Module Guide for 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**
January 17, 2024	1.0	Revision 0

2 Reference Material

This section records information for easy reference.

2.1 Abbreviations and Acronyms

symbol	**description**
AC	Anticipated Change
ED	Eating Disorder
DAG	Directed Acyclic Graph
M	Module
MG	Module Guide
OS	Operating System
R	Requirement
SC	Scientific Computing
SRS	Software Requirements Specification
UC	Unlikely Change

Contents

List of Tables

List of Figures

3 Introduction

Decomposing a system into modules is a commonly accepted approach to developing software. A module is a work assignment for a programmer or programming team (**?**). We advocate a decomposition based on the principle of information hiding (**?**). This principle supports design for change, because the "secrets" that each module hides represent likely future changes. Design for change is valuable in SC, where modifications are frequent, especially during initial development as the solution space is explored.

Our design follows the rules layed out by **?**, as follows:

- System details that are likely to change independently should be the secrets of separate modules.
- Each data structure is implemented in only one module.
- Any other program that requires information stored in a module's data structures must obtain it by calling access programs belonging to that module.

After completing the first stage of the design, the Software Requirements Specification (SRS), the Module Guide (MG) is developed (**?**). The MG specifies the modular structure of the system and is intended to allow both designers and maintainers to easily identify the parts of the software. The potential readers of this document are as follows:

- New project members: This document can be a guide for a new project member to easily understand the overall structure and quickly find the relevant modules they are searching for.
- Maintainers: The hierarchical structure of the module guide improves the maintainers' understanding when they need to make changes to the system. It is important for a maintainer to update the relevant sections of the document after changes have been made.
- Designers: Once the module guide has been written, it can be used to check for consistency, feasibility, and flexibility. Designers can verify the system in various ways, such as consistency among modules, feasibility of the decomposition, and flexibility of the design.

The rest of the document is organized as follows. Section **??** lists the anticipated and unlikely changes of the software requirements. Section **??** summarizes the module decomposition that was constructed according to the likely changes. Section **??** specifies the connections between the software requirements and the modules. Section **??** gives a detailed description of the modules. Section **??** includes two traceability matrices. One checks the completeness of the design against the requirements provided in the SRS. The other shows the relation between anticipated changes and the modules. Section **??** describes the use relation between modules.

4 General Structure of an NLP Pipeline

All NLP pipelines in this system can be thought of as following a general structure. This is not a hard description of the structure each diagnosis component will use but a soft description of what the system's NLP pipelines will look like. This pattern is likely to show up throughout the system in various locations and is included here to provide context and a conceptual model the reader can use for the rest of the document. The proposed structure is as follows:

Input → Tokenizer → Representation Model → Prediction Model → Predictions

Where the Prediciton Model can be further broken up into:

→ Transformation → ... → Transformation →

A brief description of each stage is as follows:

- Input: The input text to the pipeline.
- Tokenizer: Responsible for "cleaning" the text, can perform many different tasks: removing junk words (often referred to as stopwords) like the, and, a, etc.; expanding contractions; removing numerics and URLS. Leaves the text as text, just with the less useful features removed.

- Representation Model: Machine learning models can not directly operate on text, they need numerical inputs. The representation model converts from the text to some numerical format, typically a vector. Different representation models output different formats and encode different information into these formats.
- Prediction Model: Transforms the numerical format provided by the representation model into the mental health predictions of interest. The prediction model may transform the data mutliple times in different ways before reaching a final prediction.
  - Transformation: A specific transformation in the prediction model's pipeline
- Predictions: The final predictions produced by the pipeline.

Due to the interconnected nature of this pipeline and the research based nature of this project large changes to the system's models are both expected and unavoidable. This is because changes in one section of pipeline can require large changes in other sections, trying a new prediction model may require an entirely different representation model to be used for example.

5 Anticipated and Unlikely Changes

This section lists possible changes to the system. According to the likeliness of the change, the possible changes are classified into two categories. Anticipated changes are listed in Section **??**, and unlikely changes are listed in Section **??**.

5.1 Anticipated Changes

Anticipated changes are the source of the information that is to be hidden inside the modules. Ideally, changing one of the anticipated changes will only require changing the one module that hides the associated decision. The approach adapted here is called design for change.

AC1: What format the input data takes.

AC2: What format the output data takes.

AC3: In what format the data is stored during runtime. Different layouts can change how quickly certain information can be accessed and effect performance.

AC4: What stages there are in a task's NLP pipeline.

AC5: How the raw text data is "cleaned" (see **??**: tokenizer).

AC6: How the cleaned text data is converted into numerical data.

AC7: How many manipulations of the numerical data are performed before producing a prediction.

AC8: How the numerical data is manipulated to produce predictions.

AC9: What the optimal parameters for a pipeline will be.

AC10: How the accuracy of a pipeline will be evaluated.

AC11: How the system will integrate with the tools the operations team is developing.

5.2 Unlikely Changes

The module design should be as general as possible. However, a general system is more complex. Sometimes this complexity is not necessary. Fixing some design decisions at the system architecture stage can simplify the software design. If these decision should later need to be changed, then many parts of the design will potentially need to be modified. Hence, it is not intended that these decisions will be changed.

UC1: Input/Output devices (Input: File, Output: File).

UC2: What diagnosis tools the system will provide.

6 Module Hierarchy

This section provides an overview of the module design. Modules are summarized in a hierarchy decomposed by secrets in Table **??**. The modules listed below, which are leaves in the hierarchy tree, are the modules that will actually be implemented:

M1: Hardware-Hiding Module

M2: Depression Main Module

M3: Depression Data Processing Module

M4: Depression Feature Extraction Module

M5: Depression Output Module

M6: Depression Evaluation Module

M7: Anorexia Main Module

M8: Anorexia Parser Module

M9: Anorexia Training Module

M10: Anorexia Predictor Module

EDM1: ED Parsing

EDM2: ED Cleaning

EDM3: ED Output

EDM4: ED Pipeline Manager

EDM5: ED Trainer

EDM6: ED Data Storage

EDM7: ED Representation Model

EDM8: ED Relabeler

EDM9: ED Prediction Model

EDM10: ED Aggregator

EDM11: ED Metrics

EDM12: ED Evaluator

EDM13: ED Visualization

Level 1	**Level 2**	**Level 3**
Hardware-Hiding	None	
3*Behaviour-Hiding		
	2*Eating Disorder IO	ED Input
		ED Output
16*Software Decision	5*Depression Pipeline	Depression Main
		Depression Data Processing
		Depression Feature Extraction
		Depression Output
		Depression Evaluation
	4*Anorexia Pipeline	Anorexia Main
		Anorexia Parser
		Anorexia Training
		Anorexia Predictor
	7*Eating Disorder Pipeline	ED Pipeline Manager
		ED Trainer
		ED Data Storage
		ED Representation Model
		ED Relabeler
		ED Prediction Model
		ED Aggregator
		ED Evaluation

Table 1: Module Hierarchy

Level 3	**Level 4**
2*ED Input	ED Parsing
	ED Cleaning
ED Output	
Depression Main	
Depression Data Processing	
Depression Feature Extraction	
Depression Output	
Depression Evaluation	
Anorexia Main	
Anorexia Parser	
Anorexia Training	
Anorexia Predictor	
ED Pipeline Manager	
ED Trainer	
ED Data Storage	
ED Representation Model	
ED Relabeler	
ED Prediction Model	
ED Aggregator	
3*ED Evaluation	ED Metrics
	ED Evaluator
	ED Visualization

Table 2: Module Hierarchy cont

7 Connection Between Requirements and Design

The design of the system is intended to satisfy the requirements developed in the SRS. In this stage, the system is decomposed into modules. The connection between requirements and modules is listed in Table **??**.

8 Module Decomposition

Modules are decomposed according to the principle of "information hiding" proposed by **?**. The *Secrets* field in a module decomposition is a brief statement of the design decision hidden by the module. The *Services* field specifies *what* the module will do without documenting *how* to do it. For each module, a suggestion for the implementing software is given under the *Implemented By* title. If the entry is *OS*, this means that the module is provided by the operating system or by standard programming language libraries.

Only the leaf modules in the hierarchy have to be implemented. If a dash (–) is shown, this means that the module is not a leaf and will not have to be implemented.

8.1 Hardware Hiding Modules (M??)

Secrets: The data structure and algorithm used to implement the virtual hardware.

Services: Serves as a virtual hardware used by the rest of the system. This module provides the interface between the hardware and the software. So, the system can use it to display outputs or to accept inputs.

Implemented By: OS

8.2 Behaviour-Hiding Module

Secrets: The contents of the required behaviours.

Services: Includes programs that provide externally visible behaviour of the system as specified in the software requirements specification (SRS) documents. This module serves as a communication layer between the hardware-hiding module and the software decision module. The programs in this module will need to change if there are changes in the SRS.

Implemented By: Depression IO, Anorexia IO, Eating Disorder IO

8.2.1 Eating Disorder IO (-)

Secrets: How input and output is handled for the eating disorder diagnosis tool.

Services: Includes programs that provide reading from an input file and outputting to an output file for the eating disorder diagnosis tool.

Implemented By: ED Input, ED Output

8.2.1.1 ED Input (-)

Secrets: What format the input file is provided in.

Services: Provides functionality for reading user posts from a file and formatting them into a data structure usable by the rest of the system.

Implemented By: ED Parsing, ED Cleaning

8.2.1.2 ED Parsing (EDM??)

Secrets: What format the input file is provided in.

Services: Provides functionality for reading user posts from a file.

Implemented By: parsing code developed for this project

Type of Module: Library

8.2.1.3 ED Cleaning (EDM??)

Secrets: What the initial fields and structure of the data is.

Services: Cleans the data into a form ready to be used by the rest of the pipeline..

Implemented By: cleaning code developed for this project

Type of Module: Library

8.2.1.4 ED Output (EDM??)

Secrets: What format the output file is written in.

Services: Provides functionality for writing predictions to a file.

Implemented By: ED output code

Type of Module: Library

8.3 Software Decision Module

8.3.1 Depression Main Module (M??)

Secrets: The Natural Language processor which will combine the depression related functions such as process data, extract features, generating output, and evaluation to compute symptoms of depression in new documents.

Services: Depression training data do not provide direct interaction with the user.

Implemented By: task1_main

8.3.2 Depression Data Processing Module (M??)

Secrets: The parser for taking user posts and converting them into a format readable by the nlp model.

Services: Depresson training data do not provide direct interaction with the user.

Implemented By: task1_DataProcessing

8.3.3 Depression Feature Extraction Module (M??)

Secrets: The Natural Language processor which will analyze, compute with NLP techniques and extract the necessary features on parsed data.

Services: Depression training data do not provide direct interaction with the user.

Implemented By: task1_FeatureExtraction

8.3.4 Depression Output Module (M??)

Secrets: The Natural Language processor which will take the extracted features and generate sentences that shows symptoms of depression.

Services: Depression training data do not provide direct interaction with the user.

Implemented By: task1_Output

8.3.5 Depression Evaluation Module (M??)

Secrets: The Natural Language processor which will evaluate the output generated by measuring it against various metrics.

Services: Depression training data do not provide direct interaction with the user.

Implemented By: task1_Evaluation

8.3.6 Anorexia Main Module (M??)

Secrets: The Natural Language processor which will combine the parser, training, and predictor modules to compute anorexia risk in new documents.

Services: Anorexia training data do not provide direct interaction with the user.

Implemented By: task2_main

8.3.7 Anorexia Parser Module (M??)

Secrets: The parser for taking user posts and converting them into a format readable by the nlp model.

Services: Anorexia training data do not provide direct interaction with the user.

Implemented By: task2_parser

8.3.8 Anorexia Training Module (M??)

Secrets: The Natural Language processor which will train the model on parsed data.

Services: Anorexia training data do not provide direct interaction with the user.

Implemented By: task2_training

8.3.9 Anorexia Predictor Module (M??)

Secrets: The Natural Language processor which will analyze inputted user data to determine a risk for anorexia

Services: Anorexia training data do not provide direct interaction with the user.

Implemented By: task2_predictor

8.3.10 Eating Disorder Pipeline (-)

Secrets: The structure and design of the eating disorder NLP pipeline.

Services: Includes programs that manage the overall pipeline and others that define the specific transformations used to produce a prediction. More details on the structure of the pipeline and how modules fit together can be found in section **??**.

Implemented By: ED Pipeline Manager, ED Representation Model, ED Relabeler, ED Aggregator ED Prediction Model, ED Trainer, ED Evaluator

8.3.10.1 ED Pipeline Manager (EDM??)

Secrets: What components the NLP pipeline for the eating disorder task is comprised of.

Services: Orchestrates and runs all the components of the eating disorder NLP pipeline when used for predicting a score.

Implemented By: pipeline manager code developed for the project

Type of Module: Library

8.3.10.2 ED Trainer (EDM??)

Secrets: How the pipeline is optimized and tuned.

Services: Orchestrates the tuning of models and model parameters for the pipeline.

Implemented By: trainer code developed for the project

Type of Module: Library

8.3.10.3 ED Data Storage (EDM??)

Secrets: What format the data is stored in during runtime.

Services: Abstracts away the underlying data format and provides an interface for accessing features of the data.

Implemented By: data storage code developed for the project

Type of Module: ADT

8.3.10.4 ED Representation Model (EDM??)

Secrets: How text data is converted into a numerical format.

Services: Provides a routine for converting text data into a numerical format.

Implemented By: representation model code developed for the project

Type of Module: Library

8.3.10.5 ED Relabeler (EDM??)

Secrets: How scores for each user in the ED data are converted into scores for each post.

Services: Provides a tool for creating scores for each post in the ED data based on the scores for each user. For more details on how this relabeling is used see section **??**

Implemented By: relabeling code developed for the project

Type of Module: ADT

8.3.10.6 ED Prediction Model (EDM??)

Secrets: How the numerical representation of text from the Representation Model is converted to predictions.

Services: Orchestrates and runs all the transformations that make up the prediction model.

Implemented By: prediction model code developed for the project

Type of Module: ADT

8.3.10.7 ED Aggregator (EDM??)

Secrets: How scores for each post in a user's post history are converted to a single score for the user.

Services: Provides a tool for creating a single score for a user from scores for each of their posts.

Implemented By: aggregation code developed for the project

Type of Module: ADT

8.3.10.8 ED Evaluation (-)

Secrets: How the accuracy of model will be evaluated.

Services: Provides functions and routines for evaluating on specific metrics against certain baselines, and visualizing the data.

Implemented By: ED Metrics, ED Evaluator, ED Visualization

8.3.10.9 ED Metrics (EDM??)

Secrets: What accuracy measures the model will be evaluated on.

Services: Provides functions for evaluating predictions on different metrics.

Implemented By: metrics code written for this project

Type of Module: Library

8.3.10.10 ED Evaluator (EDM??)

Secrets: What baselines a model will be evaluated against.

Services: Provides functions for seeing the performance of baselines as compared to a model.

Implemented By: evaluator code written for this project

Type of Module: ADT

8.3.10.11 ED Visualization (EDM??)

Secrets: What methods there are to visualize the data.

Services: Provides functions for visualizing different aspects of the data through graphs and plots.

Implemented By: visualization code written for this project

Type of Module: Library

9 Traceability Matrix

This section shows two traceability matrices: between the modules and the requirements and between the modules and the anticipated changes.

Req.	**Modules**
T1FR-1	M??, M??
T1FR-2	M??, M??
T1FR-3	M??, M??, M??
T2FR-1	M??, M??
T2FR-2	M??, M??
T2FR-3	M??, M??, M??
T2FR-4	M??, M??
T3FR-1	EDM??, EDM??
T3FR-2	EDM??
T3FR-3	EDM??
T3FR-4	EDM??, EDM??, EDM??, EDM??, EDM??, EDM??, EDM??, EDM??, EDM??
T3FR-5	EDM??, EDM??, EDM??, EDM??, EDM??, EDM??, EDM??, EDM??, EDM??
T3FR-6	EDM??, EDM??, EDM??, EDM??, EDM??, EDM??, EDM??, EDM??, EDM??
GR1	M??, M??, EDM??, EDM??
GR2	M??, M??, EDM??
SR1	M??, M??, EDM??
SR2	M??, M??, EDM??

Table 3: Trace Between Requirements and Modules

AC	Modules
AC??	M??, M??, EDM??
AC??	M??, M??, EDM??
AC??	EDM??
AC??	M??, M??, EDM??
AC??	M??, M??, EDM??, EDM??
AC??	M??, M??, EDM??
AC??	M??, M??, EDM??, EDM??, EDM??
AC??	M??, M??, EDM??, EDM??, EDM??
AC??	M??, M??, EDM??
AC??	M??, M??, EDM??, EDM??, EDM??
AC??	M??, M??, EDM??

Table 4: Trace Between Anticipated Changes and Modules

10 Pipeline Structure and Use Hierarchy Between Modules

10.1 Depression

The pipeline for the depression module has M**??** Depression Main orchestrating the different stages. M**??** Data Processing processes the input data and converts to a usable form. M**??** Feature Extraction then pulls the most important features out of this data and feeds it to M**??** which writes to an output. M**??** provides evaluation metrics so the developer can see the accuracy of the model.

Figure 1: Pipeline for Depression Component

10.2 Anorexia

The anorexia pipeline uses the trainer to train the predictor on data with predictions gotten by the parser. It then makes predictions on data after it is trained

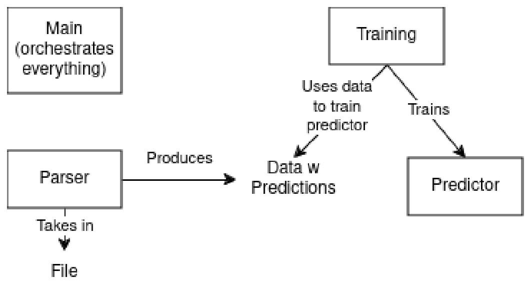

Figure 2: Anorexia Training

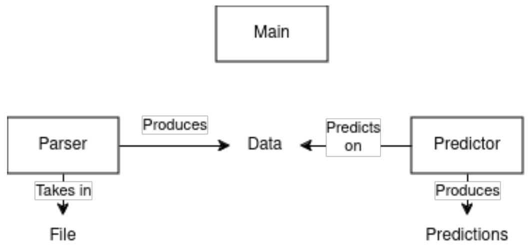

Figure 3: Anorexia Predicting

10.3 Eating Disorders

The eating disorders pipeline involves the usage of a relabeling technique. This technique takes the ED data, which original only has scores for each user, and produces scores for each post. The prediction model is then trained to predict these post level scores. Then these predictions are aggregated to user level predictions which are evaluated against what the user scores should be. The rational behind this relabeling method will be explored more in the VnV report.

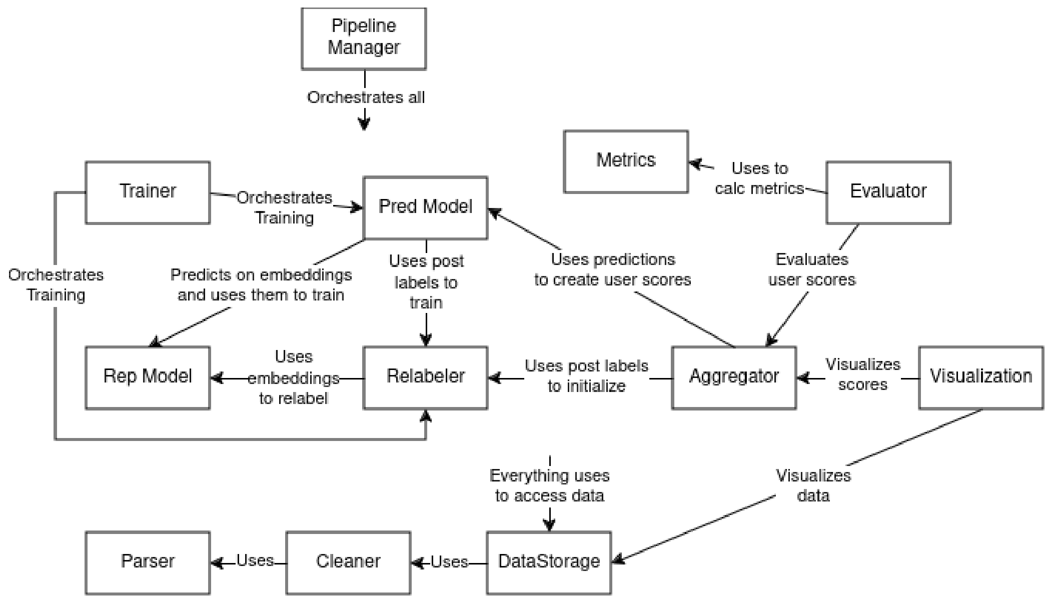

Figure 4: ED Uses Diagram

11 Timeline

- Module: EDM?? and EDM?? Finish Date: Jan 20 2024 Assignee(s): Jessica
- Module: EDM?? Finish Date: Jan 22 2024 Assignee(s): Jessica
- Module: EDM?? Finish Date: Jan 26 2024 Assignee(s): Jessica
- Module: EDM?? Finish Date: Jan 28 2024 Assignee(s): Jessica
- Module: EDM?? Finish Date: Feb 4 2024 Assignee(s): Jessica
- Module: EDM?? Finish Date: Feb 8 2024 Assignee(s): Jessica
- Module: EDM?? Finish Date: Feb 10 2024 Assignee(s): Jessica
- Module: EDM?? and EDM?? Finish Date: Feb 11 2024 Assignee(s): Jessica
- Module: EDM?? Finish Date: Feb 13 2024 Assignee(s): Jessica

- Module: EDM?? Finish Date: Feb 15 2024 Assignee(s): Jessica
- Module: M?? Finish Date: Feb 1 2024 Assignee(s): Franklin, Matthew
- Module: M?? Finish Date: Feb 1 2024 Assignee(s): Franklin, Matthew
- Module: M?? Finish Date: Feb 1 2024 Assignee(s): Franklin, Matthew
- Module: M?? Finish Date: Feb 1 2024 Assignee(s): Franklin, Matthew
- Module: M?? Finish Date: Feb 1 2024 Assignee(s): Franklin, Matthew
- Module: M?? Finish Date: Feb 1 2024 Assignee(s): Michael, Ben
- Module: M?? Finish Date: Feb 1 2024 Assignee(s): Michael, Ben
- Module: M?? Finish Date: Feb 1 2024 Assignee(s): Michael, Ben
- Module: M?? Finish Date: Feb 1 2024 Assignee(s): Michael, Ben
- Module: M?? Finish Date: Feb 1 2024 Assignee(s): All
- Testing (PyTest): M??, M??, M?? Finish Date: Feb 2 2024 Assignee(s): Franklin, Matthew
- Testing (PyTest): M??, M??, M?? Finish Date: Feb 2 2024 Assignee(s): Michael, Ben
- Testing (PyTest): M??, M?? Finish Date: Feb 3 2024 Assignee(s): Franklin, Matthew
- Testing (PyTest): M?? Finish Date: Feb 4 2024 Assignee(s): Michael, Ben
- Testing: All Modules Finish Date: Feb 4 2024 Assignee(s): All