Verification and Validation Report: Natural Language Processing for Mental Health Risk Prediction

Team 13, The Cognitive Care Crew
Jessica Dawson
Michael Breau
Matthew Curtis
Benjamin Chinnery
Yaruo Tian

April 4, 2024

1 Revision History

Date	Version	Notes
March 6th 2024	1.0	Revision 0
April 4th 2024	1.1	Made changes based on group 8 feedback

2 Symbols, Abbreviations and Acronyms

symbol	description
Precision	the proportion of positive guesses that were correct
Recall	the proportion of positive individuals that were correctly predicted
F-score	combination of precision and recall
R-PREC	represents balancing recall and precision
NDCG	measures the effectiveness of a ranked list in information retrieval
RMSE	Root Mean Square Error
MZOE	Mean Zero-One Error
MAE	Mean Absolute Error
RS	Restraint Subscale
ECS	Eating Concern Subscale
SCS	Shape Concern Subscale
WCS	Weight Concern Subscale
FRT	Function Requirement Test
NFRT	Non-functional Requirement Test

Contents

List of Tables

List of Figures

3 Functional Requirements Evaluation

3.1 Task 1

1. **FRT-T1-1**

   Control: Automatic

   Initial State: No models have been trained by the task 1 system yet, input data as jsonl files are available

   Input: The Task 1 Input data

   Output: The system will extract required sentences from the jsonl files and output data that is appropriate to be fed into Task 1.

   Test Case Derivation: The system will receive its input data, process then parse according to the Task's input format.

   How test will be performed: The tester will run the parser with the given jsonl files as input and compare the parsed data.

   Results: Passed

2. **FRT-T1-2**

   Control: Automatic

   Initial State: No models have been trained by the task 1 system yet, past years input data, training data, and golden truth values are available

   Input: The Task 1 Training Data, Input data, and Golden Truth Values from a past year eRisk Competition

   Output: The system will output sentence ranking and diagnostic metrics based on the input data.

   Test Case Derivation: The system will receive its training data and input data corresponding to a prior year. It's analysis will output RP and NDCG accuracy metrics.

   How test will be performed: The system should derive its training data and feature sets based on correlation to 21 depression symptoms of beck depression index. The outputted metrics will be compared to golden truth data to ensure the accuracy is within the desired bounds and is improving on prior year implementation.

Results: Passed

3. **FRT-T1-3**

   Control: Automatic

   Initial State: No models have been trained by the task 1 system yet, input data is available

   Input: The Task 1 Input data

   Output: The system will output sentence ranking and diagnostic metrics based on the input data to a txt file.

   Test Case Derivation: The system will make an analysis based off the provided input data and return each prediction to a test file in a specified formatting.

   How test will be performed: The system's output txt file will be verified that each entry is on its own line in the format ”{symptom_number}, Q0, {sentence-id}, {position_in_ranking}, {score}, {system_name}”.

   Results: Passed

3.2 Task 2

1. **FRT-T2-1**

   Control: Automatic

   Initial State: No models have been trained by the task 2 system yet, input data is available

   Input: The Task 2 Input data consisting of user posts

   Output: The system will output sentence ranking and diagnostic metrics based on the input data to a txt file.

   Test Case Derivation: The system will make an analysis based off the provided input data and return each prediction to a test file in a specified formatting.

   How test will be performed: The system's output txt file will be verified that each entry is on its own line in the format ”{Username} {prediction}”.

   Results: Passed

2. **FRT-T2-2** Control: Automatic

   Initial State: The task 2 system has received input data on a given individual and made a corresponding prediction.

   Input: More data corresponding to the same user analyzed in the input state.

   Output: The system will output an updated prediction taking into account all data provided.

   Test Case Derivation: The system should create a new prediction for the chosen user taking into account all data provided.

   How test will be performed: The system should provide a prediction based off the original data input, after receiving new data, the system should combine posts made by the same user to create a new prediction.

   Results: Passed

3.3 Task 3

1. **FRT-T3-1**

3.4 Task 3

FRT-T3-1 and FRT-T3-2 ensure the output format of task 3 is correct

(a) **FRT-T3-1**

Output: The system will output predictions to a txt file where each line is of the format ”{Username} {prediction for Q1} ... {prediction for Q28}”

Result: Passed

(b) **FRT-T3-2**

Output: The system will output predictions to a txt file where each prediction should be between 0 and 6

Result: Passed

3.4.1 Overview of Eating Disorder Models

Development Process

Initial exploration into models for this task looked at predicting a score directly from a user's post history. This ran into problems quickly though as there are a lot of posts for each user and not that many users meaning you are trying to train a model with relatively few training examples to predict a score from a large amount of input information, which is unlikely to give good results. Thus development in this direction halted early.

In order to find a new direction to go in a literature review was conducted of the submissions made for this task last year, of which there were three with available papers. This resulted in two general directions to go in.

The first method was to reduce the amount of information the model was taking as input in some way. This could involve finding a way to filter the user's post history to only the most relevant posts, using dedicated dimensionality reduction techniques on the numerical representations of posts, or transforming the representation of posts in some way. A team last year did something similar to how the anorexia task was tackled by our team this year where they represented a user's post history as a distribution of topics and predicted on those instead of the specific embeddings (**?**).

The second method was to create labels for each post instead of each user. This approach has the potential to be quite powerful as if done correctly it can provide a lot more data to train a model on. Last year a team did try this however their approach was very naive, they simple labeled each post with the author's score (**?**). This resulted in some posts that had nothing to do with eating disorders having a very alarming questionnaire attached to them. This area had a lot of space for improvement so it was the direction that was gone in for task 3.

After discussing with Diego of our supervision team he suggested adapting a method used by a team on a different task in a prior year. This method would relabel a post as positive or negative based on how many posts by positive and negative authors were near it in the space represented by the numerical representations of posts (**?**). This approach could not be directly applied to this task though as users have 0 to 6 scores. A modification of the approach was developed by team member Jessica using Bayes Theorem that could produce a probability for how likely it was that a post was related to one of the EDE-Q questions and convert that probability into a positive or negative score. The details of the approach are in appendix A (**??**), this appendix is suggested reading as it is cool and one of the team's members is quite proud of it. This approach was validated with the supervision team during a meeting and was thusly developed.

The next part of this approach involves training a model to predict a score for a post from it's text. After consulting with Diego he suggested building a model on top of the MentalRoBERTa pre-trained model. This is a model that has been trained on prior mental health information that can be adapted to this task and as such has the potential to pick out patterns and details a blank model might otherwise miss.

The final stage of the prediction pipeline involves aggregating from these post level scores to a user level score. This was done by simply plotting the proportion of positive posts in a user's post history against their known answers to the questionnaire for a set of train users, fitting a line to this data, and using said line to aggregate on test users.

This was the final model that was developed, however an secondary model was also developed largely through coincidence. This used the relabeling technique to make predictions on posts directly, without training the MentalRoBERTa model and then aggregated to user scores from there. This technique performed okay but deploying it would also require sharing user's post history which presented privacy concerns, as such this approach was not considered as viable. This approach is still included in the table of accuracy results for completeness's sake.

Generalizability

An important quality for these sorts of models to have is the ability to use them outside they data they were developed on. In terms of this approach building on top of MentalRoBERTa creates a model that can be shared without concerns over privacy. This type of pre-trained model also tends to be quite powerful and good at working outside it's training data. Some limitations to this are of course the question of how representative of the general population the eRisk data is, does it contain any biases, as these would impact the generalizability of the developed model. This is not a question I am capable of answering though as I didn't collect the data. One area in this pipeline that has the potential to be quite weak to these effects is the aggregation method. It's very simple currently and if it turns out the individuals in our data talk about eating disorders more or less often than the general population this can result in the aggregated user level scores being wrong. However even if this is the case the MentalRoBERTa model will still be valuable.

The following are accuracy metrics for the explored models.

	MAE	MZOE	RS	ECS	SCS	WCS	GED
zeros	2.85	**0.73**	3.17	2.80	3.42	3.02	3.03
sixes	3.15	0.76	3.86	4.23	3.74	3.85	3.86
average	2.10	0.94	1.86	1.79	1.96	1.68	1.68
secondary model	1.80	0.80	1.86	1.43	1.58	1.51	1.40
final model	**1.67**	0.75	**1.59**	**1.41**	**1.48**	**1.16**	**1.16**

Table 1: Accuracy results for task 3 models

(a) **FRT-T3-3**

Output: The system will print accuracy metrics for the model to the console where the model scores higher than the baseline zero strategy on MAE or ties MAE and improves on at least one other metric

Result: **Passed**

(b) **FRT-T3-4**

Output: The system will print accuracy metrics for the model to the console where the model scores higher than the baseline six strategy on MAE or ties MAE and improves on at least one other metric

Result: **Passed**

(c) **FRT-T3-5**

Output: The system will print accuracy metrics for the model to the console where the model scores higher than the baseline average strategy on MAE or ties MAE and improves on at least one other metric

Result: **Passed**

4 Nonfunctional Requirements Evaluation

4.1 Usability Requirements

(a) **NFRT-U-1**

Type: Dynamic, Manual

Initial State: System is completed and trained

Input/Condition: The system will be ran on a Windows desktop/laptop and macOS desktop/laptop device with the correct environment setup

Output/Result: The system should generate an expected result

How test will be performed: After setting up the environment on both macOS and Windows machines, the system will be run on both environments with the same command. Test will pass as long as an expected result is generated from both machines.

Results: Passed

4.2 Safety and Security Requirements

(a) **NFRT-SS-1**

Type: Automatic, Dynamic

Initial State: The system has been trained and is awaiting data to form predictions on.

Input/Condition: A set of data that the system can predict on.

Output/Result: The resulting predictions, in a form where no sensitive data from the input is present in the output. Test Case Derivation: Sensitive in this case refers to the post history of the subjects eRisk provides as training and test data. If these posts can be reconstructed from any part of our system outputs it is possible the identity of the individual could be discovered. This represents an unacceptable breach of privacy and can not happen.

How test will be performed: After the system has produced it's predictions a script will be run that takes all posts in the input data, forms a string out of all consecutive three word triplets (ie. "hello my good friend" forms "hello my good" and "my good friend"), both with and without processing (removal of stopwords, punctuation, etc.), and scans the output to ensure that none of these triples are present.

Results: Passed

4.3 Legal Requirements

(a) **NFRT-L-1**

Type: Static, Manual

Initial State: The source code and documentation is prepared

Input/Condition: The user reviews the entirety of the source code and related documentation

Output/Result: Copyright licenses, appropriate credits and/or MIT license must attached

How test will be performed: The testers will review the entirely of the project and check to see if appropriate copyright licenses and/or credits are included for resources that require them

Results: Passed

5 Comparison to Existing Implementation

A large portion of our validation is deploying our models for the eRisk 2024 competition to determine how well they perform compared to models from other participants. As the current competition is not over, the metrics to compare against were derived from the performances of teams from previous year eRisk competitions.

5.1 Task 1

Metrics were compared against eRisk's 2022 Task 1: Search for symptoms of depression

5.1.1 Our Model

- R-PREC: 0.219
- NDCG: 0.683

5.1.2 Best Performing Model from Previous Years

- R-PREC: 0.375
- NDCG: 0.596

5.1.3 Discussion

In comparison to the accuracy metric of the 2022's best performing model for this task, which were 0.375 for R-PREC and 0.596 for NDCG. We outperformed it in terms of the NDCG metric with a value of 0.683 but lacking in R-PREC with a value of 0.219. The greater NDCG, which shows better performance in rating relevant phrases despite the lower R-PREC, highlights how well the system prioritizes the quality of relevant sentence placement. The best metric to use will rely on the particular objectives of the NLP work connected to depression, taking into account the trade-off between recall and precision. Although the recall precision score from the prior year was greater, indicating a better balance, our system performs very well in NDCG, emphasizing the importance of quality ranking for recognizing important sentences.

5.2 Task 2

Metrics were compared against eRisk's 2021 Task 1: Early Detection of Pathological Gambling

5.2.1 Our Model

- Precision: 0.627
- Recall: 0.644
- F-score: 0.635

5.2.2 Best Performing Model from Previous Years

- Precision: 0.586
- Recall: 0.939
- F-score: 0.721

5.2.3 Discussion

Overall F-score of our model was slightly lower than the best performing model from the previous year, our model is lacking heavily compared to other models in the recall section but has the best overall precision of any model and would place second out of the six teams during that eRisk competition. It is important to note that this was early risk prediction for gambling addiction whereas ours is for anorexia and therefore cannot be directly compared and should just be used as overall reference for the broad category of positive negative classification.

5.3 Task 3

A comparison between our model and the best metric scores from last year's iteration of the task are compared. The best scores are the highest accuracy achieved on that metric across all teams, not the metrics from a "best" model.

	MAE	MZOE	RS	ECS	SCS	WCS	GED
final model	**1.67**	0.75	**1.59**	**1.41**	**1.48**	**1.16**	**1.16**
2023's best	2.19	**0.67**	**1.59**	1.92	1.90	2.00	2.00

5.3.1 Discussion

Our final model outperforms the best accuracy metrics from last year on all metrics except RS which we tie, and MZOE which was achieved by a baseline last year. Given that we also fail to outperform the baselines on MZOE this year this is not much of a loss.

6 Unit Testing

Due to the nature of our project being almost entirely output focused. Our tests do not directly test the individual modules as the code is verified directly by the fact that the metrics of the output are directly showing that the model is performing as intended. Any error in individual units would produce incorrect output or error and as a result was deemed to not be necessary.

7 Changes Due to Testing

The supervisors for this project as individuals who had participated in previous competitions were instrumental in the understanding of the project and bi-weekly update meetings led to iterative improvements to the methodology of the choices in our models as well as the performance in regards to metrics.

8 Automated Testing

Automated testing for each individual task was done using pytest by analyzing the output and checking for correct output and checking that the models are running as intended. To do this the test module calculates the metrics of the results and checks if the resulting metrics are above a certain threshold where if it was below this threshold, something is most likely going wrong with the model. We also selected pylint to be our linter which we implemented using github actions.

9 Trace to Requirements

9.1 Traceability Between Test Cases and Requirements

Requirement traceability from S2 of SRS to testing in this document.

Table 2: Traceability Between Functional Test Cases and Functional Requirements, T1FR-1 to T3FR-3

Test IDs	**Functional Requirement IDs**									
	T1FR-1	**T1FR-2**	**T1FR-3**	**T2FR-1**	**T2FR-2**	**T2FR-3**	**T2FR-4**	**T3FR-1**	**T3FR-2**	**T3FR-3**
FRT-INP-1	X			X				X		
FRT-INP-2	X			X				X		
FRT-T1-1		X								
FRT-T1-2			X							
FRT-T2-1					X		X			
FRT-T2-2						X				
FRT-T3-1									X	X
FRT-T3-2									X	X
FRT-T3-3										
FRT-T3-4										
FRT-T3-5										

Table 3: Traceability Between Functional Test Cases and Functional Requirements, T3FR-4 to T3FR-6

Test IDs	**Functional Requirement IDs**		
	T3FR-4	**T3FR-5**	**T3FR-6**
FRT-INP-1			
FRT-INP-2			
FRT-T1-1			
FRT-T1-2			
FRT-T1-3			
FRT-T2-1			
FRT-T2-2			
FRT-T2-1			
FRT-T3-2			
FRT-T3-3	X		
FRT-T3-4		X	
FRT-T3-5			X

Table 4: Traceability Between Non-Functional Test Cases and Non-Functional Requirements

Test IDs	**Non-Functional Requirement IDs**

	GR1	GR2	SR1	SR2
NFRT-U-1	X			
NFRT-SS-1			X	
NFRT-SS-2				X
NFRT-L-1		X		

10 Traceability of Test Cases and Modules

Test IDs	Module IDs								
	M2	**M3**	**EDM1**	**EDM2**	**EDM3**	**EDM4**	**EDM5**	**EDM6**	**EDM7**
FRT-INP-1	X	X	X	X					
FRT-INP-2	X	X	X	X					
FRT-T1-1	X								
FRT-T1-2	X								
FRT-T1-3	X								
FRT-T2-1		X							
FRT-T2-2		X							
FRT-T3-1			X	X					
FRT-T3-2			X	X					
FRT-T3-3					X	X	X	X	X
FRT-T3-4					X	X	X	X	X
FRT-T3-5					X	X	X	X	X

11 Code Coverage Metrics

Due to the nature of our project being almost entirely output focused. Our tests do not directly test the code as the code is verified directly by the fact that the metrics of the output are directly showing that the model is performing as intended.

Appendix A — Relabeling Method

Note: this is the calculation of a score for one post and one question, assume that we've already picked a post and question to calculate for

$$P(Pos) = probability\ that\ the\ post\ is\ positive$$

$$k = number\ of\ neighbors\ queried$$

$$D = [S_1, S_2, ..., S_k]$$

$$Where:\ S_n = score\ of\ neighbor\ n\ of\ the\ post$$

$$P(D) = probability\ of\ specific\ set\ of\ scores$$

$$P(Pos|D) = \frac{P(D|Pos)P(Pos)}{P(D)}$$

The marginal $P(D)$ is easily calculable from the probabilities of each neighbor:

$$P(D) = P(S_1 = s_1) * P(S_2 = s_2) * ... * P(S_k = s_k)$$

Which in turn are just:

$$P(S_n = s_n) = \frac{number\ of\ posts\ with\ score\ s_n}{total\ number\ of\ posts}$$

The prior $P(Pos)$ becomes a hyper parameter for the question.

$P(D|Pos)$ can be expanded similarly to $P(D)$:

$$P(D|Pos) = P(S_1 = s_1|Pos) * P(S_2 = s_2|Pos) * ... * P(S_k = s_k|Pos)$$

Where $P(S_n = s_n|Pos)$ is defined by a discrete probability distribution over the points $s_n \in 0, 1, 2, 3, 4, 5, 6$ where the parameters of the distribution become hyper parameters for the question.

Thus for each question you have the hyper parameters: k, the prior, and any parameters of the distribution used to define $P(S_n = s_n|Pos)$

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Reflection. Please answer the following question:

(a) In what ways was the Verification and Validation (VnV) Plan different from the activities that were actually conducted for VnV? If there were differences, what changes required the modification in the plan? Why did these changes occur? Would you be able to anticipate these changes in future projects? If there weren't any differences, how was your team able to clearly predict a feasible amount of effort and the right tasks needed to build the evidence that demonstrates the required quality? (It is expected that most teams will have had to deviate from their original VnV Plan.)

When the team originally created the VnVPlan, our understanding of our project and the eRisk competition requirements were much more lacking at the time and as a result we have had to change a lot of our requirements as well as the tests for those requirements to fit our project. Originally our understanding of the fundamental structure of the NLP pipeline lacked depth and as such our tests did not make much sense after we completed the code for the three tasks. In future projects we can anticipate that we currently do not know the full extent of what we are undertaking so being more clear about what we do not know in our VnV plan would lead to less need for large amounts of changes in test structure and content.