A Machine Learning Approach for Future Career Planning

Yu Lou
Computer Science
Stanford University
yulou@stanford.edu

Ran Ren
Computer Science
Stanford University
rren@stanford.edu

Yiyang Zhao
Computer Science
Stanford University
zhaoyy@stanford.edu

Abstract

In this paper, we work on the modeling of peoples career paths. We first collect a large number of people's profile and extract features from the descriptive information. Hand rules and clustering algorithm has been applied to help avoid the negative effect of natural language. We model people's career developments with Markov Chain, and present our approach to estimate the transition probability matrix. Finally, we solve the problem that given a person's current career path and his/her goal, what is the best best career development recommendation for him/her. As a conclusion, we will analyze the results and discuss possible improvements of our model.

1. Introduction

1.1. Motivation

For college students, when facing various career options upon graduation, it could be overwhelming to choose a job that better fits with his/her future career goals. Also, for current employees, it could be unclear that whether changing a job or pursuing advanced study will help to reach his/her ambition. This is when people start to looking for other people who have similar backgrounds to see what were their decisions and where did they end up. Instead of consulting only a few acquaintance, we present a way to help people learn from thousands of others with similar backgrounds, and find best career steps that enable them to reach their goals.

1.2. Problem Definition

We model a person's career path with a sequence of his/her education or working experiences in time order. Each piece of experience regarding either work or education is denoted as a node. Education is defined by university, major and degree, while working experience is represented by position and employer. 1 gives a graphical representation.

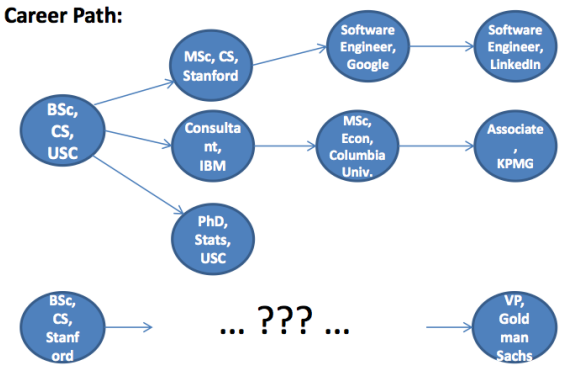

Figure 1. Sample career paths

We view each person's profile and career path as sequence with multiple nodes(steps) $(X_1, X_2, ...)$. Each node will be represented by various multinomial features $(v_{i1}, v_{i2}, ..., v_{ik})$. Given a person's current career path and his/her goal, we try to discover the optimal career path, that is, the path that has the highest probability to reach the goal node.

This project consists of various challenges. For instance, how to translate the descriptive information in one's profile into features. Also, as there are tens of thousands non-standard position titles, we need to find a way to group the ones describing the same or similar positions together, so as to avoid having billions of possible states in the model and hence under-fitting. Moreover, the algorithm for estimating the parameters and path recommendations should be carefully chosen.

The rest of this paper is organized as follows. We describes the collecting of data and data preparation in Section 2. Then in Section 3, we presents our model and approach for this problem. Section 4 shows some of the main results we get and analysis based on such results. Finally, we conclude this project and discuss possible future extensions in Section 4.

2. Data Preparation

2.1. Data Collection

We obtained about 67,000 profiles from LinkedIn as our data source. Each profile contains an average of 3.7 pieces of education or working experience. As for working experience, the raw data consists: name of the company, position, time period and optional description. Education information also breaks down into: name of the university, degree, major and time period. We first sorted the experiences of one person by their beginning date. Then, we pre-process the data in the following way in order to better model the nature of people's careers.

2.2. Data Pre-processing

Instead of company name, we use company size and industry as feature to represent the employer. Such information is obtained from LinkedIn's company pages. Using company name as a multinomial feature would not cause much problem for cases like Google or Goldman Sachs, as there are plenty of people working/worked there. However, for small companies and start-ups, it would be hard for their employees to find people with similar experience.

Similarly, in order to generalize the university feature and avoid under-fitting, we collect last year's university rank from US News[2] and divided all universities into four categories: top 10, top 50, top 100 and other. Though university rank does not necessarily reflect the reputation of a student's program (e.g. some school have bad overall rank but is quite famous for certain department, like CMU), it works well as an indicator in most cases and is relatively easy to apply.

For degree and major, hand written rules are applied to translate various representations to a standard set of degrees and majors. However, for positions, we discovered that people use too many different terms for positions. It is impossible for us to hand write some rule to categorize those tens of thousands positions into a reasonable number of categories. Therefore, we run a modified k-means clustering algorithm on position titles.

2.3. Clustering

2.3.1 Distance

The distance between each two position titles is defined by their averaged semantic similarity. The semantic similarity is generated from the WordNet::Similarity project[1], which is a Perl module that implements a variety of semantic similarity measures based on information from lexical database WordNet[3].

2.3.2 Algorithm

We use the K-means clustering algorithm and choose k to be an arbitrary number, 1000. Also, instead of updating the k means to be the centroids of each cluster, we set the new means to be the phrase that has the highest total similarity with all the phrases in the cluster.

2.3.3 Clustering Result

After clustering, many semantically similar position titles are grouped together. For example, "software development engineer", "programmer", "software developer" are all grouped together with "software engineer", and "Recruiting Coordinator", "Technical Recruiter" are grouped with "Recruiter". However, some acronyms, like "VP", instead of clustered to be with "Vice President", just stand alone as WordNet does not contain information regarding this word.

3. Model

3.1. Markov Chain

First, we assume people's current job only depends on his most recent job (or education). This is a simplified version of the problem; however, we can always index the state by considering all history information to achieve Markov property.

Thus, we can model a career path as a Markov chain, i.e. the next state depends only on the current state.

$$P(X_t = x_t | X_{t-1} = x_{t-1}, ..., X_1 = x_1)$$
$$= P(X_t = x_t | X_{t-1} = x_{t-1})$$

And we further assume the career path is a time-homogeneous Markov chain, so that the process is described by a single, time-independent transition matrix p.

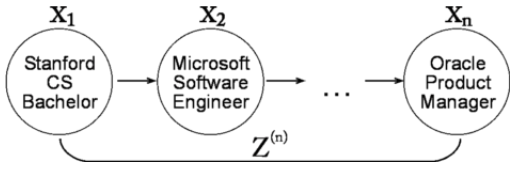

Figure 2. Markov chain

3.2. Probability Estimation

We need to estimate the transition matrix $p_{(N,N)}$, where N is the total number of different states, and entry $p_{i,j}$ represents the probability of going from state i to state j

$$p_{i,j} = Pr(X_t = j | X_{t-1} = i)$$

We have observation $z^{(n)}$ of people's profile, which is a sequence of states $x_1, x_2, ..., x_n$, and we also think a career path of length n is a random variable $Z^{(n)}$. Then the probability of $Z^{(n)}$ taking value of $z^{(n)}$

$$\begin{aligned}&Pr(Z^{(n)} = z^{(n)})\\ =&Pr(X_1 = x_1)\prod_{t=2}^{n} Pr(X_t = x_t|Z^{(t-1)} = z^{(t-1)})\\ =&Pr(X_1 = x_1)\prod_{t=2}^{n} Pr(X_t = x_t|X_{t-1} = x_{t-1})\end{aligned}$$

Now, the likelihood of all observations $Z_1^{(n_1)}$, $Z_2^{(n_2)}$,..., $Z_m^{(n_m)}$ given transition matrix p

$$\begin{aligned}&L(p)\\ =&\prod_{i=1}^{m} Pr(Z_i^{(n_i)} = z_i^{(n_i)})\\ =&\prod_{i=1}^{m} Pr(X_{i,1} = x_{i,1})\prod_{t=2}^{n_i} Pr(X_{i,t} = x_{i,t}|X_{i,t-1} = x_{i,t-1})\end{aligned}$$

If we rewrite the likelihood in terms of $p_{i,j}$, we will get

$$L(p) = \left[\prod_{i=1}^{m} Pr(X_{i,1} = x_{i,1})\right]\left[\prod_{i=1}^{N}\prod_{j=1}^{N} p_{i,j}^{n_{ij}}\right]$$

Here n_{ij} is the number of times that state i goes to states j among all observations.

Therefore, we want to maximize the log likelihood, which is

$$\begin{aligned}&\ell(p) = \log L(p)\\ =&\sum_{i=1}^{m} \log Pr(X_{i,1} = x_{i,1}) + \sum_{i=1}^{N}\sum_{j=1}^{N} n_{ij} \log p_{i,j}\end{aligned}$$

Also notice the probabilities have the property that the summation of probability of making a transition from state i is equal to 1, that is

$$\sum_{j=1}^{N} p_{i,j} = 1$$

Now we are facing a constrained optimization problem, we define the Lagrangian of this problem to be

$$\mathcal{L}(p, \beta) = \ell(p) + \sum_{i=1}^{N} \beta_i(1 - \sum_{j=1}^{N} p_{i,j})$$

By setting

$$\frac{\partial \mathcal{L}}{\partial p_{i,j}} = 0, \frac{\partial \mathcal{L}}{\partial \beta_i} = 0$$

We get

$$\sum_{j=1}^{N} p_{i,j} = 1, \frac{n_{ij}}{p_{i,j}} - \beta_i = 0$$

The solution to the problem is

$$p_{i,j} = \frac{n_{ij}}{\sum_{j=1}^{N} n_{ij}}$$

4. Path Prediction

Once we have the model, we can use the model to predict a career path according to user's requirement. We allow user to specify a concrete objective. A concrete objective is defined by exact information he wants to achieve (Software Engineer at Facebook, Trader at Morgan Stanley, etc.). We apply graph search technique to solve the problem.

4.1. Shortest Path

In the concrete objective case, we have a start state x_s representing user's background so far. The objective can also be converted into a state x_o using the information provided. Now we want to predict a path $P^* = x_{i_1}, x_{i_2}, ..., x_{i_n}$ which is most possible from $x_s = x_{i_1}$ to $x_o = x_{i_n}$, that is

$$\begin{aligned}P^* &= \arg\max_P \prod_{t=1}^{n-1} Pr(X_{t+1} = x_{i_{t+1}}|X_t = x_{i_t})\\ &= \arg\max_P \sum_{t=1}^{n-1} \log p_{i_{t+1},i_t}\\ &= \arg\min_P \sum_{t=1}^{n-1} -\log p_{i_{t+1},i_t}\end{aligned}$$

So if we consider each state as a vertex, put $-\log p_{i_{t+1},i_t}$ as weight on edge (i_{t+1}, i_t). Solving the problem is equivalent to finding a shortest path between a single source and a single destination on directed graph, and all weights are positive. The problem can be solved using Dijkstra's algorithm.

5. Results

On given queries, our model has generated plausible and logical career path recommendations.

For instance, when a person with a computer science degree intends to reach a senior developer position in a large non-IT-related company, the generated path would first recommend a developer position in junior level in a IT company, then try to switch over to that goal position, which corresponds to our conventional opinion.

Also, if a person with a computer science degree intends to land a director position in a financial firm, the model

would first recommend obtaining a degree in the financial-related major, preferably masters, then climb the ladder up through assistant manager, manager and eventually reach director. This also conforms to the common understanding that transition into another industry field is eased by obtaining a relevant degree.

However, due to the nature that the accuracy of our career path generation is largely to be measured against human common sense, it is difficult to provide means of automatic result evaluation. This is further elaborated in the conclusion section.

6. Conclusion

Our model gives convincing result on basic queries that generally results in short-stepped paths. However, despite the effort, the models logical accuracy quickly falls off when the expected length of the path increases, or in other words, the level of goal position increases. One such interesting, however undesirable result comes from querying the path from a college graduate to a large companys CEO. The model gives just a single step conversion. The falloff is most likely the result of the original data skew and our preprocessing of the data, and is further explained later.

Several reasons factored into the lack of desirable result. First of all, the feature components that are relevant to the career path module are difficult to be automatically quantified and compared. Aside from the rankings of graduating universities and the sizes of the company, both of which are more considered as ranges rather than an exact number, the values of other components, including the majors, position titles and industry fields, are all distinct and incomparable to each other, and can only be indexed or treated with binary representations. Except for the binary representation that would create sparse features with thousands of components of which only few of them are 1s, the lack of proper mathematical representation seriously limits the machine learning algorithms that are applicable to the proposed problem.

Secondly, the raw data from LinkedIn is represented in natural language format, and this added to the task of data cleaning an additional level of processing requirement. The entries in the raw data highly vary from person to person, depending on their input. Many position titles contain more details than what we desire, which is more general, e.g. computer engineer. In addition, many titles are actually meaning the same position (software developer vs. software engineer vs. programmer). Our clustering algorithm depends on similarity between words, so the actual accuracy of similarity between title phrases is yet to be examined.

Thirdly, our model is completely trained from real human input, so it may be largely skewed towards education-position transition, which in our interpretation of the model is a must-have step in that persons career path. This results a more accurate path generation in the first step when a persons education info is the query input. However, when a senior position is the goal, there may not be enough multi-step entries in the profile that outweighs the number of direct transition entries found. As a way of reducing the models final search space and easing the data preparing phase, we imposed strict format restrictions on parsing of the original data. It is likely that the more transitions a person experienced, the more likely that format is violated, thus resulting the profile filtered out. This is one of the areas that require the most urgent optimization.

Finally, as mentioned in the result section, the accuracy of the generated paths is mostly to be measured against common sense, so given a real profiles education info and current position, the generated path may differ from that persons actual experience, but this does not necessarily mean that the accuracy is bad on this query.

As for future work and modifications, we intend to first improve the quality of the transition from raw data in natural language format to the training input in mathematical format. This includes a more powerful initial parser and a better clustering algorithm to cluster major and position titles, or even company industry. Then we would seek other applicable learning algorithm to train the model, possibly on binary-component features. In addition, we would seek ways to automatically evaluate the quality and accuracy for the generated path.

References

[1] T. Pederson, S. Patwardhan, J. Michelizze, and S. Banerjee. Wordnet::similarity, 2008. http://www.d.umn.edu/ tpederse/similarity.html.

[2] National universities rankings - us news, 2010. http://colleges.usnews.rankingsandreviews.com/best-colleges/national-universities-rankings.

[3] Wordnet, 2006. http://wordnet.princeton.edu/.