such as alumni profiles, we could increase the sense of usefulness from the users perspective.

In conclusion, having in consideration the overall opinion towards the system we consider our proof-of-concept validation and evaluation as being successful.
Figure 6.11: Answers to the survey question *"Do you have any suggestions for possible improvements?"*. From top to bottom, the answers are "Filter recommendations by country or by university.", "Filter recommendations by degree.", "Disable online courses recommendations.", "Only see recommended online courses with more detail.", "Possibility to see profiles of alumni that took each of the recommended programs." and "Other".
Proof of Concept System
Chapter 7

Conclusions and Future work

7.1 Conclusions

The main goal of this dissertation was to discover how to use alumni and job offers information to recommend higher education programs appropriate to achieve a certain job.

We accomplished our goal by starting with a literature review on recommender systems, with a special focus on collaborative approaches, in order to identify the most suitable approach to the problem in hands. At the same time, we identified viable information sources, namely social networks with business purposes, explored the information that could be extracted from each one, and ended with the information storage, analysis and cleaning.

We then used that information to implement the actual recommender system with the application of a user based top-N recommendation approach. We performed several experiences to fine-tune the recommender system parameters, namely the similarity measure used, the value of $k$ and $N$.

Finally, since the recommendations are subjective, we evaluated and validated our solution by distributing a graphical interface and a small set of questions to access people opinion towards the system.

In conclusion, we achieved our main goal of discovering how to use the alumni and job offers information to recommend higher education programs. However, it is important to highlight some of the limitations and challenges of our approach and solution. The main challenge was the creation of a solid data set with information about alumni. The access to job offers information was relatively easy but the same did not occur with alumni information, where we encountered several barriers: the APIs did not provide access to the information, the access was dependent on special permissions, users’ privacy settings prevented our access to it or the access was paid. Since we could not have free access to alumni information, we decided to pay a subscription in Stack Overflow Careers and obtain access to the alumni information.

In addition, the data set was the main influence over all the decisions throughout our work, mainly due to the distribution of alumni per education. Consequently, the choice of the most suitable values for the parameters of the recommender system, $k$ and $N$, was challenging. We
Conclusions and Future work

successfully overcame the difficulties by elaborating a full decision process where results vary accordingly to the data set used and, in that way, our process has the ability to adapt to different data sets. The main drawback with this approach is that the value for $k$ has to be decided every time the data set is updated.

Regarding the similarity measure used, we did not encounter in the literature one that was suitable for our problem and, at the same time, incorporated weights into the computation. We surpassed this limitation by using one well known binary similarity measure and incorporating weights into it.

Thereafter, we conclude that the challenges faced were successfully overcome but, at the same, that our approach has some limitations caused by the data set that could disappear if we could have access to a wider range of alumni profiles.

7.2 Contributions

In this thesis we contribute with:

- An analysis on the information possible to gather from multiple social networks with business purposes;
- A data set with alumni, jobs and skills associated to them;
- A set of guidelines to properly clean and standardize jobs and alumni data;
- A higher education program top-N recommendation algorithm;
- An evaluation on the application of different existing similarity measures;
- A metric to evaluate the similarity between items with non-binary features;
- A decision process to decide the value of $k$, which results are dependent on the data set;
- A sensitive analysis on the recommender system parameters;
- An approach to trim the recommendation list accordingly to the value established for $N$;
- A prototype system to evaluate and validate the proposed solution.

7.3 Future Work

Throughout the development of the higher education recommendation system it was possible to identify potential points of future work that could create additional value.

Since our work was a proof of concept with Computer Science as the target discipline, the extent of this recommender system to all knowledge areas would be interesting from the students, universities and companies points of view. However, gathering the data needed would certainly be a much bigger challenge than what it already was in our work.
Conclusions and Future work

The user chooses an area from a closed set of options that was created through categories and subcategories available in Upwork. It would be interesting to use another approach to group jobs, namely clustering, and compare the results with the ones we obtained. In addition, increase the granularity of the area specified could be interesting. Instead of choosing an area of expertise, users could choose a job position and a specific company to work in.

On the other hand, as seen in our survey answers, it would be interesting to explore recommendation filters. For instance, give users the possibility to filter the recommended programs by country or by type of degree. Furthermore, providing a set of alumni profiles that were the foundation of each recommendation could be interesting from the users’ perspective, since they would be able to get a sense of the career paths taken by people that work in a certain area.

In addition, the data set created has tremendous potential regarding future analysis paths for exploration. The alumni behavior could be analyzed in terms of how their education had an impact on personal competences and whether additional programs were required in order to obtain the competences needed for the work environment and what education skills they acquired and never applied in a professional environment.

Regarding a certain job position or area of expertise, through alumni information it would be possible to analyze their current and past job positions to get a sense of what experience has the most positive influence on candidates to a certain position.

Universities could try to adjust their programs by understanding what are the most important skills for their students to acquire in order to be more qualified when it comes to get a certain job.
Conclusions and Future work
References


[CAL]  Susana B Cruz, Ana Aguiar, and Daniel E Lucani. Generation of Trimmed Similarity Lists with Applications in Electronic Program Guides. pages 1–6.


REFERENCES


## Appendix A

### Job categories and subcategories

<table>
<thead>
<tr>
<th>Subcategory ID</th>
<th>Subcategory Name</th>
<th>Category ID</th>
<th>Category Name</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>web &amp; mobile design</td>
<td>1</td>
<td>web, mobile &amp; software dev</td>
<td>72</td>
</tr>
<tr>
<td>2</td>
<td>other - design &amp; creative</td>
<td>2</td>
<td>design &amp; creative</td>
<td>32</td>
</tr>
<tr>
<td>3</td>
<td>graphic design</td>
<td>2</td>
<td>design &amp; creative</td>
<td>48</td>
</tr>
<tr>
<td>4</td>
<td>mobile development</td>
<td>1</td>
<td>web, mobile &amp; software dev</td>
<td>146</td>
</tr>
<tr>
<td>5</td>
<td>web development</td>
<td>1</td>
<td>web, mobile &amp; software dev</td>
<td>471</td>
</tr>
<tr>
<td>6</td>
<td>other - software development</td>
<td>1</td>
<td>web, mobile &amp; software dev</td>
<td>122</td>
</tr>
<tr>
<td>7</td>
<td>human resources</td>
<td>3</td>
<td>accounting &amp; consulting</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>ecommerce development</td>
<td>1</td>
<td>web, mobile &amp; software dev</td>
<td>49</td>
</tr>
<tr>
<td>9</td>
<td>scripts &amp; utilities</td>
<td>1</td>
<td>web, mobile &amp; software dev</td>
<td>11</td>
</tr>
<tr>
<td>10</td>
<td>qa &amp; testing</td>
<td>1</td>
<td>web, mobile &amp; software dev</td>
<td>32</td>
</tr>
<tr>
<td>11</td>
<td>game development</td>
<td>1</td>
<td>web, mobile &amp; software dev</td>
<td>27</td>
</tr>
<tr>
<td>12</td>
<td>database administration</td>
<td>4</td>
<td>it &amp; networking</td>
<td>11</td>
</tr>
<tr>
<td>13</td>
<td>customer service</td>
<td>5</td>
<td>customer service</td>
<td>6</td>
</tr>
<tr>
<td>14</td>
<td>web content</td>
<td>6</td>
<td>writing</td>
<td>15</td>
</tr>
<tr>
<td>15</td>
<td>desktop software development</td>
<td>1</td>
<td>web, mobile &amp; software dev</td>
<td>87</td>
</tr>
<tr>
<td>16</td>
<td>network &amp; system administration</td>
<td>4</td>
<td>it &amp; networking</td>
<td>56</td>
</tr>
<tr>
<td>17</td>
<td>data extraction / etl</td>
<td>7</td>
<td>data science &amp; analytics</td>
<td>4</td>
</tr>
<tr>
<td>18</td>
<td>lead generation</td>
<td>8</td>
<td>sales &amp; marketing</td>
<td>40</td>
</tr>
<tr>
<td>19</td>
<td>marketing strategy</td>
<td>8</td>
<td>sales &amp; marketing</td>
<td>14</td>
</tr>
<tr>
<td>20</td>
<td>other - sales &amp; marketing</td>
<td>8</td>
<td>sales &amp; marketing</td>
<td>33</td>
</tr>
<tr>
<td>21</td>
<td>email &amp; marketing automation</td>
<td>8</td>
<td>sales &amp; marketing</td>
<td>6</td>
</tr>
<tr>
<td>22</td>
<td>technical support</td>
<td>5</td>
<td>customer service</td>
<td>9</td>
</tr>
<tr>
<td>23</td>
<td>data entry</td>
<td>9</td>
<td>admin support</td>
<td>18</td>
</tr>
<tr>
<td>24</td>
<td>smm - social media marketing</td>
<td>8</td>
<td>sales &amp; marketing</td>
<td>22</td>
</tr>
<tr>
<td>25</td>
<td>technical writing</td>
<td>6</td>
<td>writing</td>
<td>10</td>
</tr>
<tr>
<td>26</td>
<td>electrical engineering</td>
<td>10</td>
<td>engineering &amp; architecture</td>
<td>9</td>
</tr>
<tr>
<td>27</td>
<td>other - data science &amp; analytics</td>
<td>7</td>
<td>data science &amp; analytics</td>
<td>24</td>
</tr>
<tr>
<td>28</td>
<td>data mining &amp; management</td>
<td>7</td>
<td>data science &amp; analytics</td>
<td>15</td>
</tr>
<tr>
<td>29</td>
<td>machine learning</td>
<td>7</td>
<td>data science &amp; analytics</td>
<td>11</td>
</tr>
</tbody>
</table>

65
## Job categories and subcategories

<table>
<thead>
<tr>
<th>Subcategory ID</th>
<th>Subcategory Name</th>
<th>Category ID</th>
<th>Category Name</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>other - it &amp; networking</td>
<td>4</td>
<td>it &amp; networking</td>
<td>56</td>
</tr>
<tr>
<td>32</td>
<td>article &amp; blog writing</td>
<td>6</td>
<td>writing</td>
<td>33</td>
</tr>
<tr>
<td>33</td>
<td>other - engineering &amp; architecture</td>
<td>10</td>
<td>engineering &amp; architecture</td>
<td>5</td>
</tr>
<tr>
<td>34</td>
<td>product management</td>
<td>1</td>
<td>web, mobile &amp; software dev</td>
<td>20</td>
</tr>
<tr>
<td>35</td>
<td>project management</td>
<td>9</td>
<td>admin support</td>
<td>37</td>
</tr>
<tr>
<td>36</td>
<td>resumes &amp; cover letters</td>
<td>6</td>
<td>writing</td>
<td>2</td>
</tr>
<tr>
<td>37</td>
<td>sem - search engine marketing</td>
<td>8</td>
<td>sales &amp; marketing</td>
<td>8</td>
</tr>
<tr>
<td>38</td>
<td>web research</td>
<td>9</td>
<td>admin support</td>
<td>16</td>
</tr>
<tr>
<td>39</td>
<td>erp / crm software</td>
<td>4</td>
<td>it &amp; networking</td>
<td>13</td>
</tr>
<tr>
<td>40</td>
<td>other - accounting &amp; consulting</td>
<td>3</td>
<td>accounting &amp; consulting</td>
<td>4</td>
</tr>
<tr>
<td>41</td>
<td>personal / virtual assistant</td>
<td>9</td>
<td>admin support</td>
<td>32</td>
</tr>
<tr>
<td>42</td>
<td>logo design &amp; branding</td>
<td>2</td>
<td>design &amp; creative</td>
<td>16</td>
</tr>
<tr>
<td>43</td>
<td>3d modeling &amp; cad</td>
<td>10</td>
<td>engineering &amp; architecture</td>
<td>3</td>
</tr>
<tr>
<td>44</td>
<td>data visualization</td>
<td>7</td>
<td>data science &amp; analytics</td>
<td>3</td>
</tr>
<tr>
<td>45</td>
<td>animation</td>
<td>2</td>
<td>design &amp; creative</td>
<td>8</td>
</tr>
<tr>
<td>46</td>
<td>photography</td>
<td>2</td>
<td>design &amp; creative</td>
<td>1</td>
</tr>
<tr>
<td>47</td>
<td>presentations</td>
<td>2</td>
<td>design &amp; creative</td>
<td>4</td>
</tr>
<tr>
<td>48</td>
<td>video production</td>
<td>2</td>
<td>design &amp; creative</td>
<td>14</td>
</tr>
<tr>
<td>49</td>
<td>copywriting</td>
<td>6</td>
<td>writing</td>
<td>24</td>
</tr>
<tr>
<td>50</td>
<td>other - writing</td>
<td>6</td>
<td>writing</td>
<td>7</td>
</tr>
<tr>
<td>51</td>
<td>accounting</td>
<td>3</td>
<td>accounting &amp; consulting</td>
<td>8</td>
</tr>
<tr>
<td>52</td>
<td>financial planning</td>
<td>3</td>
<td>accounting &amp; consulting</td>
<td>2</td>
</tr>
<tr>
<td>53</td>
<td>other - customer service</td>
<td>5</td>
<td>customer service</td>
<td>6</td>
</tr>
<tr>
<td>54</td>
<td>display advertising</td>
<td>8</td>
<td>sales &amp; marketing</td>
<td>7</td>
</tr>
<tr>
<td>55</td>
<td>market &amp; customer research</td>
<td>8</td>
<td>sales &amp; marketing</td>
<td>6</td>
</tr>
<tr>
<td>56</td>
<td>public relations</td>
<td>8</td>
<td>sales &amp; marketing</td>
<td>9</td>
</tr>
<tr>
<td>57</td>
<td>seo - search engine optimization</td>
<td>8</td>
<td>sales &amp; marketing</td>
<td>10</td>
</tr>
<tr>
<td>58</td>
<td>academic writing &amp; research</td>
<td>6</td>
<td>writing</td>
<td>8</td>
</tr>
<tr>
<td>59</td>
<td>management consulting</td>
<td>3</td>
<td>accounting &amp; consulting</td>
<td>1</td>
</tr>
<tr>
<td>60</td>
<td>quantitative analysis</td>
<td>7</td>
<td>data science &amp; analytics</td>
<td>3</td>
</tr>
<tr>
<td>61</td>
<td>editing &amp; proofreading</td>
<td>6</td>
<td>writing</td>
<td>10</td>
</tr>
<tr>
<td>62</td>
<td>product design</td>
<td>10</td>
<td>engineering &amp; architecture</td>
<td>10</td>
</tr>
<tr>
<td>63</td>
<td>illustration</td>
<td>2</td>
<td>design &amp; creative</td>
<td>13</td>
</tr>
<tr>
<td>64</td>
<td>telemarketing &amp; telesales</td>
<td>8</td>
<td>sales &amp; marketing</td>
<td>18</td>
</tr>
<tr>
<td>65</td>
<td>other - admin support</td>
<td>9</td>
<td>admin support</td>
<td>13</td>
</tr>
<tr>
<td>66</td>
<td>general translation</td>
<td>11</td>
<td>translation</td>
<td>3</td>
</tr>
<tr>
<td>67</td>
<td>mechanical engineering</td>
<td>10</td>
<td>engineering &amp; architecture</td>
<td>4</td>
</tr>
<tr>
<td>69</td>
<td>creative writing</td>
<td>6</td>
<td>writing</td>
<td>4</td>
</tr>
<tr>
<td>70</td>
<td>audio production</td>
<td>2</td>
<td>design &amp; creative</td>
<td>1</td>
</tr>
<tr>
<td>72</td>
<td>transcription</td>
<td>9</td>
<td>admin support</td>
<td>2</td>
</tr>
<tr>
<td>73</td>
<td>interior design</td>
<td>10</td>
<td>engineering &amp; architecture</td>
<td>4</td>
</tr>
<tr>
<td>75</td>
<td>technical translation</td>
<td>11</td>
<td>translation</td>
<td>2</td>
</tr>
<tr>
<td>77</td>
<td>other - legal</td>
<td>12</td>
<td>legal</td>
<td>2</td>
</tr>
<tr>
<td>79</td>
<td>paralegal services</td>
<td>12</td>
<td>legal</td>
<td>1</td>
</tr>
<tr>
<td>80</td>
<td>legal translation</td>
<td>11</td>
<td>translation</td>
<td>2</td>
</tr>
<tr>
<td>81</td>
<td>grant writing</td>
<td>6</td>
<td>writing</td>
<td>1</td>
</tr>
<tr>
<td>82</td>
<td>voice talent</td>
<td>2</td>
<td>design &amp; creative</td>
<td>1</td>
</tr>
<tr>
<td>84</td>
<td>civil &amp; structural engineering</td>
<td>10</td>
<td>engineering &amp; architecture</td>
<td>1</td>
</tr>
<tr>
<td>-</td>
<td>without subcategory</td>
<td>-</td>
<td>without category</td>
<td>2131</td>
</tr>
</tbody>
</table>

Table A.1: Frequency of jobs in each subcategory and association between categories and subcategories
Appendix B

Education degrees cleaning details

1. Simplify degrees that did not correspond to a higher education:
   Degrees that included 'high school', 'lyceum', 'gymnasium', 'abitur', 'alevel', 'gce',
   'advanced level', 'aslevel', '10th' or '12th' were transformed into 'high school';
   Degrees that included 'coursera' were transformed into 'coursera';
   Degrees that included 'edx' were transformed into 'edx';
   Degrees that included 'udemy' were transformed into 'udemy';
   Degrees that included 'zend' were transformed into 'zend';
   Degrees that included 'mcts', 'mcpd', 'mcp' or 'microsoft' were transformed into 'mi-
   crosoft';
   Degrees that included 'online' were transformed into 'online course';
   Degrees that included 'certificate' were transformed into 'certificate';
   Degrees that included 'ccna', 'google', 'learn', 'workshop', 'camp', 'training', 'trainee',
   'trainer', 'traine' or 'trained' were transformed into 'training';
   Degrees that included 'self', 'certified' or 'conference' were transformed into '-'.

2. Replace expressions and characters:
   Replace '&' by 'and';
   Replace characters with accent marks by the corresponding characters without accent
   marks;
   Replace degrees that were only had numbers by '-';
   Replace square brackets and brackets by parenthesis.

3. Remove characters and words:
   Remove full stops, commas, hyphens, quotation marks, apostrophes, double spaces;
   Remove 'in', 'on', 'of';
Education degrees cleaning details

Remove 'honours’, 'honors’, 'hons’;
Remove 'distinction’, 'first class’, '1st class’;
Remove 'in progress’, 'incomplete’, 'not completed’, 'not finished’, 'unfinished’;
Remove 'degree’, 'attended’, 'combined’;
Remove expressions such as '1 year’ or 'three years’.

4. **Replace plural words by their singular version:**
   - Replace 'computers' by 'computer';
   - Replace 'eletronics' by 'eletronic';
   - Replace 'physics' by 'physic';
   - Replace 'sciences' by 'science';
   - Replace 'technologies' by 'technology';
   - Replace 'telecommunications' by 'telecommunication'.

5. **Simplify expressions:**
   - Simplify 'bachelors’ to 'bachelor’;
   - Simplify 'bachelor of science’ and 'bsc’ to 'bs’;
   - Simplify 'b eng’ to 'be’;
   - Simplify 'b tech’ and 'btech’ to 'bt’;
   - Simplify 'masters’ to 'master’;
   - Simplify 'masters’ to 'master’;
   - Simplify 'master of science’ and 'msc’ to 'ms’;
   - Simplify 'mathematics’ to 'math’;
   - Simplify 'm eng’ to 'me’;
   - Simplify 'ph d’ to 'ph d’;
   - Simplify 'post doc’, 'postdoc’, 'post doctoral’ to 'postdoctoral’;
   - Simplify 'post grad’, 'postgrad’, 'post graduate’ to 'postgraduate’.

6. **Correct typos in frequent words:**
   - Find variations of the words 'engineering’ and 'bachelor’, and correct them.

7. **Manually analyse and transform degrees that started with numbers**

8. **Manually translate degrees written in Asian languages**
Appendix C

Education locations cleaning details

1. Simplify locations that did not correspond to a higher education location:
   Locations that included 'high school', 'lyceum', 'liceo' or 'secundaria' were transformed into 'high school';
   Degrees that included 'coursera' were transformed into 'coursera';
   Degrees that included 'edx' were transformed into 'edx';
   Degrees that included 'udemy' were transformed into 'udemy';
   Degrees that included 'mcts', 'mcpd', 'mcp' or 'microsoft' were transformed into 'microsoft';
   Degrees that included 'self' were transformed into 'self taught'.

2. Replace expressions and characters:
   Replace '&' by 'and';
   Replace characters with accent marks by the corresponding characters without accent marks;
   Replace degrees that were only had numbers by '-';
   Replace hyphens between words by commas.

3. Remove characters:
   Remove full stops, quotation marks, apostrophes, underscores, '@', 'o', 'a' and double spaces;
   Remove spaces before commas.

4. Simplify expressions:
   Simplify 'saint' to 'st';
   Simplify 'technical' to 'tech'.
5. **Correct typos in frequent words:**
   Find variations of the word ‘university’ and correct them.

6. **Manually translate degrees written in Asian languages**

7. **Verify equal locations that only differ on commas:**
   If yes, transform the location without commas to the version with commas because it was more readable;

8. **Verify if one location contained another:**
   If yes, transform the longer one to simpler (smaller) one, which was the most general.

9. **Verify if one location without commas contained another:**
   Follow the rule in previous step.

10. **Verify if there were equal locations without white spaces:**
    If yes, transform the location with white spaces to the version that had;
    This was especially relevant for cases such as ‘u a’ and ‘ua’.
Appendix D

Higher Education Program Recommendation

1: web & mobile design
- Web Development Immersive – General Assembly
- Certificate – Nova Scotia Community College
- Bachelor of Science in Computer Science – University of California
- Computer Science – University of Maryland
- Certificate – British Columbia Institute of Technology
- Bachelor of Science in Computer Science – California State University
- Bachelor of Science in Computer Science – Middlesex University
- Bachelor of Science in Computer Science – Politecnico di Milano
- Bachelor of Technology – Jawaharlal Nehru Technological University
- Full Stack Web Development – App Academy

2: other - design & creative
- Online Course – Codecademy
- Master of Computer Applications – Anna University
- Bachelor of Science in Computer Science – University of California
- Master of Computer Applications – Sikkim Manipal University
- Bachelor Information Technology – Macquarie University
- Bachelor of Science in Computer Science – University of Karachi
- Bachelor of Technology – Anna University
- Bachelor of Engineering in Computer Science and Engineering – Anna University
- Bachelor of Computer Applications – Guru Gobind Singh Indraprastha University
Higher Education Program Recommendation

3: graphic design

Certificate – British Columbia Institute of Technology
Bachelor of Science in Computer Science – University of California
Course – University of Buenos Aires
Computer Information Systems – Devry University
Master of Science – Birla Institute
Bachelor of Engineering – Anna University

4: mobile development

Certificate – Madison College
Computer Science – Rochester Institute of Technology
Certificate – Nova Scotia Community College
Master of Science – Birla Institute
Online Course – Mongodb University
Web Development – Dev Bootcamp
Bachelor of Science in Computer Science – Florida State University
Web Development Immersive – General Assembly
Bachelor of Science in Computer Science – University of Michigan
Bachelor of Science in Computer Science – University of Havana

5: web development

Web Development Immersive – General Assembly
Certificate – Nova Scotia Community College
Certificate – British Columbia Institute of Technology
Bachelor of Science in Computer Science – University of Michigan
Bachelor in Computer Science – Central University
Computer Science – University of Havana
Bachelor of Science in Computer Science – Middlesex University
Online course – Mongodb University
Certificate – Notre Dame University
Online course – Microsoft
Higher Education Program Recommendation

6: other - software development
   Certificate – Nova Scotia Community College
   Bachelor of Technology – Jawaharlal Nehru Technological University
   Master of Science in Software Engineering – San Jose State University
   Bachelor of Science in Computer Science – University of California
   Master of Science in Computer Science – Maharishi University of Management
   Bachelor of Science in Computer Science – University of the Philippines
   Bachelor of Science in Computer Science – University of Michigan
   Web Development Immersive – General Assembly
   Bachelor of Science in Computer Science – University of Bucharest
   Bachelor in Computer Science – Central University

8: ecommerce development
   Bachelor of Science in Computer Science – Central University
   Bachelor of Science in Computer Science – University of California
   Certificate – Nova Scotia Community College
   Web Development Immersive – General Assembly
   Computer Science – University of Texas
   Master of Computer Applications – Anna University
   Master Computer Science – Maharishi University of Management
   Certificate – Code Camp
   Certificate – Stanford University
   Bachelor of Information Technology – Macquarie University

9: scripts & utilities
   Bachelor of Science in Computer Science – University of the Philippines
   Certificate – Nova Scotia Community College
   Web Development Immersive – General Assembly
   Bachelor of Science in Information Technology – Rochester Institute of Technology
   Bachelor of Science in Computer Science – University of California
   Bachelor of Science in Computer Science – University of Michigan
   Bachelor of Science in Computer Science – University of Havana
   Bachelor of Science in Computer Science – Central University
Higher Education Program Recommendation

Bachelor of Science in Computer Science – Middlesex University
Certificate – Stanford University

10: qa & testing
Bachelor of Science in Computer Science – University of Wisconsin
Master of Science in Computer Science – University of Southern California
Bachelor of Science in Computer Science – University of the Philippines
Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – University of Illinois
Master of Science in Computer Science – University of Illinois

11: game development
Master of Science Computer Science – Georgia Institute of Technology
Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – University of Bucharest
Bachelor of Engineering – Gujarat Technological University
Bachelor of Science in Computer Science – University of Michigan
Bachelor of Science in Computer Science – Brigham Young
Bachelor of Science in Computer Science – University of the Philippines
Bachelor of Science in Computer Science – Florida State
Bachelor of Technology – National Institute
Bachelor of Science in Computer Science – University of Havana

12: database administration
Online Course – Mongodb University
Certificate – Nova Scotia Community College
Bachelor of Science in Computer Science – California State University
Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – University of Wisconsin
Bachelor of Science in Computer Science and engineering – Bangladesh University
Bachelor of Science in Computer Science – Oregon State University
Computer Science – University of Texas
Bachelor of Science in Computer Science – Brigham Young University
Computer Science – Rochester Institute of Technology
Higher Education Program Recommendation

16: desktop software development

Master of Science Computer Science – Maharishi University of Management
Certificate – Nova Scotia Community College
Bachelor of Science in Computer Science – University of Texas
Master of Science Computer Science – University of Illinois
Bachelor of Science in Computer Science – Central University
Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – University of Michigan
Bachelor in Computer Science – Central University
Bachelor of Science in Computer Science – University of Bucharest
Bachelor in Computer Science – Universita Degli Studi di Padova

17: network & system administration

Online Course – Mongodb University
Bachelor of Science in Computer Science – Central University
Master of Science Computer Science – University of Illinois
Bachelor of Science in Computer Science – Florida State University
Master of Science in Software Engineering – San Jose State University
Bachelor of Science in Computer Science – University of California
Bachelor of Science in Information Technology – Rochester Institute of Technology
Computer Science – University of Texas
Bachelor of Science in Computer Science – University of Bucharest
Computer Science – Rochester Institute of Technology

18: data extraction / etl

Bachelor of Science in Computer Science – Georgia Institute of Technology
Master in Computer Science – University of Illinois
Bachelor of Engineering – University of Mumbai
Master of Science Computer Science – University of California
Bachelor of Science in Computer Science – University of California
Master of Science Computer Science – New York University
Bachelor of Engineering – University of Mumbai
Bachelor of Engineering in Computer Science – Birla Institute of Technology
Master of Science Computer Science – Georgia Institute of Technology
Higher Education Program Recommendation

24: data entry

Web Development Immersive – General Assembly
Computer Science – University of California
Full Stack Web Development – Code Camp
Bachelor of Science in Computer Science – University of Minnesota
Online Course – Codecademy
Certificate – Code Camp
Bachelor of Science Interactive Multimedia Design – University of Ulster

27: electrical engineering

Master of Science Computer Science – Georgia Institute of Technology
Bachelor of Science in Computer Science – University of Maryland
Bachelor of Science in Computer Science – University of Warsaw
Bachelor of Science in Computer Science – Brigham Young University
Bachelor of Science in Computer Science – University of Massachusetts
Computer Science – Technical University
Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – Georgia Institute of Technology

28: other - data science & analytics

Bachelor of Engineering in Computer Engineering – Mumbai University
Bachelor of Engineering – Anna University
Bachelor of Science in Computer Science – University of Michigan
Bachelor of Science in Computer Science – University of the Philippines
Master of Science Computer Science – Georgia Institute of Technology
Master of Science – University of California

29: data mining & management

Bachelor of Engineering in Computer Engineering – Mumbai University
Master of Science Computer Science – University of Florida
Bachelor Technology – Indian Institute of Technology
Master Computer Science – Technical University
Master of Science Computer Science – New York University
Master of Science Software Engineering – San Jose State University
Higher Education Program Recommendation

Bachelor of Science in Computer Science – University of Maryland
Master of Science Computer Science – University of California
Bachelor of Technology in Computer Engineering – Nirma University
Bachelor of Science in Computer Science – University of Sao Paulo

30: machine learning

Bachelor of Science in Computer Science – University of Sao Paulo
Software Engineering – Hack Reactor
Bachelor of Science in Computer Science – University of California
Master of Science Computer Science – Georgia Institute of Technology
Master of Science Computer Science – University of California
Bachelor Technology – National Institute
Bachelor of Science in Computer Science – University of the Philippines
Online Course – Mongodb University
Certificate – Nova Scotia Community College
Bachelor – University of California

31: other - it & networking

Bachelor of Science in Computer Science – University of the Philippines
Certificate – Nova Scotia Community College
Bachelor of Science in Computer Science – California State University
Bachelor of Science in Computer Science – University of Texas
Master of Science Computer Science – University of Illinois
Web Development – Coding Dojo
Master of Science Computer Science – Maharishi University of Management
Bachelor Computer Science – Central University
Bachelor of Science in Computer Science – University of Michigan
Bachelor of Science in Computer Science – Universidade Catolica

33: other - engineering & architecture

Bachelor of Technology – National Institute
Master of Science Computer Science – Georgia Institute of Technology
Bachelor Technology – National Institute
Bachelor of Science in Computer Science – University of Illinois
Higher Education Program Recommendation

Bachelor of Science in Computer Science – University of California
Bachelor of Engineering – Visvesvaraya Technological University
Bachelor of Science in Computer Science – University of Bucharest
Master of Science Computer Science – Epitech
Bachelor of Science in Computer Science – Georgia Institute of Technology
Computer Science – University of California

34: product management

Bachelor of Science in Computer Science – University of California
Online Course – Mongodb University
Certificate – Nova Scotia Community College
Web Development – Coding Dojo
Web Development Immersive – General Assembly
Bachelor of Technology – West Bengal University of Technology
Mastery Certificate – University of California, Berkeley
Certificate – British Columbia Institute of Technology
Bachelor of Engineering – Anna University
Bachelor of Arts – University of California

35: project management

Certificate – British Columbia Institute of Technology
Bachelor of Technology – Anna University
Master of Computer Applications – Anna University
Bachelor of Technology in Computer Science – Kerala University
Bachelor of Computer Applications – Ignou

38: web research

Bachelor of Science in Computer Science – Warsaw University
Certificate – South Pacific Sil
Bachelor of Science in Computer Science – University of California
Computer Science – Arizona State University
Bachelor of Science in Computer Science – Technical University
Master of Business Administration in Information Systems – American University
Bachelor of Science in Computer Science – Belarusian State University University
Higher Education Program Recommendation

Bachelor of Science in Computer Science – University of Havana
Bachelor of Science in Computer Science – University of the Punjab
Bachelor of Science in Computer Science – University of Applied Science

39: erp / crm software

Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – University of Massachusetts
Bachelor of Science in Computer Science – University of Texas
Bachelor of Science in Computer Science – University of Texas
Computer Science – Rochester Institute of Technology
Bachelor of Science in Computer Science – California State University
Computer Science – University of Maryland
Bachelor of Science in Computer Science – Arizona State University
Bachelor of Science in Computer Science – University of Warsaw
Bachelor of Science – University of Illinois

42: logo design & branding

Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – Virtual University
Master of Computer Applications – Anna University
Master of Science in Information Technology – Punjab Technical University
Bachelor of Technology in Information Technology – Anna University
Master in Computer Applications – Sikkim Manipal
Bachelor of Engineering in Computer Science – Anna University
Master in Computer Applications – Gujarat Technological University

43: 3d modeling & cad

Bachelor of Science in Computer Science – Warsaw University
Bachelor of Science in Computer Science – University of California
Master of Science in Computer Science – University of Washington
Master of Business Administration in Management Information Systems – American University
Bachelor information Technology – Queensland University of Technology
Bachelor of Science in Computer Science – Technical University
Higher Education Program Recommendation

Bachelor of Science – Universidade de Coimbra
Bachelor of Science in Computer Science – University of Nebraska
Bachelor of Science in Computer Science – University of Applied Science
Bachelor of Science in Computer Science – University of Havana

45: animation
Bachelor of Science in Computer Science – Warsaw University
Bachelor of Science in Computer Science – University of California
Certificate – South Pacific Sil
Bachelor engineering – Anna University
Bachelor of Science in Computer Science – Technical University

48: video production
Bachelor of Engineering in Computer Science and engineering – Anna University
Master – National Technical University of Ukraine
Bachelor of Science in Computer Science – University of Illinois
Bachelor of Science in Computer Science – Indiana University
Certificate – British Columbia Institute of Technology
Bachelor of Science in Computer Science – University of Karachi
Bachelor of Science in Computer Science – University of California

62: product design
Web Development Immersive – General Assembly
Web Development – Dev Bootcamp
Web Development – App Academy
Full Stack Web Development – App Academy
Web Development – General Assembly
Web Development – Makers Academy
Computer Science – University of Maryland
Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – University of North Carolina
Certificate – Madison College
63: illustration

Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – Warsaw University
Bachelor of Science in Computer Science – National University of Computer and Emerging Sciences
Video Game Design and Development – Trios College
Bachelor of Science in Computer Science – State University
Bachelor of Science in Computer Science – Universidad Complutense de Madrid

65: other - admin support

Bachelor Technology – Jawaharlal Nehru Technological University
Certificate – Nova Scotia Community College
Bachelor of Engineering – Visvesvaraya Technological University
Master of Computer Applications – Sikkim Manipal
Bachelor engineering – Anna University
Master computer application – Gujarat Technological University
Bachelor of Science in Computer Science – University of Maryland
Bachelor of Science in Computer Science – University of California
Computer Science – Indiana University
Master Computer Applications – Anna University

67: mechanical engineering

Bachelor of Science in Computer Science – Warsaw University
Certificate – South Pacific Sil
Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – Technical University
Software Engineering – Niit
Bachelor of Science in Computer Science – Belarusian State University University
Bachelor of Science in Computer Science – University of Havana
Bachelor of Science in Computer Science – University of Bucharest
Bachelor of Science in Computer Science – Georgia State University
Bachelor engineering – Anna University
Higher Education Program Recommendation

85: business intelligence

Master of Science – University of California
Computer Science – University of California
Bachelor of Science in Computer Science – University of Michigan
Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – University of Texas
Master of Science in Computer Science – University of Texas
Bachelor Technology – Indian Institute of Technology
Bachelor of Science in Computer Science – Brigham Young University
Bachelor – University of California
Computer Science – University of Maryland

86: security

Bachelor of Science in Computer Science – University of Illinois
Master of Science in Computer Science – Johns Hopkins University
Bachelor engineering – Anna University
Bachelor of Science in Computer Science – Worcester Polytechnic Institute
Bachelor of Science in Computer Science – University of California
Bachelor Technology – Indian Institute of Technology
Bachelor of Science in Computer Science – Brigham Young University
Bachelor of Science in Computer Science – California State University
Computer Science – Technical University
Bachelor of Science in Computer Science – University of Massachusetts

87: cloud

Bachelor of Science in Computer Science – Washington State University
Bachelor of Science in Computer Science – Arizona State
Bachelor of Science in Computer Science – University of California
Master – Faculdade de Engenharia da Universidade do Porto
Master of Engineering in Computing – Imperial College
Master – Moscow State University
Bachelor of Engineering in Computer Engineering – Mumbai University
Bachelor of Science in Computer Science – University of Colorado
Higher Education Program Recommendation

Bachelor of Science in Computer Science – Universidade Catolica
Master of Science in Computer Science – University of California

**88: devops**

Online Course – Mongodb University
Bachelor of Science in Computer Science – Purdue University
Bachelor – Universidade de Sao Paulo
Certificate – Nova Scotia Community College
Bachelor of Science in Computer Science – University of Texas
Bachelor of Science in Computer Science – California State University
Bachelor of Science in Computer Science – University of California
Bachelor Computer Science – Central University
Bachelor of Science in Computer Science – Florida State
Certificate – British Columbia Institute of Technology

**89: software architecture**

Computer Science – University of California
Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – University of Maryland
Certificate – Nova Scotia Community College
Bachelor of Engineering in – Gujarat Technological University
Bachelor of Engineering in Computer Science – Anna University
Master of Science in Computer Science – Maharishi University of Management
Bachelor of Science in Computer Science – Instituto Superior Tecnico
Bachelor of Science in Computer Science – University of Michigan
Bachelor of Engineering in – Anna University

**90: infrastructure engineering**

Bachelor of Science in Computer Science – Central University
Bachelor of Science in Computer Science – University of California
Bachelor of Science in Computer Science – University of the Philippines
Bachelor information Technology – Rochester Institute of Technology
Bachelor of Engineering in Computer Science – Anna University
Master of Science in Computer Science – University of Illinois
Higher Education Program Recommendation

Bachelor Computer Science – Universita Degli Studi di Padova
Bachelor of Science in Computer Science – University of Illinois
Master of Science in Computer Science – northwestern University
Bachelor Technology – Jawaharlal Nehru Technological University

91: technical management

Bachelor of Science in Computer Science – University of California
Master of Science in Computer Science – Technical University
Computer Science – University of Waterloo
Bachelor of Science in Computer Science – Brigham Young University
Bachelor of Science – University of Toronto
Bachelor of Science in Computer Science – Wentworth Institute of Technology
Bachelor of Science in Computer Science – Universidade Catolica
Full Stack Web Development – Code Camp
Bachelor of Science in Computer Science – University of Washington
Bachelor of Information Technology in Systems Engineering – Hasso Plattner Madison
Appendix E

Survey

Figure E.1: Survey first page with questions related to the user.
Figure E.2: Part one of the survey second page with questions related to the web based system.
How useful did you consider the recommendations? *

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not useful at all</td>
<td>⬜</td>
<td>⬜</td>
<td>⬜</td>
<td>⬜</td>
</tr>
<tr>
<td>Very useful</td>
<td>⬜</td>
<td>⬜</td>
<td>⬜</td>
<td>⬜</td>
</tr>
</tbody>
</table>

Do you have any suggestions for possible improvements?

☐ Filter recommendations by country or by university.

☐ Filter recommendations by degree.

☐ Disable online courses recommendations.

☐ Only see recommended online courses with more detail.

☐ Possibility to see profiles of alumni that took each of the recommended programs.

☐ Other: ________________________________

Figure E.3: Part two of the survey second page with questions related to the web based system.