Measuring the severity of the signs of Eating Disorders using Machine Learning Techniques

Sachin Prasanna
Information Technology
National Institute of Technology
Karnataka
Surathkal, Karnataka
sachinprasanna.211it058@nitk.edu.in

Abhayjit Singh Gulati
Information Technology
National Institute of Technology
Karnataka
Surathkal, Karnataka
abhayjitsinghgulati.211ee002@nitk.edu.in

Subhojit Karmakar
Information Technology
National Institute of Technology
Karnataka
Surathkal, Karnataka
subhojit.211it071@nitk.edu.in

M Yoga Hiranmayi
Information Technology
National Institute of Technology
Karnataka
Surathkal, Karnataka
myogahiranmayi.211it038@nitk.edu.in

***Abstract*—This paper measures an individual's severity of the signs of Eating Disorders using Machine Learning techniques. It is also task 3 of the CLEF ERISK 2024: EARLY RISK PREDICTION ON THE INTERNET Shared Task. We start off by explaining how the data was cleaned and preprocessed. Then, various approaches are given to solve the problem. These approaches include Word2Vec, TF-IDF, Backtranslation, Dimensionality Reduction, etc. The results are then summarised for each approach. The results were promising and better than last year's results.**

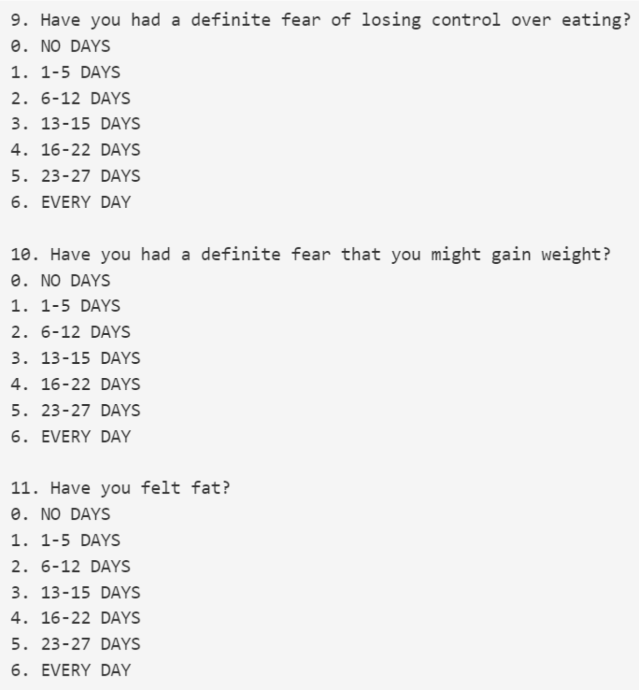
***Index Terms*—Machine Learning, Word2Vec, TF-IDF, Backtranslation, Dimensionality Reduction, Early Risk Prediction, Shared Task**

I. Introduction

Eating disorders, such as anorexia nervosa, bulimia nervosa, and binge eating disorder, are serious mental health conditions characterized by abnormal eating habits and distorted body image. These disorders often stem from a combination of genetic, psychological, and social factors. Individuals may restrict food intake, engage in binge eating followed by purging behaviors, or compulsively overeat. Eating disorders can have devastating effects on physical health, leading to malnutrition, electrolyte imbalances, and organ damage. Psychological impacts include depression, anxiety, and low self-esteem. Treatment typically involves a combination of therapy, nutritional counseling, and medical supervision to address both physical and mental aspects of the disorder.

II. Problem Statement

The task consists of estimating the level of features associated with a diagnosis of eating disorders from a thread of user submissions. For each user, the participants will be given a history of postings and the participants will have to fill a standard eating disorder questionnaire (based on the evidence found in the history of postings) [1].



```
9. Have you had a definite fear of losing control over eating?
0. NO DAYS
1. 1-5 DAYS
2. 6-12 DAYS
3. 13-15 DAYS
4. 16-22 DAYS
5. 23-27 DAYS
6. EVERY DAY

10. Have you had a definite fear that you might gain weight?
0. NO DAYS
1. 1-5 DAYS
2. 6-12 DAYS
3. 13-15 DAYS
4. 16-22 DAYS
5. 23-27 DAYS
6. EVERY DAY

11. Have you felt fat?
0. NO DAYS
1. 1-5 DAYS
2. 6-12 DAYS
3. 13-15 DAYS
4. 16-22 DAYS
5. 23-27 DAYS
6. EVERY DAY
```

Fig. 1. Questions from the EDE-Q

The questionnaires are defined from Eating Disorder Examination Questionnaire (EDE-Q) is a 28-item self-reported questionnaire adapted from the semi-structured interview Eating Disorder Examination (EDE). We will only use questions 1-12 and 19-28. It is designed to assess the range and severity of features associated with a diagnosis of eating disorder using 4 subscales (Restraint, Eating Concern, Shape Concern and Weight Concern) and a global score.

This task aims therefore at exploring the viability of automatically estimating the severity of multiple symptoms associated with eating disorders. Given the user's history of writings, the algorithms have to estimate the user's response to each individual question [1].

III. Related Work

Detecting mental health disorders from social media posts has gained significant traction as a research focus in recent years. The shared task was introduced last year in the same competition. Four teams had taken part in it.

The UMU team presented a run based on fine-tuning a pre-trained sentence transformer model. They processed the dataset and performed emoji feature extraction to enhance the model's performance [2].

The RiskBusters team leveraged a transformer-based topic modeling method. The team customized the BERTopic framework to obtain topic distributions at the user level. These topic distributions were then utilized as input features for downstream classification tasks. To improve the quality of embeddings, they adapted MentalBERT, a transformer-based language model, to the eating disorder domain [3].

The BFH-AMI team employed a logistic regression model that incorporated user and question embeddings derived from the Large Language Models (BERT and GPT-Large). By incorporating user and question embeddings the model could effectively leverage the semantic information and context within the social media writings [4].

IV. Dataset

The dataset consisted of 46 training instances and 28 testing instances. The dataset was given in the form of XML files which had to be cleaned for further processing. The XML files consisted of the user names, the posts they had posted and their timestamps. The answers given by the subjects (true labels) were given as a separate text file.

A. Data Cleaning

Several XML had loading issues and were incorrectly formatted. So as a first step, these issues were resolved and the XML file for each user was converted to CSV files for further cleaning and usage. It was also found that column names had a mismatch and was fixed.

Further cleaning involved the removal of emojis from the posts of users using the *emoji* library. It was also found that several posts were enclosed with starting with **b”** and ending with **”**. These were removed accordingly. Unicode representations were replaced with their actual representations. For example, *\xe2\x80\x99* means **’** and instances such as these were found and replaced.

B. Data Preprocessing

After cleaning, all posts from a single user were concatenated into one single chunk and preprocessed using standard preprocessing methods. This text was lowercased and URLs were removed. All kinds of punctuation were removed. Stop words were removed so that the machine learning models are not heavily influenced by the effects of these words. Finally, the words were lemmatized using the *WordNetLemmatizer* function from the *nltk.stem* library.

V. Data Visualisation

A critical step in any data project is to understand your data. Thus, data visualisation plays an important role. A simple user versus text length plot tells us the split of characters per user. This plot showed that there was indeed an imbalance in the length of texts of each user. The figure can be found in Figure 2.

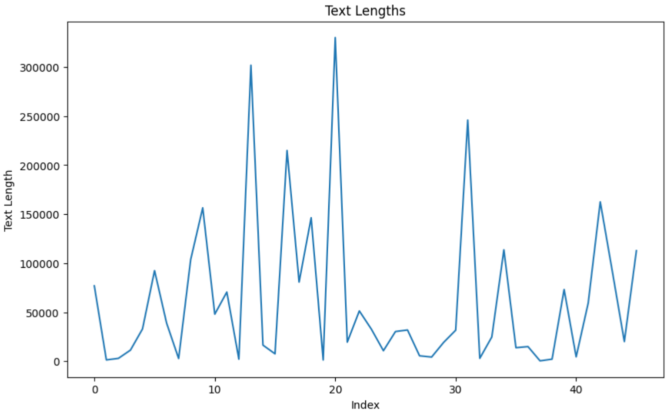


Fig. 2. Text length plotted with each user

Another interesting visualisation was to check how many posts had each person post on the reddit platform. It was again seen that the number of posts per user was not uniform and showed significant variation. The maximum number of posts of a person was 1210, while the minimum was 5. The average and median number of posts were 449.56 and 187.5 respectively. The pictorial representation can be found in Figure 3.

An ngram analysis was also done to find out the most common 2gram words in the entire dataset. The results are shown in Figure 4.

VI. Methodology

Five different approaches are proposed in our solution to the problem. They are described and explained below.

A. Different Models for Each Question

This approach consisted of fitting a model to each question. Since there were 22 questions answered by each subject, we made 22 different models to learn the distribution of each question's answers. Four different approaches were tried, they were - **Multinomial Naive Bayes**, **Linear Support Vector**

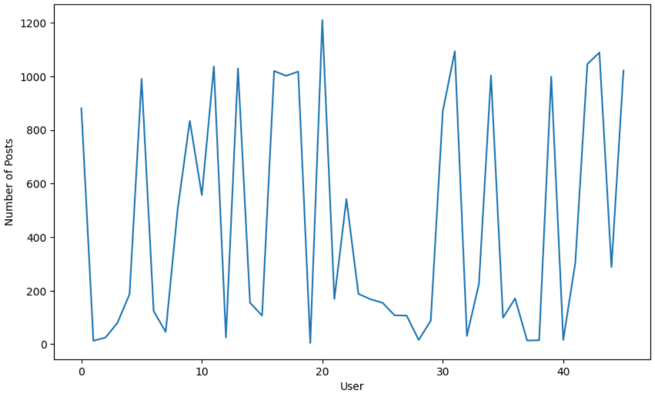
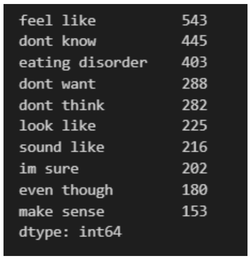


Fig. 3. Number of Posts posted by each user



```
feel like          543
dont know          445
eating disorder    403
dont want          288
dont think         282
look like          225
sound like         216
im sure            202
even though        180
make sense         153
dtype: int64
```

Fig. 4. Most common 2gram words

Machine, **Logistic Regression**, **Neural Network**. Each approach was taken and 22 different trained models are built for each question.

The **Multinomial Naive Bayes** model is a simple yet effective algorithm for text classification tasks. It works well with text data by modeling the probability of observing word counts in documents belonging to different classes. Despite its assumption of feature independence, it often performs satisfactorily in practice, particularly on large datasets.

The **Linear Support Vector Machine** model implemented here utilizes stochastic gradient descent with a hinge loss function for maximum-margin classification. It applies L2 regularization to prevent overfitting and employs hyperparameters for regularization strength, reproducibility, the maximum number of iterations and the tolerance of stopping criteria. It is particularly suited for large datasets and online learning scenarios.

Logistic regression is a widely-used linear classification algorithm that estimates probabilities using the logistic function. The model's parameters are optimized using the training data to maximize the likelihood of the observed labels. In this configuration, the logistic regression classifier employs a regularization parameter to control overfitting.

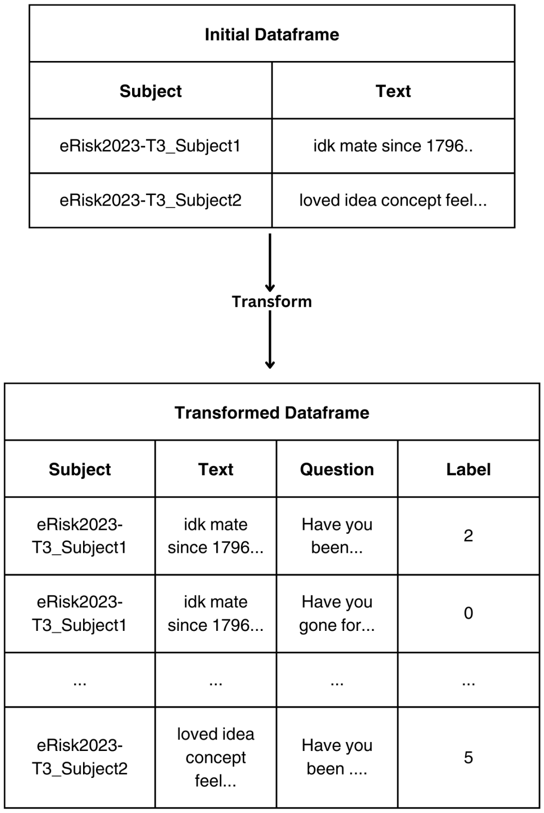
A **Neural Network** model was also employed for the task. This architecture consisted of two fully connected layers: the first layer with ReLU activation function and dropout regularization, followed by the second layer mapping to the number of output classes. The model utilized log softmax activation for output, making it suitable for multiclass classification. It was trained for 5 epochs, with the *Adam* optimiser and the *Cross Entropy Loss*.

For word embeddings for input to the models, a pipeline was constructed. The first component was the *CountVectorizer*, which converts the collection of text documents into a matrix of token counts. Then, the standard *TF-IDF* approach is taken and the corresponding matrix is constructed. These embeddings are taken for each document and fed into the machine learning algorithms and neural network for each question.

Although a good approach, the importance of the questions was not given as inputs to the models. It was just a case of learning 22 arbitrary distributions and then predicting the same. This emerged as a drawback as the questions were not given any importance in the predictions. Also, since the training set was small, there was a chance that the training was not sufficient for predicting the validation labels correctly.

B. *Extending Dataset and using Questions*

To overcome the drawback of the previous approach, the dataset was reshaped to include the text of the question in a separate column. This also includes the size of the dataset and makes use of a single model for training and predicting, rather than 22 models as in the previous approach. A pictorial representation of the transformation is shown in Figure 5.



Initial Dataframe

Subject	Text
eRisk2023-T3_Subject1	idk mate since 1796..
eRisk2023-T3_Subject2	loved idea concept feel...

Transformed Dataframe

Subject	Text	Question	Label
eRisk2023-T3_Subject1	idk mate since 1796...	Have you been...	2
eRisk2023-T3_Subject1	idk mate since 1796...	Have you gone for...	0
...	...	...	...
eRisk2023-T3_Subject2	loved idea concept feel...	Have you been	5

Fig. 5. Transformation of the Dataframe to include questions

After the transformation, the questions were also preprocessed using the same techniques as mentioned in Section IV. Subsequently, the text and question column were concatenated to a single column to be used for training models. **Multinomial Naive Bayes**, **Linear Support Vector Machine** and **Logistic Regression** models were used for training and predicting. The same approach of *TF-IDF* as mentioned in the previous section was adopted to convert the concatenated text into word embeddings.

Since the questions were very small in length and they were just concatenated at the end of the text, they did not make much of a difference. Also, better approaches could be implemented in terms of capturing the relationships and semantic meaning of the text rather than using *TF-IDF*.

C. Using Word2Vec

Word2Vec is a popular word embedding technique used to represent words as dense vectors in a continuous vector space. It learns these representations by training neural networks on large text corpora, aiming to capture semantic relationships between words based on their co-occurrence patterns. Word2Vec typically offers two approaches: Continuous Bag of Words (CBOW) and Skip-gram.

Pretrained Word2Vec models, offer precomputed word embeddings trained on vast amounts of text data, such as Google News articles as used in the project. The loaded model, pretrained on Google News corpus (3 billion running words) word vector model (3 million 300-dimension English word vectors), captures intricate semantic nuances and gives numerical meaning to text.

The same approach of concatenating the text and question was followed and then the concatenated text was converting to word embeddings in order to use both the text and question to predict the answers of the subjects.

Since each word is represented by a 300 size vector, the average vector of all the words was taken as the final representation of each subject. If a word was not present in the Word2Vec corpus, then its vector was taken as 0.

Using these new embeddings - **Logistic Regression**, **Linear Support Vector Machine**, **Random Forest** and **Gradient Boosting** algorithms were used to train the data.

The **Random Forest** model used here is an ensemble learning technique widely employed for classification tasks. Comprising multiple decision trees, each trained on random subsets of data and features, it offers robust predictions through a voting or averaging mechanism. With 100 trees, and a random state for reproducibility, this RandomForestClassifier provides effective classification across diverse datasets, especially in high-dimensional feature spaces.

The **Gradient Boosting** model utilized in this implementation is a powerful ensemble learning method widely used for classification tasks. It works by sequentially adding weak learners, typically decision trees, to correct the errors made by preceding models. Each subsequent learner focuses on the residual errors of the previous model, gradually improving the overall predictive performance. The key hyperparameters include the number of estimators and the learning rate, which control the model's complexity and the rate at which each additional learner contributes to the ensemble.

A clear problem in this approach was seen in the predictions of the model. It was observed that all 4 models were extremely biased towards predicting either 0 or 6. The Linear SVM model and the Logistic Regression model predicted only 0s and 6s for all the training instances. This called for a way to balance the training dataset and avoid the models from being over-biased to learning only 0s and 6s. It was observed that the training data set had an imbalance in the labels as shown in Figure 6. Another problem was again not giving enough weightage to the question, because we are the concatenating the text and question together. Both these issues are addressed in the following approach.

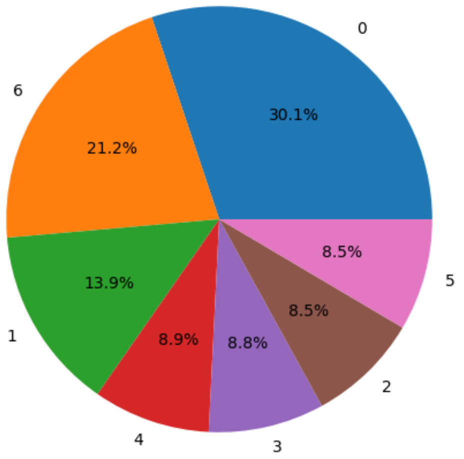


Fig. 6. Label distributions of the Training Set

D. Using Word2Vec with Backtranslation

The imbalance in training data was taken care of by Backtranslation. Backtranslation is the process of translating text from one language to another and then translating it back to the original language. We translated our text from English to French and vice versa. This technique is commonly used in natural language processing for data augmentation and improving the robustness of machine learning models.

Our implementation handles cases where the input text may exceed character limits specified by GoogleTranslate API (4999 characters) by splitting it into smaller chunks. It utilizes the Google Translator library to perform translation tasks efficiently.

Backtranslation was performed for only the text column and only for those instances with labels 2,3,4 and 5. These were then added to the dataset as new instances, which increased the weightage of the minority classes. This can be pictorially seen in Figure 8.

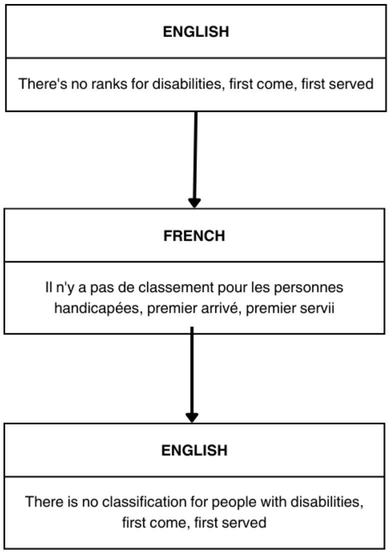

ENGLISH
There's no ranks for disabilities, first come, first served
FRENCH
Il n'y a pas de classement pour les personnes handicapées, premier arrivé, premier servii
ENGLISH
There is no classification for people with disabilities, first come, first served

Fig. 7. Backtranslation using the French Language

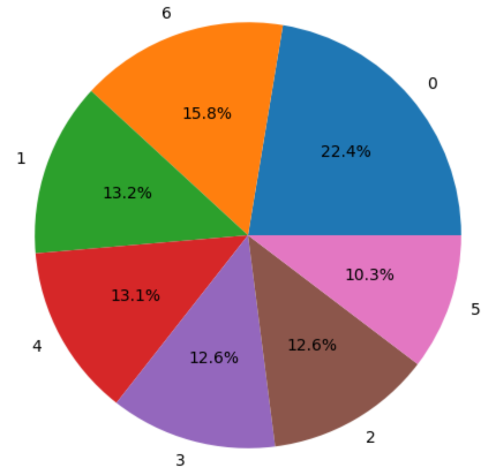


Fig. 8. Label distributions of the Training Set after Backtranslation

The second problem of not giving enough weightage to the question was taken care of by generating separate word embeddings for the text and question. Each of these are 300 dimensional vectors, and were concatenated to form the final embedding vector which was used for training which turned out to be a 600 dimensional vector. By this way, we ensured that both the text and question are given equal weightage during prediction of labels. An illustration of our methodology is shown in Figure 9.

After the generation of the word embeddings, they were fed to **Logistic Regression**, **Linear Support Vector Machine**, **Random Forest** and **Gradient Boosting** models for training.

Since the questions were much smaller in text size as compared to the text column as mentioned before, giving an equal weightage to both might be slightly overshooting the bar. An alternate approach is discussed regarding the same on the next subsection.

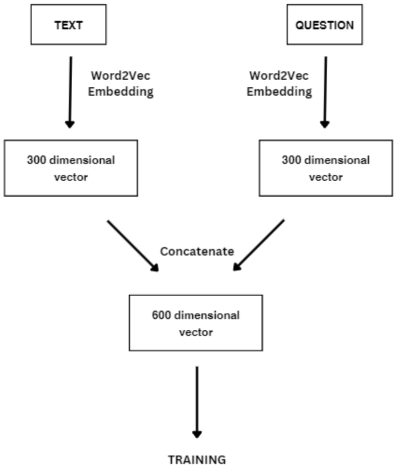


Fig. 9. Pictorial representation of Word2Vec with Backtranslation

E. Using Word2Vec with Backtranslation and Dimensionality Reduction

An innovative method to reduce the weightage of the question embeddings is proposed using dimensionality reduction. Principle Component Analysis (PCA) was used for the purpose of dimensionality reduction.

PCA (Principal Component Analysis) is a dimensionality reduction technique that transforms high-dimensional data into a lower-dimensional representation while preserving most of its variance. It achieves this by identifying the principal components, which are orthogonal directions in the original feature space that capture the maximum variance in the data. It was used to reduce to the dimensions of the question embeddings from size 300 to size 100.

This ensured that the importance of text to question is in the ratio of 3:1. The approach is illustrated in Figure 10.

VII. Evaluation Metrics

Evaluation is based on the following effectiveness metrics, and these are taken directly from the Shared Task organiser's paper [1].

A. Mean Zero-One Error (MZOE)

Mean Zero-One Error between the questionnaire filled by the real user and the questionnaire filled by the system (i.e. fraction of incorrect predictions).

$$\mathrm{MZOE}(f, Q) = \frac{|\{q_i \in Q : R(q_i) \neq f(q_i)\}|}{|Q|}$$

where f denotes the classification done by an automatic system, Q is the set of questions of each questionnaire, q_i is the i-th question, $R(q_i)$ is the real user's answer for the i-th question, and $f(q_i)$ is the predicted answer of the system for the i-th question. Each user produces a single MZOE score,

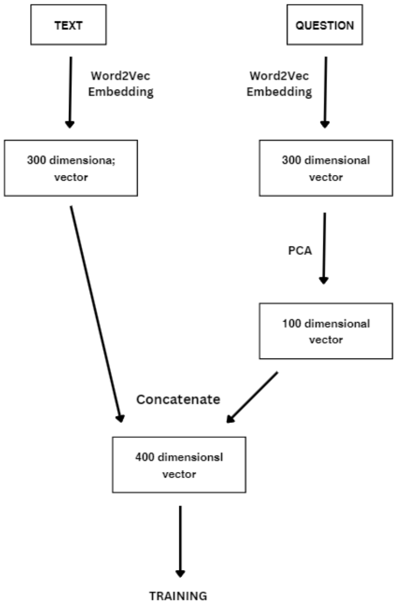


Fig. 10. Pictorial representation of Word2Vec with Backtranslation and Dimensionality Reduction

and the reported MZOE is the average over all MZOE values (mean MZOE over all users).

B. Mean Absolute Error (MAE)

Mean Absolute Error (MAE) between the questionnaire filled by the real user and the questionnaire filled by the system (i.e. average deviation of the predicted response from the true response).

$$MAE(f,Q) = \frac{1}{|Q|} \sum_{q_i \in Q} |R(q_i) - f(q_i)|$$

C. Macroaveraged Mean Absolute Error (MAE$_{macro}$)

Macroaveraged Mean Absolute Error (MAE$_{\text{macro}}$) between the questionnaire filled by the real user and the questionnaire filled by the system

$$MAE_{macro}(f,Q) = \frac{1}{7} \sum_{j=0}^{6} \sum_{q_i \in Q_j} \frac{|R(q_i) - f(q_i)|}{|Q_j|}$$

where Q_j represents the set of questions whose true answer is j (note that j goes from 0 to 6 because those are the possible answers to each question). Again, each user produces a single MAE_{macro} score, and the reported MAE_{macro} is the average over all MAE_{macro} values (mean MAE_{macro} over all users).

D. Restraint Subscale (RS)

Given a questionnaire, its restraint score is obtained as the mean response to the first five questions. This measure computes the RMSE between the restraint ED score obtained from the questionnaire filled by the real user and the restraint ED score obtained from the questionnaire filled by the system. Each user u_i is associated with a real subscale ED score (referred to as $RRS(u_i)$) and an estimated subscale ED score (referred to as $fRS(u_i)$). This metric computes the RMSE between the real and an estimated subscale ED scores as follows:

$$RMSE(f,U) = \sqrt{\sum_{u_i \in U} \frac{(R_{RS}(u_i) - f_{RS}(u_i))^2}{|U|}}$$

where U is the user set.

E. Eating Concern Subscale (ECS)

Given a questionnaire, its eating concern score is obtained as the mean response to the following questions (7, 9, 19, 21, 20). This metric computes the RMSE between the eating concern ED score obtained from the questionnaire filled by the real user and the eating concern ED score obtained from the questionnaire filled by the system.

F. Shape Concern Subscale (SCS)

Given a questionnaire, its shape concern score is obtained as the mean response to the following questions (6, 8, 23, 10, 26, 27, 28, 11). This metric computes the RMSE between the shape concern ED score obtained from the questionnaire filled by the real user and the shape concern ED score obtained from the questionnaire filled by the system.

G. Weight Concern Subscale (WCS)

Given a questionnaire, its weight concern score is obtained as the mean response to the following questions (22, 24, 8, 25, 12). This metric computes the RMSE between the weight concern ED score obtained from the questionnaire filled by the real user and the weight concern ED score obtained from the questionnaire filled by the system.

H. Global ED (GED)

To obtain an overall or 'global' score, the four subscale scores are summed, and the resulting total is divided by the number of subscales (i.e., four). This metric computes the RMSE between the real and an estimated global ED scores as follows:

VIII. Results

A. Different Models for Each Question

This approach provided very fruitful results with the Linear Support Vector Machine Model. In all the metrics except the Mean Zero-One Error, this model gave the least error and hence the best performance. These results were better than the last year's shared task submissions as well.

The Neural Network was the poorest performing model, which could it be owed to the hyperparameters chosen like number of layers, number of epochs, etc.

TABLE I
PERFORMANCE METRICS FOR SECTION A

Metric	MNB	LSVM	LogReg	NN
MAE	2.341	**1.974**	2.044	2.586
MZOE	0.675	0.701	**0.672**	0.765
MAE_m	1.745	**1.560**	1.627	1.783
GED	1.838	**1.361**	1.542	2.337
RS	2.579	**1.987**	2.432	2.745
ECS	3.017	**1.627**	2.148	2.801
SCS	1.914	**1.418**	1.623	2.240
WCS	1.914	**1.418**	1.623	2.240

B. Extending Dataset and using Questions

This approach concluded as the approach with the poorest results produced when compared to other approaches. This could be due to the poor capturing of semantic meaning of the text using the *TF-IDF* method. Another plausible reason could also be imbalance in the labels of the dataset. Linear Support Vector Machine was again the best model in terms of results in this approach.

TABLE II
PERFORMANCE METRICS FOR SECTION B

Metric	MNB	LSVM	LogReg
MAE	3.360	**2.127**	2.399
MZOE	0.815	**0.619**	0.731
MAE_m	2.286	1.974	**1.791**
GED	3.677	**2.570**	2.613
RS	3.694	3.071	**2.992**
ECS	3.185	**2.714**	2.581
SCS	4.278	**2.487**	2.907
WCS	3.823	**2.497**	2.524

C. Using Word2Vec

Results improved with the use of Word2Vec and the capture of semantic meaning of the text. Multinomial Naive Bayes could not be used here as it requires positive values in its features. Instead Random Forests and Gradient Boosting methods were introduced. The best performing models were Logistic Regression and Gradient Boosting.

TABLE III
PERFORMANCE METRICS FOR SECTION C

Metric	LogReg	LSVM	RF	GB
MAE	**2.185**	2.370	2.401	2.464
MZOE	**0.630**	0.657	0.687	0.784
MAE_m	1.992	2.035	2.005	**1.780**
GED	2.660	2.743	2.730	**2.576**
RS	3.000	3.071	3.025	**2.758**
ECS	2.888	2.672	2.527	**2.288**
SCS	**2.735**	2.999	2.946	3.082
WCS	**2.494**	2.634	2.878	2.647

D. Using Word2Vec with Backtranslation

Backtranslation balanced the training set in terms of label distribution. This approach also gave equal weights to both the text and question embeddings. This approach worked, and a further reduction in the error was seen. Another advantage was that only one model was being used here. The Gradient Boosting Algorithm gave the best results, and were comparable to the Linear SVM result of the first approach.

TABLE IV
PERFORMANCE METRICS FOR SECTION D

Model	LogReg	LSVM	RF	GB
MAE	2.005	1.950	1.974	**1.849**
MZOE	0.696	**0.672**	0.755	0.756
MAE_m	1.542	1.593	1.548	**1.413**
GED	1.580	1.512	1.607	**1.387**
RS	2.222	2.245	2.091	**1.975**
ECS	2.547	2.423	2.022	**1.585**
SCS	**1.439**	1.511	1.701	1.446
WCS	1.422	**1.416**	1.551	1.666

E. Using Word2Vec with Backtranslation and Dimensionality Reduction

In the final approach, a decrease in the weightage of the question embeddings was considered using dimensionality reduction. The results were again good and comparable to the results of Section D. Again, the Gradient Boosting's algorithm performance was particularly impressive when compared to the other models.

TABLE V
PERFORMANCE METRICS FOR SECTION E

Metric	LogReg	LSVM	RF	GB
MAE	2.125	2.026	1.943	**1.875**
MZOE	0.703	**0.682**	0.760	0.756
MAE_m	1.638	1.668	1.580	**1.502**
GED	1.582	1.545	1.756	**1.492**
RS	**2.145**	2.323	2.425	2.432
ECS	2.642	2.353	2.046	**1.600**
SCS	1.567	**1.430**	1.672	1.532
WCS	1.717	1.515	1.778	**1.403**

TABLE VI
MODEL ABBREVIATION GLOSSARY

Abbreviation	Full Form
LogReg	Logistic Regression
NMB	Multinomial Naive Bayes
LSVM	Linear Support Vector Machine
NN	Neural Network
RF	Random Forest
GB	Gradient Boosting

IX. ACKNOWLEDGEMENT

This project titled "Measuring the severity of the signs of Eating Disorders using Machine Learning Techniques" was an opportunity for us to explore the field of Machine Learning, Deep Learning and its applications in Natural Language Processing. Advantages and disadvantages of several algorithms

and methods were analyzed and evaluated. We would like to extend our heartfelt thanks to **Dr. Anand Kumar Madasamy** for providing us with invaluable guidance and support throughout the course of this project. His constant encouragement and constructive feedback helped us stay focused and motivated. All code associated with this project can be found in GitHub repository.

References

[1] Javier Parapar, Patricia Martin-Rodilla, David E. Losada, Fabio Crestani - Overview of eRisk at CLEF 2023: Early Risk Prediction on the Internet (Extended Overview)

[2] R. Pan, J. A. G. Díaz, R. Valencia-Garcia, UMUTeam at eRisk@CLEF 2023 shared task: Transformer models for early detection of pathological gambling, depression, and eating disorder, in: Working Notes of CLEF 2023 - Conference and Labs of the Evaluation Forum, Thessaloniki, Greece, September 18-21, 2023.

[3] D.-N. Grigore, I. Pintilie, Transformer-based topic modeling to measure the severity of eating disorder symptoms, in: Working Notes of CLEF 2023 - Conference and Labs of the Evaluation Forum, Thessaloniki, Greece, September 18-21, 2023.

[4] G. Merhbene, A. R. Puttick, M. Kurpicz-Briki, BFH-AMI at eRisk@CLEF 2023, in: Working Notes of CLEF 2023 - Conference and Labs of the Evaluation Forum, Thessaloniki, Greece, September 18-21, 2023.