Looking for Jobs? Matching Adults with Autism with Potential Employers for Job Opportunities

Author 1 and Author 2*

Abstract

Adults with autism face many difficulties when finding employment, such as struggling with interviews and needing accommodating environments for sensory issues. Autistic adults, however, also have unique skills to contribute to the workplace that companies have recently started to seek after, such as loyalty, close attention to detail, and trustworthiness. To work around these difficulties and help companies find the talent they are looking for we have developed a job-matching system. Our system is based around the stable matching of the Gale-Shapely algorithm to match autistic adults with employers after estimating how both adults with autism and employers would rank the other group. The system also uses filtering to approximate a stable matching even with a changing pool of users and employers, meaning the results are resistant to change as the result of competition. Such a system would be of benefit to both adults with autism and employers and would advance knowledge in recommender systems that match two parties.

keywords: Autism, job search, stable mapping, algorithm, recommender system

Topics: Information retrieval, recommender systems

1 Introduction

Adults with autism are among the most under employed demographics, with a recent report claiming that 85% of them are unemployed [15]. This statistic, however, is not due to an inherent lack of ability of adults with autism, as is proven by the fact that some intervention can improve rates of employment. Assisting adults with autism find employment provides benefits both to the individual and to society. For the individual, gainful employment leads to financial independence which in turn leads to increased opportunities for adults with autism. Meaningful employment also leads to increased self-esteem and general well-being, even leading to increased cognitive ability [9]. For society, increased independence also results in less social expenditure and employment increases tax revenue [10]. More importantly although individuals with autism have special skills to contribute to the workplace if applied to the right job, this talent is currently not being utilized. It is for this reason that there is an urgent need to find a solution to this problem.

One technique that has proven to be successful in improving rates of employment and retention is job matching [4]. Resources for job matching for adults with autism, however, are limited. The problem we tackling is to develop an algorithm to provide an automated job-matching system for adults with autism and potential employers who are interested in hiring adults with autism. Job matching has proven to be a successful technique in helping adults with autism find employment and remain employed. Existing programs that utilize job matching for adults with autism include Swedish corporation *Samhall* (samhall.se) and American corporation *Daivergent* (daivergent.com). Samhall's system includes matching client's abilities with employer's demands using a system

*Double blind

that measures 25 different traits (covering sensory function, intellectual ability, mental ability, social ability, and physical ability) on 3 levels (limited, good, and high ability on the client side corresponding to low, medium, and high requirements on the employer side) [16]. Daivergent uses artificial intelligence to match vetted candidates with jobs that they extracted from descriptions using machine learning [3]. Unfortunately, there is little public information about how these corporations provide their matching beyond what details they choose to share with the public, both limiting their services to their clientele and restricting potential research contributions from studying their systems.

To solve this problem, we develop a *job-matching* system so that both users, i.e., adults with autism, who look for work, and employers can autonomously create profiles and then be automatically matched. Like Samhall, this matching is done by quantifying the skills of employees and demands of work on multiple axes. The goal is to a pick a match that *minimizes* the discrepancy between the user's skills and the employer's demands as this will minimize the amount of skills the user would need to develop and accommodations that the employer would have to make. In addition to measuring the job tasks itself, demands for the application process such as required interview skills are also included as part of the work demands.

Our job-matching system takes into account not only the *skills* of the user, but also their *interests*. This not only respects the desires of the individual, but also leads to much greater productivity [13]. As we assume that the user's preference is based principally on their *interest* while an employer's preference is based principally on the ability for a worker to perform the task *effectively* according to their *skills*, *interest* aspects are measured separately from *skill* aspects. The fact that the user's preference and the employer's preference can differ significantly necessitates a system for finding a compromise between the two. From the *interest* aspects, an automated ranking of employers is generated for each user, and from the *skill* aspects, an automated ranking of users is generated. These different rankings are combined into a single match using the Gale-Shapely stable algorithm [6], which finds a *stable match*. **Stable** means that no two participants may both have a higher ranked match with each other than who they were already paired with in the stable match.

The proposed job-matching system advances both technology for assisting adults with autism in finding jobs and the knowledge of matching systems in general. The fact users act autonomously in this system means that the labor costs associated with current job-matching systems can be reduced. Positions are also extremely limited in existing job-matching systems that are geared towards adults with autism, so this system acts as a potential starting point for increasing access to many more adults with adults. It can potentially be extended to serve other populations as well. Moreover, our job-matching system differs from existing systems in that it is open to use for anyone who wishes, no manual vetting is required. It is also open source, so it may be built upon for further research.

We proceed to present our job-matching system as follows. In Section 2, we discuss existing work that are related to job-search approaches and their significant differences with ours. In Section 3, we introduce our job-matching algorithm and detail the design methodology of our matching approach. In Section 4, we include a study based on simulation that analyzes the performance of our proposed job-matching system. In Section 5, we give a concluding remark.

2 Related Work

While numerous companies use their own job-matching algorithms, academic research on the subject exists as a specific application in the broader field of recommendation algorithms. CASPER

[14], one of the proposed job-recommendation algorithms, focuses on clustering users based on their activities while reviewing jobs so that collaborative filtering may be applied. Malinowski et al. [12], on the other hand, use the content-based filtering approach based on profiles that are manually entered by users. Much of the latest research works on job-matching relate to processing data from resumes and other sources so it can be used, with Resumatcher introduced by Guo et al. [8] in particular who seek to match similar profiles based on extracting data from unstructured resumes and job descriptions. Others, which focus on comparing unstructured data so that similar profiles may be matched, include the self-reinforcing model proposed by Koh et el. [11] and the collective-learning approach developed by Cing [2].

Even though research works on algorithms for matching autistic adults with employers is minimal, there is substantial works on the subject of autistic employment in general. Of particular interest is the work of Dreaver et al. [4] who tackle the problem from an employer's perspective. In addition to suggesting matching, they emphasize the importance of external supports and employers understanding autism. Indeed, most research works focus on the perspective adults, with numerous studies supporting the efficiency of Behavior Skills Training (BST), especially when combined with prompting and audio cues. Grob et el. [7] managed to achieve a 100% success rate at teaching skills using BST combined with prompting, while Burke et el. [1] found that BST combined with audio cues is six times as effective as BST by itself.

We observe that although existing job-recommender algorithms cannot assign multiple users to the same job, existing job-matching strategies can. Moreover, existing job-matching approaches attempt to automate the recruitment stage in the hiring process [2], which aims to make suggestions to many interested job seekers instead of selecting one. However, this process has failed to adequately serve the autistic population, and for this reason we seek to work around it. In addition to specifically tailor to the autistic population, our job-matching algorithm differs from existing ones in that it relaxes the restriction on the one-to-one matching between users and employers so that those who perform poorly in the existing system can still have unique jobs suggested to them. Instead of looking at the similarity between profiles in a single vector space, ours looks at the similarity in two different vectors spaces that cover *aptitude* and *interest* separately and uses this information to define a *stable matching.*

3 Our Job-Matching Algorithm

The central part of our job-matching algorithm is an extension of the Gale-Shapely stable-matching algorithm [6] so it may be applied in cases where the basic algorithm cannot be. While our algorithm is not the first extension of the Gale-Shapely algorithm, it is the first to use the algorithm to generate a *ranking* rather than a single match. While such a ranking is not useful or even meaningful to all applications of the Gale-Shapely algorithm, it works well with the assumptions made in this problem of matching adults with autism with potential employers. The idea of generating a *ranking* makes sense in this context, since it is based on the assumption that the information the model has is mostly accurate information, but may have incomplete information relating to the preference of the user making their choice. Users then complete the missing information by selecting their preferred employer from the ranking. We can assume that getting this information after the ranking is done does not violate the integrity of the results because the algorithm is strategy proof from the perspective of the users [5]. This means it is in the best interest of the users to accurately give their interests as far as they can. Our algorithm could also potentially be applied to other problems where the objective is to approximate a stable solution in a two-sided market. Technology that could use such problems includes applications ranging from dating apps to tools analyzing various financial markets. This novel ranking idea may help fill in missing information that was missed during other parts of an automated process when tackling these problems.

In this section, we first present the client and server sub-systems (in Section 3.1). Hereafter, we introduce the novel extensions of the Gale-Shapely Algorithm (in Section 3.2) and our stable-matching approach (in Section 3.3). Afterward, we proceed to discuss profile creation and matching (in Section 3.4) and detail the design of our matching algorithm and ranking approach (in Section 3.5 and 3.6), respectively.

3.1 Server and Client Specifications

Records containing user profiles and employer profiles, as well as additional information associated with them, are stored on the server. The record for a particular user contains the user's username, password, a text description detailing whatever the user is offering, a binary flag specifying if they are an employer or not, a sequence of double precision floats representing the user's *profile vector*, and a flag specifying if the user has finished creating their account (see Figure 1 for an example). Similarly, an employer profile contains the corresponding information about an employer. It is on this server that our job-matching algorithm is performed. The server is designed so that a user (an employee, respectively) can communicate with it using a specifically designed client program, and it supports six different contexts through which the client can send messages. These contexts are *Home, Register, Login, Delete, Update*, and *Match* (see details in Section 3.1.1).

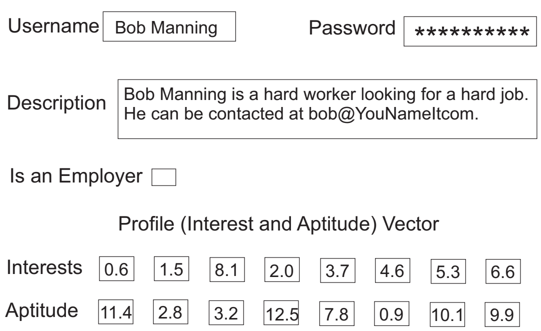


Figure 1: A graphical representation of a system record

3.1.1 The Server's Contexts

Sending a message to **Home** is used to establish a secure connection with the server. (See Figure 2 for the layout of the client-server architecture of the proposed system.) Messages to all other contexts are encrypted as they, in the very least, include the user's password, and often contain other potentially sensitive information as well.

The **Register** command is used to create an account by sending to the server the username and password for the account that the user wishes to create. As long as the username is not associated with any existing records, a default record will be created containing that username and password. By default, the text description is an empty string, the user is not an employer, the vector is set to random values, and the account is not complete.

The **Login** command retrieves the record corresponding with the username that is sent to the server and the user's record is forwarded to the client if the password matches what is in the record. All the other remaining commands will also only be undertaken if the *password* matches what is in the record for the username that is given.

Delete causes the server to remove the record corresponding with the username that is passed to it, and **Update** replaces the record for the user with one corresponding to a serialized record sent with the request, and all values in the record other than username and password are set.

Finally, **Match** returns the usernames and descriptions of the user's top ranked matches. Before the user's Match request may be satisfied, their account must be marked as being completed, and only completed accounts will be considered when running the matching algorithm.

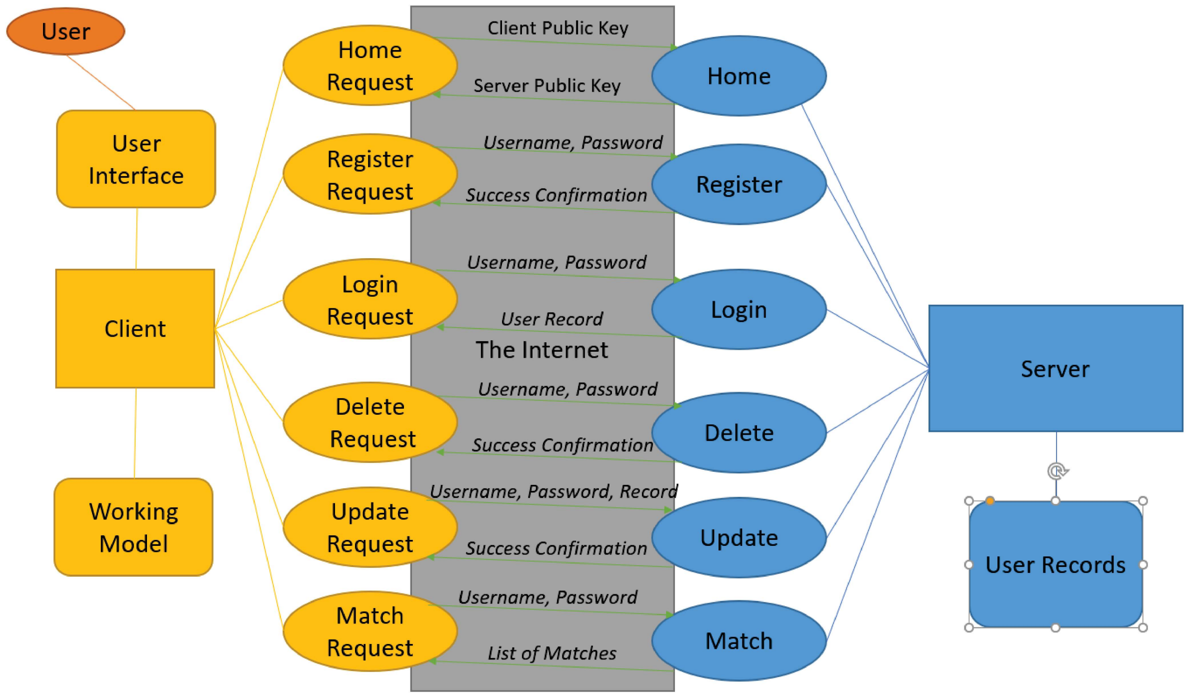


Figure 2: The user's computer (on the left) communicates with the server (on the right) over the Internet through the client. Messages (in italics) over the Internet are encrypted.

3.1.2 Overview of the Client

The client can be run from a JavaFx application and communicates with the server on behalf of the user. This application provides a user interface with buttons and text fields so the user may provide the client the information it needs to send its requests in a format that the user understands. The application also maintains a working model of the user's intent based on user input. To guide the user through creating and using their profile, the application is divided into several pages, i.e., XML documents, that are navigated through by pressing buttons in the user interface (see the configuration in Figure 3). Each page defines what the user sees, including text, buttons, and text fields. Some button presses also cause the client to send a request to the server, in which case the application must wait for the server response before it changes pages.

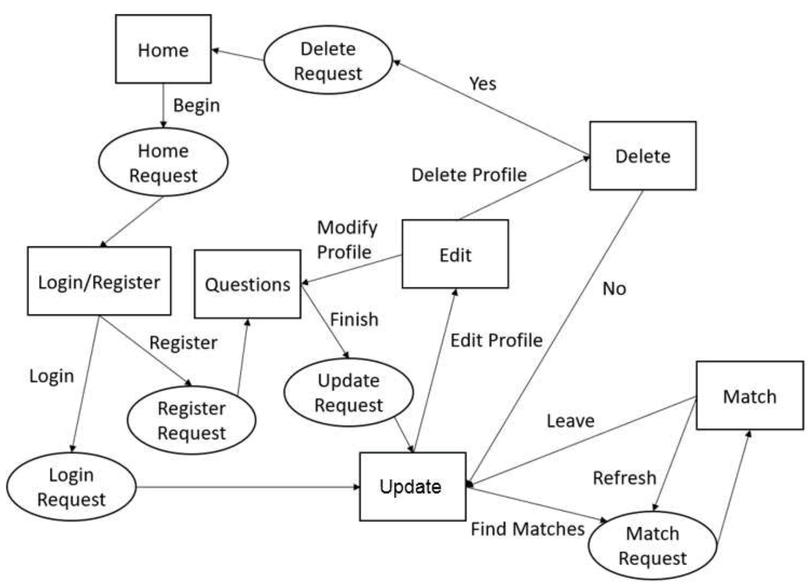


Figure 3: The structure of the client app. *Rectangles* are pages, i.e., XML documents, *ovals* are requests to the server (one for each context), and *arrows* are buttons.

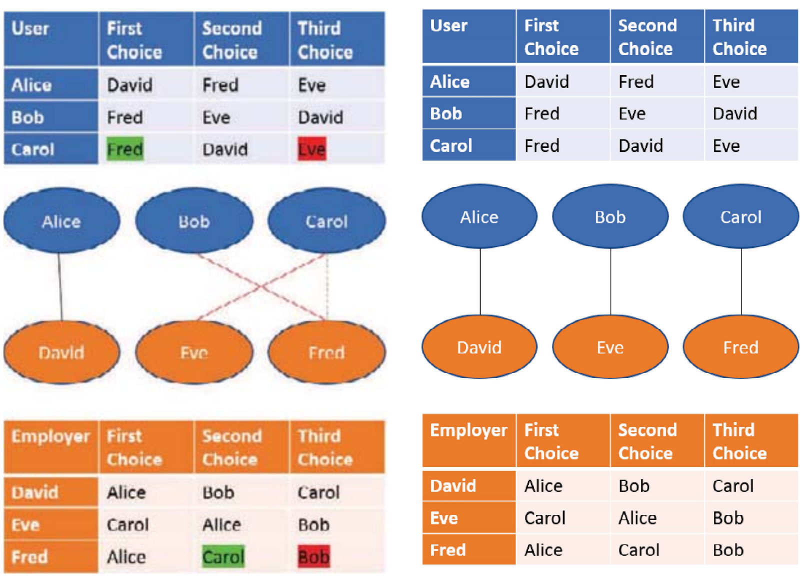
3.2 Novel Extensions to the Gale-Shapely Algorithm

Our implementation of the Gale-Shapely algorithm, which is designed for matching students with universities for college admissions and matching couples for marriage, is augmented with two novel extensions. This includes (i) a routine for recursively applying the algorithm in order to generate a *ranking* for a user instead of just a single match, and (ii) a *filter* so that the algorithm can both run faster and run even when the number of users and employers is <u>not</u> equal. As there may be some inaccuracies in the stable matching due to an inability to perfectly capture all information about a user or employer that may be of interest to the opposite party, a *ranking* of matches is given rather than just a single match so that the user may choose for themselves along the matches. Our method of augmenting the Gale-Shapely algorithm so that it may be used for ranking is unique to our work. This ranking method requires a *filtering* system, which is another augmentation to that algorithm that is also needed to ensure that the Gale-Shapely algorithm can be applied even with a *dynamic* set of users and employers. This relaxes another restriction on the Gale-Shapely algorithm, which requires a static set of users.

3.3 Stable Matchings

While this is not the first matching algorithm to be applied to helping adults with autism find employment [3, 16], it is the first where all the implementation details are publicly available, and the matching algorithm used is *novel*. It is based on the *Gale-Shapely algorithm*, but it is augmented with original features.

A matching algorithm can fulfill different criteria, with the Gale-Shapely algorithm finding the *single stable matching* which is optimal for one of the two parties that it is matching [6]. A matching is defined as a one-to-one map between two parties, with the pairing between a user and an employer called a *match*. Furthermore, a matching is *stable* if <u>no</u> two participants may both have a *higher ranked match* with each other than who they were already paired with in the stable matching. (Figure 4 gives examples of an unstable and stable matching for the same data set.) The reason we are choosing to find this stable matching rather than fulfill a different criterion is because our matches are *non-binding*, with either partying being free to accept or reject the match, since the user still needs to apply for the job afterwards and it's still up to the employer's discretion to accept the application. If a stable matching is accurate, then both the user and employer should have no reason not to accept the match, since they would not be able to find a better partner they were



(a) An unstable matching (b) A stable match

Figure 4: The red bold lines in 4(a) show an example of an unstable matching for a given set of rankings. It is unstable, since *Carol* and *Fred* would rather match with each other (dashed line) than their given match (Eve and Bob, respectively), whereas 4(b) is a stable match for the same data set.

matched with and who would also reciprocate their choice. If the recommended employer for a user is not a pairing from a stable matching, it may be in the advantage of the user or employer to ignore their matching, defeating the purpose of suggesting that match.

For a given dataset, *multiple* stable matchings may exist, and it is possible to find the *optimal* stable matching according to arbitrary objectives [19], but we are choosing to just use the one found by the Gale-Shapely algorithm. The linear programming algorithm necessary to find other stable matchings is both harder to implement and slower than the Gale-Shapely algorithm, so there must be a compelling reason to optimize a different objective in order to justify using the more complex algorithm. We consider finding the optimal stable matching from the perspective of the user's to be a good objective for the sake of benefiting the autistic community and using the Gale-Shapely algorithm is sufficient to reach it.

3.4 Creating Profiles for Matching

Before users can be matched with employers, individuals in both parties need to create profiles for themselves within the system. When making a profile, someone first specifies if they are looking for or offering employment, which determines if they are a user or an employer, respectively. In either case, the user or employer will be walked through more questions to continue building their profile. Users will be asked questions to figure out both what jobs interest them and what skills they have. (See Appendix A for the sample set of questions for user and employers created for the job-matching system.) Based on their responses to the questions, a *numerical record* will be generated with different fields corresponding with different tasks. This numerical record has two parts, an *aptitude* portion corresponding with *skills* and an *interest* portion corresponding with interests. Employers, on the other hand, will be asked questions about the job they are offering to determine the qualities required to obtain and excel at the job, and what qualities the job has which may be of interest to a user. A *numeric record* is generated for them as well whose fields directly correspond with those in a user's record—for requirements, the larger the number in the employer's field means the more of the corresponding skill is demanded from the user. This defines the employer's *aptitude vector*. Similarly, the same number in the user's corresponding field means it matches a user's *interest* in that respect, defining the employer's interest vector. The system for guiding users through creating their profiles and storing the corresponding records is detailed in Section 3.1. When employers and users are matched with each other, users will evaluate employers with similar interest profiles, and vice versa.

3.4.1 Searching for Matches

After a user has created their profile, s(he) has the option to look for jobs by pressing a button. This will trigger the *matching* algorithm which will then return a list of ranked jobs to be displayed for the user. The list includes the *name* of the employer offering the *job* and information about the job. The user can always re-press the button to *refresh* the list of jobs in case activity from other users and employers has changed the results, which just runs the matching algorithm again for whatever user and employer profiles are currently in the system, but there is also a second button to *reject* all the jobs in the list. Choosing the latter will change the user's record to include flags that specify that the user is <u>not</u> interested in those jobs and will not include them in further matches, and also *update* the user's interest components of their profile to be further away from the rejected profiles. The specific details in how this is done is described in Section 3.6. When running the matching algorithm, users will *rank* employers with similar *interest vectors* higher, whereas employers will rank users with similar *aptitude vectors* higher. The *interests vectors* (*aptitude*

Table 1: Sample questions used for creating the **aptitude** vectors

User's Feedback			**Objectives**	**Employer's Feedback**		
Question	*Response*	*Score*		*Question*	*Response*	*Score*
How are your interview skills?	Poor	1.2	**Interview Skills**	How important is presentation during interview?	Very	3
How well can you function in a noisy environment?	Manageably	2.3	**Noise Tolerance**	How noisy is your workplace?	A little bit	2.5
How loyal are you?	Extremely	4.9	**Loyalty**	How important is company loyalty?	Slightly	2

Table 2: Sample questions used for creating the **interest** vectors

User's Feedback			**Objectives**	**Employer's Feedback**		
Question	*Response*	*Score*		*Question*	*Response*	*Score*
How much do you like task consistency?	I like strong consistency	2.4	**Task Consistency**	How consistent are the job tasks?	They are quite repetitive	3.1
What social culture do you desire in a workplace?	I prefer not to interact with others while working	0.9	**Social Culture**	What is the social culture in your job like?	There is a moderate social culture	2.2

vectors, respectively) of the employers (users, respectively) measure how much the users (employers, respectively) are satisfied with the employers' requirements (users' qualifications, respectively) as used by our job-matching algorithm.

3.4.2 Filtering Profiles

Our *matching algorithm* is based around the Gale-Shapely algorithm, but includes some additional steps to ensure that prerequisites for using the Gale-Shapely algorithm are satisfied, and so that it can be used to generate a *ranking* rather than just a single match. First, *stable matchings* are only defined when the two parties being matched have the same number of participants, so the Gale-Shapely algorithm itself requires that there be the same number of user and employer profiles being matched. To ensure this, a *filtering* scheme that guarantees an equal number of users and employers is applied so only the user and employer profiles which are predicted to be most likely to impact who the target user is matched with are considered for matching. Specifically, the filtered employers will consist of jobs the target user is most likely to *apply* for based on being closest to what he is most *interested* in, and jobs the target user is most likely to *succeed* at based on their *aptitude* meeting the employer's *requirements*. (Table 1 shows a number of sample questions used for creating the *aptitude* vectors.) Meanwhile, the filtered users will be the users the target user is most likely to compete with when going after jobs they are interested in based on having similar *interests*. (Table 2 includes a number of sample questions aimed for finding the *interest* of a potential employer and user.) Our filtering scheme is based on both requesting the n (≥ 1) different employer profiles representing the jobs the target user is estimated to be the mostly likely to succeed at, and the m (≥ 1) employer profiles that the target user is predicated to be the most interested that are not already being considered with the previous request. Multiple studies have confirmed that with randomly generated rankings the expected ranking for the final match is $log(\textit{Number of Profiles})$ [18]. For this reason, we advise setting n and m to be around log of what is the estimated maximum number of users. In our implementation, we set $n = 100$ and $m = 50$, arbitrary values from a range estimated to be high enough to obtain accurate results and small enough to run in a reasonable amount of time.

To calculate the n jobs the user is most likely to *succeed* at, we first calculate the discrepancy between a user and an employer as taking the sum of squared differences in all the fields (represented as components of a vector) in the employer's profile where the employer's requirement exceeds the user's skill as denoted by the corresponding field in their profile and as shown in Equation 1.

$$Div_{App}(U, E) = \sum ReLU(E_{A_i} - U_{A_i})^2 \tag{1}$$

where $ReLU(X) = X$, if $X \geq 0$, otherwise, $ReLU(X) = 0$, and E_{A_i} is the i^{th} component of the Employer's *aptitude vector*, and U_{A_i} is the i^{th} component of the User's *aptitude vector*.

Example 1 Assume that under a particular scheme, a user vector with *aptitude* components is <1.2, 2.3, 4.9>, which represents their *interview skills*, *noise tolerance*, and *loyalty*, while an employer with *aptitude* vector is <3, 2.5, 2>, meaning the employer puts significant emphasis on *presentation during interviews*, the environment is *moderately noisy*, and (s)he weakly desire *company loyalty*. The user's fit for the job would be interpreted that (s)he is *sufficiently loyal*, and would only need *minor accommodations* at most to handle the noise in the environment, but (s)he would need to work on his/her *interview skills* significantly to likely to succeed at applying to the job. This corresponds with component-wise discrepancy of 3.24, 0.04, and 0, which sums to 3.28 for the *total discrepancy.* □

As shown in Example 1, the discrepancy represents extra labor the user must expend to reach the demands of the job, or additional accommodations the employer must make to be accessible to the user, since a higher discrepancy means a user is a worse fit for the job. We can then take those n employers with *minimal* discrepancy between them and the target user by ranking the profiles in order of *increasing* discrepancy and keeping only the top n in the ranking. In the case of a tie where two employers have the same discrepancy from the target user, the first of the two employer profiles to be created is given priority in the ranking, assisting employers who have been waiting longer be reached in the system. The top m profiles that the target user is predicted to be the most *interested* in are calculated in a similar way, but with a different formula for discrepancy. This is done by comparing the fields (again represented as components of a vector) related to *interest* instead of those relating to *skill*, and including all fields in the sum, not just those where the employer's value is greater than the user's as shown in Equation 2.

$$Div_{Int}(U, E) = \sum ReLU(E_{I_i} - U_{I_i})^2 \tag{2}$$

where E_{I_i} is the i^{th} component of the Employer's interest vector, and U_{I_i} is the i^{th} component of the User's interest vector.

Equation 2 is equivalent to the Euclidean distance between the *interest* fields of the profiles as modeled as vectors in a normed space. This *discrepancy* represents the divergence between a user's ideal job and the given job. A potential scheme for *interest* includes consistency of tasks, and the social culture of the workplace, with higher values denoting more consistency in tasks and a more prominent social culture.

Example 2 Assume that the *interest* vector of an adult with autism is <2.4, 0.9> under our encoding scheme, meaning (s)he would like *strong consistency* in their tasks and would prefer not to *interact with others* while working. It is further assume that the *interest vector* of an employer is <3.1, 2.2>, suggesting that the work is quite *repetitive* and that there is a *moderate social culture* in the workplace. The component-wise discrepancy is 0.49 and 1.69, which sums to 2.18. □

If there are less than $n+m$ (= 150 in our implementation) employer profiles in the system, then all of them will be considered, and these calculations for *discrepancy* to obtain the top n and m employer profiles can be skipped. As a result, either $n+m$ employer profiles will be considered after filtering is applied, or no filtering will be applied and all of them will be considered, in which case we define E as the total number of employer profiles.

$$N = min(n + m, E) \tag{3}$$

where N is the number of filtered employer profiles and is also the number of filtered user profiles to be considered so that we can ensure that the number of users being considered is equal to the number of employers.

Next the N-1 user profiles with the most similar *interests* to the target user are considered so that, together with the target user, N user profiles will be considered, ensuring that the same number of user and employer profiles are considered after filtering. An example of the result of filtering with $n = 2$ and $m = 1$ is shown in Figure 5.

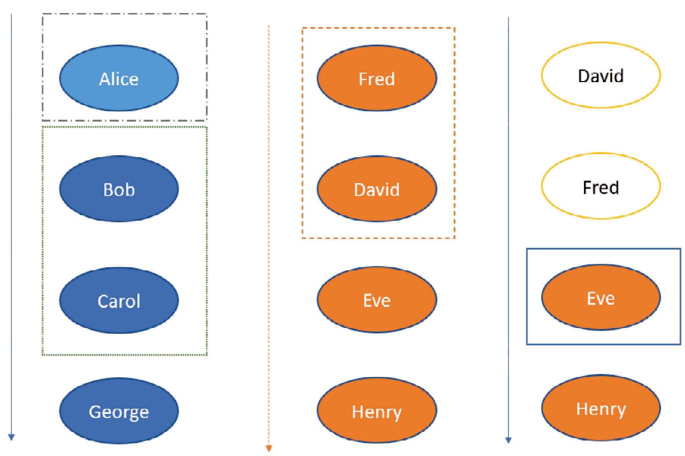


Figure 5: Users (blue) and employers (orange) being considered for matching are enclosed in rectangles. Each column is a sorted list, with the arrow on the left showing if the list is sorted by *interest discrepancy* (blue/solid) or *aptitude* discrepancy (orange/dashed). It is assumed that Alice is the target user, so all discrepancy is measured relative to her. Note that $n = 2$ in this case, so Fred and David are enclosed in the orange (dashed) rectangle, and $m = 1$, so Eve is enclosed in the blue (solid) rectangle after David and Fred are passed over due to already being considered. Alice herself is the grey (dot and dashed) rectangle, and $n + m$ - 1 users closest to Alice are in the green (dotted) rectangle. Together, three users and three employers are being considered, so a matching is defined.

The reason users with a similar *interest* are considered is because they are the most likely to compete with the target user for their preferred job and thus affect the results of the Gale-Shapely algorithm. The corresponding *discrepancy* in *interest* is calculated in the same way as the m employers the user is most likely to be interested in, by calculating the Euclidean distance between their interest records, and keeping those with the lowest scores. To ensure N-1 users can always be filtered, $n+m$-1 mock user profiles are included within the system in addition to real user profiles. These mock profiles do not correspond with any individuals and just exist to ensure the algorithm can be applied. They are randomly generated, but form a distribution that matches that of real user profiles, including potential profiles that would correspond with non-autistic individuals in order to simulate wider competition. While these mock profiles may influence the results of the algorithm as they simulate competition, they will never be target users, and will never compete with actual users when users apply for jobs after being matched. The fact that the filtering never returns more than the requested number of profiles means that the matching that follows will run in *constant time* relative to number of profiles, improving over the *quadratic time* of the unconstrained Gale-Shapely algorithm, though the computational time for filtering grows *linearly* with the number profiles. As a result, the overall time complexity for matching a single user is *linear*.

3.5 The Modified Gale-Shapely Algorithm

The next pre-requisite the modified *Gale-Shapely algorithm* needs before it can run is to be provided information about how each user being considered will rank each employer profile being considered, and vice versa. For this, users rank employers by how *interested* they are in them, whereas employers rank users by how likely they are estimated to *succeed*. These ranks are calculated in the same way they were calculated during the *filtering* process, by sorting the calculated discrepancy so that those with the *lowest discrepancy* are most preferred. With the rankings generated, the modified Gale-Shapely algorithm is now applied to find a *stable matching*.

When the algorithm starts, users are labeled as being not considered matched, but all users are labeled as being considered matched when it stops. The modified Gale-Shapely algorithm consists of applying the following loop, called the **Gale-Shaley Loop**, that matches and un-matches users until every user is considered to be matched with an employer. At that point these matches are now considered as the official matches which are returned.

1. Every user who is not considered to be matched is considered as a potential match to their current top ranked employer.
2. Each employer who becomes matched to their top ranked user is being considered as a potential match. The other users who were being considered to them will no longer be considered to be matched, and will now consider their next top ranked employer as their current top ranked employer.

Example 3 Figure 6 shows a complete run of the modified Gale-Shapely algorithm on a simple dataset. Step 1 is the initiation, and every following step alternates between Step 1 and Step 2 in the Gale-Shapely loop. *Potential matches* are blue (asterisk/dotted), *matches* are green(plain/solid), and *rejected matches* are red(xed/dashed). This example does not show any cases of former matches becoming rejected (going from green to red), but such behavior is possible. □

3.6 Generating Ranked Results

The employer the target user is matched with is returned as their top suggested employer. To generate the rest of the ranking, assume that the user was not interested in the most recent employer that was suggested to them. To take into account this scenario, the components in the vector representation that measure the user's interest would be updated to be further away from the components in that employer's record. This is done by subtracting a weight[1] multiple of the component of a employer's profile vector from the corresponding component in the user's profile vector for each component representing interest as shown in Equation 4.

$$U_I' = U_I - w \times (E_I - U_I) \tag{4}$$

[1]The value of the weight is determined empirically with the goal of having users choose higher-ranked matches. This can be determined by finding an approximately optimal solution to minimizing the aggregated rank users' choices based on experimental data.

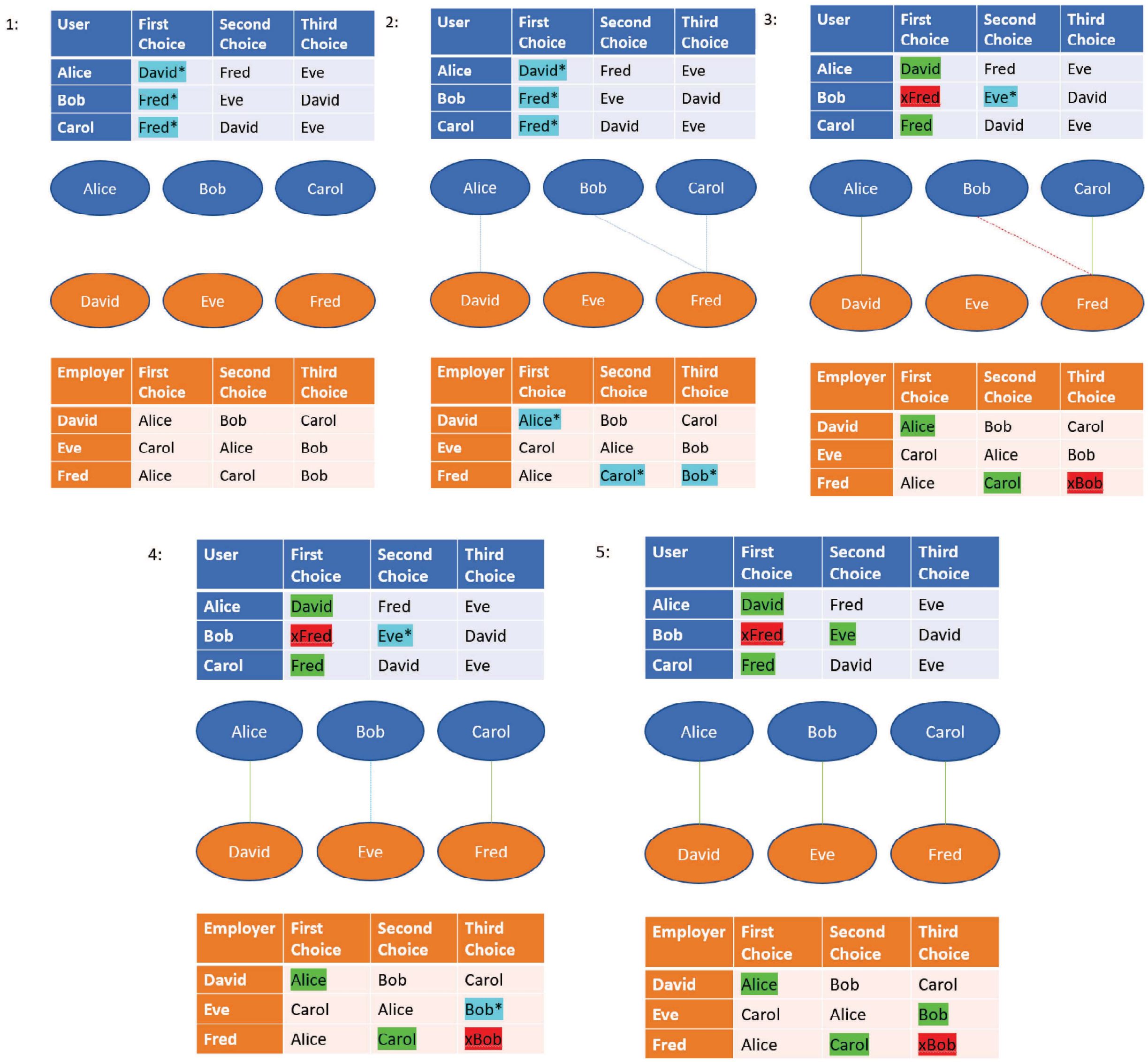
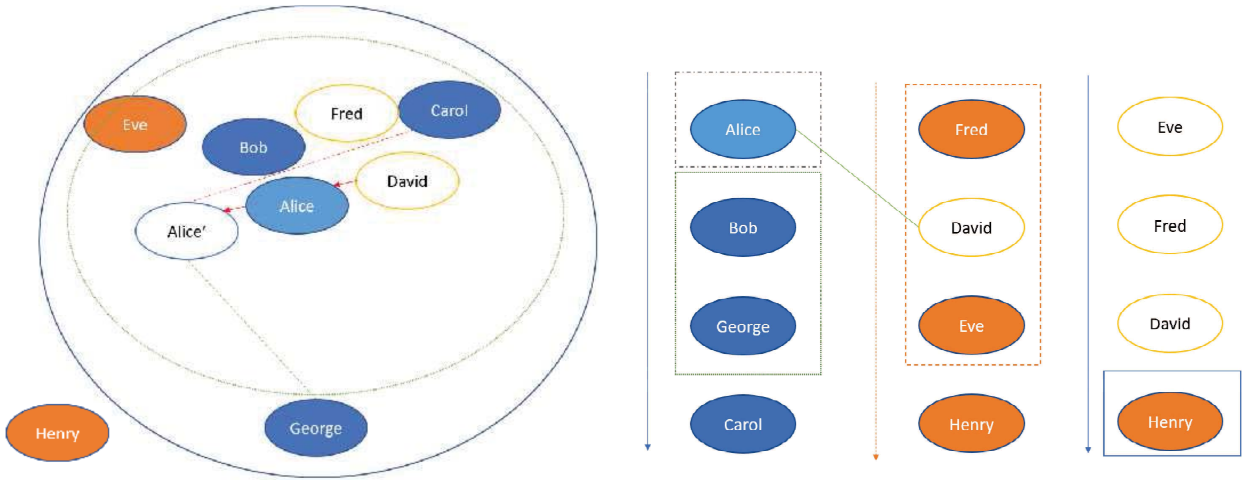


Figure 6: An example of running the modified Gale-Shapely algorithm

where U_I is the target user's *interest* vector, E_I is the matched employer's interest vector, w is the weight given to negative feedback, and U_I' is the target user's updated interest vector.

Example 4 Consider the *interest* vector of an adult with autism as shown in Example 2, which is <2.4, 0.9>, and the *interest* vector of the matched employer, which is <3.1, 2.2>. Further assume that the weight w is 0.5. Hence, the difference between the interest vectors of the two profiles is <0.7, 1.3>, and the updated user interest vector in his profile is <2.4, 0.9> - 0.5 × <0.7, 1.3> = <2.05, 0.25>. A visual representation of this change is show in Figure 7(a). □

The matched employer will also not be further considered when filtering employers. With this in mind, the rest of the matching algorithm can be re-applied with the updated information, i.e., updated interest fields and ignoring the last recommended profile, in which case it will generate a new matching and return a new match. (See Figure 7(b) for a new set of candidates to be considered for matching.) This new match can be returned as the next suggested employer. The process can in theory be repeated until every employer has been ranked, but it only needs to be



(a) Updates of the Interest vectors

(b) Changes of profiles

Figure 7: (a) The green (dotted) circle encloses the closest users in terms of interest to Alice according to her original interest vector, while the blue (solid) circle encloses the closest employers. Alice' interest vector is at the position of the new vector after moving away from David, her top-ranked match. Alice's interest vector is closer to George than Carol, changing who is considered in the next match. However, Eve remains closer to Alice than to Henry. (b) The filtered results for the second match. Note that the *interests* sorting has changed from the first match, but not the *aptitude sorting*. Since David was already found as a match, he is no longer being considered, changing who is considered in both the n closest by *aptitude* and the m closest by *interest*.

applied until the specified number of employers to be displayed to the user have been ranked, at which point they are displayed to the user. This same process is used to update a user's profile if the user rejects all the matches, making it so that if k results are listed at a time, then rejecting the results would cause ranked results $k{+}1$ through $2k$ according to the original profile vector to be displayed instead, pressing it again would display results $2k{+}1$ through $3k$, and so on. This process of iteratively applying the modified Gale-Shapely algorithm on filtered results to create a ranking is *novel*. The optimal value of k depends on what is practical to display on the screen to the user while still retaining ease of use, we have chosen $k = 5$. Shown below is the modified *Gale-Shapely* algorithm.

Algorithm. Modified Gale-Shapely Matching Algorithm (User, Employers, Users) Begin

1. Let *Target User* be *User*, *Potential Employers* be *Employers*, and *Potential Users* as *Users* excluding the *Target User*

2. For ($i = 0$; $i \leq k$; i++) DO

   (a) Apply filter on *Potential Employers, Potential Users, Target User* to obtain N *filtered-employers* and N *filtered-users*.

   (b) Call *Gale-Shapely algorithm (Filtered Employers, Filtered Users)* [5] to obtain the *Stable-Matching map*.

   (c) Select the match for the *Target User* from the *Stable-Matching map* and return the *Matched Employer*.

   (d) Let the i^{th} ranked match as the *Matched Employer*.

(e) Update the interest vector of *Target User* as $TargetUser_I$ - $w \times (Match_I - TargetUser_I)$, where X_I denotes the interest vector of the user or employer, and w is the weight.

(f) Let *Potential Employers* as *Potential Employers* without the *Matched Employer*.

3. Return the ranked matches

End

4 System Simulation and Results

While we are supposed to test our solution on end users to see if the designed system is superior to other solutions in practice, we prove that some aspects of our solution are superior to other solutions instead, at least under certain conditions. Specifically, we have tested our matching algorithm in a simulated environment, showing that our matching algorithm is superior to comparable algorithms for the case that was simulated. This simulation was coded in Java and ran in the IntelliJ coding environment.

4.1 The Simulation

For the simulation, we created a workable system where users and employers continuously enter the system and leave when they are either hired or hire someone, respectively. Users in the system request matches and apply to one of the matches suggested to them. After a user applies to the job that an employer is advertising, the employer will decide whether to hold, reject an applicant, or eventually hire an applicant that they held. If a user is hired, they will be removed, along with the employer, and the count of successful hires will be incremented. The simulation ceases after a set number of steps, which was 100 in our simulation, and the number of successful matches is returned. This count is used to compare different matching schemes, which are treated as individual algorithms in this empirical study.

During the simulation process, we considered four different matching schemes, denoted "Matched", "Interest", "Aptitude", and "Mixed". **Matched** refers to ranking users using our own matching algorithm, whereas **Interest** and **Aptitude** rank users by interest and aptitude divergence, respectively, and **Mixed** refers to ranking users by the sum of aptitude and interest divergence when they are being recommended to a target user. As these different schemes are based on making matches from the same distribution of vector spaces, they are comparable, and we know that differences between these results must be due to the matching algorithm itself rather than due to what data they utilized. Our matching scheme differs both from the comparable schemes and other existing matching schemes as mentioned in Section 2 in that the former looks at the competition along the *two* aspects to find stable machines, whereas the latter just try to optimize the discrepancy between employer and employee. For each matching scheme, we ran the simulation 100 times and calculated the *average* number of successful matches, as well as the variance of successful matches in the sample, so that *null hypothesis* testing could be performed. Based on these statistics, the null hypothesis that our matching scheme performs equally well or worse to comparable matching algorithms in terms of average successful matches under the parameters of the simulation is *rejected*, proving that our method is *superior* in the case that was simulated.

Every time the *step event* is executed, new users or employers will be generated from the same *geometric distribution* to simulate users continuously entering the system. This geometric distribution is defined by its *stopping probability*, which is 0.3 in our simulation[2]. Each time after a user or an employer is generated, an event for making their profiles will be queued.

[2] 0.3 is the probability that generation will stop after each user or employer is generated and the step event ends.

Once a user has created a profile, the user will queue an event to request matches. Out of the list of given matches, the user applies for the one that (s)he like the most in terms of interest as based on the true vector's values. The user will then wait until (s)he has received the rejected or hired confirmation. If rejected, the user will request matches again by queuing another match event and the application process will repeat. A constraint is that a user will not apply to the same job twice. If a user has applied to every job, (s)he will wait before refreshing the job list by queuing another match event to be executed next step.

In addition to the *aptitude* vector, an employer will have a *threshold* for each component of the vector, and these thresholds denote the requirements for a job. After an employer creates his/her profile, the employer will queue a *wait event* and stick around until (s)he has users in their application queue. If an employer has applications in their queue, (s)he will test the first user in the queue according to their threshold and remove him/her from the application queue. If a user is tested and the user does not meet the threshold in some component, meaning the threshold for that component is greater than the value in the user's true *aptitude vector*, the user will be rejected. If the employer rejects a user while not holding any users, then the employer will have to lower the threshold to be between the previous value and the value of the rejected user's component according to a *linear function*, simulating the employer becoming more willing to accommodate as they are unable to find someone who can fulfill the requirements. In our simulation, the *adjustment weight* was set at 0.5, meaning that new threshold is set halfway between the old threshold and the user's component. If the tested user meets all the thresholds, (s)he will be held if no user is currently being held.

Once an employer has held an applicant, (s)he will wait a fixed number of steps before hiring the applicant. In our simulation, the wait lasts ten steps. If a tested user both has a lower discrepancy than the held candidate and is accepted based on the thresholds, then (s)he will be held and the previously held user will be rejected. However, if a tested user has a lower discrepancy than the held candidate but is rejected due to unable to meet the threshold in some component, the employer will lower the threshold, simulating the employer moving to accommodate users that are generally better for the position, but are not currently being accommodated. Once a user is hired, both the user and the employer will be removed from the system and will not be considered by the matching algorithm.

4.2 The Simulation Results

The results obtain from our simulation run are shown in the following table:

Scheme	**Matched**	**Interest**	**Aptitude**	**Mixed**
Average	69.72	48.99	42.10	48.47
Variance	130.57	82.68	84.20	96.82
Z-score for Matched $\geq$ *Null Hypothesis*	N/A	4.49	5.96	4.46
Probability of z-score $<$ *Null Hypothesis*	N/A	3.62×10^{-6}	2.0×10^{-9}	4.22×10^{-6}

Psychology uses a p-value of 5, meaning that if the probability of the z-score under null hypothesis is less than 0.05, then the results are *statistically significant*, and the null hypothesis should be rejected. In this case, all the probabilities are well below 0.05, and thus we can safely claim that our method is more effective than comparable methods for the given parameters.

4.3 Observation

While the simulation was only run for the given parameters, these parameters were chosen arbitrarily within a simulation scheme designed to emulate the behavior of real-world agents. Since we lack real-world data to fill in the parameters, and it is not possible to run the simulation on every possible combination of parameters, we can assume that the results from this set of parameters are as valid as any other arbitrary set of parameters. This is enough to establish the potential superiority our method may have it applied in the real-world.

5 Conclusions

Anecdotal reports from companies with programs that bring in workers with autism show that having such workers changes the attitudes of their co-workers [17]. It opened the minds of these workers to not only being accepting of a more diverse population, but to also consider different problem-solving strategies. The fact that employees with autism are now financially independent also decreases financial strain on relatives and their generally increased well-being reflects positively on everyone they interact with. In this paper, we have proposed a job-matching algorithm for adults with autism and potential employers that connects adults with autism with the necessary job skill required by the potential employers and contribute to the autistic community and companies who can benefit from hiring diverse work force.

Since our job-matching algorithm is agnostic to whether or not a user actually has a diagnosis of autism, it can also be used to assist individuals who do not have a diagnosis but face similar obstacles to finding employment. Some other disorders were potential employees with these disorders may have similar concerns include sensory processing disorder, social (pragmatic) communication disorder, language disorder, obsessive compulsive personality disorder, attention deficit disorder, and social anxiety disorder. While our proposed job-matching system is designed with autism in mind, it could be expanded to include questions relating to even more disorders. As the system is designed to be competitive, anyone could potentially sign up for it, so it could potentially serve as an alternative means of finding employment for adults with other disability in general.

References

[1] R. Burke, M. Andersen, S. Bowen, M. Howard, and K. Allen. Evaluation of Two Instruction Methods to Increase Employment Options for Young Adults with Autism Spectrum Disorders. *Research in Developmental Disabilities*, 31(6):1223–1233, 2010.

[2] C. Cing. Development of Job Matching Algorithm with Collective Learning. Master of Computer Science, Universiti Tunku Abdul Rahman, August 2013.

[3] Daivergent. https://daivergent.com/data-services/recruitment.

[4] J. Dreaver, C. Thompson, S. Girdler, M. Adolfsson, M. Black, and M. Falkmer. Success Factors Enabling Employment for Adults on the Autism Spectrum from Employers' Perspective. *Journal of Autism and Developmental Disorder*, 50:1657–1667, 2020.

[5] E. Dubins and D. Freedman. Machiavelli and the Gale-Shapley Algorithm. *The American Mathematical Monthly*, 88(7):485–494, August-September 1981.

[6] D. Gale and L. Shapley. College Admissions and the Stability of Marriage. *The American Mathematical Monthly*, 69(1):9–15, January 1962.

[7] C. Grob, D. Lerman, C. Langlinais, and N. Villante. Assessing and Teaching Job-related Social Skills to Adults with Autism Spectrum Disorder. *Journal of Applied Behavior Analysis*, 52:150–172, 2019.

[8] S. Guo, F. Alamudun, and T. Hammond. RésuMatcher: A Personalized Résumé-job Matching System. *Expert Systems With Applications*, 60:169–182, 2016.

[9] D. Hendricks. Employment and Adults with Autism Spectrum Disorders: Challenges and Strategies for Success. *Journal of Vocational Rehabilitation*, 32:125–134, 2010.

[10] A. Jacob, M. Scott, M. Falkmer, and T. Falkmer. The Costs and Benefits of Employing an Adult with Autism Spectrum Disorder: A Systematic Review. *PLOS ONE*, pages 1–15, October 2015.

[11] M. Koh and Y. Chew. Intelligent Job Matching with Self-learning Recommendation Engine. *Procedia Manufacturing*, 3:1959–1965, 2015.

[12] J. Malinowski, T. Keim, O. Wendt, and T. Weitzel. Matching People and Jobs: A Bilateral Recommendation Approach. In *Proceedings of the* 39^{th} *Annual Hawaii International Conference on System Sciences (HICSS'06)*, pages 137c–137c, 2006.

[13] M. Parsons, D. Reid, J. Reynolds, and M. Bumgarner. Effects of Chosen Versus Assigned Jobs on the Work Performance of Persons with Severe Handicaps. *Journal of Applied Behavior Analysis*, 23(2):253–258, Summer 1990.

[14] R. Rafter, K. Bradley, and B. Smyth. Personalized Retrieval for Online Recruitment Services. In *Proceedings of the BCS/IRSG* 22^{nd} *Annual Colloquium on Information Retrieval (IRSG 2000)*, pages 1–13, March 2000.

[15] A. Roux, J. Rast, K. Anderson, and P. Shattuck. National Autism Indicators Report: Developmental Disability Services and Outcomes in Adulthood. Indepth Report, Drexel Univ., May 2017. https://drexel.edu/autismoutcomes/publications-and-reports/publications/National-Autism-Indicators-Report-Developmental-Disability-Services-and-Outcomes-in-Adulthood/.

[16] Samhall. Samhall 2014 Annual and Sustainability Report. https://samhall.se/wp-content/uploads/2020/04/Samhall_anual-_report2014.pdf, 2014.

[17] E. Shein. Hiring from the Autism Spectrum. *Communications of the ACM*, 63(6):17–19, June 2020.

[18] G. Shi, Y. Kong, B. Chen, G. Yuan, and R. Wu. Instability in Stable Marriage Problem: Matching Unequally Numbered Men and Women. *Complexity*, 2018:5 pages, 2018.

[19] J. Vande Vate. Linear Programming Brings Marital Bliss. *Operations Research Letters*, 8:147–153, June 1989.

A The Aptitude and Interest Scheme

Question	Employee	First Answer	Second Answer	Third Answer	Min Value	Lower Value	Upper Value	Max Value
How much do you like task consistency	Yes	I can't stand repetitive tasks.	I like some sense of routine, but with some variety as well.	I prefer to just perform a single task.	0	3	6	9
How consistent are the job tasks?	No	Everyday is different at our company.	While the bulk of the work is similar from day to day, new problems are always arising the need to be solved.	Our work is extremely repetitive.	2	4	8	10
How are your interview skills?	Yes	I perform poorly during interviews.	I am not skilled at interviews, but do not consider the interview process to be a major obstacle.	I am highly skilled at demonstrating my soft skills during interviews.	0	1	5	10
How important is presentation during interview?	No	For us, interviews are merely a formality and we do not evaluate a candidate based on their interview.	Interviews are useful for accessing the fitness of an employee, but are strictly secondary to resumes, portfolios, and referrals when assessing a candidate.	We closely monitor the body language of candidates and other features of the presentation during interviews in order to access soft skills.	0	2	7	10
What social culture do you desire in a workplace?	Yes	I prefer to work alone.	I am fine with working alone but enjoy the company of others.	I thrive off interacting with my coworkers.	0	2	7	10
What is the social culture in your job like?	No	People mostly keep to themselves.	Coworkers tend to maintain a casual relationship with each other.	Workers are expected to form intimate bonds with one another through their work.	1	2	8	10
How well can you function in a noisy environment?	Yes	I cannot function in a noisy environment.	Noise can distract or bother me, but I can manage.	I can effectively follow multiple conversations in a noisy environment without any difficulty.	-2	2	5	12

How noisy is your workplace?	No	The workplace is quiet, and workers can wear headphones if they please.	There is frequently ambient conversation and workers are expected to remain alert, but there is nothing out of the ordinary.	The workplace is filled with loud noises at all times	0	1	7	12
How loyal are you?	Yes	I am only looking for a job for the money, and I'll leave when I can find something better.	I am as loyal as the job needs me to be.	I am known as someone who will never give up on a commitment.	-1	3	5	10
How important is company loyalty?	No	We recognize that employers just see us a stepping stone in their career, and that's fine.	We invest a lot in our employees, and expect them to return the favor.	Company loyalty is one of our core values.	0	4	7	9

The Interest and Aptitude schema include the *Interest* and *Aptitude* components. The **Interest** component consists of *Social Culture, Repetition, Salary, Location* and *AQ* score, whereas the **Aptitude** component contains *Light Sensitivity, Sound Sensitivity, Interview Skills, Loyalty, Trustworthiness, Attention to Detail, Office Skills-Social, Office Skills-Technical, Fine Motor Skills*, and *Gross Motor Skills*. The table shown above depicts some of the sample quizzes of the two schema.