**Smarter Job Recommending System**

CSE 6242 Project proposal

Team: Blue Fly Mining

Contact: Jingwei Qi

Email: qijingwei@gatech.edu

**I. Background and justification**

Job search engine sites, such as linkedin, careerbuilder, indeed etc., enable users to search online sources of job listings by keyword and location. The popularity of these sites among job seekers has grown tremendously over the past few years, and it has become imperative for the websites to provide more precise and accurate job recommendations to their users. On the other hand, the data generated on job search websites contain certain characteristics that make it challenging to build a good recommender using traditional methods. In this project, we plan to build a better job recommendation system that provides a list of jobs that a given user is interested in. In particular, we will take advantage of the rich text data from job-postings to build beyond a elaborate user-job feature space, and apply learning-to-rank method to provide more accurate recommendation.

**II. Literature Review**

The current generation of  recommender systems can be classified into three categories:

Content-based recommender systems, Collaborative- based recommender systems, Hybrid-Based recommender systems.

1. **Content-based recommender systems** focus on recommending items containing textual information, such as documents, and news messages. Content in these systems is usually described with keywords[1]. Our project will use text-mining technologies to acquire job information, and find potential job for the user.  But the Content-based recommender system cannot be applied to those potential jobs that the user is unfamiliar with.
2. **Collaborative-based recommender systems** are try to predict the utility of jobs for a particular user based on jobs previously applied by other users. It can be grouped into two general classes: memory-based( also called heuristic-based) and model-based.
3. **Memory-based** algorithms rating predictions based on the entire collection of previous applied jobs by users.It can also be divided into two groups.  User-based and Item-based.

User-based method consider the similarity to other users and select top K similar users, and recommend the jobs which are applied by them[2]. Our project mainly use similar users’ application information to recommend new jobs for user.

Item-based, on the other hand, recommends the jobs similar to the users applied before and predict the potential the user will apply. It is much easier for calculation than user-based.  But it doesn’t consider the difference of different users, the accuracy is less than user-based method.

1. **Hybrid-Based Recommendation** combines content-based and collaborative recommender system has higher performance has higher accuracy. One common way is combine Collaborative technology with other technologies to overcome cold-start problem[1][3]. To improve the accuracy, our  project will combine content-based and user-based together[6].
2. **Latent Dirichlet Allocation**

LDA is a generative topic model that allows sets of observations to be explained by groups that explain why some parts of data are similar. [7] Documents are modelled as mixture over a set of topics.[8] LDA uses latent variable to explain the observed co-occurrence of words in documents. It assumes that each document is associated with a mixture of active topics, and that each word is kind of generated by one of these topics. Topic models like LDA possess a rich representational power since whey allow for documents to be comprised of words from different topics rather than just one. This increased power allows us to scale up the features to make our recommendation more accurate. A spectral algorithm based on the method of moments.[9][10] It attempts to match the observed moments with those posited by the model. This methods is simple and efficient to implement which meets our demands for the small course project.

**6. Word Embedding**

Also, learning word representations from large collections of unstructured text is an effective way of capturing word similarity information. Word embedding techniques allows words to be represented by vectors of real numbers in low dimensional space, relative to a continuous space. Using these techniques can boost the performance of mining features and tags from text. A neural probabilistic language model [11] is provided to fight curse of dimensionality by learning a distributed representation for words, which allows each training sequence to inform the model about an exponential number of semantically neighboring sentences. We can quickly build and train this model using methods of previous research[12][13]

**III. Hem’s Question**

1. Our task is to build a recommendation system that provides a given user a list of jobs that he or she is expected to be interested in. Formally, we define our problem as follows:

Let U be a collection of user profile;

Let J be a collection of job posting.

Let A(u,j) be a matrix indicating whether a given user u has applied to a given job j. (A(u,j) = 1 if user u has applied to job j.

Given U,J,A, we will derive a function f(u,j) reflecting the probability that user u will apply to job j in the future. For a given user ui, our recommender will return the top-k j’s that result in the highest f(ui,j) score for that ui.

**2.**

As introduced in detail in the literature review section, most recommender systems take either of two basic approaches: collaborative filtering or content-based filtering. Content-based method tries to place each job posting into a feature space, and calculate a similarity for each pair of job. Collaborative filtering arrives at a recommendation that's based on a model of prior user behavior.

Traditional recommendating systems have certain limitations. First of all, the assumption that “similar users are more likely to apply for similar job” is often not valid in the job search scenario. For example, a history major in New York having applied to a job in finance does not mean that all history majors are more likely to be interested in finance. The complexity of the decision making process for career interest demands a more sophisticated learning model than a simple similarity metric[5].That’s why we need to implement collaborative mechanism. In addition to that, the sparseness of  the application matrix makes it very hard for the traditional recommendation system to predict users’ interest accurately[6].

1. Since the previous methods are too naïve only making use of relevance between data, we want to create a more sophisticated model utilizing the raw text in job description.

We devise our approach by first observing an important characteristic of our particular problem: unlike in other recommendation tasks (movies, music, etc.), where customers and items are two *distinct* feature sets, in the job recommendation scenario, the semantic overlap between user profiles and job postings, (i.e the “match” of certain keywords in the user’s resume and in job description ) is actually a more important measurement of relevance than relevance inferred by similarity between two users or between two jobs. Thus, we plan to use text-mining and NLP techniques such as LDA and word embedding to construct a rich feature vector space based on the combination of user profile and job posting. Our recommending model first computes the similarities between subsets of the user’s feature vector and subsets of the job’s feature vector. The similarity vectors will then be weighted by training a supervised learning model based on user application data.

**4.**

A more accurate job search engine allows for more efficient allocation of resources and will benefit job search websites, job seekers and recruiters, among others.

1. If our method is successful, we will see an improvement in click-through-rate and application per impression rate on the job search websites.
2. Data availability and quality are the biggest risks we’re facing. In particular, our model requires data for user profiles, which is hard to get through API’s. Fortunately, we were able to find a complete set of user and job data from career builder.com. If our model shows significant improvement over the existing method, we can collaborate with job search websites who owns the proprietary user data and scale up our model.
3. We will utilize open source technologies and there will be no monetary costs.
4. The project can be done in 10 weeks.
5. At the end of the 5th week, along with a progress report, we will have a trained recommender model. In the following weeks the model will be tested and validated, and at the meantime we will start to build a user interface. At the end of 10th week, a functional recommender with interactive user interface will be competed.

**IV. Project Activities plan**

**a)**     Work Distribution

Jingwei:

        Team leader, engineer, mainly works on applying and analysing machine learning/data ming/raking/recommendation algorithm, building models, visualization coding, also helps on visualization design, model testing/evaluation.

Wenying:

        Front end designer, analyst, mainly works on visualization design and data analytics, discovering innovative ways to present data, also helps on model evaluation.

Mengfan:

        Tester, quality controller, mainly works on testing the robustness and correctness of the codes and models, also helps on model building, visualization design.

**b)**    Timing Frame

Project Proposal (Sept. 11-Oct. 10)

        Brainstorming, come up with innovative ideas that worth trying.

Read papers, do surveys, make plans, write a proposal.

Start making prototypes.

Progress Report (Oct. 11-Nov. 06)

Collect, clean data

Build, train models

Start visualization scratching

Finish the first iteration of the whole project

Project Final Presentation (Nov. 06-Dec. 04)

Modify algorithms and improve model performance

Visualize the result as well as the process

Finish several iterations of the project

Make presentation videos and posters

**c)**    Environment and Tools

Python, Javascript, D3, etc.

**V. Reference**

[1] Gediminas Adomavlclus, Alexander Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions” . IEEE Transactions on knowledge and data engineering. Vol 17, No. 6, June 2005.

[2] A. Ansari, S. Essegaier, and R. Kohli, “Internet Recommendations Systems,” J. Marketing Research, pp. 363-375, Aug. 2000.

[3] M. Pazzani and D. Billsus, “Learning and Revising User Profiles:The Identification of Interesting Web Sites,” Machine Learning,

vol. 27, pp. 313-331, 1997.

[4] A. Nakamura and N. Abe, “Collaborative Filtering Using Weighted Majority Prediction Algorithms,” Proc. 15th Int’l Conf. Machine Learning, 1998

[5] G. Shani, R. Brafman, and D. Heckerman, “An MDP-Based Recommender System,” Proc. 18th Conf. Uncertainty in Artificial Intelligence, Aug. 2002.

[6] Soboroff and C. Nicholas, “Combining Content and Collaboration in Text Filtering,” Proc. Int’l Joint Conf. Artificial Intelligence Workshop: Machine Learning for Information Filtering, Aug. 1999.

[7] David M. Blei, Andrew Ng, and Michael Jordan. Latent dirichlet allocation. JMLR, 3:993–1022, 2003.

[8] M.D. Hoffman, D.M. Blei, and F. Bach. Online learning for latent dirichlet allocation. In NIPS, 2010.

[9] K. Pearson. Contributions to the mathematical theory of evolution. Phil. Trans. of the Royal Society, London, A., 1894.

[10] Anandkumar, Anima, et al. "A spectral algorithm for latent Dirichlet allocation."*Advances in Neural Information Processing Systems*. 2012.

[11] Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language

model. Journal of Machine Learning Research, 3:1137–1155, 2003.

[12] Yoshua Bengio and Jean-Sebastien Sen ´ ecal. Quick training of probabilistic neural nets by importance sampling. In AISTATS’03, 2003.

[13] Yoshua Bengio and Jean-Sebastien Sen ´ ecal. Adaptive importance sampling to accelerate training of a neural probabilistic language model. IEEE Transactions on Neural Networks, 19(4):713–722, 2008.