CREATING A PERSONALITY MODEL USING GENETIC ALGORITHMS, BEHAVIORAL PSYCHOLOGY, AND A HAPPINESS DATASET

Ciprian-Ionuț NUȚESCU[1], Mariana MOCANU[2]

In this paper, we proposed a genetic algorithm based on behavioral psychology developed by Carl Gustav Jung (16 Personalities model), in which we describe the person's behavioral features related to his personality. The model used for inference is based on 40 years of psychology research and studies from the book "The 16 personality types that determinate how we live, love and work" by Otto Kroger and Janet M. Thuesen, published in 1988. To generate one's personality model, we are using genetic algorithms. To improve the personality mode, we are reinforcing it using weight-based fitness functions with inference extracted from the HappyDB dataset. In this dataset, people around the world express the most valuable/happy moment in their life. We conducted experiments on 25 individuals by generating their personality models using the proposed method above and collected feedback about how precise and accurate the model was presented. We obtained a total accuracy of 80% based on user feedback about their personality.

Keywords: Genetic algorithms, Evolutionary algorithms, Behavioral psychology, Personality model

1. Introduction

A person's personality is a model definition from their experiences, knowledge, and education developed across their entire life. Based on this approach we can relate that if we take two random persons from the entire world population, they will have similar personality characteristics, but not identical. Carl Gustav Jung was a Swiss psychiatrist and psychoanalyst who founded analytical psychology. He is the father of personality archetypes and based on his work the personality model was derived (MBTI - Myers-Briggs Personality Type Indicator), dividing the personality of a person among side 4 different axes: introversion-extroversion, intuitive-sensing, thinker-feeler, and judger-perceiver. The internet is full of this type of test, where you answer some relatively simple questions and obtain a certain personality indicator. Based on human analysis [1], it is discussed that a person is not 100% introverted or extroverted (for this example, we have taken into consideration the introversion-extroversion ax). A person is defined as a percentage of introversion and extroversion, where they will display certain personality characteristics based on circumstances.

[1] Ph.D student. Eng, Dept. of Computer Science, University POLITEHNICA of Bucharest, Romania, e-mail: cipinutescu@yahoo.com

[2] Professor, Dept. of Computer Science, University POLITEHNICA of Bucharest, Romania, e-mail: mariana.mocanu@cs.pub.ro

For this paper, we extracted the human behavioral model from the book "16 Personalities type that determinate how we live, love and work" [1] and include it in our proposed algorithm. The model contains various circumstances and behavioral indices based on a person's personality traits. As an example (well-known), introverts prefer small groups of people and are depleted when socializing, while extroverts prefer big groups of people and are energized when socializing. As another example, sensor people prefer specific answers to specific questions, while intuitive people tend to think about several things at once. As another example thinker, people would prefer to settle a dispute based on what is fair and trustworthy than make people happy, while feeler people will overextend themselves to meet other people's preferences. We extracted all these preferences for each axis, for different behavioral circumstances (thinking and concentration, social interaction and communication, reflection and self-image, work and creation). Even if the psychological model described in this book is 40 years old, this may be applied to today's world.

To reinforce the model described here, we opted to extract inference from a contemporary dataset, the HappyDB moments [2] using a BOIT-TFIDF method proposed in [3] from the moments, to use it as inference in our extracted model. Using this approach, we want to improve the accuracy and precision of the personality model to better match the user. The input for the algorithm will represent how the user perceives himself alongside the 4 axes of the personality model and information about gender, marital status, parenthood, and age. The algorithm and proposed hypothesis will be explained in the next section.

A related work on this domain is "The Myers-Briggs Personality Type and Its Relationship to Computer Programming" [10], which experimented on program-ming students to see how their personality influences their programming skills. It is empirically demonstrated that sensing students scored significantly higher than intuitive students on writing programs and judging students achieved higher grades on computer programs than did perceiving students. Introvert-extrovert personality characteristics did not affect students' programming skills. Most of the research on MBTI and Jung's personality theory is not centered on obtaining one's personality using machine learning but on determinating various patterns between personality and intelligence [11], problem-solving behaviors [12], or business success [13]. The determination of a person's personality (as implemented nowadays) is answering a suite of 50-100 questions (based on how accurate is the model) and receiving the personality results. The state of the art of this paper is the approach we are considerating: using machine learning and genetic algorithms to generate a person's personality model using less input (not answering 50-100 questions as in other methods of determination) and obtaining a good accuracy (based on experimental results).

2. Proposed algorithm and hypothesis

To determine a person's personality, we used a genetic algorithm [4]. Genetic algorithms are structured based on natural evolution, representing a subsection of evolutionary algorithms [5]. It resembles the same processing from nature as a cellular molecule: crossing-over, mutation, natural selection, population, and generation. It starts with an initial number of individuals containing a certain chromosome. After the crossing-over stage, new individuals are created by mixing the chromosomes from each parent. New individuals are exposed to a mutation stage, where the chromosome can mutate at certain genes (with probability). New individuals created are then filtered to a natural selection stage, where only the most fitted individuals are selected for the new population, and the rest of the individuals are discarded. This way, the algorithm converges to a better solution for each iteration (based that the fitness function is correct). In the figure below, we illustrate the stages of a genetic algorithm.

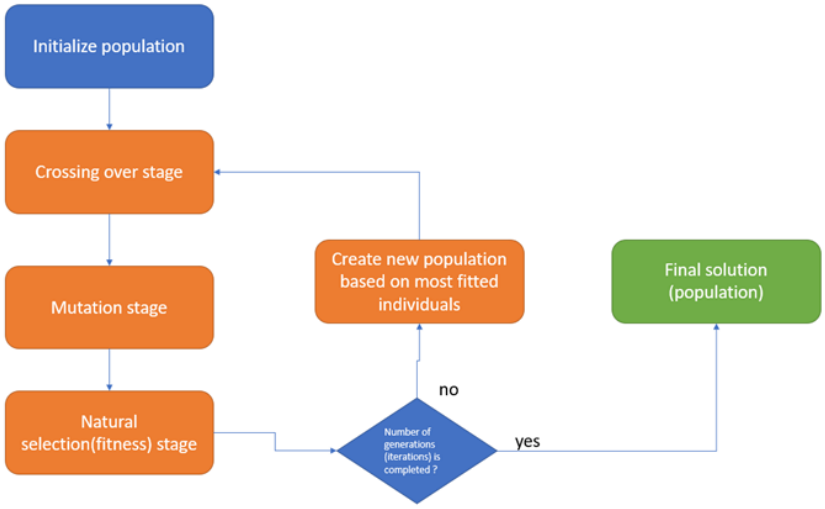

Fig. 1. General genetic algorithm stages

From [1], we extracted the preferences model for each type of personality axis (it is based on psychological and clinical studies). The model takes into account all 4 axes of personality and for each, behavioral preferences in different situations are described. The 4 axes are the following:

- Introvert-extrovert
- Sensor-intuitive
- Thinker-feeler
- Judger-perceiver

From these 4 axes, there are 4x4=16 personality types that define human behavior [1]. The behavior preferences extracted for each personality axis are linked to the environment(circumstances) a person reacts:

- Social interaction and communication
- Reflection and self-image
- Thinking and concentration
- Work and creation

This preference tells us how a person would react based on the circumstances. The model is extremely big and has a lot of circumstances and behavioral preferences. We exemplified some in this paper. Let's take some examples from the model (all extracted from [1]):

If you are an extrovert, you probably:

a) tend to talk first, think later (social interaction and communication)

b) know a lot of people, and count them as friends (social interaction and communication)

If you are introverted, you probably:

a) rehearse things before saying them (social interaction and communication)

b) enjoy the peace of quiet of having time to yourself (reflection and self-image)

If you are a sensor, you probably:

a) prefer specific answers to a specific question (thinking and concentration)

b) like to concentrate on what you are doing at the moment, and don't wonder about what's next (work and creation)

If you are intuitive, you probably:

a) tends to think about several things at once (thinking and concentration)

b) find the future very intriguing and are more excited about the future than the present (thinking and concentration)

Also, from [1], we extracted the personality temperaments and various trivia about the personality choices of fashion, jobs, and mating advice. We added this to our model to improve the matching to one personal behavior (but we didn't apply any machine learning techniques or genetic algorithms generation to it, only matching personality).

These preferences will represent the chromosome in our genetic algorithm. We will need an initial population for the genetic algorithm to start computing the best personality match for a person. For this, the person inputs a percentage for each axis, for example, 0.4 extrovert and 0.6 introvert, 0.2 perceiver, and 0.8 judgers. We opted for a percentage-based approach because based on the personality and psychology studies done in recent years, a person's personality is not

100% introverted or extroverted. We as humans, cannot be categorized like computers with 1 or 0, black or white, a more philosophical way of definition is multiple shades of gray. The input taken from the user is based on what he thinks about himself and how he/she seems. This is not a personality test with questions, where the personality is determined based on answering some questions that are based on the score achieved, we determine the personality of that person. The Internet is full of those types of questionnaires and tests. This is a new method using genetic algorithms to determine a person's preferences and behavioral models based on their personality. Also, as input, we required the gender, marital status, parenthood (if the user has kids or not), and age. We will this information to better match with our tf-idf maps extracted from [2] using the BOIT-TFIDF method.

After the input is taken from the user, the initial population is generated taking into account the percentages as follows: if 0.6 introverts and 0.4 extroverts, then the chromosome/individual (represented in our problem by the personality pref-erence lists) resulted will have 0.6 percentage introvert preferences and 0.4 ex-trovert preferences. This is applied for all 4 axes in the personality schema as shown in fig. 1, meaning that a chromosome/individual will have the follow-ing:

- Extrovert-introvert preferences
- Sensor-intuitive preferences
- Thinker-feeler preferences
- Judfer-perceiver preferences

This will represent the working unit for our genetic algorithm (and will be tak-en into consideration in the genetic algorithm stages).

For the crossing-over stage, we opted for a one-point crossover, meaning that from a two-parent chromosome, we select a point (and index), where the lists will be combined. This is applied the same (with the same index) for all 4 axes (lists) of preferences (explained above). For the mutation stage, we opted for a 0.1 mu-tation chance of a chromosome, this will mean that we select randomly a prefer-ence from one axis and exchange it with one from the model (this will be ran-dom). For the natural selection (fitness stage), we will use the tf-idf model extracted from HappyDB[2] represents various tfidf maps based on age, accomplishment, gender, marital status, and parenthood. To explain better the model, we defined its maps as follows:

- wholeIdf map (map<string,double>)
- category tf-idf map (map<integer, map<string, double>>, where the integer is the category index)
- age tf-idf map (map<integer, map<string, double>>, where the integer is the age index)
- parenthood tf-idf map (map<integer, map<string, double>>, where the inte-ger is the parenthood index)

- marital tf-idf map (map<integer, map<string, double>>, where the integer is the marital index)
- gender tf-idf map (map<integer, map<string, double>>, where the integer is the gender index)

The integers (indexes) are just mapping (one-hot encoders) to map a label fea-ture to a number for easier storage and computing. The map<string, double> contains words with their BOIT-TFIDF [3] computed weight for various labels sub-datasets extracted from the whole dataset. These weights will be taken into con-sideration when computing the fitness of the chromosome as follows: the fitness score of a certain chromosome (which is represented by 4 preferences axis lists) will be the eight sums for all the words that make those preferences. The process of computing the weight of one word is the following, if is not present in any tf-idf map, then is considered 0, otherwise, it is computed as a ponderate weight from each map (category, age, gender, marital, parenthood) having the following parameters/tunings:

w_Age = 0.04081202332448342d;
w_Gender = 0.1962045208095352;
w_Marital = 0.36419008095450284;
w_Parenthood = 0.3987933749114786;

The category will have the parameter/tunning equal to 1 so our formula becomes:

```
score += 1 * idfCategory.getOrDefault(word, 0.d)+
      w_Age * idfAge.getOrDefault(word,0.d) +
      w_Gender * idfGender.getOrDefault(word, 0.d) +
      w_Marital * idfMarital.getOrDefault(word, 0.d) +
       w_Parenthood*idfParenthood.getOrDefault(word,0.d);          (1)
```

where:

- idfCategory is the tf-idf map word-weight for category label/property
- idfAge is the tf-idf map word-weight for age label/property
- idfGender is the tf-idf map word-weight for gender label/property
- idfMarital is the tf-idf map word-weight for marital label/property
- idfParenthood is the tf-idf map word-weight for the parenthood la-bel/property

The above parameters/tunning are not taken randomly. They were computed using an information gain extracting formula used in ID3/C4.5 decision trees algorithms from [6] and [7]. The parameter for each feature (age, gender, marital, status, parenthood) represents how important is the HappyDB dataset (more precisely how it improves informational gain by reducing entropy in the dataset). For the category label, we are not capable of computing it because it

represented the class and not the label in our dataset, so we are not using a parameter/tunning for it. This way, the psychological model extracted from [1] is enforced in the genetic algorithms with the informational gain from [2] (which defines a more recent weight-word calibration model). The algorithm uses 10 individuals per generation and will iterate for 10 (minimum) - 30 (maximum) generations (based on how fast it will converge). The most fitted individual in the final population will represent the personality preferences of each user.

3. Implementation and experimental results

For the implementation of the hypnosis, we worked on two phases. The first phase was the extraction of inferences and models from [1] and [2]. There were two big models extracted and compiled for our paper:

The personality model from [1], was extracted manually by reading the book and extracting and parsing certain information (the book is very old and NLP techniques were not suitable for extracting the text). The model was object-oriented based, which means it was implemented as a personality library directly in JAVA (the programming language we used to implement the online application). The personality-word-weight model from [2], was extracted using Python, with an extraction algorithm based on [3]. The model was saved in a JSON-like object to be exported and used in our online application.

After having these two models apriori, we opted for a JAVA application client-server using Spring Framework and Bootstrap web UI. The application contains a tomcat container that deploys a web application on port 8080, from which a web interface is used to interact with the application.

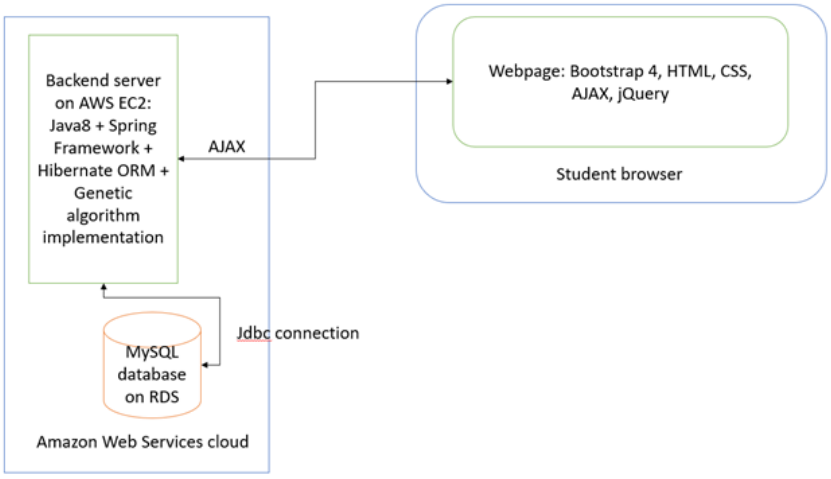

Fig. 2. Application architecture and technologies used in the implementation

We opted for this model because it is the simplest way to extract experimental results for our paper. The JAVA application had the model in JSON and object-oriented language loaded at startup before initializing the Spring context and tomcat server. Because of that, we had some issues with the RAM especially, because the working threshold of our JAVA application was 3GB because of the loaded models directly into memory (meaning that with less RAM the application won't start).

We opted to load everything in RAM and not read it wrong each time the ge-netic algorithm needed information because of the latency that would occur. The genetic algorithm is already CPU-intensive, reading weights and personality pref-erences each time from disk will result in an IO-intensive application. The genet-ic algorithm is custom-made and implemented from scratch by us in JAVA (we didn't opt for any Machine Learning library in JAVA like weka). After that, we deployed the application on the AWS cloud on t2.medium instances with gp2 disks. The t2.medium instance is not a FREE instance on AWS (not on the free tier) and has a 3.3 GHz Xeon Scalable Processor with 2 cores and 4 GB RAM, with 30 GB SSD on EBS disk with gp2 provision. The network performance was low-moderate (as AWS describes in the specifications), meaning that not a lot of traffic is recommended between the front end and back end of the application. The application used static IP (EIP- Elastic IP) on AWS, without any DNS or load-balancers (to reduce the costs of the deployment).

We tested the application on 25 students (24-25 years old) in, the fourth-year Faculty of Computer Science, University of Politehnica Bucharest. The student's input was anonymous (because this is regarded as individual information) and was deleted after the experiment (only the feedback was preserved). After the student filled in the percentages, the personality preferences were displayed. After that, the feedback form was displayed to take the opinion of the student on how well the algorithm matched their personality. The feedback questions that were on the form (they had to select 1 to 5) :

- How did the preferences match with your persona?
- How well did the introvert-extrovert preferences match with you?
- How well did the sensitive-intuitive preferences match with you?
- How well did the perceiver-judger preferences match with you?
- How well did the thinker-feeler preferences match with you?
- What do you think about the other descriptions(temperament, fashion style, recommended jobs, etc...)?
- Would you recommend this test to other people?

Only after submitting the feedback, the results were saved into the database (and by results, we defined the feedback, personality type, and temperament):

Table 1

Feedback form results

Id	Personality	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Tempera-ment	Average score
1	INFP	4	4	3	3	3	3	3	NF	3.28
2	ENTJ	2	4	4	4	2	3	4	NT	3.28
3	INTP	3	5	5	3	4	4	5	NT	4
4	INTJ	5	5	5	5	5	5	5	NT	5
5	ESTP	4	4	4	3	4	4	4	SP	3.85
6	ENTJ	4	5	5	5	5	4	5	NT	4.71
7	INFP	4	4	4	4	4	4	5	NF	4.14
8	INTJ	4	3	4	2	3	2	4	NT	3.14
9	ENTJ	4	5	3	3	5	4	3	NT	3.28
10	ESTP	5	5	5	5	5	5	5	SP	5
11	INTJ	5	3	5	2	4	4	2	NT	3
12	INTJ	4	5	5	3	3	4	4	NT	4
13	ISTP	4	5	4	4	4	3	4	SP	4
14	ENTJ	5	3	4	5	5	4	5	NT	4.42
15	ESTP	3	4	5	5	4	4	4	SP	4.14
16	ISTJ	4	3	4	3	3	3	4	SJ	3.42
17	INTJ	5	5	4	4	4	5	3	NT	3.85
18	ENFP	3	4	4	4	4	4	5	NF	4
19	INTJ	5	4	5	5	5	3	5	NT	4.57
20	ENTJ	3	3	3	4	5	4	5	NT	3.85
21	ESTP	5	5	5	5	5	5	5	SP	5
22	INTJ	3	2	2	3	2	3	3	NT	2.57
23	ISFJ	4	4	4	5	3	3	3	SJ	3.57
24	ESTP	4	4	2	3	4	4	4	SP	3.57
25	INTJ	4	3	3	2	4	3	2	NT	3

Table 2

Feedback from average, minimum, and maximum scores for each question

	Q1	Q2	Q3	Q4	Q5	Q6	Q7
AVG	4.04	4.04	4.04	3.76	3.96	3.76	4.04
MIN	2	2	2	2	2	2	2
MAX	5	5	5	5	5	5	5

The gender, marital status, parenthood, and age were not saved into the dataset because it represents personal information from the users (even if we had their agreement to process it and store it).

As we can see from the results, we had a medium score for all questions of 3.76-4.04 (from 1-5) meaning that the proposed algorithm matched 75% - 81% of the personality preferences model for our students. The experimental results are based on real humans, NOT created artificially (that is why we have so few because people often do not want to participate in psychology experiments that reveal details about their personality, behavior, and selves). We plot the distribution of personality and temperaments from our feedback data:

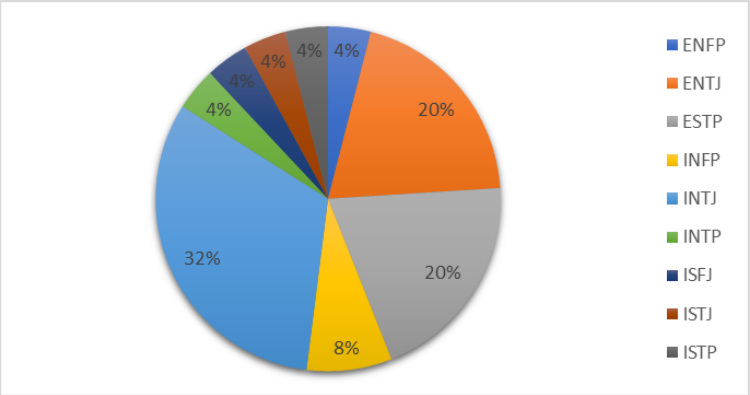

Fig. 3. Distribution of personality

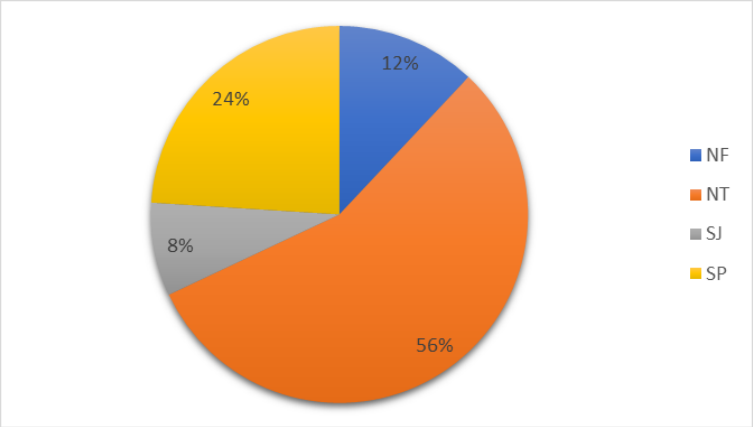

Fig. 4. Distribution of temperaments

There are 4 types of temperaments described in the user model. They are NT (intuitive-thinker), NF (Intuitive-Feeler), SP(sensitive-perceiver), and SJ (Sensi-tive-Perceiver). We see from the figures above that the most dominant personali-ty types are introverts, thinkers, and intuitive types (which match the description of job personality). For the temperaments, the most predominant by far is the NT (Intuitive-Thinker), which is a good match for our type of faculty (Computer Sci-ence). For the personality type, the distribution is on more intuitive personality types: ENTJ (Extrovert-Intuitive-Thinker-Judger), ESTP (Extrovert-

Sensitive-Thinker-Periceiver), INFP (Introvert-Intuitive-Feeler-Perceiver), INTJ (Introvert-Intuitive-Thinker-Judger), INTP (Introvert-Intuitive-Thinker-Perceiver), ISFJ (Introvert-Sensitive-Feeler-Judger) than on sensitive personality types: ESTP (Extrovert-Sensitive-Thinker-Perceiver), ENTFP (Extrovert-Intuitive-Feeler-Perceiver). We extracted an average score of our question answers results in the below figure:

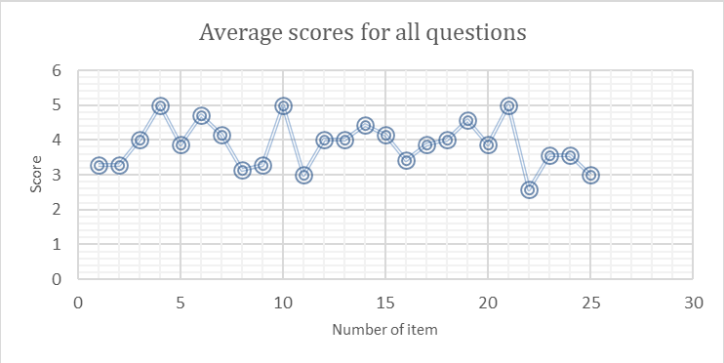

Fig. 5. Average scores for all questions

Based on the above graphic and table 2, the average score is between 2(minimum) and 5(maximum), with an average of 3.86 (77%) for all questions asked in the feedback form. Based on the above graphics, the general matching personality model has a maximum score of 5 and a lower score of 2, having an average of 4 (80%). For this question, the student had to say from 1 to 5 how well the algorithm matches his/her personality characteristics. This is by far the most important question in the questionnaire. We did a similar experiment of the personality model in [8], but the overall matching score for question 1 (how well the personality model matches the persona) was 75%. We see an increase of 5% from switching the inference datasets from [9] to [2] for our inference model. For the introvert-extrovert preferences axis (question 2), we have a minimum of 3 and a maximum of 5, with an average of 4.04 (81%). For the sensor-intuitive preferences axis (question 3), we have a minimum of 2, a maximum of 5, and an average of 4.04 (81%). For the thinker-feeler preferences axis (question 4), we have a minimum of 2, a maximum of 5, and an average of 3.76 (75%). For the perceiver-judger preferences axis (question 5), we have a minimum of 2, a maximum of 5, and an average of 3.96 (79%). All these results on the 4 axes of per-sonalities represent a good accuracy of the model proposed. For the temperament, fashion style, and other personality trivia (question 6), we have a minimum of 2 and a maximum of 5 scores, with an average of 3.76 (75%). This trivia was offered directly from the personality model from [1] without any machine learning computation. We see that the matching score is way better (80% - question 1) when using machine learning techniques to better match the personality models. For question 6, we have a minimum of 2, a maximum of 5,

and an average of 4.04. This means that 81% of the students who experimented would recommend it to another person or friend.

4. Conclusions

In our paper, we describe a method to determine personality based on a genetic algorithm and behavioral psychology with an 80% overall matching (based on experimental results). The algorithm used a behavioral psychological model based on each personality type and its preferences in different circumstances and it was reinforced using a weight word model extracted using TFIDF-BOIT from the HappyDB dataset and used in the fitness function of the genetic algorithm alongside the parameters/tunning extracted using information gain to better ponder the fitness score. The results are promising, improving the matching score from 75% from the experiment [8] where we used an MBTI dataset [9] for model enforcement to 80% using the HappyDB dataset [2] and informational gain ex-traction to reduce entropy in our dataset [6][7].

R E F E R E N C E S

[1]. *Otto Kroeger and Janet M. Thuesen*, "The 16 personality types that determinate how we live, love and work", Tilden Press Book, 1988

[2]. HappDB dataset, https://www.kaggle.com/code/ydalat/happydb-what-100-000-happy-moments-are-telling-us (downloaded 10.10.2022)

[3]. *Long Cheng, Yang Yang, Kang Zhao and Zhipeng Gao*, Research and Improvement of TF-IDF Algorithm Based on Information Theory, Advances in Intelligent Systems and Computing, AISC, volume 905, 2019

[4]. *H. Muhlenbein, M. Gorges-Schuler, and O. Krpji*, MER, Evolution algorithms in com-binatorial optimization, 1987

[5]. *Thomas Bäck,, Hans-Paul Schwefel*, An Overview of Evolutionary Algorithms for Pa-rameter Optimization, , 1993, MIT press

[6]. *Yan-yan Song and Ying Lu*, Decision tree methods: applications for classification and prediction

[7]. *Colin G. DeYoung and Yanna J. Weisberg*, Information Gain Directed Genetic Algo-rithm Wrapper Feature selection for Credit Rating, 2018

[8]. *Nutescu Ciprian, Mariana Mocanu*, Creating personality model using genetic algorithms and behavioral psychology, ECAI 2023 (accepted)

[9]. MBTI dataset, https://www.kaggle.com/datasets/zeyadkhalid/mbti-personality-types-500-dataset (downloaded 10.10.2022)

[10]. *Adrian Furnham, Joanna Moutafi & Laurence Paltiel*, Intelligence in Relation to Jung's Personality Types, University of London 2005

[11]. *Fumiyo Hunter and Nissim Levy*, Relationship of Problem-Solving Behaviors and Jung-ian Personality Types, 2016

[12]. *George H. Rice and David P. Lindecamp*, Personality types and business success of small retailers, 1989