PAN at FIRE: Overview of the PR-SOCO Track on Personality Recognition in SOurce COde

Francisco Rangel
Autoritas Consulting
Valencia, Spain
francisco.rangel@autoritas.es

Fabio A. González
MindLab Research Group
Universidad Nacional de Colombia
fagonzalezo@unal.edu.co

Felipe Restrepo-Calle
MindLab Research Group
Universidad Nacional de Colombia
Bogotá, Colombia
ferestrepoca@unal.edu.co

Manuel Montes
INAOE
Mexico
mmontesg@inaoep.mx

Paolo Rosso
PRHLT Research Center
Universitat Politècnica de València
prosso@dsic.upv.es

ABSTRACT

Author profiling consists of predicting some author's characteristics (e.g. age, gender, personality) from her writing. After addressing at PAN@CLEF mainly age and gender identification, and also personality recognition in Twitter[1], in this PAN@FIRE track on Personality Recognition from SOurce COde (PR-SOCO) we have addressed the problem of predicting author's personality traits from her source code. In this paper, we analyse 48 runs sent by 11 participant teams. Given a set of source codes written in Java by students who answered also a personality test, participants had to predict personality traits, based on the big five model. Results have been evaluated with two complementary measures (RMSE and Pearson product-moment correlation) that have permitted to identify whether systems with low error rates may work due to random chance. No matter the approach, openness to experience is the trait where the participants obtained the best results for both measures.

Keywords

personality recognition; source code; author profiling

1. INTRODUCTION

Personality influence most, if not all, of the human activities, such as the way people write [5, 25], interact with others, and the way people make decisions. For instance, in the case of developers, personality influence the criteria they consider when selecting a software project they want to participate [22], or the way they write and structure their source code. Personality is defined along five traits using the Big Five Theory [7], which is the most widely accepted in psychology. The five traits are: extroversion (E), emotional stability / neuroticism (S), agreeableness (A), conscientiousness (C), and openness to experience (O).

Personality recognition may have several practical applications, for example to set up high performance teams. In software development, not only technical skills are required, but also soft skills such as communication or teamwork. The possibility of using a tool to predict personality from source code, in order to know whether a candidate may fit in a team, may be very valuable for the recruitment process. Also in education, to know students' personality from their source codes may help to improve the learning process by customising the educational offer.

In this PAN@FIRE track on Personality Recognition from SOurce COde (PR-SOCO), we have addressed the problem of predicting an author's personality from her source code. Given a source code collection of a programmer, the aim is to identify her personality traits. In the training phase, participants have been provided with source codes in Java, written by computer science students, together with their personality traits. At test, participants have received source codes of a few programmers and they have to predict their personality traits. The number of source codes per programmer will be small reflecting a real scenario such as the one of a job interview: the interviewer could be interested in knowing the interviewee degree of conscientiousness by evaluating just a couple of programming problems.

We suggested participants to investigate beyond standard n-grams based features. For example, the way the code is commented, the naming convention for identifiers or indentation may also provide valuable information. In order to encourage the investigation of different kinds of features, several runs per participant were allowed. In this paper, we describe the participation of 11 teams that sent 48 runs.

The reminder of this paper is organised as follows. Section 2 covers the state of the art, Section 3 describes the corpus and the evaluation measures, and Section 4 presents the approaches submitted by the participants. Section 5 and 6 discuss results and draw conclusions, respectively.

2. RELATED WORK

Pioneers research works in personality recognition were carried out by Argamon *et al.* [27], who focused on the identification of extroversion and emotional stability. They used support vector machines with a combination of word categories and relative frequency of function words to recognize these traits from self-reports. Similarly, Oberlander and Nowson [21] focused on personality identification of bloggers. Mairesse *et al.* [20] analysed the impact of different

[1]http://pan.webis.de/

set of psycholinguistic features obtained with LIWC[2] and MRC[3], showing the highest performance on the openness to experience trait.

Recently, researchers have focused on personality recognition from social media. In [14, 24, 6], the authors analysed different sets of linguistic features as well as friends count or daily activity. In [18], the authors reported a comprehensive analysis on features such as the size of the friendship network, the number of uploaded photos or the events attended by the user. They analysed more than 180,000 Facebook users and found correlations among these features and the different traits, specially in case of extroversion. Using the same Facebook dataset and similar set of features, Bachrach *et al.* [1] reported high results predicting extroversion automatically.

In [26], the authors analysed 75,000 Facebook messages of volunteers who filled a personality test and found interesting correlations among words usage and personality traits. According to them, extroverts use more social words and introverts use more words related to solitary activities. Emotionally stable people use words related to sports, vacation, beach, church or team; whereas neurotics use more words and sentences referring to depression.

Due to the interest on this field and with the aim at defining a common framework of evaluation, some shared tasks have been organised. For example, *i)* the Workshop on Computational Personality Recognition [5]; or *ii)* the Author Profiling task at PAN 2015 [25] with the objective of identifying age, gender and personality traits of Twitter users.

Regarding programming style and personality, in [3] the authors explored the relationship between cognitive style, personality and computer programming style. More recently, the authors in [16] also related personality to programming style and performance. Whereas the 2014 [10] and 2015 [11] PAN@FIRE tracks on SOurce COde (SOCO) where devoted to detect reuse, in 2016 we aimed at identifying personality traits from source code.

3. EVALUATION FRAMEWORK

In this section we describe the construction of the corpus, covering particular properties, challenges and novelties. Finally, the evaluation measures are described.

3.1 Corpus

The dataset is composed of Java programs written by computer science students from a data structures course at the *Universidad Nacional de Colombia*. Students were asked to upload source code, responding to some functional requirements of different programming tasks, to an automated assessment tool. For each task, students could upload more than one attempted solution. The number of attempts per problem was not limited / discouraged in any way. There are very similar submissions among different attempts and also some of them contain compilation-time or runtime errors.

Although in most of the cases students uploaded the right Java source code file, some of them erroneously uploaded the compiler output, debug information or even the source code in other programming language (e.g.: Python). A priori this seems to be noise for the dataset and a sensible alternative could have been to remove these entries. However, we decided to keep them due to the following reasons: firstly, participant teams could remove them easily if they decide to do so; secondly, it is possible that this kind of mistakes is related to some personality traits, so this information can be used as a feature as well. Finally, although we encouraged the students to write their own code, some of them could have reused some pieces of code from other exercises or even looked for code excerpts on books or the Internet.

In addition, each student answered a Big Five personality test that allowed us to calculate a numerical score for each one of the following personality traits: extroversion, emotional stability / neuroticism, agreeableness, conscientiousness, and openness to experience.

Overall, the dataset consists of 2,492 source code programs written by 70 students along with the scores of the five personality traits for each student, which are provided as floating point numbers in the continuous range [20,80]. The source codes of each student were organized on a single text file with all her source codes together with a line separator among them. The dataset was split in training and test subsets, the first one containing the data for 49 students and the second one the data of the remaining 21. Participants only have access to the personality traits scores of the 49 students in the training dataset.

3.2 Performance measures

For evaluating participants' approaches we have used two complementary measures: Root Mean Square Error (RMSE) and Pearson Product-Moment Correlation (PC). The motivation to use both measures is to try to understand whether a committed error is due to random chance.

We have calculated RMSE for each trait with Equation 1:

$$RMSE_t = \sqrt{\frac{1}{n}\sum_{1}^{n}(y_i - \hat{y}_i)^2} \quad (1)$$

where $RMSE_t$ is the Root Mean Square Error for trait t (neuroticism, extroversion, openness, agreeableness, conscientiousness); y_i and $\hat{y}_i$ are the ground truth and predicted values respectively for author i. Also for each trait, PC is calculated following Equation 2:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2} \quad (2)$$

where each x_i and y_i are respectively the ground truth and the predicted value for each author i; $\bar{x}$ and $\bar{y}$ the average values.

4. OVERVIEW OF THE SUBMITTED APPROACHES

Eleven teams participated in the Personality Recognition in SOurce COde[4] shared task. They sent 48 runs with different approaches, and 9 of them have submitted the working notes describing their approaches. Following, we briefly highlight the different systems.

- *besumich* [23] experimented with two kinds of features, bag of words and character n-grams (with n=1,2,3). In both cases, they experimented with lowercase and

[2]http://www.liwc.net/

[3]http://www.psych.rl.ac.uk/

[4]http://www.autoritas.es/prsoco/

original case, and three representations, binary (presence/absence), term frequency (TF) and TF-IDF. The authors trained linear, ridge and Lasso regressions. The final configuration used to send their runs combined lowercased unigrams weighted with TF-IDF (with and without space characters) with different values for the alpha parameter of the Lasso regression.

- *bilan* [2] started with analysing the code structure with the Antlr Java Code Analyzer[5]: it parses the program code and produces a parse tree of it. Then, they use each single node of the output tree (nodes represent different code categories, like classes, loops or variables) and count the frequency distribution of these nodes (around 200 features are taken into consideration). Apart from the Antlr, they obtain a set of custom features for the source code, such as the length of the whole program, the average length of variable names, the frequency of comments, their length, what indentation the programmer is using, and also the distribution and usage of various statements and decorators. They also extract features from the comments such as the type/token ratio, usage of punctuation marks, average word length and a TF-IDF vector. They trained their models with two approaches, learning from each single source code, and from the whole set of source codes per author.

- *castellanos* [4] used also Antlr with the Java grammar to obtain different measures from the analysis of the source code. For example, the amount of files, the average lines of code, the average number of classes, the average number of lines per class, average attributes per class, average methods per class, average static methods, and so on, combined with Halstead metrics [15] such as bugs delivered, difficulty, effort, time to understand or implement, and volume. For prediction, he experimented with support vector regression, extra trees regression, and support vector regression on averages.

- *delair* [8] combined style features (e.g. code layout and formatting, indentation, headers, Javadoc, comments, whitespaces) with content features (e.g. class design problems, method design problem, annotations, block checks, coding, imports, metrics, modifiers, naming conventions, size violations). They trained a support vector machine for regression, gaussian processes, M5, M5 rules and random trees.

- *doval* [9] approached the task with a shallow Long Short Term Memory (LSTM) recurrent neural network. It works at the byte level, meaning that at each time step a new byte from the input text is processed by the network in an ordered manner. Bytes belonging to a particular source code package in an input text file are considered as a sequence, where the processing of some byte at time step t is influenced by the previous time steps *t-1, t-2, ... , 0* (initial time step). The network learning criterion is a smoothed mean absolute error which uses a squared term if the absolute element-wise error falls below 1.

[5]https://github.com/antlr

- *gimenez* [13] proposed two different approaches to tackle this task. On the one hand, each code sample from each author was taken as an independent sample and vectorized using word n-grams; on the other hand, all the codes from an author was taken as a unique sample vectorized using word n-grams together with hand-crafted features (e.g. number of codes that implemented the same class, the appearance of pieces of code suspicious of plagiarism, number of developed classes, number of different classes). Regardless of the approach, a logistic regression model was trained.

- *hhu* [19] extracted structure (e.g. number of methods per class, length of function names, cyclomatic complexity) and style (e.g. length of methods per class, number of comments per class) features but ignored layout features (e.g. indentation) because they may be easily modifiable by the programming IDE. They used variance and range besides mean to aggregate the frequencies and then, constructed a separate model for each trait training both linear regression and nearest neighbour models.

- *kumar* [12] used multiple linear regression to model each of the five personality traits. For each personality trait, they have used four features: *i)* the number of genuine comment words in multi-line comments, i.e., between */** and ***/* found in the program code; *ii)* the number of genuine single-line comment words in single line comments, i.e., comments following "//". Both in the previous feature and in this one, they have not considered the cases where lines of code are commented and the feature value is normalized by dividing it by the total number of words in the program file; *iii)* the number of lines containing non-existent spaces, e.g., *for (int i=1; i<=cases; i++)* as opposed to *for (int i = 1; i< = cases; i++)*, since the presence of spaces is supposed to be a good programming practice (this feature value is normalized by dividing it by the total number of lines in the program file); *iv)* the number of instances where the programmer has imported the specific libraries only (e.g. cases of import java.io.FileNotFoundException as opposed to import java.io.*) as this is supposed to be a good programming practice. This feature value was also normalized with respect to the total number of lines in the program file.

- *uaemex* [28] obtained three types of features related with: *i)* Indentation: space in code, space in the comments, space between classes, spaces between source code blocks, space between methods, spaces between control sentences, and spaces in clustering characters "( ), [ ], { }"; *ii)* Identifier: the presence of underscore, uppercase, lowercase and numbers characters in the identifier, and the length of the identifier. These characteristics were extracted for each class, method and variable names. Also, the percentage of number of initialized variables was extracted; and *iii)* Comments: the presence of line and block comments, the size of the comments, and the presence of comments with all letters in uppercase. They have experimented with symbolic regression, support vector machines, k-nearest neighbours, and neural networks.

Although *montejo* have not sent a working note, they sent us a brief description of their system. They have used ToneAnalyzer[6], an IBM Watson module that proposes a value for each big five trait for a given text. The authors used ToneAnalyzer with the source code as it is and rescaled the output to fit the right range for the traits. Similarly, *lee* sent us the description of their system. They set a hypothesis that according to the personality, there will be differences in the steps of the source codes. Given a *ith* coder and n source codes for a coder c_i, the authors sorted codes by length and naming c_i^0 to c_i^{n-1}. They transformed each code to a vector v_i^j using skip-thought encoding [17], then calculated *n-1* difference vectors d_i^j using equation $d_i^j = v_i^{j+1} - v_i^j$. The authors plot each coder to a feature space $Sum(d_i)$ and $Avg(d_i)$, and then apply logistic regression algorithm to train a model.

Furthermore, we have provided with two baselines:

- *bow*: a bag of character 3-grams with frequency weight.
- *mean*: an approach that always predicts the mean value observed in the training data.

5. EVALUATION AND DISCUSSION OF THE SUBMITTED APPROACHES

Results are presented in Table 1 in alphabetical order. Below the participants' results, a summary with the common descriptive statistics is provided for each trait. In the bottom of the table, results for the baselines are also provided. Figures 1 to 3 show the distribution of the two measures: RMSE and Pearson correlation for all the participants except the baselines. In Figure 1 we can appreciate that there are many runs with anomalous RMSE values (outliers), whereas in Figure 2 we have removed these outliers. Looking at these figures and at the table of results, we can observe that:

- The mean is between 10.49 and 12.75 (a difference of 2.26), corresponding the lowest value to openness and the highest one to neuroticism.
- The median is between 8.14 and 10.77 (a difference of 2.63), corresponding again the lowest value to openness and the highest one to neuroticism.
- The lowest difference between mean and median was obtained for conscientiousness (1.75), followed by neuroticism (1.98). The highest difference was obtained for extroversion (2.72), agreeableness (2.36) and openness (2.35).
- In all the cases, the mean is higher than the median, and also than the 3rd quartile ($q3$), showing the effect of the outliers.
- The minimum and maximum values were obtained for openness trait (6.95 and 33.53 respectively).
- When removing outliers, the maximum value was obtained for extroversion (16.67).
- The lowest quartiles, both 1st and 3rd quartiles ($q1$ and $q3$), correspond to openness (7.54 and 9.58 respectively).
- The narrowest inter quartile range corresponds to conscientiousness (1.22), followed by neuroticism (1.84) and openness (2.04). The widest correspond to extroversion (3.23), followed by agreeableness (2.28).

In Figure 3 the distribution of the Pearson correlations is shown. Looking at this figure and at the table of results, we can observe that:

- There is only one outlier in agreeableness trait (0.38). Regretfully, this correlation corresponds to a high value in the RMSE (25.53).
- The mean is between -0.01 and 0.09 (a difference of 0.10), corresponding the lowest value to conscientiousness and agreeableness, and the highest one to openness. In any case, values very close to the random chance.
- The median is between -0.03 and 0.08 (a difference of 0.11), corresponding the lowest value to agreeableness and the highest one to extroversion.
- The lowest difference between mean and median was obtained for conscientiousness (0), followed by neuroticism (0.01), and extroversion, agreeableness and openness (0.02).
- The mean is higher than the median in case of openness (0.09 vs. 0.07) and agreeableness (-0.01 vs. -0.03). The other occurs in case of neuroticism (0.04 vs. 0.05), extroversion (0.06 vs. 0.08), and conscientiousness (in both -0.01).
- The minimum value was obtained for the extroversion trait (-0.37), very close to openness (-0.36), and the maximum for openness (0.62), followed by extroversion (0.47), agreeableness (0.38), neuroticism (0.36) and conscientiousness (0.33).
- Nevertheless the goodness of the maximum values, they correspond in most cases with high RMSE: openness (23.62), extroversion (28.80), agreeableness (25.53), and conscientiousness (22.05). Only in case of neuroticism the maximum Pearson correlations corresponds to a low value of RMSE (10.22).
- The highest $q3$ corresponds to openness (0.28) and extroversion (0.21), followed by conscientiousness (0.14 and neuroticism (0.14). The lowest one corresponds to agreeableness (0.07).
- The narrowest inter quartile range corresponds to agreeableness (0.18), followed by neuroticism (0.22), conscientiousness (0.28), extroversion (0.31) and openness (0.33).

We can conclude that, in general, systems performed similarly in terms of Pearson correlation for all the traits. However, there seem to be higher differences with respect to RMSE, where the systems obtained better results for openness than for the rest. The distributions show that the lowest sparsity occurs with conscientiousness in case of RMSE and agreeableness in case of Pearson correlation, meanwhile the highest sparsity occurs with extroversion in case of RMSE and openness in case of Pearson correlation.

[6]https://tone-analyzer-demo.mybluemix.net/

Results for **neuroticism** are plotted in Figure 4. This figure represents each system's results by plotting its RMSE in x axis and Pearson correlation in y axis. It is worth to mention that the system proposed by *delair* in their 4th run obtained one of the highest values for Pearson correlation (0.29) although with a high RMSE (17.55). This system consists in a combination of style features (code layout and formatting, indentation...) and content features (class design, method design, imports...), trained with random trees. We can also observe a group of five (actually six due to two systems that obtained the same results) in the upper-left corner of the chart. These systems obtained the highest correlations with the lowest error, and they are detailed in Figure 5. We can see that all of them (except *lee* which used skip-thought encoding) extracted specific features from the source code, such as the number of methods, the number of comments per class, the type of comments (/* */ vs. inline), type of naming variables, and so on. We can see that some of these teams obtained similar results for two of their systems. For example, *kumar* with their 1st and 2nd runs (they used linear regression for both runs, but they tried to optimise run 2 by removing from the training set the three files which obtained the highest error in training), or *hhu* that obtained the best results for their 2nd and 4th run (they both used k-NN with a different combination of features). *Uaemex* obtained their best result with run 3 that used neural networks. We can conclude that for neuroticism, specific features extracted from the code (*kumar*, *hhu*, *uaemex*) worked better than generic features such as n-grams (*besumich*, that obtained low RMSE but without correlation in most cases), byte streams (*doval*, that obtained low RMSE but with negative correlations in most cases) or text streams (*montejo*, that obtained high RMSE with low correlations).

In Figure 6 results for **extroversion** are shown. We can see that *doval* in their 4th run obtained both the highest Pearson correlation (0.47) but with the worst RMSE (28.80). They trained a LSTM recurrent neural network by converting the input at byte level, that is, without the need of performing feature engineering. In the upper-left corner of the figure we can see the group of the best results both in RMSE and Pearson correlation, that is detailed in Figure 7. We can highlight the superiority of *besumich* run 5 (lowercased character unigrams weighted with TF-IDF and training a Lasso regression algorithm with alpha 0.01), which obtained a correlation of 0.38 with a RMSE of 8.60, and *kumar* run 1 (code specific features with logistic regression without optimisation), with a correlation of 0.35 and a RMSE of 8.60. It is worth to mention that *lee* obtained high results with four of their approaches that use skip-thought encoding, and similar occurred with *gimenez*. The last one used a combination of word n-grams with specific features obtained from the code (the number of code that implemented the same class, the appearance of pieces of code suspicious of plagiarism, the number of classes developed, and the number of different classes developed), trained with ridge runs 1 (8.75 / 0.31) and 2 (8.79 / 0.28), and logistic regression run 4 (8.69 / 0.28). In case of extroversion we can see that common features such as n-grams (*besumich*) obtained good results. Also *gimenez* used word n-grams in combination to other features, what supports this conclusion. However, byte streams (*doval*) again produced high RMSE with high correlation, or text streams (*montejo*) produced high RMSE but with low correlation. In some cases, specific features obtained low RMSE but with negative correlation (*bilan*, *hhu*, *uaemex*). Although the *bow-based baseline* is not in the top performing methods, it obtained low RMSE (9.06) with over the median correlation (0.12).

Similarly, **openness** results are presented in Figure 8. It is noticeable that two systems presented by *delair* obtained the highest correlations but with quite high RMSE. Concretely, run 1 obtained the highest correlation (0.62) with high RMSE (23.62), and run 3 obtaining the second highest correlation (0.54) with a little lower RMSE (20.28). They used M5rules and M5P respectively. Systems in the upper-left corner are shown in detail in Figure 9. We can see that the best result for both RMSE and Pearson correlation was obtained by *uaemex* in their 1st run. This run was generated using symbolic regression with three types of features: indentation, identifiers and comments. The authors optimised this run by eliminating the source codes of five developers according to the following criteria: the person who had high values in all the personality traits, the person who had a lower values in all the personality traits, the person who had an average values in all the personality traits, the person who had more source codes and the person who had few source codes. They also obtained high results with their 3rd run, where they trained a back propagation neural network with the whole set of training codes. Systems presented by *bilan* also obtained high results in different runs. Concretely, using Antlr parser to obtain features in combination with features extracted from comments and so on, they trained gradient boosted regression and multinomial logistic regression. Similarly, *castellanos* who used also Antlr combined with Halstead measures and trained extra tree regressor (run 2) and support vector regression on averages (run 3); *kumar* with combinations of structure and style features trained with linear regression (2nd run optimised by eliminating training files); and *hhu* also with combinations of structure and style features with k-NN in both runs. For openness the best performing teams used specific features extracted from the code (*uaemex*, *kumar*, *hhu*), even with the help of code analysers such as Antlr (*castellanos*, *bilan*). Common features seem to obtain good level of RMSE but with low (or even negative) correlations (*besumich*, *bow-based baseline*).

In case of **agreeableness**, as shown in Figure 10 we can see that *doval* with their 4th run obtained the highest correlation (0.38), but with a high RMSE (25.53). Systems in the upper-left corner are shown in detail in Figure 11. We can say that the best result in both measures was obtained by *gimenez* in their 3rd run. The team used ridge to train their model with a subset of code style features. It is worth mentioning that the provided baseline consistent in character n-grams appears as one of the top performing methods for this trait. For this trait is more difficult to differentiate between common and specific features since there are many different teams that, although obtained low RMSE, have negative correlations. For example *besumich* with character n-grams, *bilan* and *castellanos* with specific features obtained with Antlr (among others), or *delair* with a combination of style and content features. However, it is worth to mention that the *bow baseline* obtained top results both in RMSE and Pearson correlation.

Finally, with respect to **conscientiousness** results are depicted in Figure 12. We can see that four runs obtained high values for Pearson correlation but also obtained high

RMSE. Concretely, *delair* obtained the highest correlation (0.33) with the second highest RMSE (22.05) with their 1st and 3rd runs (M5rules and M5P respectively), and also a high correlation (0.27) with a little lower RMSE (15.53) with their 5th run (support vector machine for regression). Similarly, *doval* with their 4th run obtained high correlation (0.32) but with high RMSE (14.69) by using LSTM recurrent neural network with a byte level input. Systems in the upper-left corner are represented in Figure 13. In this case, the best results in terms of RMSE are not the best ones in terms of Pearson correlation: with respect to the first ones, *hhu* with runs 1, 2 and 3 or *uaemex* with run 1. With respect to the second ones, *lee* with runs 2, 4 and 5, *bilan* with runs 4 and 5, and *doval* with run 3. It is noticeable that again the provided *baseline* obtained one of the best results. In this case the second better RMSE with one of the top 5 correlations. In case of conscientiousness, systems that used n-grams (*besumich*, *gimenez*), byte streams (*doval*) and text streams (*montejo*) performed worst in case of Pearson correlation, with negative values in most cases, whereas the best results were achieved by combinations of structure, style and comments (*hhu*, *uaemex*) or features obtained by analysing the codes (*bilan*). However, again the *bow baseline* achieved top positions, specially in RMSE.

To sum up, depending on the trait, generic features such as n-grams obtained different results in comparison with specific features obtained from the code. In case of generic features, their impact is specially on correlation: they may obtain good levels of RMSE but without a good correlation. As it was expected, the mean-based baseline obtained no correlation, since it seems more a random value. However, its RMSE was better than the average results and the median results in most cases. This result supports the need of using also a measure like Pearson correlation in order to avoid low RMSE due to random chance.

6. CONCLUSION

This paper describes the 48 runs sent by 11 participants to the PR-SOCO shared task at PAN-FIRE 2016. Given a set of source codes written in Java by students who answered a personality test, the participants had to predict values for the big five traits.

Results have been evaluated with two complementary measures: RMSE, which provides an overall score of the performance of the system, and Pearson product-moment correlation, which indicates whether the performance is due to the random chance. In general, systems showed to work quite similarly in terms of Pearson correlation for all traits. Higher differences where noticed with respect to RMSE. The best results were achieved for openness (6.95), as it was previously reported by Mairesse *et al.* [20], as well as this was one of the traits with the lower RMSE at PAN 2015 [25] for most languages.

Participants have used different kinds of features: from general ones such as word or character n-grams to specific ones obtained by parsing the code, analysing its structure, style or comments. Depending on the trait, generic features obtained competitive results compared with specific ones in terms of RMSE. However, in most cases the best RMSE obtained with these features obtained low values of the Pearson correlation. In these cases, some systems seemed to be less robust, at least for some of the personality traits.

Finally, in line with the above comments, it is worth mentioning that approaches that took advantage of the training distributions (such as the baseline based on means did), obtained low RMSE. However, this may be due to random chance. This supports the need of using complementary measures to RMSE such as Pearson correlation, in order to avoid misinterpretations due to a biased measure.

7. ACKNOWLEDGMENTS

Our special thanks go to all of PR-SOCO participants. The work of the first author was partially supported by Autoritas Consulting and by Ministerio de Economía y Competitividad de España under grant ECOPORTUNITY IPT-2012-1220-430000. The work of the fifth author was partially supported by the SomEMBED TIN2015- 71147-C2-1-P MINECO research project and by the Generalitat Valenciana under the grant ALMAMATER (PrometeoII/2014/030).

8. REFERENCES

[1] Y. Bachrach, M. Kosinski, T. Graepel, P. Kohli, and D. Stillwell. Personality and patterns of facebook usage. In *Proceedings of the ACM Web Science Conference*, pages 36–44. ACM New York, NY, USA, 2012.

[2] I. Bilan, E. Saller, B. Roth, and M. Krytchak. Caps-prc: A system for personality recognition in programming code - notebook for pan at fire16. In *Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016*, CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[3] C. Bishop-Clark. Cognitive style, personality, and computer programming. *Computers in Human Behavior*, 11(2):241–260, 1995.

[4] H. A. Castellanos. Personality recognition applying machine learning techniques on source code metrics. In *Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016*, CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[5] F. Celli, B. Lepri, J.-I. Biel, D. Gatica-Perez, G. Riccardi, and F. Pianesi. The workshop on computational personality recognition 2014. In *Proceedings of the ACM International Conference on Multimedia*, pages 1245–1246. ACM, 2014.

[6] F. Celli and L. Polonio. Relationships between personality and interactions in facebook. In *Social Networking: Recent Trends, Emerging Issues and Future Outlook*, pages 41–54. Nova Science Publishers, Inc, 2013.

[7] P. T. Costa and R. R. McCrae. The revised neo personality inventory (neo-pi-r). *The SAGE handbook of personality theory and assessment*, 2:179–198, 2008.

[8] R. Delair and R. Mahajan. Personality recognition in source code. In *Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016*, CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[9] Y. Doval, C. Gómez-Rodríguez, and J. Vilares. Shallow recurrent neural network for personality recognition in source code. In *Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation,*

Kolkata, India, December 7-10, 2016, CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[10] E. Flores, P. Rosso, L. Moreno, and E. Villatoro-Tello. Pan@ fire: Overview of soco track on the detection of source code re-use. In *Notebook Papers of FIRE 2014, FIRE-2014, Bangalore, India*, 2014.

[11] E. Flores, P. Rosso, L. Moreno, and E. Villatoro-Tello. Pan@ fire 2015: Overview of cl-soco track on the detection of cross-language source code re-use. In *Proceedings of the Seventh Forum for Information Retrieval Evaluation (FIRE 2015), Gandhinagar, India*, pages 4–6, 2015.

[12] K. Ghosh and S. Kumar-Parui. Indian statistical institute, kolkata at pr-soco 2016 : A simple linear regression based approach. In *Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016*, CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[13] M. Giménez and R. Paredes. Prhlt at pr-soco: A regression model for predicting personality traits from source code - notebook for pr-soco at fire 2016. In *Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016*, CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[14] J. Golbeck, C. Robles, and K. Turner. Predicting personality with social media. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*, pages 253–262. ACM, 2011.

[15] M. H. Halstead. Elements of software science. operating and programming systems series, vol. 2, 1977.

[16] Z. Karimi, A. Baraani-Dastjerdi, N. Ghasem-Aghaee, and S. Wagner. Links between the personalities, styles and performance in computer programming. *Journal of Systems and Software*, 111:228–241, 2016.

[17] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, 2015.

[18] M. Kosinski, Y. Bachrach, P. Kohli, D. Stillwell, and T. Graepel. Manifestations of user personality in website choice and behaviour on online social networks. *Machine Learning*, pages 1–24, 2013.

[19] M. Liebeck, P. Modaresi, A. Askinadze, and S. Conrad. Pisco: A computational approach to predict personality types from java source code. In *Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016*, CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[20] F. Mairesse, M. A. Walker, M. R. Mehl, and R. K. Moore. Using linguistic cues for the automatic recognition of personality in conversation and text. *Journal of Artificial Intelligence Research*, 30(1):457–500, 2007.

[21] J. Oberlander and S. Nowson. Whose thumb is it anyway?: classifying author personality from weblog text. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 627–634. Association for Computational Linguistics, 2006.

[22] O. H. Paruma-Pabón, F. A. González, J. Aponte, J. E. Camargo, and F. Restrepo-Calle. Finding relationships between socio-technical aspects and personality traits by mining developer e-mails. In *Proceedings of the 9th International Workshop on Cooperative and Human Aspects of Software Engineering*, pages 8–14. ACM, 2016.

[23] S. Phani, S. Lahiri, and A. Biswas. Personality recognition working note: Team besumich. In *Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016*, CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[24] D. Quercia, R. Lambiotte, D. Stillwell, M. Kosinski, and J. Crowcroft. The personality of popular facebook users. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, pages 955–964. ACM, 2012.

[25] F. Rangel, P. Rosso, M. Potthast, B. Stein, and W. Daelemans. Overview of the 3rd author profiling task at pan 2015. In *Cappellato L., Ferro N., Jones G., San Juan E. (Eds.) CLEF 2015 labs and workshops, notebook papers. CEUR Workshop Proceedings. CEUR-WS.org, vol. 1391*, 2015.

[26] H. A. Schwartz, J. C. Eichstaedt, M. L. Kern, L. Dziurzynski, S. M. Ramones, M. Agrawal, A. Shah, M. Kosinski, D. Stillwell, M. E. Seligman, et al. Personality, gender, and age in the language of social media: The open-vocabulary approach. *PloS one*, 8(9):773–791, 2013.

[27] S. A. Sushant, S. Argamon, S. Dhawle, and J. W. Pennebaker. Lexical predictors of personality type. In *In Proceedings of the Joint Annual Meeting of the Interface and the Classification Society of North America*, 2005.

[28] E. Vázquez-Vázquez, O. González-Brito, J. Armeaga-García, M. García-Calderón, G. Villada-Ramírez, A. J. Serrano-León, R. A. García-Hernández, and Y. Ledeneva. Uaemex system for identifying traits personality in source code. In *Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016*, CEUR Workshop Proceedings. CEUR-WS.org, 2016.

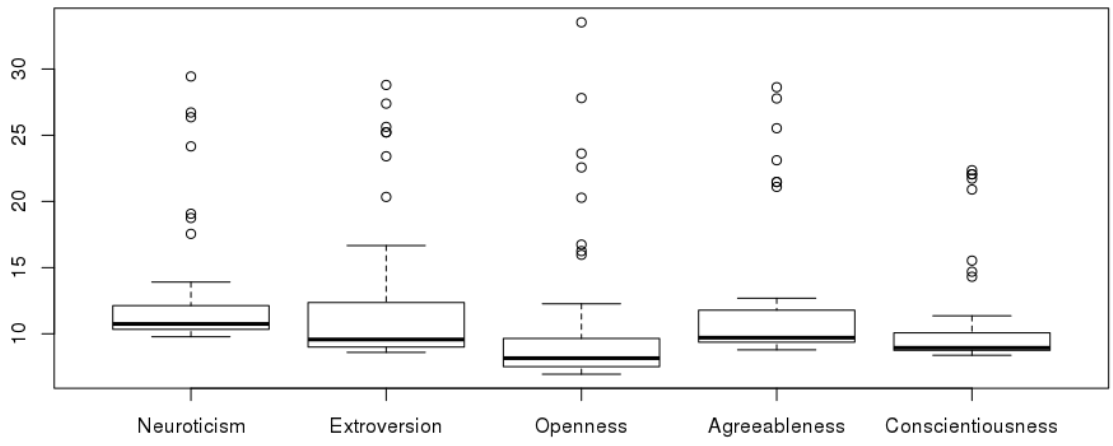

Figure 1: RMSE distribution.

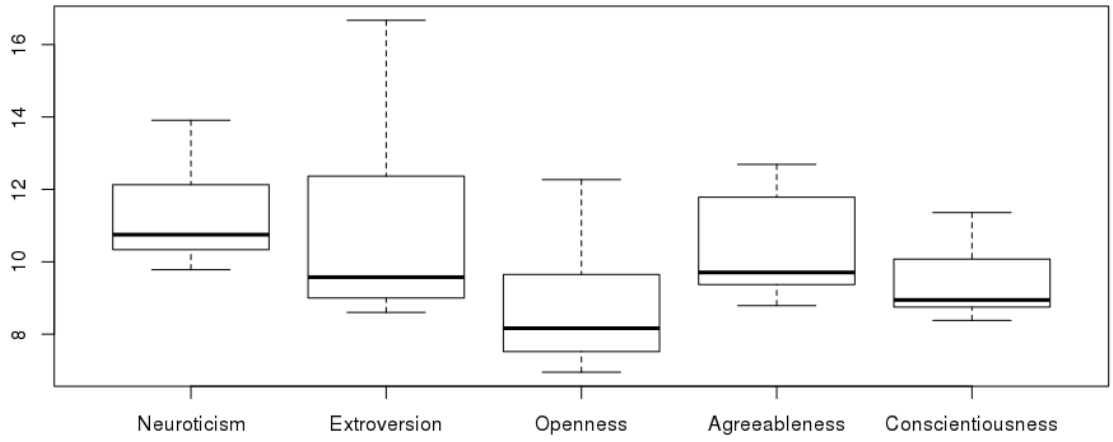

Figure 2: RMSE distribution (without outliers).

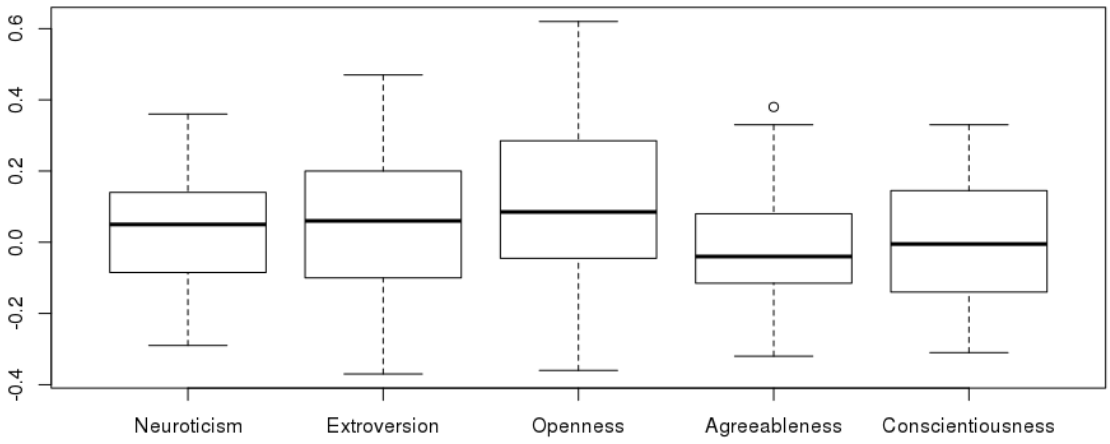

Figure 3: Pearson correlation distribution.

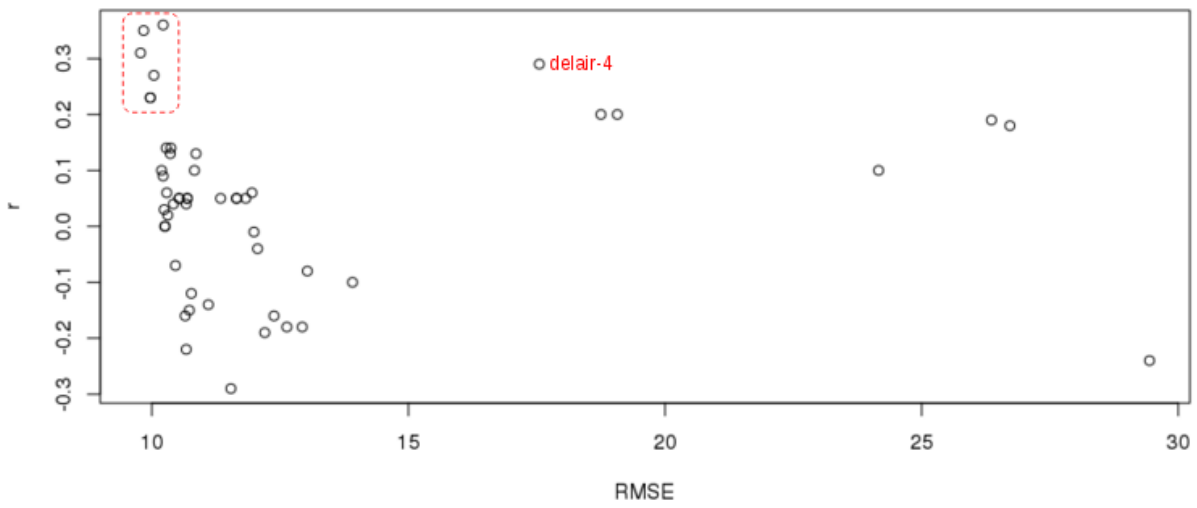

Figure 4: RMSE vs. PC for neuroticism.

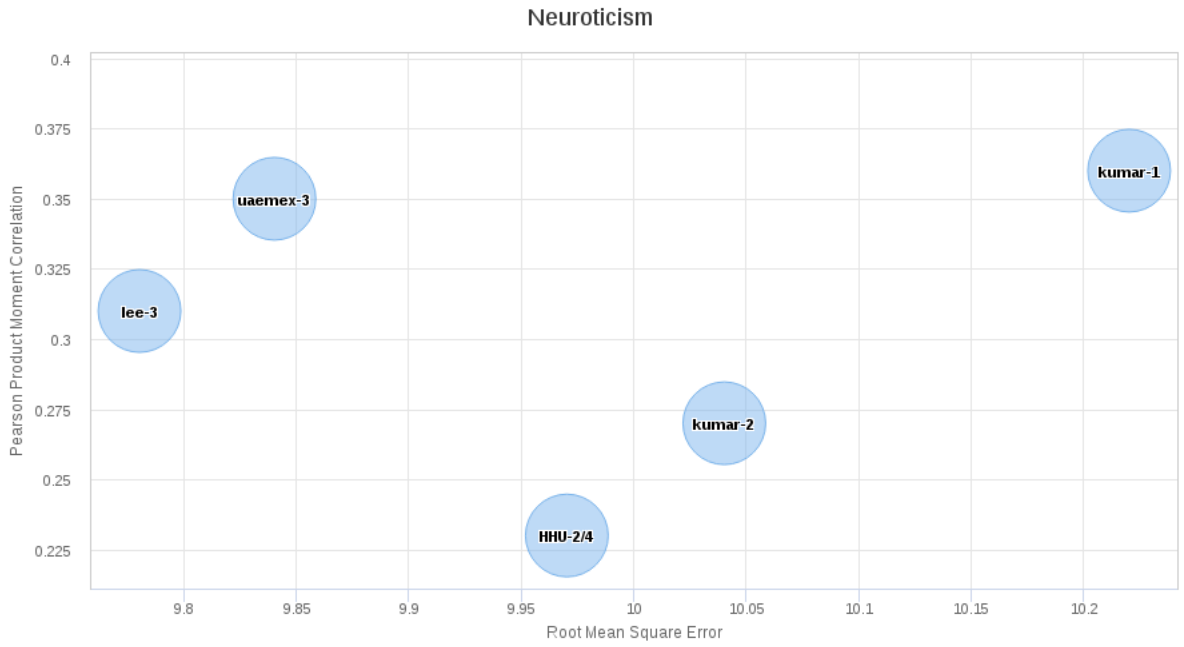

Figure 5: RMSE vs. PC for neuroticism (detailed).

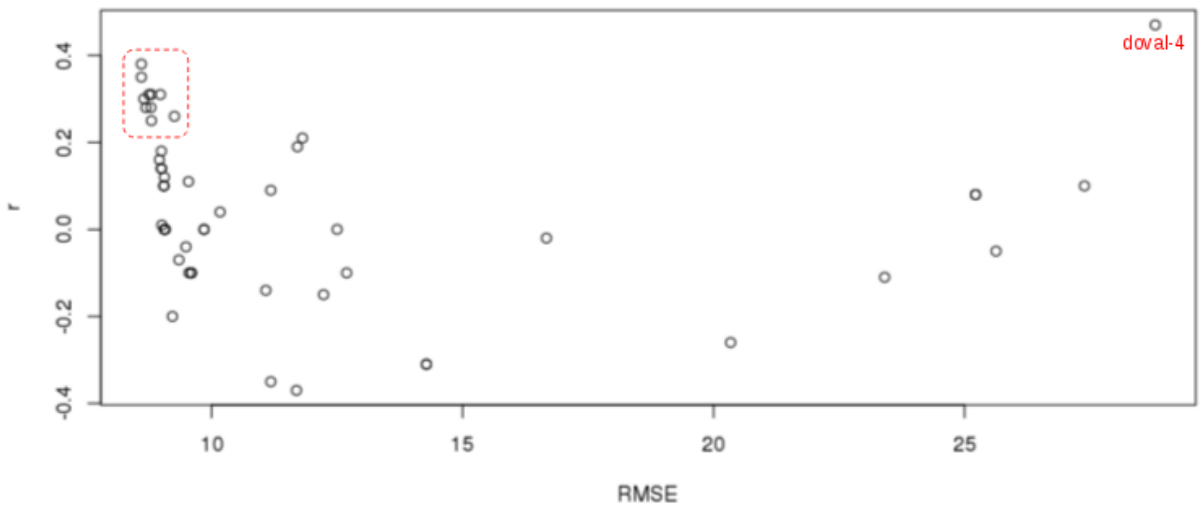

Figure 6: RMSE vs. PC for extroversion.

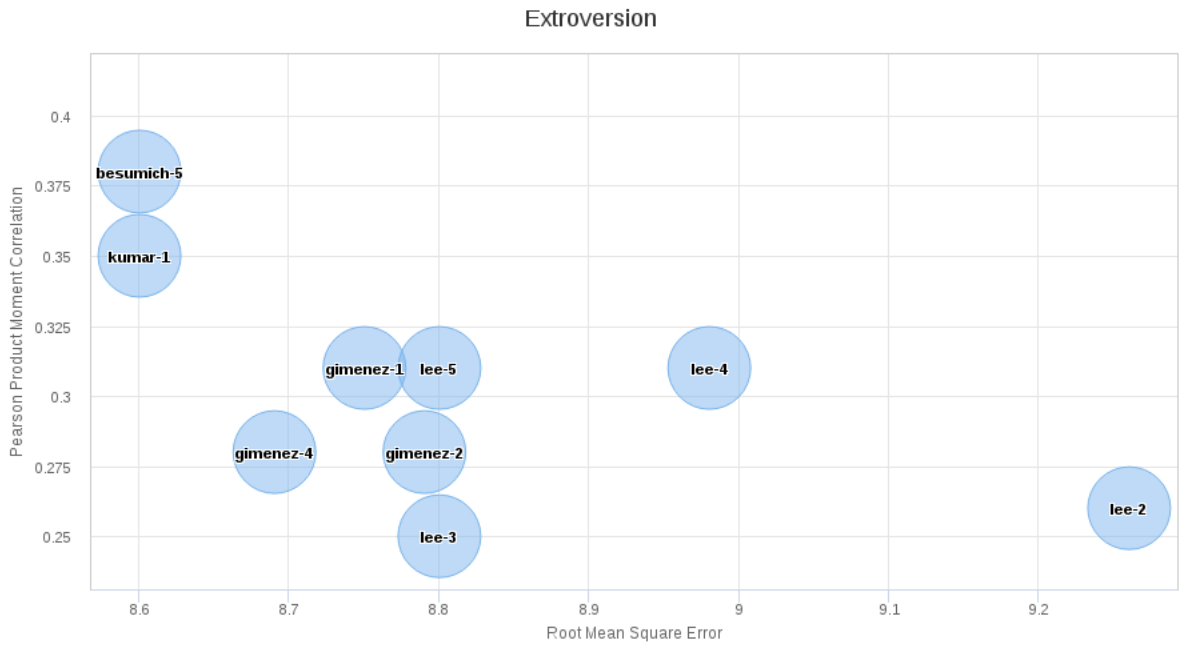

Figure 7: RMSE vs. PC for extroversion (detailed).

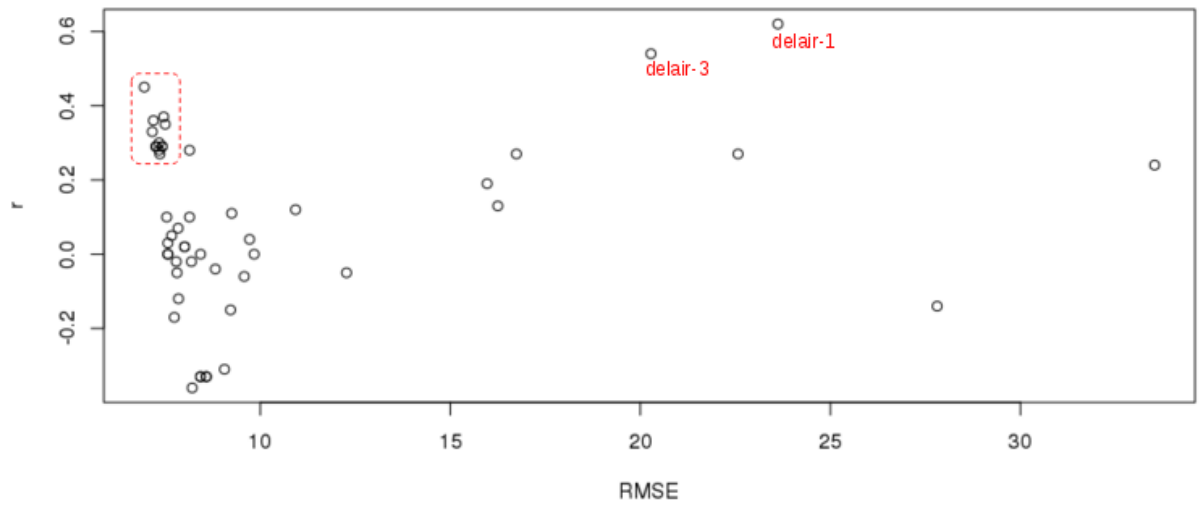

Figure 8: RMSE vs. PC for openness.

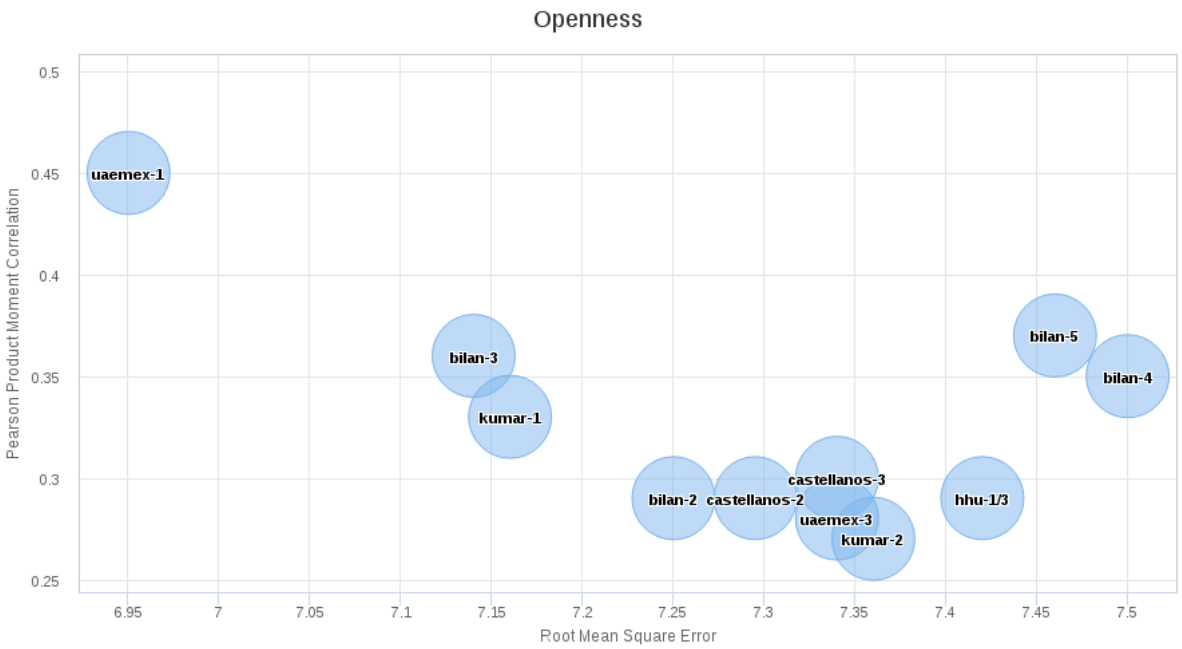

Figure 9: RMSE vs. PC for openness (detailed).

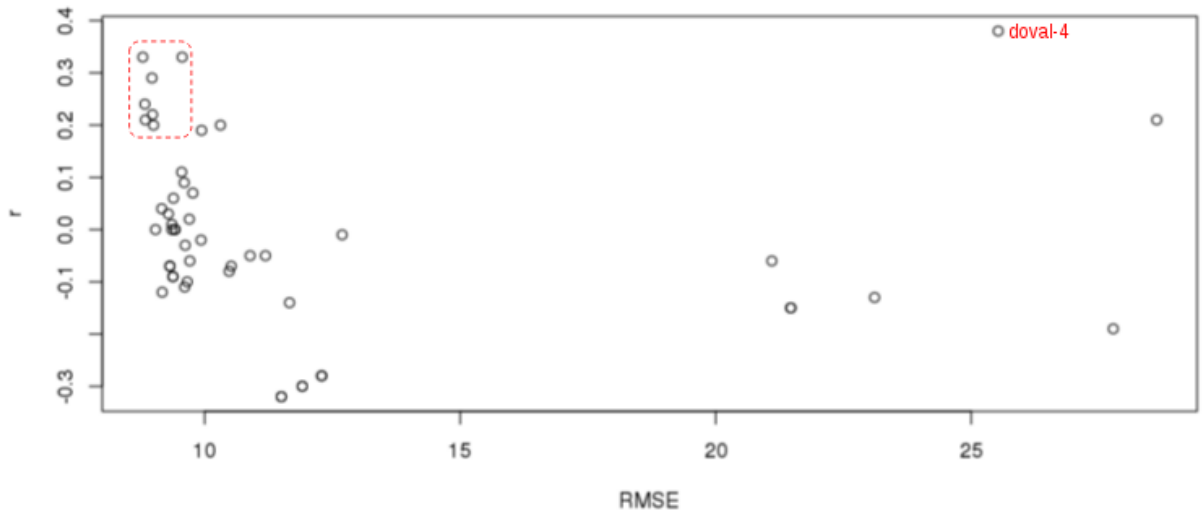

Figure 10: RMSE vs. PC for agreableness.

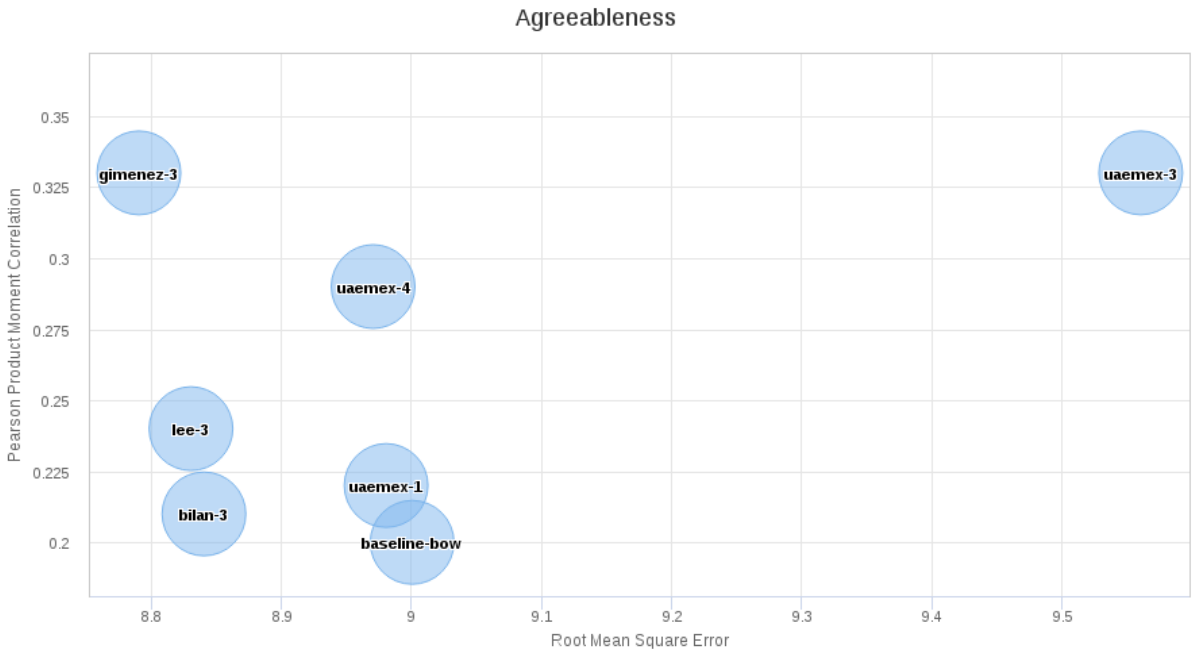

Figure 11: RMSE vs. PC for agreableness (detailed).

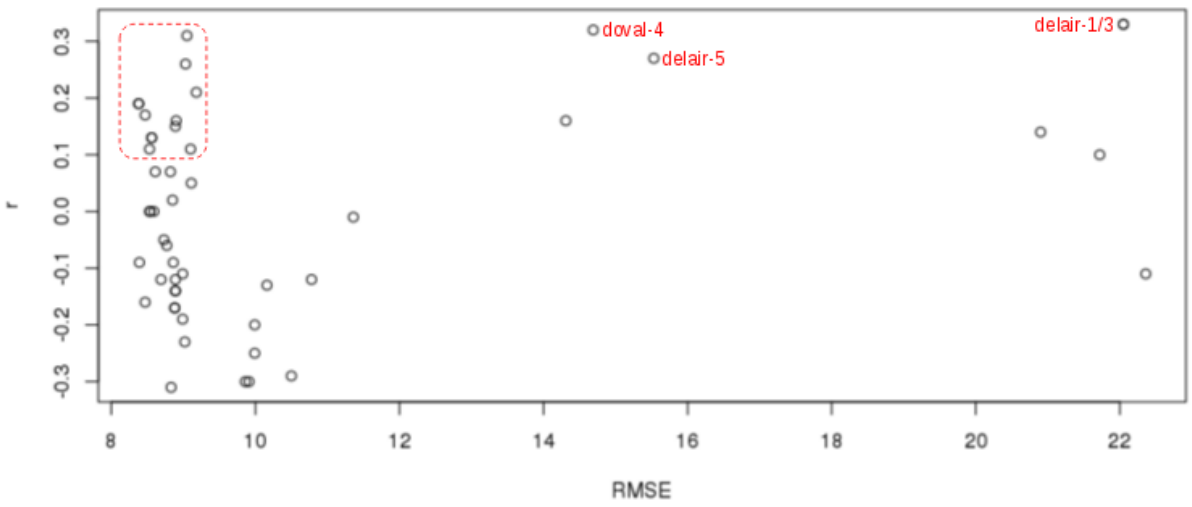

Figure 12: RMSE vs. PC for conscientiousness.

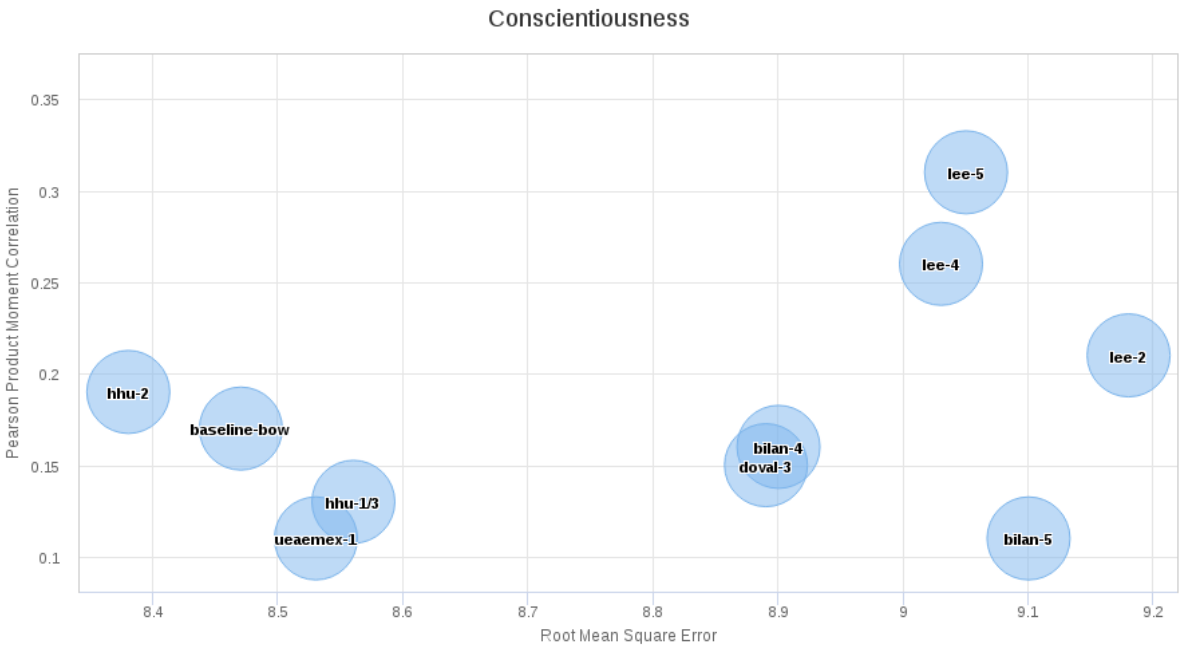

Figure 13: RMSE vs. PC for conscientiousness (detailed).

Table 1: Participants' results in terms of root mean square error and Pearson product moment correlation.

Team	Run	Neuroticism	Extroversion	Openness	Agreeableness	Conscientiousness
besumich	1	10.69 / 0.05	9.00 / 0.14	8.58 / -0.33	9.38 / -0.09	8.89 / -0.14
	2	10.69 / 0.05	9.00 / 0.14	8.58 / -0.33	9.38 / -0.09	8.89 / -0.14
	3	10.53 / 0.05	9.05 / 0.10	8.43 / -0.33	9.32 / -0.07	8.88 / -0.17
	4	10.53 / 0.05	9.05 / 0.10	8.43 / -0.33	9.32 / -0.07	8.88 / -0.17
	5	10.83 / 0.10	**8.60** / 0.38	9.06 / -0.31	9.66 / -0.10	8.77 / -0.06
bilan	1	10.42 / 0.04	8.96 / 0.16	7.54 / 0.10	9.16 / 0.04	8.61 / 0.07
	2	10.28 / 0.14	9.55 / -0.10	7.25 / 0.29	9.17 / -0.12	8.83 / -0.31
	3	10.77 / -0.12	9.35 / -0.07	7.19 / 0.36	8.84 / 0.21	8.99 / -0.11
	4	12.06 / -0.04	11.18 / -0.35	7.50 / 0.35	10.89 / -0.05	8.90 / 0.16
	5	11.95 / 0.06	11.69 / -0.37	7.46 / 0.37	11.19 / -0.05	9.10 / 0.11
castellanos	1	11.83 / 0.05	9.54 / 0.11	8.14 / 0.28	10.48 / -0.08	8.39 / -0.09
	2	10.31 / 0.02	9.06 / 0.00	7.27 / 0.29	9.61 / -0.11	8.47 / -0.16
	3	10.24 / 0.03	9.01 / 0.01	7.34 / 0.30	9.36 / 0.01	9.99 / -0.25
delair	1	19.07 / 0.20	25.22 / 0.08	23.62 / **0.62**	21.47 / -0.15	22.05 / **0.33**
	2	26.36 / 0.19	16.67 / -0.02	15.97 / 0.19	23.11 / -0.13	21.72 / 0.10
	3	18.75 / 0.20	25.22 / 0.08	20.28 / 0.54	21.47 / -0.15	22.05 / **0.33**
	4	17.55 / 0.29	20.34 / -0.26	16.74 / 0.27	21.10 / -0.06	20.90 / 0.14
	5	26.72 / 0.18	23.41 / -0.11	16.25 / 0.13	27.78 / -0.19	15.53 / 0.27
doval	1	11.99 / -0.01	11.18 / 0.09	12.27 / -0.05	10.31 / 0.20	8.85 / 0.02
	2	12.63 / -0.18	11.81 / 0.21	8.19 / -0.02	12.69 / -0.01	9.91 / -0.30
	3	10.37 / 0.14	12.50 / 0.00	9.25 / 0.11	11.66 / -0.14	8.89 / 0.15
	4	29.44 / -0.24	28.80 / **0.47**	27.81 / -0.14	25.53 / **0.38**	14.69 / 0.32
	5	11.34 / 0.05	11.71 / 0.19	10.93 / 0.12	10.52 / -0.07	10.78 / -0.12
gimenez	1	10.67 / -0.22	8.75 / 0.31	7.85 / -0.12	9.29 / 0.03	9.02 / -0.23
	2	10.46 / -0.07	8.79 / 0.28	7.67 / 0.05	9.36 / 0.00	8.99 / -0.19
	3	10.22 / 0.09	9.00 / 0.18	7.57 / 0.03	**8.79** / 0.33	8.69 / -0.12
	4	10.73 / -0.15	8.69 / 0.28	7.81 / -0.05	9.62 / -0.03	8.86 / -0.09
	5	10.65 / -0.16	8.65 / 0.30	7.79 / -0.02	9.71 / -0.06	8.89 / -0.12
HHU	1	11.65 / 0.05	14.28 / -0.31	7.42 / 0.29	12.29 / -0.28	8.56 / 0.13
	2	9.97 / 0.23	9.60 / -0.10	8.01 / 0.02	11.91 / -0.30	**8.38** / 0.19
	3	11.65 / 0.05	14.28 / -0.31	7.42 / 0.29	11.50 / -0.32	8.56 / 0.13
	4	9.97 / 0.23	9.22 / -0.20	7.84 / 0.07	11.50 / -0.32	**8.38** / 0.19
	5	10.36 / 0.13	9.60 / -0.10	8.01 / 0.02	11.91 / -0.30	8.73 / -0.05
	6	13.91 / -0.10	25.63 / -0.05	33.53 / 0.24	12.29 / -0.28	14.31 / 0.16
kumar	1	10.22 / **0.36**	**8.60** / 0.35	7.16 / 0.33	9.60 / 0.09	9.99 / -0.20
	2	10.04 / 0.27	10.17 / 0.04	7.36 / 0.27	9.55 / 0.11	10.16 / -0.13
lee	1	10.19 / 0.10	9.08 / 0.00	8.43 / 0.00	9.39 / 0.06	8.59 / 0.00
	2	12.93 / -0.18	9.26 / 0.26	9.58 / -0.06	9.93 / -0.02	9.18 / 0.21
	3	**9.78** / 0.31	8.8 / 0.25	8.21 / -0.36	8.83 / 0.24	9.11 / 0.05
	4	12.20 / -0.19	8.98 / 0.31	8.82 / -0.04	9.77 / 0.07	9.03 / 0.26
	5	12.38 / -0.16	8.80 / 0.31	9.22 / -0.15	9.70 / 0.02	9.05 / 0.31
montejo	1	24.16 / 0.10	27.39 / 0.10	22.57 / 0.27	28.63 / 0.21	22.36 / -0.11
uaemex	1	11.54 / -0.29	11.08 / -0.14	**6.95** / 0.45	8.98 / 0.22	8.53 / 0.11
	2	11.10 / -0.14	12.23 / -0.15	9.72 / 0.04	9.94 / 0.19	9.86 / -0.30
	3	9.84 / 0.35	12.69 / -0.10	7.34 / 0.28	9.56 / 0.33	11.36 / -0.01
	4	10.67 / 0.04	9.49 / -0.04	8.14 / 0.10	8.97 / 0.29	8.82 / 0.07
	5	10.25 / 0.00	9.85 / 0.00	9.84 / 0.00	9.42 / 0.00	10.50 / -0.29
	6	10.86 / 0.13	9.85 / 0.00	7.57 / 0.00	9.42 / 0.00	8.53 / 0.00
min		9.78 / -0.29	8.60 / -0.37	6.95 / -0.36	8.79 / -0.32	8.38 / -0.31
q1		10.36 / -0.08	9.00 / -0.10	7.54 / -0.05	9.38 / -0.11	8.77 / -0.14
median		10.77 / 0.05	9.55 / 0.08	8.14 / 0.07	9.71 / -0.03	8.99 / -0.01
mean		12.75 / 0.04	12.27 / 0.06	10.49 / 0.09	12.07 / -0.01	10.74 / -0.01
q3		12.20 / 0.14	12.23 / 0.21	9.58 / 0.28	11.66 / 0.07	9.99 / 0.14
max		29.44 / 0.36	28.80 / 0.47	33.53 / 0.62	28.63 / 0.38	22.36 / 0.33
		Neuroticism	Extroversion	Openness	Agreeableness	Conscientiousness
baseline	bow	10.29 / 0.06	9.06 / 0.12	7.74 / -0.17	9.00 / 0.20	8.47 / 0.17
	mean	10.26 / 0.00	9.06 / 0.00	7.57 / 0.00	9.04 / 0.00	8.54 / 0.00