Recommendations to Support Staffing Decisions

Abigail Gertner
The MITRE Corporation
202 Burlington Rd
Bedford, MA 01730
+1 781-271-3130
gertner@mitre.org

Susan Lubar
The MITRE Corporation
202 Burlington Rd
Bedford, MA 01730
+1 781-271-2860
slubar@mitre.org

Beth Lavender
The MITRE Corporation
202 Burlington Rd
Bedford, MA 01730
+1 781-271-6362
lavender@mitre.org

ABSTRACT

We have developed a recommender system to assist project managers at MITRE with identifying and learning about potential candidates for open project positions. Features were added to the user interface for the recommender as a result of a pilot study in which the recommender was used to suggest job candidates for open positions. In this paper we explain how the recommender is implemented, describe the pilot study, and discuss the improvements to the user interface that were made to support the study.

1. INTRODUCTION

The MITRE Corporation, a not for profit that operates multiple federally funded research and development centers (FFRDCs), has approximately 7,000 employees working on projects ranging in length from a few days to a few years. The largest concentrations of employees are located in Bedford, MA and McLean, VA. However, there are additional employees at smaller sites or working as teleworkers all over the world.

Due to the nature of MITRE's work, new projects frequently arrive which need to be staffed. The employee base covers diverse skills such as health care knowledge, software development, signal processing, and knowledge of civilian and military programs and systems.

As part of the internal corporate website, a site called "Tech Stature" is supplied for employees to enter information about themselves, including educational background, publications, professional activities, and semantic tags describing their skills and expertise. A separate electronic system is used for entering job requisition information for projects with unfilled positions. This system also includes a field for semantic tags describing the skills desired for the particular job. Other data sources available for our use include employees' availability over a six-month period as well as their contact information, job level, organization within MITRE, location, and scheduled hours.

We have implemented a recommender system called MaPP (Matching People to Projects) which attempts to match people to job requisitions based on the tags they have entered about themselves. This is intended to be an aid to help the project leader. However, we have found that there are many other pieces of information about an employee such as location, staff level, or availability, which also impact how good of a match they are for a particular job. We therefore present the recommendations to the user via an interface that includes the ability to sort and filter the information according to the search they are performing. The recommender is available as a prototype service on our internal corporate network.

In this paper, we describe the design of the MaPP recommender GUI, as well as a pilot effort that had the goal of matching open job requisitions with appropriate staff.

Approved for Public Release; Distribution Unlimited. Case Number 14-2592

2. RELATED WORK

Previous work on recommending people in an enterprise setting has often focused on recommendations in the context of a social network [1][2]. For example, Chen et. al. [1] present a recommender system that finds potential "friends" for users on the IBM social network by looking for people with similar contribution content as well as common relationships in the social network.

Another related body of work is that of expertise finding [3][4], which aims to help users find an expert within an organization to collaborate with or ask for assistance. The MITRE Expert Finder [3] is one such system that drew from information on the corporate network to recommend staff members with expertise relevant to a user's query. The MITRE Expert Finder relied on written documents being contributed by the experts in order to identify them. The task of the MaPP staff recommender is different because often a project leader will be searching for staff members with skills such as specific programming languages or familiarity with a particular technical domain area about which they may not have produced written documents. Tech Stature tags therefore provide a much more reliable source of information to match employees based on their declared skills.

Because the MaPP staff recommender is primarily a decision support tool, assisting users in making staffing decisions, the user interface requirements are different from those of a typical expert finder. A project leader looking for staff has several diverse constraints that must be satisfied in finding an ideal candidate, beyond simply identifying a good skill match. These interface requirements will be discussed in Section 5.

3. RECOMMENDER IMPLEMENTATION

The goal of the MaPP staff recommender is to find staff members who have tagged themselves with skills that match the requirements of a job opening. The recommender is triggered when a set of query terms taken from the job requisition are entered. It was clear that it would not be effective to use a straight keyword search to find matches since the tags entered by employees in Tech Stature, and those in the job requisition system, are unconstrained and therefore a single concept may be represented by any number of synonyms and related terms in the tags. For example, a job requiring skills in "cloud computing"

might be a fit for staff members with tags such as "web services", "virtualization", or "Hadoop". We determined that using a recommender system approach would allow us to find matches to staff with tags that are related to the query terms even when they are not an exact match.

3.1 Recommendation Display

When displaying the recommendation results we aimed for a minimalist aesthetic that would provide the information needed in an uncluttered display, making it easy for users to scan through the results. For each employee in the recommender output, we include a photo, contact information, organization, location, job level, business title, scheduled hours, and their Tech Stature tags (see Figure 1). Any tags that are exact matches for the search are highlighted in blue. The employee's availability for the next six months is shown as a small bar chart.

Figure 1: Display of a single recommendation result

The full recommender display is shown in Figure 2. While results are sorted by the relevance score from the recommender, we chose not to include any indication of the score itself in the display because we felt that it would be potentially distracting and was not really necessary for the user to process the results. Based on the pilot study, we now feel that it may be useful to include some representation of the strength of the match in order to help project leaders determine which recommendations could be most valuable to pursue further.

We have considered limiting the number of results per page to improve readability; however, the desire to see all possible results at one time was deemed to be of greater priority.

3.2 Recommender Algorithm

The staff recommendation algorithm is based on the insight that rather than recommending *items* for a *user*, as in a typical recommender system, we are recommending *users* for an *item*. Or, more precisely, we are recommending users for a group of items – the set of tags in the query. We therefore constructed a recommender that uses a standard collaborative filtering recommendation algorithm, but with the users and items flipped so that it is recommending users for a given item rather than items for a given user. Our recommender is implemented using the Apache Mahout collaborative filtering recommendation library [6].

The staff tags in Tech Stature do not have any score or priority value attached to them, so we use a recommendation algorithm designed for binary ratings. To compute the similarity between two tags we use the Tanimoto similarity coefficient to compare the sets of users associated with each tag. The Tanimoto coefficient, also known as the Jaccard coefficient, is a metric that measures the similarity between two sample sets by computing the ratio of the elements the sets have in common to the total size of the sets. When comparing two tags T_1 and T_1, the similarity of the tags is measured by comparing the sets of users U_1 and U_2 that have tagged themselves with T_1 and T_2 respectively. The Tanimoto coefficient for the two sets of users is:

$$_1(_2 =) \quad \frac{\left| \cap _2 \right|}{\left| \cup _2 \right|}$$

The recommender uses a nearest neighbor approach, finding the 30 nearest neighbors to the query tag based on their Tanimoto similarity. This gives us a set of tags that are related via co-occurrence with the query tags. We then find the employees that are associated with the tags in the neighborhood and return them as the recommended staff for the query.

Recommending staff for a query is complicated by the fact that we are searching for a match for multiple tags. This is essentially the same as the group recommendation problem presented in [5] in which the goal is to recommend movies to a group of people. O'Connor et. al. discuss two different approaches to recommending for a group. The first option is to generate recommendations for each member of the group individually and then combine them. This approach has the advantage that it produces results that are directly related to the individual members of the group, but it is less likely to generate serendipitous results that arise from the combination of group members' preferences. The second option, which is the one we use, is to create an artificial "pseudo-user" representing the combined preferences of the group members. Since our recommender data model is flipped, with the tags as the "users" and the staff members as the items being recommended, we create a "pseudo-tag" representing the combination of tags in the search query. The pseudo-tag is associated with every user who has used at least one of the tags in the original search query. We then run the recommendation algorithm outlined in the previous paragraph using the pseudo-tag as the recommendation target. We look for tags in the neighborhood of the pseudo-tag based on how many users they have in common, and then recommend the staff members with the highest expected preference for the pseudo-tag.

The final difference between the MaPP staff recommender and a typical recommender system is that we want to recommend staff members who match any of the query terms exactly in addition to the staff who are related to the query but are not an exact match. Most recommender systems do not recommend items that match exactly because those are the items that the user has already rated and therefore presumably already knows about and doesn't need to see a recommendation for. In a group recommendation context, on the other hand, it may be desirable to recommend items that one or more of the members of the group has already rated because those items may still be of interest to the group as a whole. Therefore, after computing a set of recommendations for new users who are not associated with any of the query tags, we add in all of the users who are associated with any of the individual query tags and compute the expected preference for those users as well. The resulting set of recommendations is sorted based on the expected preference for each result.

4. PILOT STUDY

A trial staffing effort was initiated to determine the feasibility of a matching tool. Project leaders looking for staff were asked to enter electronic job requisitions in an online system separate from our recommender. Although a tag field exists in the requisition form, prior to the pilot study described here, it was rarely used. As part of this effort, project leaders were encouraged to add tags describing job requirements to each form. To ensure that tags

entered in the requisitions would match known tags in Tech Stature, the job requisition system was updated to allow users to select tags from the existing set of Tech Stature tags. A total of 44 requisitions were opened for the positions. Of these, 38 had tags, and were thus used for our study.

4.1 The Tag Dataset

A second part of the pilot study was a push to encourage staff to tag themselves in Tech Stature. While the Tech Stature interface is very easy to use, employees often do not understand the purpose of the tags and may need external motivation and guidance in order to provide useful tags. We have provided several tools to make the tagging process easy, including autocomplete suggestions in the tag entry field and recommendations based on the employee's job category and department. Prior to the start of the pilot study the average tag rate for departments participating in the study was 60%. The goal of the pilot was to get to a tag rate of over 85% for each participating department. This goal was actually surpassed over the course of the study, with an average tag rate of 90% being achieved by the end.

As of this writing, there are approximately 17,000 unique tags in the Tech Stature system and a total of 80,000 tags for employees. About 75% of employees have at least one tag. The median number of tags per employee is 9. The tags exhibit a typical long tail distribution, with the most frequent tag ("systems engineering") being used by 1333 staff members, while about 9,000 tags, or 54% are only used by a single staff member.

4.2 Recommendation Generation

To facilitate the generation of recommendations based on job requisition tags, a link was added to the job requisition pages which when clicked will automatically open the staff recommender, populated with the tags from the requisition. A project leader or other user who is looking at an open job requisition only has to click on that link to receive a list of recommended candidates for the position.

For the pilot study, a member of our team, knowledgeable about the corporation and about our recommender, ran each set of requisition tags through our system. For some requisitions, it was found that the tags provided by the project leader were too general. For example, tags such as "systems engineering", "software engineer" or "project manager" describe the general job category that is desired to fill the position but does not give enough domain specific information related to the particular job to be filled. Additionally, these general tags tend to be the most frequently used by employees and therefore result in a large number of matches to be considered in the result set. In these cases, the team member added more specific tags when it was clear from the text of the requisition what they should be, and in other cases called the project leader to discuss what tags to add.

The use of very specific tags in the job requisition could also pose a challenge for the recommender because those tags are often used by just one or two employees, and therefore the recommender has very little information to go on to find related tags. In this case, our team member again added appropriate search terms based on the written job description and a discussion with the project leader in order to produce a more useful result.

For the purpose of this pilot study, we found it useful to add an export capability so that our recommender results could be explored and shared in a spreadsheet. For each employee in the recommendation results, we include their contact information, job title, staff level, location, organization, scheduled hours, availability over the next six months, and their Tech Stature tags. Using this spreadsheet, our team member and the project leaders were able to pivot the employees based on various factors such as location and organization. Our team member edited the spreadsheets to remove suggested employees that appeared to be a bad match based on their tags, or those whose tags did match, but their role in the company made them not a good candidate. In the end, spreadsheets were generated for 25 of the 38 open requisitions. The remaining 13 either had no matching staff available, or had specific requirements such as working at a remote location, for which we had no data. On average, spreadsheets sent to project leaders contained 30 suggested employees.

4.3 Study Findings

Based on the spreadsheets of results, project leaders were able to determine a set of employees that might be good candidates for a particular position. They then followed up by contacting these employees to determine interest, and to interview those that were.

We are in the process of collecting feedback from project leaders about the pilot. We know that successful matches were made through the use of our recommender, and that some project leaders adopted use of the recommender directly rather than going through our team member. Several project leaders indicated that, once presented with a list of recommendations, they felt the need to interview every employee listed in a spreadsheet; a time consuming undertaking. Efforts to filter results more, or to supply a score for each employee could alleviate this issue.

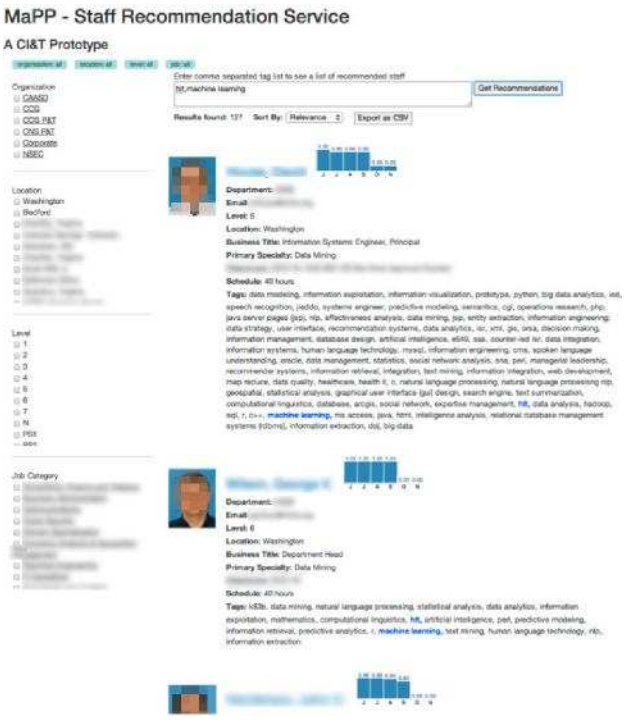

Figure 2: Screenshot of the staff recommender interface

5. INTERACTIVE RECOMMENDATION NAVIGATION

The full recommender interface is shown in Figure 2. Our initial design included a just single field in which to enter tags describing a particular job. We quickly found that, although the results generated based on tags were good, there were additional

attributes of employees unrelated to their skills or experience that made them better or worse matches for a position. For example, it may be a job requirement for the employee to be located at a particular site, or to be at a particular job level. Allowing the user to take these attributes into account while exploring the recommendations returned by our system required additional interactive features that we had not initially included in the interface design. We therefore added sorting and filtering capabilities to the recommender interface as described below. These features give the user much greater flexibility to identify recommendations that match all of their requirements.

5.1 Sorting Recommendations

Recommendation results are initially sorted by relevance to the input query, as calculated by the recommendation algorithm. We found during the pilot that additional features such as availability, location, or job level are also useful for sorting and examining recommended employees. We implemented a sort menu so that users can reorder recommendation results according to any of these attributes. This allows the recommender interface itself to function similarly to the exported spreadsheets described in Section 4, which users can use to pivot on any attribute of interest.

5.2 Faceted Filtering of Recommendations

In addition to sorting results, it is also helpful to remove results from the recommendation list that are not viable staffing options due to constraints such as location, job level, etc. To allow for this filtering of results, we added a set of facet menus along the left side of the page, with checkboxes to restrict searches by organization, location, job level, and employee job category. For each facet menu, one or more checkboxes could be selected to restrict the search. If no boxes are checked under a particular facet, all results are allowed through for that facet. Additionally, a list of "breadcrumbs" are included at the top of the page showing what filters have been selected for each category.

Figure 3: Facet menu for location – partial listing shown here

We considered two approaches to applying the selected filters to our searches. One approach would be to apply the filters to our employee base prior to generating recommendations, and use the restricted set of employees to generate recommendations. However, this would also restrict the set of nearest neighbor tags used to make the recommendations. Thus we chose to follow a second approach where all employees are considered for generating recommendations, and the results are post-filtered to match the selected criteria.

6. CONCLUSIONS AND FUTURE WORK

We have described the design and implementation of a recommender system to identify staff members to fill open job positions. Recommendations are generated based on tags entered by employees describing their knowledge and skills. The recommender was initially designed to simply return a list of results sorted by relevance. During a pilot study of the recommender we identified and implemented a number of additional interface features that enhance the utility of the recommender as a tool for making staffing decisions. These include the ability to sort and filter recommender results based on secondary attributes such as location and job level.

The use of a collaborative filtering recommender for matching users based on their tags depends on the ability to measure tag similarity based on co-occurrence. This works quite well to produce meaningful recommendations, but it is also likely to miss certain relationships due to the nature of users' tag selection. A given individual will probably select tags that are related to each other because their interests encompass a range of related topics. However they are less likely to include two tags that are direct synonyms of each other, or a phrase and its acronym, or two slightly variants of the same concept. Our next steps for the staff recommender therefore, will involve developing additional methods for measuring the similarity between two terms and incorporating these more comprehensive similarity scores into the recommendation algorithm.

After completing the pilot study described in this paper, we are expanding the use of the recommender to additional organizations within the company. We plan to collect more quantitative data on the effectiveness of the recommender in providing useful staff recommendations.

7. REFERENCES

[1] Chen, J., Geyer, W., Dugan, C., Muller, M. and Guy, I. 2009. Make new friends, but keep the old: recommending people on social networking sites. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '09). ACM, New York, NY, USA, 201-210.

[2] Kautz, H., Selman, B. and Shah, M. 1997. Referral Web: combining social networks and collaborative filtering. *Commun. ACM* 40, 3 (March 1997), 63-65.

[3] Maybury, M., D'Amore, R., and House, D., 2002, Awareness of Organizational Expertise. *International Journal of Human-Computer Interaction*, Vol. 14, Issue. 2.

[4] McDonald, D. W. 2003. Recommending collaboration with social networks: a comparative evaluation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '03). ACM, New York, NY, USA, 593-600.

[5] O'Connor, M., Cosley, D., Konstan, J. A. and Riedl, J.. 2001. PolyLens: a recommender system for groups of users. In Proceedings of the seventh conference on European Conference on Computer Supported Cooperative Work (ECSCW'01). Kluwer Academic Publishers, Norwell, MA, USA, 199-218.

[6] Owen, S., Anil, R., Dunning, T., and Friedman, E. *Mahout in Action*, Manning, 2011