A Machine-based Personality Oriented Team Recommender for Software Development Organizations

Murat Yilmaz[1] and Ali Al-Taei[2] and Rory V. O'Connor[3]

[1] Çankaya University, Turkey
myilmaz@cankaya.edu.tr
[2] University of Baghdad, Baghdad, Iraq
alitaei@mtu.edu.iq
[3] Dublin City University, Ireland
roconnor@computing.dcu.ie

Abstract. Hiring the right person for the right job is always a challenging task in software development landscapes. To bridge this gap, software firms start using psychometric instruments for investigating the personality types of software practitioners. In our previous research, we have developed an MBTI-like instrument to reveal the personality types of software practitioners. This study aims to develop a personality-based team recommender mechanism to improve the effectiveness of software teams. The mechanism is based on predicting the possible patterns of teams using a machine-based classifier. The classifier is trained with empirical data (e.g. personality types, job roles), which was collected from 52 software practitioners working on five different software teams. 12 software practitioners were selected for the testing process who were recommended by the classifier to work for these teams. The preliminary results suggest that a personality-based team recommender system may provide an effective approach as compared with ad-hoc methods of team formation in software development organizations. Ultimately, the overall performance of the proposed classifier was 83.3%. These findings seem acceptable especially for tasks of suggestion where individuals might be able to fit in more than one team.

Keywords: Organizational Improvement, MBTI, Personality Profiling, Personnel Recommendation System, Neural Networks, Multilayer Perceptron.

1 Introduction

Software development is concerned with the systematic production of quality software on a limited budget and time, which stills depend on complex human interactions to create an economic value. In more recent years, a significant number of researchers suggest that the major issues encountered in software development becomes are more sociological in their nature [1]. It is therefore becoming increasingly difficult to ignore the fact that *software development is a social activity* [2, 3]. Software practitioners are usually work in collaborative groups in all stages of software development where working on such a team is an inherently social activity, which is important to sustain the software development organization's structure.

Social aspects of software development is an emerging field of interest which adds new kind of capabilities to software development organizations [4]. Consequently, the personality characteristics of software practitioners receive an increasing level of attention [5]. In fact, revealing the personality types of software practitioners allows us to understand the software organizations. It may help us to manage its development process and strengthen its evolution [6]. However, there has been few empirical efforts that attempt to deal with the factors affecting software development efforts based on software practitioners behaviors and their personality types.

The process of personality typing has been used for nearly thousands of years dating back to Greek archetypes. Myers-Briggs Type Indicator (MBTI) is a personality typing assessment system. It was developed by psychologists Myers and Briggs as a self-report instrument. During our previous research [7], an MBTI compatible personality assessment tool was constructed to reveal the personality types of individuals who are working on software development organizations, which was developed particularly for software practitioners so as to build better software development teams. Ultimately, the goal of this study is to build a preliminary model for a personnel recommendation system for software development organizations from an industrial perspective.

This study seeks to address the following research questions: "*Is it possible to explore subjective characteristics such as personality types of software practitioners to classify participants to improve the social structure of software teams using a machine learning approach?*"

The working mechanism of such a recommendation system is planned on predicting a set of compatibility structures based on personality types of software practitioners of a software development organization, which could also offer recommendations regarding the most suitable members of a software team in terms of their personality preferences. The goal of this exploratory study is to investigate the possible combinations of software practitioners in software teams as regards to their job title. To this end, a neural network based personality classifier is employed. The classifier is trained with the real data which was collected from a software development organizations. Furthermore, proposed approach is tested by classifying a group of individuals based on this into teams using this information pattern.

Using a hybrid methodology; the results of individual's personality test, the data collected from previous encounters in which the structure of current teams are analyzed. Furthermore, a set of compatibility options is predicted by the recommendation system approach. In light of these remarks, it has been thought that better results from previous studies are expected where the collected information will be a valuable asset for resolving multi-dimensional issues in the service of building more effective team structures.

The remainder of the paper is organized in the following manner. In section 2, we cover the background including brief details of the personality typing, the definitions of the proposed methods of machine learning (i.e. artificial neural networks), and their applications in the machine-based classification literature. Section 3 explains the details of the research methodology, which we use to conduct this study. In the next section, we discuss the results, which was used to classify software practitioners to the software development teams. In the final section, we draw some conclusions based on our preliminary results and detail the possible improvements for further research.

2 Background

Personality is considered as a set of relatively permanent traits (i.e. a set of patterns of behaviors) that can be found unique for a person. According to MBTI, this can be explained by four dichotomies namely E-I, S-N, T-F, J-P. Extroversion (E) versus introversion (I) shows how an individual regenerate his or her energy. Sensing (S) and intuition (N) is about how individuals make sense of their environment (i.e. process the information about the world). Sensing type of persons trusts more on their five senses while intuitive types are inclined to listen their subconscious and trusts their insights. Thinking (T) and feeling (F) preferences shows the decision-making style where an individual can be either objective thinker or a value oriented characteristics that focus on people and relationships. Lastly, Judging (J) and Perceiving (P) show the difference in individuals regarding their life style. Judging (J) persons prefer to see what lies ahead, organization and control while perceiving (P) individuals favor flexibility and keeping their options more open.

Typically, an MBTI assessment shows an individuals' inclination through one bipolar personality characteristic. Consequently, an MBTI type survey is usually conducted to collect the data where participants are asked to choose their highest preference (i.e. dominant function). To find an MBTI personality type, researcher should cross-reference the four dichotomies, which produces sixteen personality types.

Accordingly, a combination of these 16 personality types can be formed as shown in Table 1 [8].

ISTJ	ISFJ	INFJ	INTJ
ISTP	ISFP	INFP	INTP
ESTP	ESFP	ENFP	ENTP
ESTJ	ESFJ	ENFJ	ENTJ

Table 1. 16 MBTI personality types

In particular, MBTI has an important difference from other psychometric assessments: It does not suggest or recommend any preferred type. Instead, it reveals a person's place on four distinctive continuums of bipolar personality type scales. Such an approach is beneficial to software managers to understand the individual differences between software practitioners. It helps managers to avoid and resolve conflicts, identify gaps in software teams, improve team-based communication (i.e. encourage software team members to understand their teammates). It is therefore helpful to build better people skills, interactions as well as team configurations.

2.1 Neural Networks

A neural network (NN) can be similar to a network of parallel micro processing units, which are inspired from the model of a nerve cell in humans [9]. These units are based on a group of regression models which are chained to produce a combination of outputs by having a set of inputs embedded in a single mathematical approach [10]. Traditionally, after defining a particular problem a NN is trained to solve it. Most interestingly, however, the NN program does not know anything about the problem it addresses where the produced answers are usually emerged from the interaction between nodes by its evolutionary process [9].

The NN model consists of several neurons, i.e. island-like structures (or nodes) which are added such a way that only appropriate neighbors nested with them. Therefore, its algorithm is utilized to find the closest node that can fit by using the regression equations associated with it [10]. The collection of these nodes or neurons are called the neural net where these nodes have a number of inputs with associated weights, and a threshold value which determines either it is fired or not [9].

A neural net can be used to solve several kinds of problems such as classification, prediction, pattern recognition, etc. Multi layer perceptron (MLP) network is one of the most popular and commonly used neural network particularly for such processes [11]. MLP network consists of three distinctive kinds of layers: an input layer, one (or more) hidden layer(s), and an output layer. These layers are connected together by a set of weighted connections. The number of nodes in input layer is equal to the number of attributes in the input vector [10]. Hidden layer(s) and nodes in each layer are up to the designer's point of view as they can vary and should be managed carefully for better efficiency. And the final output from the output layer nodes represents the predicted outputs where each node in output layer represents a single output [11]. Figure 1 shows the typical MLP with one hidden layer.

Similar to a behavioral conditioning mechanism, an important approach used with MLP is the back propagation (BP) algorithm [10]. Basically, researcher starts with an input and propose a desired output. Consequently, the network is rewarded for close outputs to the desired input while the nodes are punished for an incorrect output. These activities are used to update the weights of the network so as to improve the results it produces. The MLP networks are frequently used with BP algorithm [11]. MLP initiates with small random weights, and

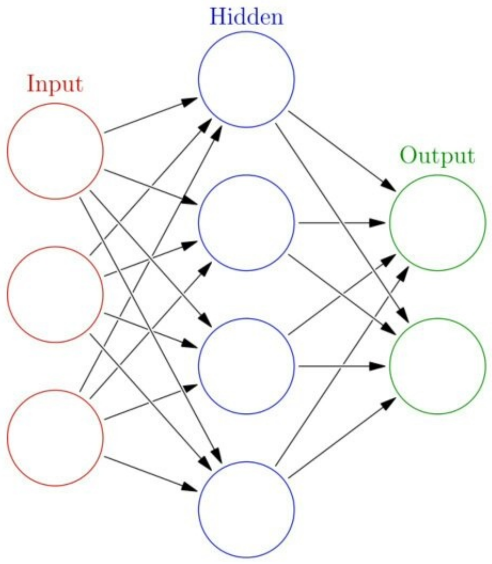

Fig. 1. Typical design of MLP artificial neural network [9]

a desired error rate. The learning process is achieved by using input: desired-output pair vectors to adjust the weights in order to minimize the error rate (i.e. calculate the difference between the real error and the desired error rates for all nodes in output layer). Next, the back propagation algorithm adjusts the weights [9].

2.2 Machine Learning in Personality Assessment Literature

The investigation of psychometric properties (e.g. traits, motivation, and personal preferences) of individuals have been studied in many different disciplines including but not limited to software engineering [7], game development [12], and economics courses [13]. In particular, several different methods (e.g. MBTI [8], Keirsey's Temperament Sorter [14], Big Five [15], etc.) have been utilized to assess personality types of participants. A search of the literature revealed a few studies which includes machine based classification of personality characteristics as follows.

Mairesse et al. [16] used linguistic cues for the purpose of predicting personality automatically from text and conversation. The suggested method was based on exploring Big Five personality traits depending on conversation and text of individuals, and the self and testers personality rating from observation. Accordingly, different methods were employed to test three the performance of different models. They used three approaches of machine learning methods: classification algorithms, regression, and ranking (e.g. Support Vector Machines (SVM), decision tree (DT), and nearest neighbor (NN)). Their results indicated that the performance of ranking models was the highest. Furthermore, performance of classification models trained using observed personality dataset was better than the models trained using self personality rating. In addition, it was observed that personality traits was the most important factor for extracting feature set.

Celli et al. [17] conducted experiments using six different machine learning methods (i.e SVM, NN, DT, naive Bayes, logistic regression, and rule learner) for the purpose of personality type and interaction style recognition based on profile pictures of Facebook users (N=100) and self-assessed personality test of Big Five traits. Feature extraction process was carried out by bag-of-visual-words (BoVW) technique. Results showed that the accuracy of each classifier depended on personality traits. The average performance was 66.5%.

Cowley et al. [18] claimed that employing machine learning methods to explore game-play experience and player personality type is still in early stages. In their study, they utilized two different decision tree methods (i.e. CART and C5.0) and used DGD player taxonomy on Pac-Man gamers to select appropriate rules for a classification. Training set contained 100 instances, while the testing set contained 37 instances. Ultimately, the validation testing performance of classifier was about 70%.

Aruan et al. [19] built a virtual tutor agent (VTA), which was developed for multiple users for the goal of problem-based learning in cooperative environments. It was inspired from massively multiplayer online games (MMOG). Both conceptual issues of learning using interface-supported cooperative environment and technological issues of deploying and dealing with massive users from MMOG perspective were combined together. In addition, some applications and coding have been used to achieve the goal, and the result was acceptable.

Golbeck et al. [20] proposed a model to predict personality of Twitter users. The model depended on information that publicly available in profiles of Twitter users and Big Five personality test. In particular, the Big Five test was administered to 279 of the users, and 2000 of their most common tweets were gathered. Next, feature was extracted through text analysis tools. Lastly, two regression methods (i.e Gaussian Process and ZeroR) were used to predict personality traits. Results showed that both of the methods performed similarly and the accuracy was reasonable.

To reduce the costs of monitoring and analyzing player personality, Kang et al. [21] proposed an automated system for the analysis of MMOG players' behaviors using trajectory (non-parametric) clustering algorithm with simple data. At first, they classified the data hierarchically, and then used trajectory clustering algorithm to analyze behaviors. The system was tested on world of warcraft (WoW) game environment and the results were good in both analyzing player's behavior and creating players' experience insights and profiles automatically.

Lotte et al. [22] reviewed a number of most common used classification methods (e.g. SVM, MLP, Hidden Markov Model (HMM)) and compared their performance to find the proper classification algorithm(s) for brain-computer interface

(BCI) using *electro encephalo graphy* (EEG) dataset. The results and efficiency of each classifier were analyzed and compared among other classifiers to present a concrete base of knowledge that can be regarded when choosing the proper classifier for a specific task. In general, for the area of BCI using EEG dataset, it was found that SVM performs better than other classifiers. However, the performance of MLP was also acceptable for such a task. It is therefore likely to see the notion of neural networks, which are commonly used in BCI area of research.

This paper attempts to show that a machine-based personality classifier, which is planned as a recommendation system while selecting personnel for actual software teams. To date, studies investigating machine learning based personality classification have produced equivocal results. Most studies found in the literature in the field of personality based classification have mostly focused Big Five personality traits and SVM and DT as machine learning methods. However, this exploratory study aims to suggest software practitioners based on their MBTI personality types using a novel approach.

3 Method

Based on the data collected from 52 software practitioners, we aim to explore the patterns between personality types and roles of software practitioners who are working in teams. After revealing such relationships, the collected attributes were used to train a neural network (i.e. multi layer perceptron), and ultimately the goal is to create an initial version of a personality type based team configurations. Authors believe that software managers benefit from sucj social aspects while searching for a suitable team for their software practitioners.

In our previous study [7], we have already conducted an MBTI-like assessment for five teams of software practitioners from a software development organization. The personality types and roles of five software teams (total 52 people) were singled out. In addition, all selected teams were considered as productive teams, which consist of individuals with a minimum of five years of industrial experience. The software practitioners were also selected from the individuals who worked together for more than two years in software development projects.

After the data was transformed to binary values, the MLP was trained based on the patterns that were extracted from these team's roles and personality types. Accordingly, 10 input nodes were formed. 4 input nodes represents the personality types and 6 input nodes (e.g. role one is represented like 000001, and role two like 000010, etc.) to represent practitioners' job role. In the output layer, 5 output nodes represents 5 software teams which are based on the initial parameters, e.g. Team 1 represented as 00001, team 2 represented like 00010, etc.

Figure 2 illustrates the suggested MLP model where $\{I_1, I_2, I_3 ... I_{10}\}$, are the nodes (e.g. personality types and roles) that are shown in the input layer, and $\{T_1, ... T_n\}$, are the possible team formations.

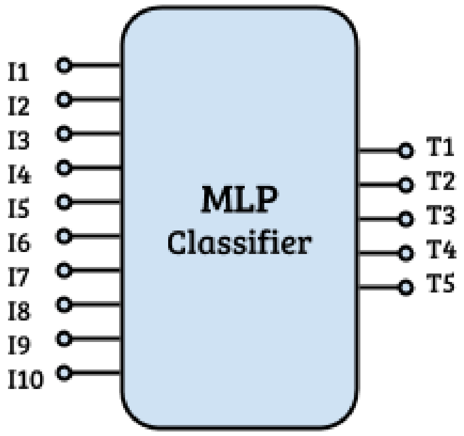

Fig. 2. The suggested model for MLP-based Team Recommnender.

To build a team classifier, we built a perceptron with three layers, which was equipped with back propagation algorithm. During the study, several outputs were examined to decide the best configuration parameters for classification. The findings were detailed through the next section.

4 Results

The vectors of input data (N=52) with their desired output were fed into the MLP model during the training process. The weights values were firstly created randomly between -1 and 1 and were iteratively updated until convergences toward the desired output, and decreasing the error rate until the minimum. Momentum parameter was used to increase the convergence [23]. The training mode continues until the number of epochs reaches 1 million or the error rate is equal or less than 0.01.

Leave one out cross validation (LOOCV) is one of the methods used in machine learning studies for validating model performance [24]. In this method data is splitted into N samples and perform N rounds of train/test processes (e.g. N-1 samples for training and 1 sample for testing). Then, the estimated performance is calculated as the average of testing samples [25].

To avoid over-fitting, LOOCV technique with 10-folds was used during the training process. Many configurations (e.g. error rate, learning rate) were investigated to explore the best performance within the predefined conditions. Accordingly, the optimal number of nodes in hidden layer was 15.

To test the classification performance of the proposed model, 12 software practitioners were selected who fed into the model for finding suitable teams. The overall performance of the suggested model was 83.3%. The result seems acceptable especially for such tasks of suggestion (i.e. persons might fit in more than one team). Table 2 shows the performance and error rate of the model for both training and testing processes (learning rate= 0.7, momentum= 0.69, number of epoch= 1396).

Training Process (N=52)		Testing Process (N=12)	
Correctly Classified	Incorrectly Classified	Correctly Classified	Incorrectly Classified
52 (100%)	0 (0%)	10 (83.3%)	2 (16.6%)
RMSE= 0.002		RMSE= 0.2	

Table 2. Training-testing results for the classifier.

To evaluate the face validity [26], the suggested personnel for five selected teams were shared with the software management group. Next, we interviewed the research manager of the company where we collected the empirical data (i.e. personality types and roles of software practitioners). The interviewee suggested that such an approach could be useful as a complementary tool for the personnel recruitment process. In fact, we have found that there was a sense of agreement for the benefits of the model between interviewee and the authors.

> ***Interview quotation:*** *"I believe that building an effective software team is such a challenging task. It is also hard to do a manual reconfiguration especially after the initial declaration of software team members. Therefore, seeing more possibilities of team configuration is necessary. I found a tool that helps to predict a possible position for a team member is quite useful strategy. However, it would be more beneficial for us [the company] if you could suggest a team member who may fit for more than one team."*

4.1 Limitations

Using personality assessments to investigate people's type of personality does not always yield very accurate results for many reasons, and therefore, they should be regarded as indicators for individuals' preferences and temperaments rather than solid evidence for their exact type [8]. Furthermore, artificial neural networks (ANN) and other similar methods of machine learning and pattern recognition have many parameters affecting model performance such as error rate, preparation process of datasets, actual size of data, and quality of training and testing sets. Therefore, they do not always provide the optimal results and they should be designed carefully [22].

The MLP team classifier was operationalized by using the empirical data that was collected from a single company. In fact, the personality types and the software practitioners' roles of the teams found in that company may not represent all possible software engineering team patterns or structures. Therefore, our results are limited in a (specific) software development company's identified patterns. To design a team recommender for another company, a new MLP should be trained accordingly.

5 Conclusions and Future Work

The main aim of this preliminary study is to explore the possibility of building a team recommender mechanism. It can be used to suggest a set of suitable software practitioners for the actual software teams regarding their possible roles and personality types. Based on a set of empirical data, we planned a suggestion mechanism for improving team management activities. Although the current study was based on a small sample of participants, the findings suggest that team-based personality patterns can be highlighted for improving team building activities or building a novel team configuration process.

Despite its exploratory nature, this study offers some insight into social aspects of software development. Firstly, software managers could likely to benefit from a machine-based team recommender approach. However, further experimental investigations are needed to estimate more team configurations. Secondly, we believe that the proposed method to achieve the actual results are reasonable as we aim to investigate the possibilities of a set of team formations in terms of their personality types of a selected population. Considerably, more work will need to be done to evaluate the effectiveness of a personality-based software team recommender.

Returning to the research question posed at the beginning of this study, authors confirm that personality types of software practitioners along with their team-based roles are useful information for observing (social) software team patterns. However, it would be interesting to assess the effects of personality types on software team formations on a more large scale. In light of these remarks, authors confirm that machine learning techniques can create a significant advantage for addressing problems of software engineering and process improvement research.

References

1. DeMarco, T., Lister, T.: Peopleware: productive projects and teams. Dorset House Publishing Company (1999)
2. Acuna, S.T., Juristo, N., Moreno, A.M., Mon, A.: A Software Process Model Handbook for Incorporating People's Capabilities. Springer-Verlag (2005)

3. Dittrich, Y., Floyd, C., Klischewski, R.: Social thinking-software practice. The MIT Press (2002)
4. Yilmaz, M., O'Connor, R.: An approach for improving the social aspects of the software development process by using a game theoretic perspective: towards a theory of social productivity of software development teams. In: 6th International Conference on Software and Data Technologies. Volume 1., SciTePress (2011) 35–40
5. Beecham, S., Baddoo, N., Hall, T., Robinson, H., Sharp, H.: Motivation in software engineering: A systematic literature review. Information and Software Technology **50** (2008) 860–878
6. Yilmaz, M., O'Connor, R.: Towards the understanding and classification of the personality traits of software development practitioners: Situational context cards approach. In: Software Engineering and Advanced Applications (SEAA), 2012 38th EUROMICRO Conference on, IEEE (2012) 400–405
7. Yilmaz, M.: A software process engineering approach to understanding software productivity and team personality characteristics: an empirical investigation. PhD thesis, Dublin City University (2013)
8. Myers, I.B., McCaulley, M.H., Most, R.: Manual: A guide to the development and use of the Myers-Briggs Type Indicator. Consulting Psychologists Press Palo Alto, CA (1985)
9. Kodicek, D.: Mathematics and physics for programmers. Cengage Learning (2005)
10. Garson, G.D.: Neural networks: An introductory guide for social scientists. Sage (1998)
11. Bishop, C.M.: Neural networks for pattern recognition. Clarendon press Oxford (1995)
12. Bartle, R.A.: Designing virtual worlds. New Riders (2004)
13. Borg, M.O., Stranahan, H.A.: Personality type and student performance in upper-level economics courses: The importance of race and gender. The Journal of Economic Education **33** (2002) 3–14
14. Keirsey, D.: Please Understand Me II: Temperament, Character, Intelligence Author: David Keirsey, Publisher: Prometheus Nemesis Book C. Prometheus Nemesis Book Co (1998)
15. John, O.P., Donahue, E.M., Kentle, R.L.: The big five inventoryversions 4a and 54. Berkeley: University of California, Berkeley, Institute of Personality and Social Research (1991)
16. Mairesse, F., Walker, M.A., Mehl, M.R., Moore, R.K.: Using linguistic cues for the automatic recognition of personality in conversation and text. Journal of artificial intelligence research (2007) 457–500
17. Celli, F., Bruni, E., Lepri, B.: Automatic personality and interaction style recognition from facebook profile pictures. In: Proceedings of the ACM International Conference on Multimedia, ACM (2014) 1101–1104
18. Cowley, B., Charles, D., Black, M., Hickey, R.: Real-time rule-based classification of player types in computer games. User Modeling and User-Adapted Interaction **23** (2013) 489–526
19. Aruan, F., Prihatmanto, A., Hindersah, H., et al.: The designing and implementation of a problem based learning in collaborative virtual environments using mmog technology. In: System Engineering and Technology (ICSET), 2012 International Conference on, IEEE (2012) 1–7
20. Golbeck, J., Robles, C., Edmondson, M., Turner, K.: Predicting personality from twitter. In: Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third

Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on, IEEE (2011) 149–156

21. Kang, S.J., Kim, Y.B., Park, T., Kim, C.H.: Automatic player behavior analysis system using trajectory data in a massive multiplayer online game. Multimedia tools and applications **66** (2013) 383–404
22. Lotte, F., Congedo, M., Lécuyer, A., Lamarche, F., Arnaldi, B., et al.: A review of classification algorithms for eeg-based brain–computer interfaces. Journal of neural engineering **4** (2007)
23. Hagan, M.T., Demuth, H.B., Beale, M.H., et al.: Neural network design. Volume 1. Pws Boston (1996)
24. Priddy, K.L., Keller, P.E.: Artificial neural networks: an introduction. Volume 68. SPIE Press (2005)
25. Cawley, G.C., Talbot, N.L.: Efficient leave-one-out cross-validation of kernel fisher discriminant classifiers. Pattern Recognition **36** (2003) 2585–2592
26. Lewis-Beck, M., Bryman, A.E., Liao, T.F.: The Sage encyclopedia of social science research methods. Sage Publications (2003)