**REPUBLIC OF TURKEY**
**YILDIZ TECHNICAL UNIVERSITY**
**DEPARTMENT OF COMPUTER ENGINEERING**


# EMOJI PREDICTION AND MODEL ANALYSIS WITH NATURAL LANGUAGE PROCESSING

19011018 — Bedrettin Şamil Öztürk

**SENIOR PROJECT**

Advisor
Prof. Dr. Mehmet Fatih AMASYALI

Mayıs, 2024

# ACKNOWLEDGEMENTS

I would like to express my thanks and respect to Prof. Mehmet Fatih Amasyalı, who has provided me with his help and valuable guidance for any situation I may need since the selection of the project. Especially his in-depth knowledge and experience in the field that is the subject of the project has been a horizon-opener for me. This project has been an important stepping stone for me to progress in the developing fields of artificial intelligence and natural language processing. Even the possibility of playing a role in the development of these groundbreaking technologies is very exciting. I am grateful to him for stooding behind me so that I could have a share in this.

Bedrettin Şamil Öztürk

# TABLE OF CONTENTS

# LIST OF ABBREVIATIONS

| | |
|---|---|
| BERT | Bidirectional Encoder Representations from Transformers |
| CNN | Convolutional Neural Network |
| GPT | Generative Pre-trained Transformer |
| LSTM | Long Short-Term Memory |
| NLP | Natural Language Processing |
| RNN | Recurrent Neural Network |
| SemEval | Semantic Evaluation |
| SVM | Support Vector Machine |
| TF-IDF | Term Frequency — Inverse Document Frequency |
| UROP | Undergraduate Research Opportunities Program |
| API | Application Programming Interface |
| bi-LSTM | Bidirectional Long Short-Term Memory |

# LIST OF FIGURES

# LIST OF TABLES

# ABSTRACT

## Emoji Prediction and Model Analysis with Natural Language Processing

Bedrettin Şamil Öztürk

Department of Computer Engineering
Senior Project

Advisor: Prof. Dr. Mehmet Fatih AMASYALI

In our digitalised age, our communication tools make it increasingly difficult to communicate our emotional reactions. The aim of this project is to achieve model successes that will overcome this challenge. By using Natural Language Processing methods and machine learning algorithms, it is aimed to reveal and predict the emotion hidden under the language. In this way, the actual emotions intended to be conveyed in written communication will be understood more clearly and the inefficiencies in emotion analysis methods will be eliminated.

In addition to models and classification methods such as BERTurk, bi-LSTM, KNN, and Multinomial Naive Bayes, various representation methods such as TF-IDF and CountVectorizer are also examined. In order to increase the performance, data augmentation methods were applied and as a result, an increase in success was observed. Thanks to the Hugging Face and OpenAI interfaces where data generation was performed, the existing dataset was enriched and a much better ground was prepared for training. The effects of these data generation methods on success were also measured.

Data augmentation, representation methods and model types were analysed separately by creating a control group and comparisons were made in the light of different success metrics. It was understood in which situations models and datasets tend to give more successful results. These findings have made a great contribution to existing sentiment analysis studies. It has become a source of analysis for future

studies in this field.

**Keywords:** Natural Language Processing, emoji prediction, sentiment analysis, machine learning, data generation, deep learning, Transformers models

# ÖZET

## Doğal Dil İşleme ile Emoji Tahmini ve Model Analizleri

Bedrettin Şamil Öztürk

Bilgisayar Mühendisliği Bölümü
Bitirme Projesi

Danışman: Prof. Dr. Mehmet Fatih AMASYALI

Dijitalleşen çağımızda iletişim araçlarımız duygusal tepkilerimizi iletmeyi gittikçe zorlaştırmaktadır. Bu projenin amacı, bu zorluğun üstesinden gelecek model başarılarına ulaşmaktır. Doğal Dil İşleme yöntemleri ve makine öğrenmesi algoritmaları kullanarak dilin altında saklanan duyguyu ortaya çıkarmak ve tahmin etmek hedeflenmektedir. Böyle yazılı iletişimde aktarılmak istenen asıl duygular daha net anlaşılacak ve duygu analizi yöntemlerindeki etksikliklikler giderilecektir.

Çalışmada BERTurk, bi-LSTM, KNN ve Multinomial Naive Bayes gibi model ve sınıflandırma yöntemlerinin yanında TF-IDF ve CountVectorizer gibi çeşitli temsil yöntemleri de incelemeye alınmıştır. Performansın artırılması hususunda ise veri artırım yöntemlerine başvurulmuş ve sonuç itibariyle başarıda artış gözlenmiştir. Veri üretiminin gerçekleştirildiği Hugging Face ve OpenAI arayüzleri sayesinde mevcut veri kümesi zenginleştirilmiş, eğitim için çok daha iyi bir zemin hazırlanmıştır. Bu veri üretimi metotlarının başarıya olan etkileri de ayrıca ölçülmüştür.

Veri artırımı, temsil yöntemleri ve model çeşitlerinin ayrı ayrı kontrol grubu oluşturularak incelemeye alınması neticesinde farklı başarı metrikleri ışığında kıyaslamalar yapıldı. Modellerin ve veri kümelerinin hangi durumlarda daha başarılı sonuç vermeye eğimli oldukları anlaşıldı. Bu bulgular, mevcut duygu analizi çalışmalarına büyük bir katkı sağladı. Gelecekte bu alanda yapılacak çalışmalar için bir analiz kaynağı oldu.

**Anahtar Kelimeler:** Doğal Dil İşleme, emoji tahmini, duygu analizi, makine

öğrenmesi, veri üretimi, derin öğrenme, Transformers modelleri

# 1
# Introduction

People use tone of voice, gestures and mimics rather than verbal expressions when communicating. Thus, the transfer of what needs to be transferred is transferred much more clearly. According to research, 90% of emotional expressions are conveyed through these non-verbal expressions. Emojis are indispensable for expressing our emotions in our written communication. Because most of the time, without emoji, a message is often emotionally incomplete. For this reason, it is necessary to be aware of how important emojis really are in written communication. Especially in a complex language like Turkish, the place of emojis in the language is even more distinct.

In studies in the field of Natural Language Processing(NLP), a classification such as positive, negative and neutral is usually encountered. But the phenomenon we call emotion is much more than that. A text contains much more than being positive, negative or neutral. Therefore, it is necessary to unravel this emotional complexity and run appropriate classification models accordingly. Therefore, in this study, basic emotions and corresponding emoji were preferred. Since it is obvious that the emotional intention in texts is best expressed through emojis, the labelling process was also done with these emojis.

Correct prediction of the emoji tags selected for this study was considered as a priority task. This classification and prediction mechanism is aimed to work correctly and to increase the success values. This requires in-depth analysis and comparison of machine learning models, various data augmentation methods and vectorisation methods. In this study, necessary analyses have been made in line with all these objectives and relevant findings have been obtained

## 1.1 Necessity and Objectives of the Project

Detailed emotional analyses of text data are of great importance, especially in the use of chatbots and customer analysis of companies. However, despite this importance,

unfortunately, it is difficult to find a model that works well enough for Turkish and meets the requirements. The importance of making both pre-processing steps and training steps suitable for the Turkish language has increased even more. In this way, knowing how different factors affect model success will give direction and speed to these studies.

The main purpose of this project is to recognize a wide variety of people's emotions through text data and to make detailed analyzes accordingly. Recognizing many different emotions, not just positive, negative and neutral labels, is very important for people's general use of artificial intelligence.

## 1.2 Chapters Overview

- Preliminary Review:

  In the preliminary review section, existing studies on the subject were examined and the academic point reached was determined.

- Feasibility:

  In this section, the project is analyzed and planned in terms of applicability. It includes business time, software, hardware, legal and economic feasibility.

- System Analysis:

  In this chapter, after the requirements of the project were thoroughly determined, decisions were made about which technologies would be used and the plan became clear. .

- System Design: At this design stage, the previously planned road map started to be realized and the findings of the prediction models started to be received.

  - Software Design: The libraries and models to be used in software design, which is the most important part of the design phase, have been selected. Various use cases were examined and requirements were designed to be applicable for this study.
  - Database Design: This part covers the preparation processes from obtaining the data set, which is a factor that directly affects the success results, to model training.
  - Input-Output Design: This part includes obtaining data from the necessary development environment and making conclusions available through visualizations of the relevant controls. Analysis is made based on the given outputs.

- Implementation:

  This is the section where all plans have been finalized and it is now time to put all processes into code. As a result of the relevant controls and configurations, visualizations are obtained and it is observed that the code works correctly with different parameters.

- Experimental Results:

  This section includes how the codes run and the trained models give results in different scenarios and how they are compared in different ways. Visualized graphics and outputs were screenshotted and added to the report.

- Performance Analysis:

  In this section, the structures and factors compared in the experimental results are discussed and comments are made.

- Conclusion:

  In this section, clear and concrete inferences are made on the findings of all these comparisons and analyses. The results have become clear about which factors affect success and how.

# 2
# Literature Review

The use of emojis in digital communication has taken a large place in NLP research. Because we can only understand ideas that cannot be understood directly from the text through emojis. There are many studies carried out in the academy in such an important field.

## 2.1 Current Approaches and Methodologies

Machine learning techniques such as Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) have been widely applied in emoji prediction research. However, the introduction of transformer-based models like BERT (Bidirectional Encoder Representations from Transformers) has shifted the study field toward more sophisticated, context-aware systems.

In their study "Emoji Prediction in Tweets using BERT,"[1] Muhammad Osama Nusrat and colleagues from Fast Nuces, Islamabad, and Université Bretagne Sud, Vannes, offer a transformer-based technique using BERT for emoji prediction.

"The Duluth UROP team at SemEval-2018 Task [2] employed ensemble learning techniques incorporating classifiers like Naive Bayes, Logistic Regression, and Random Forests to predict emojis from tweets in many languages. Their approach focused on applying oversampling techniques to address data skewness, utilizing bigram and unigram properties.

"The EPUTION project"[3], presented at SemEval-2018, suggested a novel user adaption technique that enhanced emoji prediction accuracy by more than 6%, in contrast to standard models.

This research investigates the effectiveness of multiple models, including logistic regression, LSTM (Long Short-Term Memory), Random Forest Regressor, and SVM (Support Vector Machine), in order to predict the frequency of the most popular

emojis in tweets for both English and Spanish. Emphasizing the challenges of emoji prediction, the other study[4] notes that some emojis—like a Christmas tree—are more predictable than others that are used more frequently.

Another paper[5] describes a deep learning method that leverages LSTM networks enhanced with a context-aware self-attention mechanism to predict emojis in English tweets.

The paper "The Dabblers at SemEval-2018 Task 2: Multilingual Emoji Prediction"[6] by Larisa Alexa and colleagues looks into the use of Recursive Neural Networks and Naive Bayes for emoji prediction in tweets. The study highlights how challenging it is to comprehend emojis due to their wide variations between user settings, languages, and nations.

Francesco Barbieri and colleagues report the first shared task on emoji prediction, "SemEval 2018 Task 2: Multilingual Emoji Prediction,"[7]. Predicting the most probable emoji from text-only tweets in Spanish and English is the challenge at hand.

The Tübingen-Oslo team's strategy[8] for the SemEval-2018 Task 2, which concentrated on multilingual emoji prediction in texts written in both Spanish and English, is presented in this work.

This paper presents the The Tübingen-Oslo team's approach[8] for the SemEval-2018 Task 2, which focused on multilingual emoji prediction in texts written in both Spanish and English.

Another work[9] uses pre-trained language models to examine gender identification from Turkish tweets, highlighting the understudied field of gender categorization from brief social media posts in the Turkish language.

The UMDSub at SemEval-2018 Task 2[10] employed a multi-channel Convolutional Neural Network (CNN) that leverages subword embeddings as its textual representation technique to address the problem of emoji prediction from tweets.

For SemEval-2018 Task 2, YNU-HPCC[11] employed a bi-directional gated recurrent unit (Bi-GRU) enhanced with an attention mechanism to predict emojis in tweets posted in both English and Spanish.

## 2.2 Identification of Gaps and Contributions of the Project

The majority, and almost all, of natural language processing and emoji prediction studies have focused on the English language. However, interest in the Turkish language has remained very limited and this project aims to contribute to the academy in terms of model training and analysis in Turkish.

This study also supports emoji prediction tasks by testing and comparing different vectorization methods separately. The methods used can then be adapted to other languages.

# 3
# Feasibility

The feasibility section of this study analyzes how much resources the project needs while measuring its feasibility. It includes technical feasibility, labor and time planning, legal vs economic feasibility sections.

## 3.1 Technical Feasibility

It examines whether the technical requirements and competencies are met to carry out the project as planned. It is divided into software and hardware

### 3.1.1 Software Feasibility

Choosing the Python language in the project is based on the reasons that it includes libraries of machine learning algorithms, provides ease of operation, and can be easily implemented in the Google Colab development environment. Powerful libraries such as Pytorch, tensorflow, scikit learn are the main reasons to use Python.

Hugging Face's Transformers library was used as a source for this emoji prediction task and all nlp processes.

The environment in which the project was developed, including its inputs and outputs, was chosen as Google Colab Pro. Thus, it is possible to perform training operations that require large processing power much faster.

In addition, by using the HuggingFace Inference API and Pro membership, it was possible to use large language models in our code through the API. It played a major role in meeting the requirements of the project.

Later, the OpenAI API service selected for another data augmentation method was also integrated, thus obtaining the necessary data to learn how the results of the models would be affected.

### 3.1.2 Hardware Feasibility

Due to the high processing power requirements of deep learning models, powerful GPUs provided by GOogle Colab Pro were used and ensured the continuity of the experiments.

A personal computer with 16 GB RAM and an Intel i7th generation processor was used for local development.

Thus, it became clear that sufficient and necessary resources were obtained for the entire technical scope targeted in the project.

## 3.2 Labor and Time Planning

It involves planning which works need to be done and in how long so that the project can achieve its intended goals. The project plan predicts that the project will be completed within 12 weeks if followed.

**Figure 3.1** displays the labor and time planning for the project and lists the precise steps and due dates required to finish each stage of the project successfully.

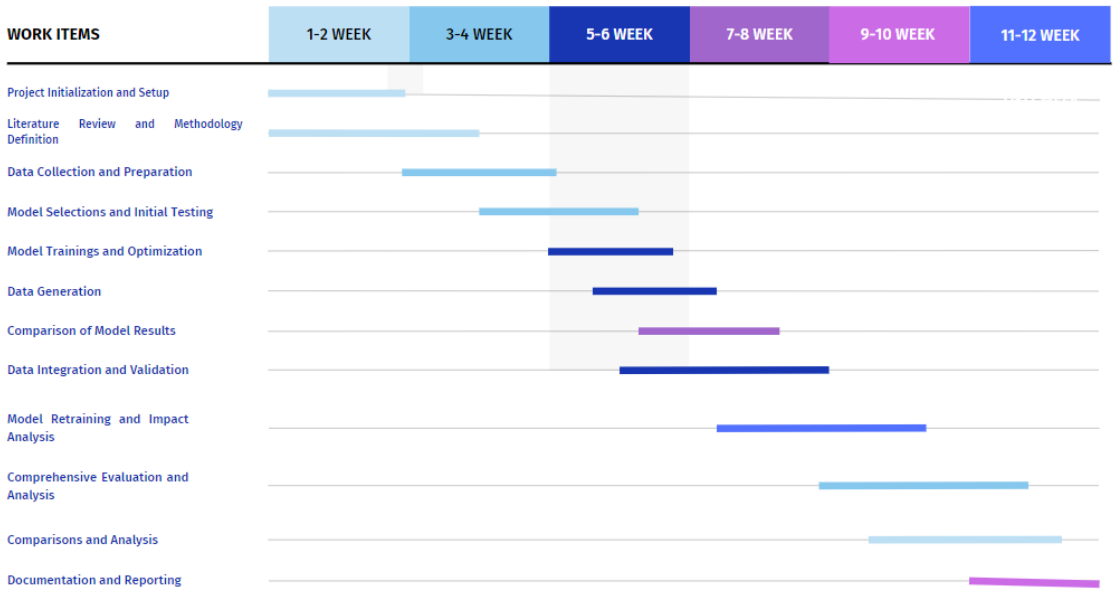


**Figure 3.1** Gantt Chart

- **Week 1: Project Initialization and Setup** Google Colab Pro, the environment in which the emoji prediction task would be carried out, was used, thus enabling all necessary integration processes.

- **Weeks 1-2: Literature Review and Methodology Definition** Bu alanda emoji tahmini üzerinde yapılan araştırmaların incelenmesi ve mevcut çalışmayla kıyaslanması, son gelişmelerden haberdar olunması.

- **Weeks 2-3: Data Collection and Preparation** Appropriate datasets labeling different emojis for Turkish were scanned from rich sources such as Hugging Face and Kaggle. Then, after selecting the dataset, the processes of making this dataset ready for the models were examined.

- **Weeks 3-4: Model Selections and Initial Testing** After preparing the dataset, the models to be analyzed were examined and applied to the existing dataset in turn. Their performances will be compared according to success metrics.

- **Weeks 4-6: Model Training and Optimization** After the models were run, their performances were examined and further investigations were carried out on which parameters to change and how to increase this success.

- **Weeks 6-7: Data Generation** To better examine how expanding and growing the existing dataset affects the success of the models, data was generated using the OpenAI API service and Hugging Face's Inferece APIs and the results were examined.

- **Week 7: Comparison of Model Results** The models were compared and the reasons for the differences in their performance were investigated and examined.

- **Week 8: Data Integration and Validation** The generated data was added to the existing data set and thus the retraining processes were progressed.

- **Week 9: Model Retraining and Impact Analysis** After different methods were tried and the sizes of the data sets were adjusted, how these factors affected the success of the models was examined in detail and final reports were prepared.

- **Weeks 10-11: Comprehensive Evaluation and Analysis** Modellerin farklı perfonmansları başarı metrikler çerçevesinde daha yakından incelendi.

- **Week 11: Model Comparison and Final Selection** After testing existing models and methods and comparative analysis, the most suitable models and methods for practical use were found.

- **Week 12: Documentation and Reporting** The reporting phase of all these processes involved examining the codes and analyzing the outputs. When the latest findings became clear, they were added to the conclusion section.

## 3.3 Legal Feasibility

Since all resources used in this study are open source, there is no violation, including the protection of personal data. All processes are carried out legally.

## 3.4 Economic Feasibility

Deep learning models and complex structures running in the project require large volumes of processing power. For this purpose, processing units from the Google Colab Pro product were purchased because personal computers were not powerful enough. GPU and RAM upgraded.

# 4
# System Analysis

This section includes planning the course of the project. It clarifies what path to follow towards which goals. Details the requirements.

## 4.1 Project Objectives and Requirements Identification

The aim of this project is to increase the accuracy of emoji predictions in Turkish datasets and to use machine learning algorithms in the most accurate way. Efforts are made to find and eliminate existing deficiencies.

To achieve this, data augmentation methods were examined. New compatible data was generated based on the existing dataset via the OpenAI API service and the Hugging Face Inference API. In this way, the impact of existing models on success was measured. This measurement was clarified with different metrics.

In addition, in this study, different vectorization methods such as TF-IDF and CountVectorizer were compared, and the SMOTE method, which can be used in case of imbalance in the data set, was also tried and its effect was measured. In addition, KNN, Multinomial Naive Bayes and bi-LSTM models were run with different parameters and different dataset settings to observe how the results would change.

## 4.2 System Modules

The project is broken up into several main modules, each of which is designed to carry out a certain task in the pipeline for emoji prediction. Together, these modules process data, train models, and prediction emojis based on Turkish text data by utilizing the latest advancements in machine learning and natural language processing.

- **Data Collection and Preprocessing Module:** This module involves finding the dataset and making sure that it meets the appropriate conditions. In our study,

the PsychExp dataset used by DeepMoji was selected. This is because it contains the appropriate number of different emotion labels.

- **Text Vectorization and Encoding Module:** The stages of preparing the dataset before model training are very important for success results. The tokenizer class and BERT representations of the Transformers library are prominent methods in this regard. TF-IDF and CountVectorizer methods were also examined in this study.
- **Model Training and Evaluation Module:** It involves training the selected models within the framework of accuracy, precision, recall, and F1-score metrics and performing this training with appropriate configuration parameters.
- **Emoji Prediction Engine:** In this module, the prediction skills of the trained models are tested and graphs are created. The most important of these visualizations are confusion matrices. In this way, it can be seen how well the model works in which classes. .

These modules form the outline of the whole project and this road map is followed. Organizing work this way is important for accurately measuring project success.

The diagram visualizing the data flow steps is as in **Figure 4.1**

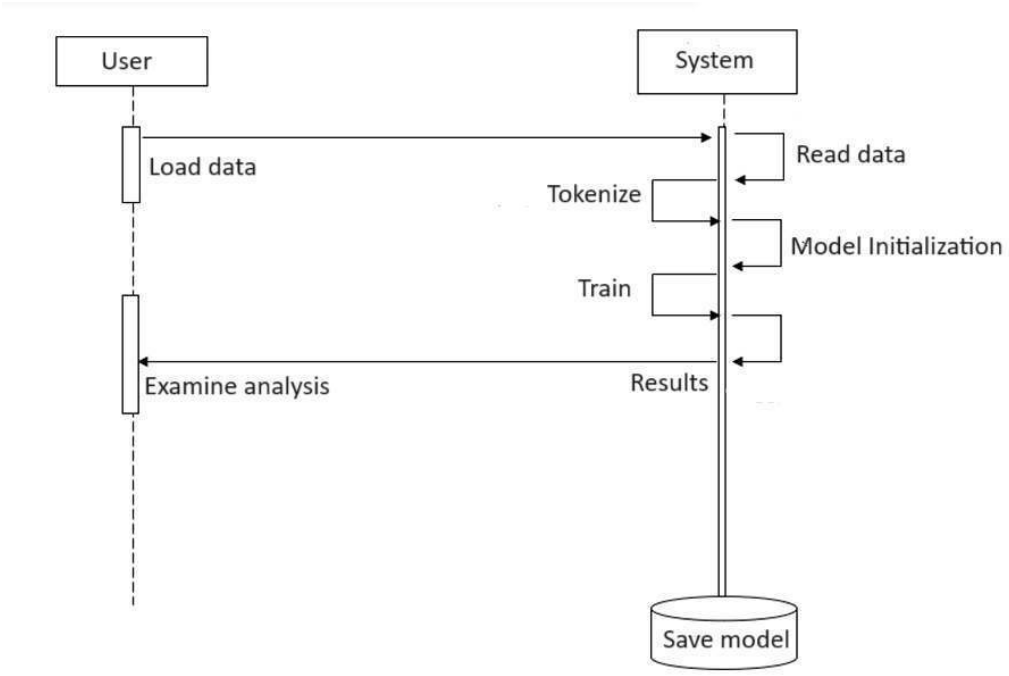


**Figure 4.1** Data Flow Diagram

## 4.3 Performance Metrics

The effectiveness of the emoji prediction models will be assessed in this study using a number of crucial performance metrics.

### 4.3.1 Confusion Matrix and Its Importance

The confusion matrix is a very useful visual tool in measuring the predictions of trained models. It helps to see the performance of the algorithms most clearly by comparing the real labels with the predicted labels.

### 4.3.2 How the Confusion Matrix Works

The confusion matrix lays out the predictions in a matrix format with:

- **True Positives (TP):** Correctly predicted positive observations.
- **True Negatives (TN):** Correctly predicted negative observations.
- **False Positives (FP):** Incorrectly predicted as positive (Type I error).
- **False Negatives (FN):** Incorrectly predicted as negative (Type II error).

The confusion matrix, which examines actual and predicted values, is presented in **Figure 5.7**.

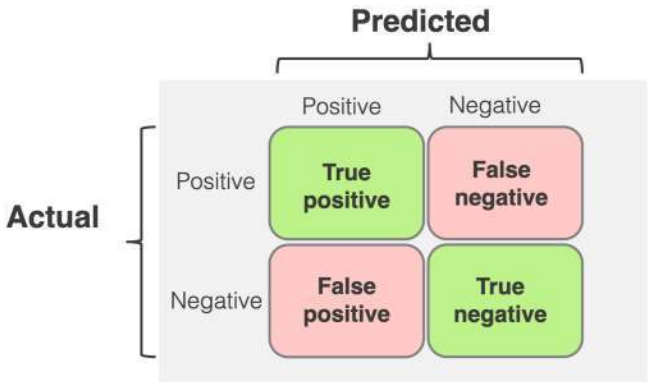


**Figure 4.2** Confusion Matrix [12]

### 4.3.3 Accuracy

This statistic evaluates by calculating the ratio of correctly predicted data to all available data. High accuracy rate is a primary goal for every algorithm.

### 4.3.4 Precision

This gives a percentage by calculating the proportion of truly positive predictions among all positive outcomes predicted by the model.

### 4.3.5 Recall

Sensitivity is what it is called, finding each relevant class, finding the proportion of data it correctly identifies among all real classes.

### 4.3.6 F-1 Score

Recall and sentivity are close in value and are useful when only one statistic is needed. It will also be used in this study.

### 4.3.7 Support

This metric simply indicates the number of times the class is in the dataset.

The Venn diagram illustrating the relations of the values of True Negatives (TN), True Positives (TP), False Negatives (FN), and False Positives (FP) is depicted in **Figure 4.3**

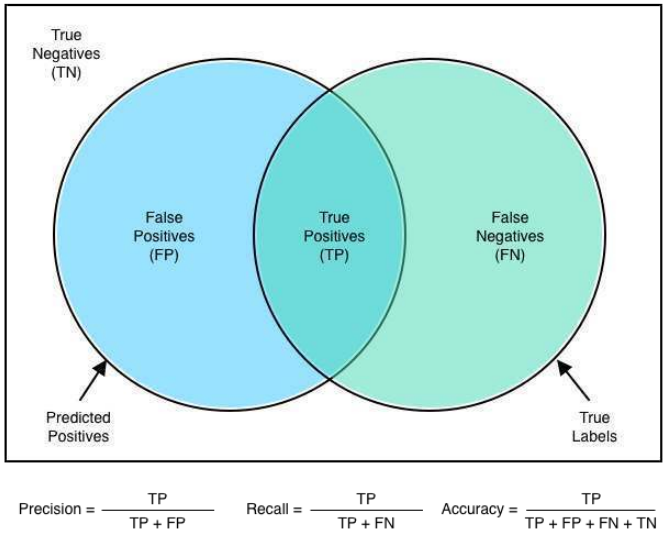


**Figure 4.3** Performance Metrics Venn Diagram [13]

Each of these metrics can be affected by various parameters such as traint test separation, dataset dimensions, vectorization methods, data generation model preferences. Throughout this project, analyzes will be conducted on these metrics.

# 5
# System Design

The purpose of this section includes planning and designing all processes ready for implementation, from obtaining data to obtaining results.

## 5.1 Software Design

The software part of the design software constitutes the most important part of the project. Examples such as how to take the data, how to prepare it, how to divide it into which ratios, which models to read and how, etc. provide answers and prepare them for the implementation phase.

The entire process of developing the emoji prediction system is shown in the below, which covers everything from initial data collecting to final analysises. This diagram aids in understanding the system's architecture and the sequential steps taken to develop and enhance the emoji prediction models. Planning of all these processes is shown in **Figure 5.1**

1. **Data Flow and Process Design:** Necessary tokenization and vectorization processes are initially carried out to remove noise in the data and make its format ready for machine learning algorithms. Preprocess steps are skipped thanks to Transformers Tokenizers and there is no loss of success. GPT and BERT representations are also used at this stage.

2. **Machine Learning Model Integration:** A pre-trained model was selected to apply fine tune. BERT was chosen for this because it was observed that this model was more suitable for Turkish. Make sure that the necessary configuration adjustments are made.

3. **Algorithm and Parameter Optimization:** This stage includes using KNN, Multinomial Naive Bayes and bi-LSTM models, as well as trying vectorization

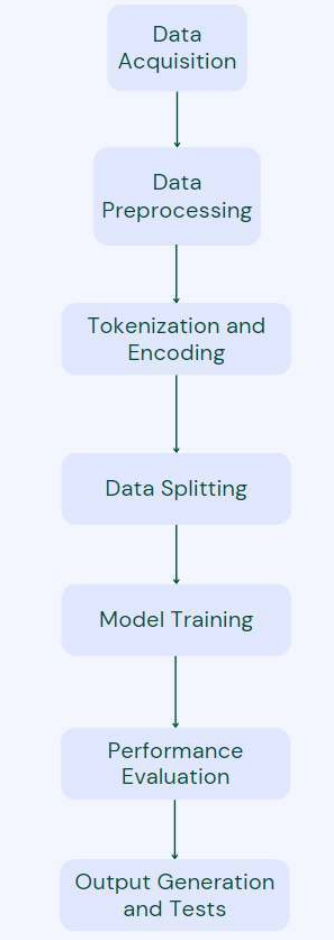


**Figure 5.1** Software Architecture Flowchart

methods such as TF-IDF and CountVectorizer sequentially and measuring their effect on the results. It is aimed to gain generalization ability on data.

4. **Integration of Performance Metrics:** In order to view the relevant performance metrics, the classification report creation phase comes after the training and prediction phases. Thus, the effect of the model and other factors can be seen as direct output and compared.

5. **Data Generation and Impact on Performance:** This phase, which constitutes an important part of the project, involves generating new data by adhering to emotion labels over the existing data set. As a result, it was observed to work well and new data sets were produced.

## 5.2 Database Design

The PsychExp dataset used by DeepMoji was preferred after the dataset in Hugging Face, which is in Turkish and fed from datasets that allow training. This is because

these datasets were used in training 1.2 billion tweets. Sample text - tag pairs are as in **Figure 5.2**

```
df1.sample(10)
```

| | Text | Label |
|---|---|---|
| 1449 | Yanlışlıkla ön kamera açıldı irkildim | 😰 |
| 1730 | Artık be diyebilirim ki kelimelerimden de önce... | 😔 |
| 1473 | Ben şok olmuştum gidemez diye ağlamıştım ya in... | 😜 |
| 2679 | Son görülme saatin endişe verici rakamlara ula... | 😰 |
| 775 | Aziz Yıldırım defol git hayatımızdan küflü pey... | 😠 |
| 3125 | Bugün benim doğum günüm 21 de bitti | 😊 |
| 3727 | Antidepresan kullanmaya başladığımdan beri dah... | 😠 |
| 2108 | Bu kadar twite nasıl tt olmamış insan gerçekte... | 😜 |
| 1169 | Sinir akıyor kanımdan yemin ederim kafayı yiye... | 😠 |
| 1571 | Konyaspor adına üzüldüm hakem rakip gibi davra... | 😔 |

**Figure 5.2** Dataset Samples

The dataset simply consists of two columns. Text and its corresponding emotion tag, thanks to these tags, it can be understood which emoji corresponds to that text.

**Figure 5.3** displays a word cloud, which offers insight and a way to comprehend the data set's content.

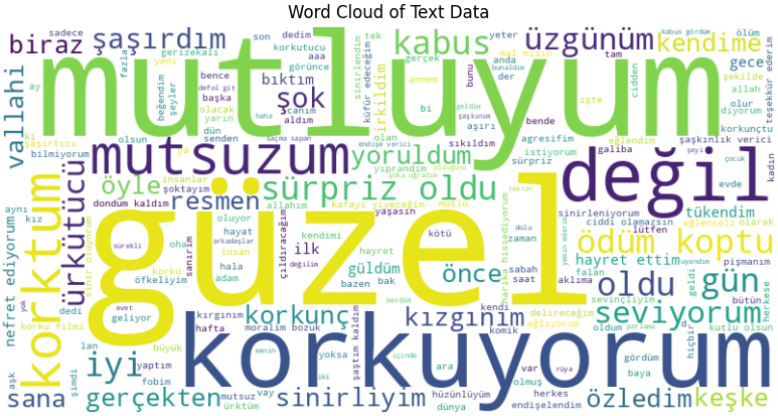


**Figure 5.3** Word Cloud

Supervised learning method is the most suitable method for this project. As a matter of fact, learning which emotion tag corresponds to which text samples allows the model to make accurate predictions on the test data.

The length of the text contents in the dataset is as shown in in **Figure 5.4**

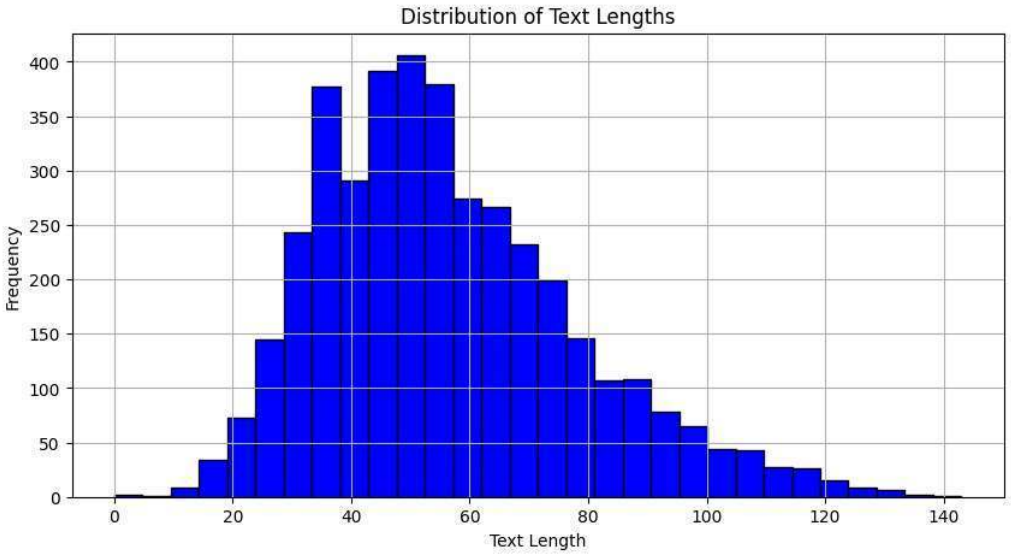


**Figure 5.4** Text Lengths

## 5.3 Input-Output Design

This part of the project explains the format in which input and output will be received and displayed. This is important for the analysis to be done clearly.

### 5.3.1 Input Design

Due to ease of access and security issues, Google Drive was chosen as the storage location for the data. In addition, Google Colab Pro, which offers powerful processing units, was preferred and GPUs and TPUs were accessed. Thus, it was possible to complete very long training sessions in a much shorter time.

Instead of the initially preferred five-emoji dataset, a richer and more useful dataset containing seven emotion tags was preferred. The effect of dataset choice on success was also examined later.

In order for the algorithms to produce the expected outputs, the label(emoji) distribution was shared equally as seen in **Figure 5.5**. 5 emojis, 800 of each emoji, were studied with a total of 4000 data.

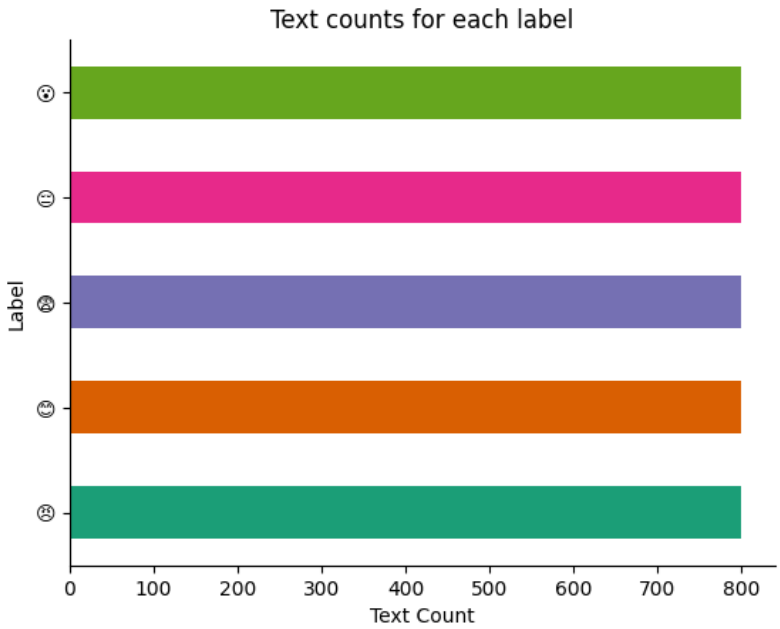


**Figure 5.5** Text Counts

**Figure 5.6** displays the label (emoji) distribution. This figure shows the distribution of the dataset's seven emotion labels, which total 7481. This ensures that the model training is efficient and well-balanced.

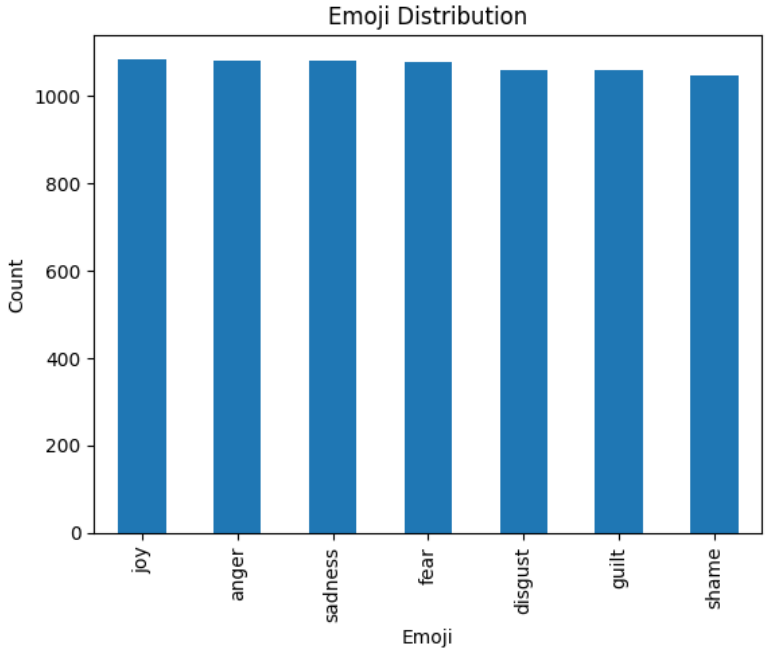


**Figure 5.6** Emoji Distribution Plot

### 5.3.2 Output Design

It is important to see and know how the modules and tools used in the project give output because the project findings depend on them. Therefore, various libraries and visualization tools were used in this output design. For example, the confusion matrix plays a key role here.

The heat map example of the confusion matrix inferred by comparing the predicted values with the actual values is shown in **Figure 5.7**

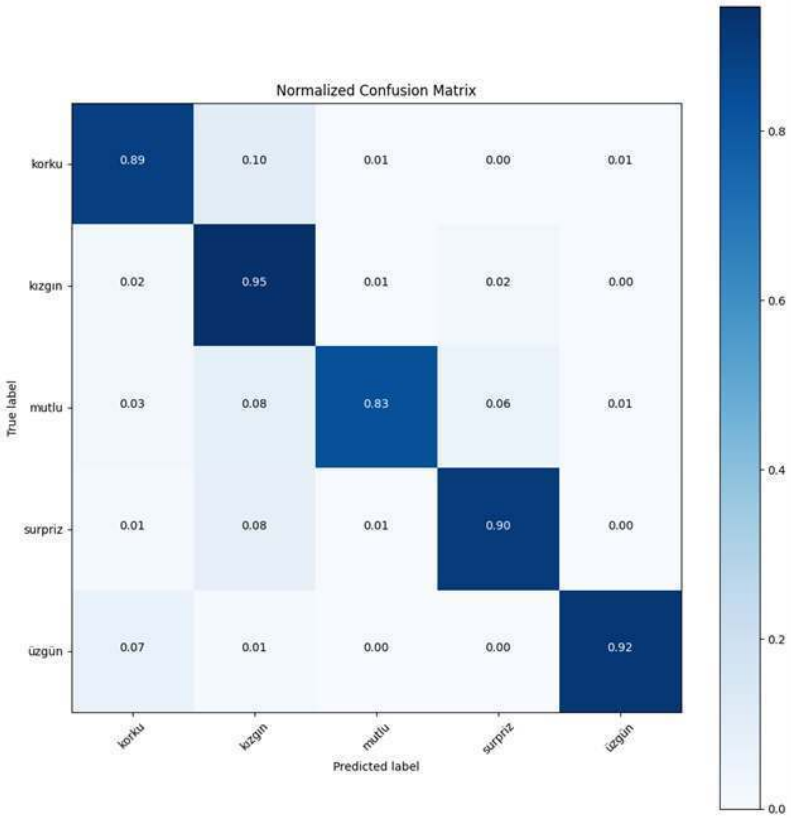


**Figure 5.7** Sample Confusion Matrix

Additionally, an example of a classification report, demonstrating the detailed evaluation metrics, is shown in **Figure 5.8**.

Bidirectional LSTM Report

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| anger | 0.37 | 0.19 | 0.25 | 223 |
| disgust | 0.47 | 0.46 | 0.47 | 200 |
| fear | 0.51 | 0.75 | 0.60 | 217 |
| guilt | 0.50 | 0.52 | 0.51 | 208 |
| joy | 0.69 | 0.69 | 0.69 | 229 |
| sadness | 0.57 | 0.53 | 0.55 | 224 |
| shame | 0.27 | 0.29 | 0.28 | 195 |
| accuracy | | | 0.49 | 1496 |
| macro avg | 0.48 | 0.49 | 0.48 | 1496 |
| weighted avg | 0.49 | 0.49 | 0.48 | 1496 |

**Figure 5.8** Classification Report

# 6
# Implementation

In this section, the previously planned and designed processes are put into code. It includes screenshots of running the code and provides ideas of what the intermediate outputs look like:

## 6.1 Environment Setup

Preparing the development environment plays an important role in accurately measuring success, the necessary Python libraries are installed. including tensorflow, scikit-learn, and transformers. Google Drive connection is established.

## 6.2 Data Collection and Preparations

It goes through a series of preparation stages to put the received data into a format where the model can work. This includes tokenization and vectorization steps. BERT and GPT embeddings are used and are the most basic elements of the data preparation phase.

The **Figure 6.1** illustrates the process of gaining access to the dataset on the drive and storing it as a dataframe.

```
# Access the dataset and load it as a dataframe
drive.mount('/content/drive/')
df = pd.read_excel('/content/drive/MyDrive/Colab Notebooks/Bitirme Projesi/dataset/turkish_dataset.xlsx')
df['Emoji'] = df['Label'].map(emoji_mapping)
df
```

| | Text | Label | Emoji |
|---|---|---|---|
| 0 | Aşık olduğumuz dönemde, her karşılaştığımızda ... | joy | |
| 1 | Bir trafik kazasına karıştığım zaman, | fear | |
| 2 | Birkaç gün süren yoğun çalışmanın ardından eve... | anger | |
| 3 | Benim için en çok anlam ifade eden kişiyi kayb... | sadness | |
| 4 | Bir geyiği devirdiğim zaman, hayvanın yaraları... | disgust | |
| ... | ... | ... | ... |
| 8475 | Karanlıkta yalnız kaldığımda tuhaf sesler duym... | fear | |
| 8476 | Yönetimin aldığı kararlar halkı haklı olarak ö... | anger | |
| 8477 | Kedim kayboldu, çok üzgünüm.. | sadness | |
| 8478 | Çöp kokusuyla dolu bir çöp konteynerini açmak... | disgust | |
| 8479 | Bir hata yaptığını fark ettiğinde içinde derin... | shame | |

8480 rows × 3 columns

**Figure 6.1** Loading Dataset

Data labels were made more readable by displaying each text's label equivalent as an emoji in the dataframe, as shown in **Figure 6.2**.

```
# Define the emoji mapping
emoji_mapping = {
    'joy': emoji.emojize(':smile:', language='alias'),
    'anger': emoji.emojize(':angry:', language='alias'),
    'sadness': emoji.emojize(':cry:', language='alias'),
    'fear': emoji.emojize(':fearful:', language='alias'),
    'disgust': emoji.emojize(':nauseated_face:', language='alias'),
    'guilt': emoji.emojize(':pensive:', language='alias'),
    'shame': emoji.emojize(':flushed:', language='alias')
}
emoji_mapping
```

```
{'joy': '😄',
 'anger': '😠',
 'sadness': '😢',
 'fear': '😨',
 'disgust': '🤢',
 'guilt': '😔',
 'shame': '😳'}
```

**Figure 6.2** Emoji Mapping

The procedure for partitioning the dataset into train and test subsets, together with the corresponding sample sizes, is illustrated in **Figure 6.3**. For the data to be used to make meaningful conclusions, this procedure is crucial and required.

```
# Split the dataset
X_train, X_test, y_train, y_test = train_test_split(X_vectorized, encoded_labels, test_size=0.2, random_state=42)

# Sizes of the splits
print(f'Training set size: {X_train.shape[0]} samples')
print(f'Test set size: {X_test.shape[0]} samples')

# The distribution of labels in training and testing sets
print(f'Training set label distribution: {pd.Series(y_train).value_counts().to_dict()}')
print(f'Test set label distribution: {pd.Series(y_test).value_counts().to_dict()}')
```

```
Training set size: 5984 samples
Test set size: 1496 samples
Training set label distribution: {2: 861, 1: 857, 0: 857, 4: 855, 5: 855, 6: 850, 3: 849}
Test set label distribution: {4: 229, 5: 224, 0: 223, 2: 217, 3: 208, 1: 200, 6: 195}
```

**Figure 6.3** Splitting Dataset

## 6.3 Data Generation

Based on the existing data set for the data set size and ratios, which are among the factors affecting model success, new data suitable for the corresponding labels for the texts was generated. For this, first the OpenAI API connection was established and the gpt-3.5-turbo model was selected as the model. The temperature value was also adjusted according to the dataset and purpose. **Figure 6.4** contains the pertinent code snippet.

```
#Preparing the generator model with proper prompt to generate compatible data
completion = client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages=[
        {"role": "system", "content": "You are a helpful assistant. You will generate a tweet for each prompt."},
        {"role": "user", "content": "Create a new sentence expressing the emotion of 'fear'. Respond in Turkish. Example text: Bir trafik kazasına karıştığım zaman.
    ],
    temperature=0.7,
    max_tokens=100,
    )
```

```
print("Sample generated text for label 'fear':")
completion.choices[0].message.content
```

```
Sample generated text for label 'fear':
Amcam ve komşum polis nezaretinde eve geldiğimde ne yapacağımı bilemedim.
```

**Figure 6.4** Preparing the Generator

The function that takes the text and label columns of the data set as parameters and generates new data according to the appropriate prompt is given in **Figure 6.5**.

```python
# Function to generate data according to the given text and label pairs.
def generate_text_based_on_label(text, label):
    prompt = f"Create a new sentence expressing the emotion of '{label}'. Respond in Turkish. Example text: {text}.\nNew Text:"

    response = client.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=[
            {"role": "system", "content": "You are a helpful assistant. You will generate a tweet for each prompt."},
            {"role": "user", "content": prompt}
        ],
        temperature=0.5,
        max_tokens=150
    )

    new_text = response.choices[0].message.content.strip()
    return new_text
```

**Figure 6.5** Generate Function

A loop was created so that all these functions could run at once, and automatic saving was added without losing data. The loop code and sample outputs are as in **Figure 6.6**.

```python
# Loop through the dataset and save data to excel file.
for i, row in df.iterrows():
    text, label = row['Text'], row['Label']
    new_text = generate_text_based_on_label(text, label)
    generated_texts.append({"Text": new_text, "Label": label})

    # Print the progress
    print(f"Generated {i + 1}/{len(df)}: {new_text}")

    # Save at intervals
    if (i + 1) % save_interval == 0:
        save_generated_texts(generated_texts, save_path)

# Save any remaining texts after the loop
if len(generated_texts) % save_interval != 0:
    save_generated_texts(generated_texts, save_path)
    print(f"All entries processed and saved.")
```

```
Generated 950/1000: Yemekteki çürümüş peynir kokusu midemi bulandırdı.
Data saved to /content/drive/MyDrive/Colab Notebooks/Bitirme Projesi/dataset/generated_texts_openai.xlsx
Generated 951/1000: Geçen hafta iş toplantısında sunum yaparken bir hata yaptım ve tüm patronlarım önünde rezil oldum. O anki u
Generated 952/1000: İş arkadaşımın projede benim yerime geçmesine izin verdim ve şimdi pişmanım.
Generated 953/1000: Yeni doğmuş bir bebek gibi sevinç doluyum! 🌞
Generated 954/1000: Akşam karanlığında tek başıma yürürken aniden bir gürültü duydum ve yüreğim ağzıma geldi. Hızla evime döndü
Generated 955/1000: Birisi bana yalan söylediğinde içimde büyük bir öfke hissettim, bu durum beni gerçekten sinirlendirdi.
Generated 956/1000: Bugün en sevdiğim kitabı bitirdim ve karakterlerin hikayesinin sona erdiğini bilmek beni çok üzdü.
Generated 957/1000: Sinema salonunda otururken, yanımdaki kişi birdenbire ayaklarını çıkarıp koklayarak rahatladı.
Generated 958/1000: Bir iş toplantısında hatamı kabul etmekte zorlandım ve sonunda patronum karşısında utanç içinde hissettim.
Generated 959/1000: Arkadaşımın doğum gününü unutmuş olmam beni çok suçlu hissettirdi.
Generated 960/1000: Uzun bir bekleyişin sonunda sevdiklerinle buluşmak..
Data saved to /content/drive/MyDrive/Colab Notebooks/Bitirme Projesi/dataset/generated_texts_openai.xlsx
```

**Figure 6.6** Generate Loop

Examples of the model that generates text data according to emotion labels inspired by the existing dataset are shown in **Figure 6.7**.

| | generated_tweet | label |
|---|---|---|
| 0 | Ailem neden böyle yapıyor?! Arkadaşlarımla birlikte olmak istiyorum, neden izin vermiyorlar?! | anger |
| 1 | Ailemizin en mutlu günü! 10 yıl sonra ağabeyim ve eşiyle birlikte tekrar bir araya geldik. Mutluluklarımızı paylaşmak için sabırsızlanıyorum! | joy |
| 2 | 200 kişilik bir kalabalığa konuşma yapmak zorunda kalmak... Tarihi bir olayı anlatmak... Kalbim küt küt atıyor, ama korkuyu yenmek için derin bir nefes alıyorum! | fear |
| 3 | 3 hafta önce sadece 1 kez görüştüğüm büyükbabam birkaç ay sonra gitti. Onun gibi iyi bir insanı kaybetmek çok zor. Maddi imkanım olmasaydı yanında kalacak, belki de son anlarını birlikte geçirirdik. Çok üzgünüm | sadness |
| 4 | İlk defa erkek arkadaşımla birlikte evden ayrıldım ve nerede kalacağımız konusunda yalan söyledim. Şimdi vicdan azabı içinde kıvranıyorum. Neden hep böyle yapıyoruz? | guilt |
| 5 | İlk başta tiksinti, sonra üzüntü... Çöp kutularını karıştıran yaşlı adamın hikayesi, sigara izmaritlerinin peşinden giden bir insanı gösteriyor. | disgust |
| 6 | Utandırıcı anılar... 15 yaşında iken annem odamda yasak şeyler buldu; alkol, esrar, doğum kontrol hapları... O günden beri kendimi suçlu hissediyorum. | shame |
| 7 | Mutluluğumdan uçuyorum! Bana başvurduğum Yaz işi teklif eden bir mektup aldığımda, tüm emeklerimin karşılığını aldığımı hissettim! | joy |

**Figure 6.7** Generated Data Examples

Among the language models fine-tuned for the Turkish language and especially specialized for the text generation task, "ytu-ce-cosmos/Turkish-Llama-8b-v0.1" and "turkish-gpt2-large-750m-instruct-v0" are the leading ones. .1" models were also used in the data generation phase and the results were also examined as it is in **Figure 6.8**.

```
r = text_generator(text)
print("Generated sample texts for the label 'anger' (ytu-ce-cosmos/Turkish-Llama-8b-v0.1)")
print(r[0]['generated_text'])
```

```
Generated sample texts for the label 'anger' (ytu-ce-cosmos/Turkish-Llama-8b-v0.1)
1. "Bu adil değil! Her zaman her şeyi doğru yapmam gerekiyor ama diğerleri bunu yapmıyor."
2. "Neden hep benim başıma gelen şeyler oluyor? Neden bu kadar çok kötü şansa maruz kalıyorum?"
3. "Bütün bunlar beni gerçekten sinirlendiriyor ve kızdırıyor; sanki kimse bana yardım etmek istemiyormuş gibi geliyor."
```

**Figure 6.8** Generation Examples Using ytu-ce-cosmos/Turkish-Llama-8b-v0.1 Model

Similar generation operations were performed using Hugging Face's inference API and mistralai/Mistral-7B-Instruct-v0.3 model. The data created with different generation models will be merged into the existing data set, and how it affects success will be examined and compared from different angles.

## 6.4 Model Training

TThe code equivalents of BERT, bi-LSTM, KNN with TF-IDF, and CountVectorizer models are explained in this section. It details how the training setups took place. It is understood which parameters need to be changed to increase model performances.

As seen in **Figure 6.9**, the prediction process of the KNN classifier model has been implemented to yield good classification results using neighborhood relations.

```
# Training the classifier and making predictions
best_k = k_range[np.argmax(scores)]
knn = KNeighborsClassifier(n_neighbors=40)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
```

**Figure 6.9** KNN Classifier

Creating a bi-LSTM neural network model using Keras includes:

- Embedding Layer: Creates dense vectors with a size of 50 from input tokens.
- SpatialDropout1D Layer: In order to avoid overfitting, it applies dropout to the embedding space.
- Bidirectional LSTM Layer: To gather dependencies from both past and future contexts, 128 units with dropout are used.
- Dense Layer: Produces a probability distribution across the potential emoji classes using a softmax activation.

It was observed that bi-LSTM worked much better instead of LSTM to increase emoji prediction accuracy, and further experiments and observations were made on this model. **Figure 6.10** summarizes the model.

```
# LSTM Model building
model = Sequential([
    Embedding(input_dim=10000, output_dim=50, input_length=40),
    SpatialDropout1D(0.2),
    Bidirectional(LSTM(128, dropout=0.2, recurrent_dropout=0.2)),
    Dropout(0.2),
    Dense(len(emoji_mapping), activation='softmax')
])

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 40, 50)            500000

 spatial_dropout1d_1 (Spati  (None, 40, 50)            0
 alDropout1D)

 bidirectional_1 (Bidirecti  (None, 256)               183296
 onal)

 dropout_1 (Dropout)         (None, 256)               0

 dense_1 (Dense)             (None, 7)                 1799

=================================================================
Total params: 685095 (2.61 MB)
Trainable params: 685095 (2.61 MB)
Non-trainable params: 0 (0.00 Byte)
```

**Figure 6.10** LSTM Model Build

## 6.5 Model Evaluation

It gives outputs and makes evaluations within the framework of success metrics so that we can understand how correctly and well the codes executed here work by finally giving an output. It helps to understand the advantages and disadvantages of factors. **Figure 6.11** displays the measurement and visualization of loss and accuracy data during model training.

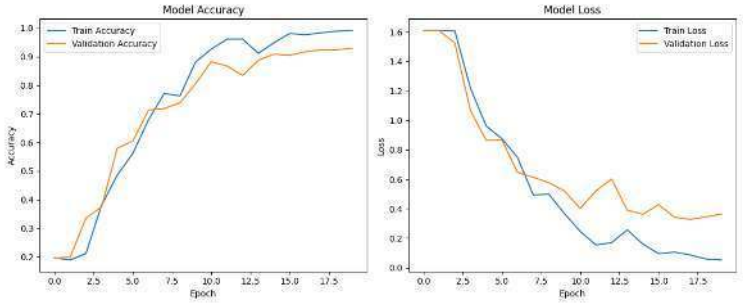


**Figure 6.11** Model Loss and Accuracy Changes Throughout the Epochs

## 6.6 Results Visualization

It gives outputs and makes evaluations within the framework of success metrics so that we can understand how correctly and well the codes executed here work by finally giving an output. It helps to understand the advantages and disadvantages of factors.

The confusion matrix produced as illustrated in **Figure 6.12** is produced by comparing the predicted tag values with the actual tags. The values were made easier to grasp by being displayed as a heat map.

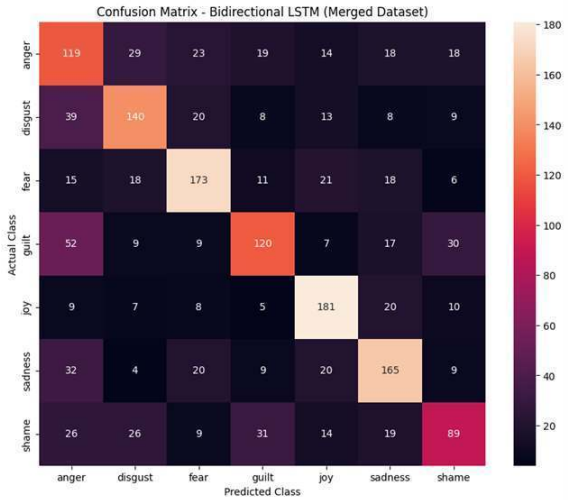


**Figure 6.12** Confusion Matrix for bi-LSTM model

The classification report, which is a thorough outcome report for drawing the required conclusions, analyses, and comparisons of the model, is as follows: **Figure 6.13** with its implementation once the confusion matrix has been created.

```
# Classification Report
from IPython.display import display

report = classification_report(y_test, y_pred, target_names=label_encoder.classes_, output_dict=True)
report_df = pd.DataFrame(report).transpose()


print("Classification Report")
display(report_df)
```

Classification Report - With TfidfVectorizer

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| anger | 0.410526 | 0.349776 | 0.377724 | 223.000000 |
| disgust | 0.476190 | 0.550000 | 0.510441 | 200.000000 |
| fear | 0.689655 | 0.552995 | 0.613811 | 217.000000 |
| guilt | 0.757895 | 0.346154 | 0.475248 | 208.000000 |
| joy | 0.463918 | 0.786026 | 0.583468 | 229.000000 |
| sadness | 0.428986 | 0.860714 | 0.520211 | 224.000000 |
| shame | 0.534247 | 0.200000 | 0.291045 | 195.000000 |
| accuracy | 0.499332 | 0.499332 | 0.499332 | 0.499332 |
| macro avg | 0.537345 | 0.492238 | 0.481707 | 1496.000000 |
| weighted avg | 0.535154 | 0.499332 | 0.484802 | 1496.000000 |

**Figure 6.13** Classification Report

The model's predictions against the entered text samples are shown in **Figure 6.14** below.

```
sample_texts = [
    "Az önce harika bir haber aldım ve çok sevindim!",
    "Son zamanlarda kendimi çok kötü hissediyorum.",
    "İnsanların zamanıma saygı göstermemesi beni çok kızdırıyor.",
    "Uçağa binmekten korkuyorum.",
    "Korkuyorum ama nedense üzgün hissediyorum, hüzün üzgün, üzülüyorum"
]

for text in sample_texts:
    result = classifier(text)[0]['label']
    predicted_emoji = emoji_label_mappings[result]
    print(f"Sample text: '{text}'")
    print(f"Predicted emoji: {predicted_emoji} (Model Label: {result})\n")
```

```
{'LABEL_0': '😊', 'LABEL_1': '😔', 'LABEL_2': '😠', 'LABEL_3': '😨', 'LABEL_4': '😜'}
Sample text: 'Az önce harika bir haber aldım ve çok sevindim!'
Predicted emoji: 😊 (Model Label: LABEL_0)

0.9999473094940186
Sample text: 'Son zamanlarda kendimi çok kötü hissediyorum.'
Predicted emoji: 😔 (Model Label: LABEL_1)

0.9998984336853027
Sample text: 'İnsanların zamanıma saygı göstermemesi beni çok kızdırıyor.'
Predicted emoji: 😠 (Model Label: LABEL_2)

0.9999550580978394
Sample text: 'Uçağa binmekten korkuyorum.'
Predicted emoji: 😨 (Model Label: LABEL_3)

0.9999651908874512
Sample text: 'Korkuyorum ama nedense üzgün hissediyorum, hüzün üzgün, üzülüyorum'
Predicted emoji: 😔 (Model Label: LABEL_1)

0.8653164505958557
```

**Figure 6.14** Predicting Results

# 7
# Exprimental Results

The results of emoji predictions as a result of different scenarios and configurations are shown in this section. Here, by carrying out different combinations and providing a control group, the net effect of the specific factor examined on success is observed.

## 7.1 Data Generation and Its Impact

The impact of data generation on this project was truly significant; Its success was measured by adding it separately to the existing training data set. For accurate comparison, the initial dataset and the test dataset in the merged datasets were kept separate and common. Therefore, in three separate cases, the effects of data generation were observed on the same test set. This comparative analysis was done on the bert-base-turkish-128k-uncased model, which is based on BERT and has a higher success rate than others.

The inclusion of generated data resulted in a significant improvement in the model's predictive capabilities, as shown in the classification reports in **Figure 7.1**, **Figure 7.2** and **Figure 7.3**.

BERT Classification Report for initial dataset:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| sadness | 0.67 | 0.67 | 0.67 | 224 |
| fear | 0.67 | 0.75 | 0.71 | 217 |
| joy | 0.78 | 0.89 | 0.83 | 229 |
| shame | 0.67 | 0.48 | 0.56 | 195 |
| guilt | 0.64 | 0.48 | 0.55 | 208 |
| disgust | 0.64 | 0.61 | 0.63 | 200 |
| anger | 0.48 | 0.61 | 0.53 | 223 |
| accuracy | | | 0.65 | 1496 |
| macro avg | 0.65 | 0.64 | 0.64 | 1496 |
| weighted avg | 0.65 | 0.65 | 0.64 | 1496 |

**Figure 7.1** Classification Report for Initial Dataset

BERT Classification Report for merged dataset (gpt-3.5-turbo)

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| sadness | 0.71 | 0.70 | 0.70 | 224 |
| fear | 0.79 | 0.74 | 0.76 | 217 |
| joy | 0.89 | 0.85 | 0.87 | 229 |
| shame | 0.50 | 0.66 | 0.57 | 195 |
| guilt | 0.68 | 0.63 | 0.65 | 208 |
| disgust | 0.66 | 0.78 | 0.72 | 200 |
| anger | 0.65 | 0.49 | 0.56 | 223 |
| accuracy | | | 0.69 | 1496 |
| macro avg | 0.70 | 0.69 | 0.69 | 1496 |
| weighted avg | 0.70 | 0.69 | 0.69 | 1496 |

**Figure 7.2** Classification Report for Merged Dataset with gpt-3.5-turbo Generations

BERT Classification Report for merged dataset (Mistral)

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| sadness | 0.75 | 0.69 | 0.72 | 224 |
| fear | 0.77 | 0.77 | 0.77 | 217 |
| joy | 0.88 | 0.89 | 0.88 | 229 |
| shame | 0.63 | 0.60 | 0.61 | 195 |
| guilt | 0.64 | 0.70 | 0.67 | 208 |
| disgust | 0.73 | 0.70 | 0.71 | 200 |
| anger | 0.63 | 0.66 | 0.64 | 223 |
| accuracy | | | 0.72 | 1496 |
| macro avg | 0.72 | 0.72 | 0.72 | 1496 |
| weighted avg | 0.72 | 0.72 | 0.72 | 1496 |

**Figure 7.3** Classification Report for Merged Dataset with Mistral-7B-Instruct-v0.3 Generations

### 7.1.1 Comparative Analysis

1. Initial Dataset

   The 'Joy' and 'Fear' categories stand out with particularly high precision and recall values, indicating that the model can reliably predict these emotions. Lower values are seen in the 'Shame' and 'Guilt' categories, which indicates that the model needs improvement in these categories.

2. Merged Dataset (Mistral-7B-Instruct)

   Significant improvements are seen in all categories. There are especially significant improvements in the 'Sadness', 'Fear' and Disgust' categories. This shows that when the Mistral model is used for data augmentation, the model becomes more reliable and widely applicable.

3. Merged Dataset (GPT-3.5-turbo)

   'Joy' and 'Fear' categories again stand out with their high precision and recall values. Lower performance was observed in the 'Anger' and 'Shame' categories. This shows that data produced with GPT-3.5-turbo may introduce a certain variability in the model.

Comparative analysis reveals that data augmentation is more successful than existing data and increases generalization ability. Both the Mistral model and the gpt-3.5-turbo model increased and improved the average success and accuracy.

## 7.2 Impact of Increased Emotion Labels

The efficacy of the model was assessed on datasets including varying quantities of emotion labels. The model was trained using two different datasets: one with five emotion labels and the other with seven. Below are **Figure 7.4 and Figure 7.5** the confusion matrices for both configurations.

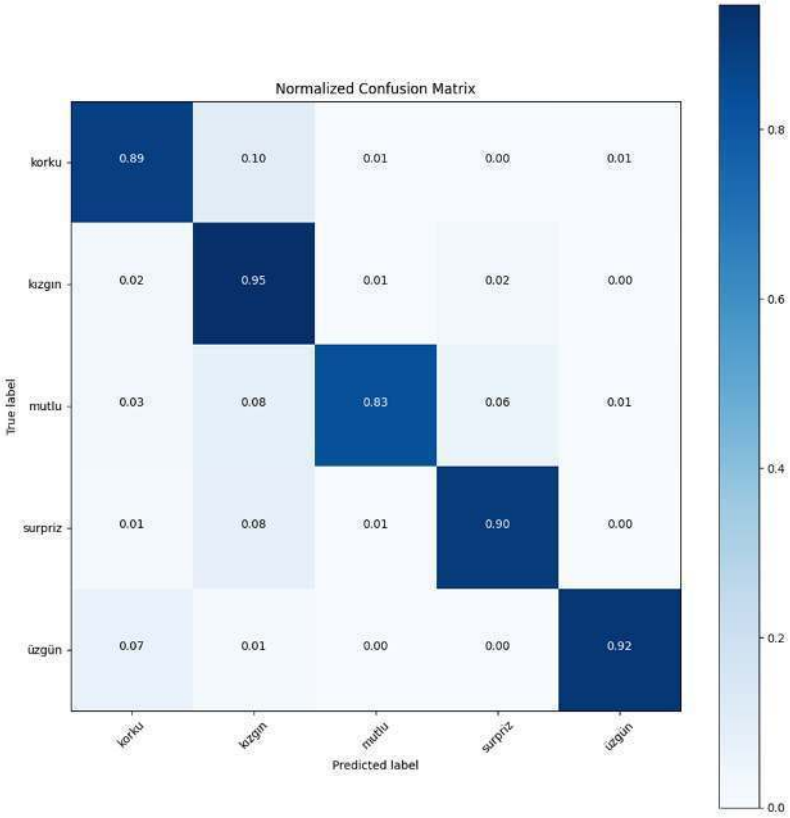


**Figure 7.4** Confusion Matrix for 5 Emotion Labels

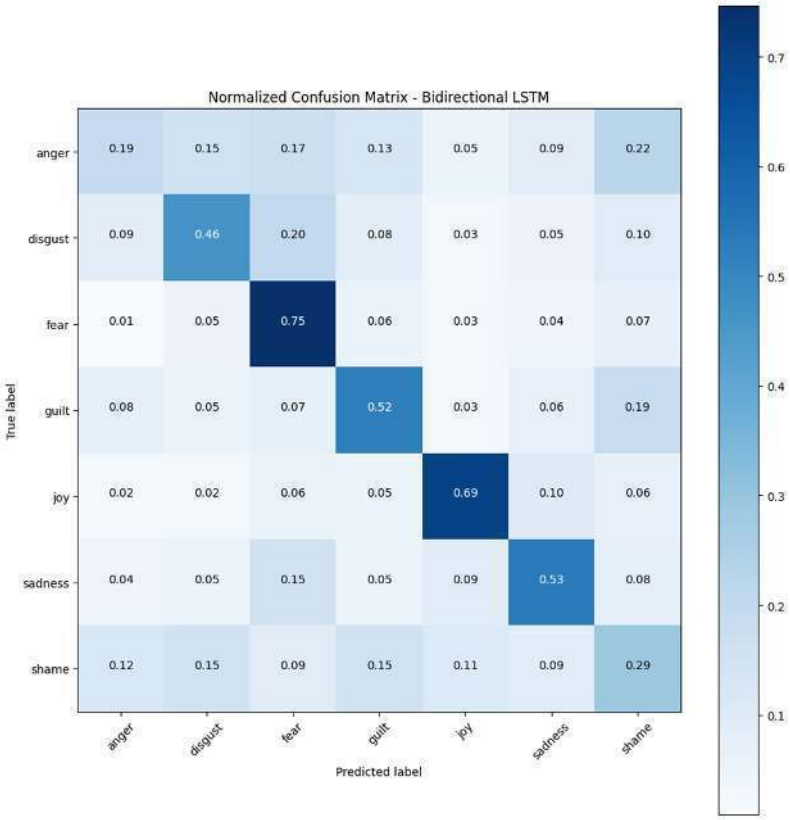


**Figure 7.5** Confusion Matrix for 7 Emotion Labels

## 7.3 Comparison of KNN Performance with TF-IDF and CountVectorizer

Comparison of TF-IDF and CountVectorizer methods was performed via KNN. This comparison using the same dataset showed clear differences

1. TF-IDF Vectorization

   TF-IDF is generally used to convert text data into numerical attributes, where the relevance of terms is measured and given importance.

   The classification report presents the results, which indicate that the majority of the labels produce relatively superior precision, recall, and F1-scores when employing TF-IDF. The 'joy' label, on the other hand, displays a recall of 0.78 and an F1-score of 0.58, whilst the 'fear' label obtains a precision of 0.69 and an F1-score of 0.61. TF-IDF's total accuracy is approximately 0.50, as **Figure 7.6** illustrates.

Classification Report - With TfidfVectorizer

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| anger | 0.410526 | 0.349776 | 0.377724 | 223.000000 |
| disgust | 0.476190 | 0.550000 | 0.510441 | 200.000000 |
| fear | 0.689655 | 0.552995 | 0.613811 | 217.000000 |
| guilt | 0.757895 | 0.346154 | 0.475248 | 208.000000 |
| joy | 0.463918 | 0.786026 | 0.583468 | 229.000000 |
| sadness | 0.428986 | 0.660714 | 0.520211 | 224.000000 |
| shame | 0.534247 | 0.200000 | 0.291045 | 195.000000 |
| accuracy | 0.499332 | 0.499332 | 0.499332 | 0.499332 |
| macro avg | 0.537345 | 0.492238 | 0.481707 | 1496.000000 |
| weighted avg | 0.535154 | 0.499332 | 0.484802 | 1496.000000 |

**Figure 7.6** Classification Report for KNN with TF-IDF Vectorizer

2. CountVectorizer

   Unlike TF-IDF, CountVectorizer creates a frequency-based structure by simply counting the number of each word in the document. It obviously performs worse than TF-IDF in the classification report. The 'fear' label has an accuracy of 0.22 and an F1-score of 0.27, which are much lower. Comparably, the 'joy' label exhibits less successful classification with a recall of 0.39 and an F1-score of 0.37. With CountVectorizer, the overall accuracy is roughly 0.27. The result is shown in **Figure 7.7**.

Classification Report - KNN (CountVectorizer):

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| anger | 0.318841 | 0.098655 | 0.150685 | 223.000000 |
| disgust | 0.216285 | 0.425000 | 0.286678 | 200.000000 |
| fear | 0.218750 | 0.354839 | 0.270650 | 217.000000 |
| guilt | 0.785714 | 0.105769 | 0.186441 | 208.000000 |
| joy | 0.348837 | 0.393013 | 0.369610 | 229.000000 |
| sadness | 0.281654 | 0.486607 | 0.356792 | 224.000000 |
| shame | 0.222222 | 0.010256 | 0.019608 | 195.000000 |
| accuracy | 0.272059 | 0.272059 | 0.272059 | 0.272059 |
| macro avg | 0.341758 | 0.267734 | 0.234352 | 1496.000000 |
| weighted avg | 0.341954 | 0.272059 | 0.238526 | 1496.000000 |

**Figure 7.7** Classification Report for KNN with Count Vectorizer

Upon closer comparison, the TF-IDF vectorization method performed much better than CountVectorizer. TF-IDF measured the values of words in context better and was thus more successful

## 7.4 Comparison of the Models

Analyzes of three different methods, whose success was measured with the same vectorization methods on the same data set, were performed.

1. **bi-LSTM**

   The model performed well in the 'Joy' (Precision: 0.69, Recall: 0.69) and 'Fear' (Precision: 0.51, Recall: 0.75) categories. It performed poorly in the 'Anger' (Precision: 0.37, Recall: 0.19) and 'Shame' (Precision: 0.27, Recall: 0.29) categories.

   While the Bi-LSTM model exhibits high performance in some categories, it has difficulties especially in the 'Anger' and 'Shame' categories. Overall, this is a reliable model for the dataset, but improvement is needed in some categories. The report is shown in **Figure 7.8**

Bidirectional LSTM Report

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| anger | 0.37 | 0.19 | 0.25 | 223 |
| disgust | 0.47 | 0.46 | 0.47 | 200 |
| fear | 0.51 | 0.75 | 0.60 | 217 |
| guilt | 0.50 | 0.52 | 0.51 | 208 |
| joy | 0.69 | 0.69 | 0.69 | 229 |
| sadness | 0.57 | 0.53 | 0.55 | 224 |
| shame | 0.27 | 0.29 | 0.28 | 195 |
| accuracy | | | 0.49 | 1496 |
| macro avg | 0.48 | 0.49 | 0.48 | 1496 |
| weighted avg | 0.49 | 0.49 | 0.48 | 1496 |

**Figure 7.8** Classification Report for bi-LSTM

2. **Multinomial Naive Bayes**

   It showed high performance in the 'Joy' (Precision: 0.67, Recall: 0.74) and 'Fear' (Precision: 0.71, Recall: 0.68) categories. It performed relatively poorly in the 'Anger' (Precision: 0.48, Recall: 0.44) and 'Shame' (Precision: 0.45, Recall: 0.44) categories. The Multinomial Naive Bayes model achieved generally high accuracy and F1 scores. However, there is confusion between the categories 'Disgust' and 'Anger'. This model is a reliable choice for the dataset. Classification report for Multinomial Naive Bayes is as **Figure 7.9**

Multinomial NB Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| anger | 0.48 | 0.44 | 0.46 | 223 |
| disgust | 0.62 | 0.53 | 0.57 | 200 |
| fear | 0.71 | 0.68 | 0.69 | 217 |
| guilt | 0.47 | 0.61 | 0.53 | 208 |
| joy | 0.67 | 0.74 | 0.70 | 229 |
| sadness | 0.65 | 0.58 | 0.61 | 224 |
| shame | 0.45 | 0.44 | 0.44 | 195 |
| accuracy | | | 0.58 | 1496 |
| macro avg | 0.58 | 0.57 | 0.57 | 1496 |
| weighted avg | 0.58 | 0.58 | 0.58 | 1496 |

**Figure 7.9** Classification Report for Multinomial Naive Bayes

3. **K-Nearest Neighbors (KNN)**

   It showed high performance in the 'Joy' (Precision: 0.46, Recall: 0.79) and 'Fear' (Precision: 0.69, Recall: 0.55) categories. It performed poorly in the 'Anger' (Precision: 0.41, Recall: 0.35) and 'Shame' (Precision: 0.53, Recall: 0.20)

categories. The KNN model performed quite well in the 'Joy' category. However, there is a memory weakness in the 'Fear' and 'Guilt' categories. Overall, it is a preferable model for this dataset, but it has a high misclassification rate and is shown in **Figure 7.10**

Classification Report - KNN

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| anger | 0.410526 | 0.349776 | 0.377724 | 223.000000 |
| disgust | 0.476190 | 0.550000 | 0.510441 | 200.000000 |
| fear | 0.689655 | 0.552995 | 0.613811 | 217.000000 |
| guilt | 0.757895 | 0.346154 | 0.475248 | 208.000000 |
| joy | 0.463918 | 0.786026 | 0.583468 | 229.000000 |
| sadness | 0.428986 | 0.660714 | 0.520211 | 224.000000 |
| shame | 0.534247 | 0.200000 | 0.291045 | 195.000000 |
| accuracy | 0.499332 | 0.499332 | 0.499332 | 0.499332 |
| macro avg | 0.537345 | 0.492238 | 0.481707 | 1496.000000 |
| weighted avg | 0.535154 | 0.499332 | 0.484802 | 1496.000000 |

**Figure 7.10** Classification Report for KNN

Bi-LSTM: Although the model is strong in 'Joy' and 'Sadness' categories, it fails in the categories of 'Fear' and 'Guilt'. Generally, this is a dependable model for the dataset but there are possibilities of improvement.

Multinomial Naive Bayes: By and large, the accuracy and F1 scores were quite high with this model. However, these two groups are confused for each other 'Disgust' versus 'Anger'. This choice also applies to the data set.

KNN: As for KNN model, it worked good with "Joy" category, but turned out to be bad for "Fear" and "Guilt". Firstly it is generally favored by most models but however they have higher mis-classification rates.

## 7.5 BERT Model Performance Analysis

In comparison to other models, the BERT model—more precisely, the BERTurk version—performed better on emoji prediction tests. It fared better than models like

KNN, Naive Bayes, and LSTM, achieving an accuracy of roughly 72% along with F1 Score, Precision, and Recall metrics around 72%, as it is demonstrated in **Figure 7.11**

BERT Classification Report

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| sadness | 0.75 | 0.69 | 0.72 | 224 |
| fear | 0.77 | 0.77 | 0.77 | 217 |
| joy | 0.88 | 0.89 | 0.88 | 229 |
| shame | 0.63 | 0.60 | 0.61 | 195 |
| guilt | 0.64 | 0.70 | 0.67 | 208 |
| disgust | 0.73 | 0.70 | 0.71 | 200 |
| anger | 0.63 | 0.66 | 0.64 | 223 |
| accuracy | | | 0.72 | 1496 |
| macro avg | 0.72 | 0.72 | 0.72 | 1496 |
| weighted avg | 0.72 | 0.72 | 0.72 | 1496 |

**Figure 7.11** Classification Report for BERT

**Reasons for Superior Performance of BERT**

1. Contextual Understanding

   Thanks to its bidirectional structure, one can better understand the meaning of a word in context by examining it before and after.

2. Pretraining on Large Turkish Corpus

   Since it has received extensive preliminary training on Turkish data, it does not overlook the points specific to the language.

3. Transformer Architecture

   It better evaluates the values of the word in the number of transfarmator structure and increases its sensitivity.

The summarized result comparison table of the compared models is as the **Table 7.1** below.

**Table 7.1** Model Comparisons Table

| **Model** | **Accuracy** |
|---|---|
| Bidirectional LSTM | %49 |
| Multinomial Naive Bayes | %58 |
| K-Nearest Neighbors | %49 |
| BERT | %72 |

# 8
# Performance Analysis

All these processes and experimental results are evaluated within the scope of project analysis and the efficiency of the system is examined. It involves examining experimental findings in more detail.

## 8.1 Performance Testing Methods

The primary methods used to evaluate the system's performance include:

1. Accuracy, Precision, Recall, and F1 Score:

   These metrics provide a comprehensive view of the model's performance across different classes.

2. Confusion Matrices:

   Visual tools to understand the model's predictions, showing the true versus predicted labels.

3. Data Generation Impact:

   Assessing how additional generated data influences model performance.

### 8.1.1 Robustness Analysis

The robustness and consistency of the model were calculated with the changes made to the data sets during the research process.

### 8.1.2 Scalability Considerations

As the dataset grows, scalability concerns arise. This is important to measure and take into account. For example, in this study, one can get an idea about the measurability of the model by looking at how different models are affected by the size of the data set.

# 9
# Conclusion

## 9.1 Project Overview and Methods

As a result of these study plans, it seems that the aim of inferring emoji predictions from Turkish texts by using natural language processing techniques has been achieved. However, on the way to this result, it was learned which model could work better in which scenarios. Other actions are as follows:

1. **Data Collection and Preperation:**

   Using datasets from Hugging Face and the PsychExp benchmark, text data was prepared through tokenization and vectorization.

2. **Data Generation:**

   Generating additional text data using Hugging Face and OpenAI APIs to enrich the training dataset.

3. **Model Training and Evaluation:**

   Implementing and evaluating multiple models such as BERTurk, LSTM, Multinomial Naive Bayes, and KNN with various vectorization techniques.

## 9.2 Results

The experimental findings showed how several strategies affected the accuracy of emoji predictions:

- Data Generation: The performance of the model used improved greatly, especially with the addition of the dataset generated from the Mistral model, and the F1 score increased from 0.65 to 0.72.

- Vectorization Techniques: As a result of various experiments applied on the KNN model, it was concluded that the generalization and success ability of the TF-IDF method was by far better than the CountVectorizer.

- Model Comparisons:

  In conclusion, the comparison of various models for emoji prediction shows that BERTurk is better than others with a weighted F1-score of 0.72 and is characterized by high accuracy making it the most efficient model for this purpose. In addition, Bi-LSTM model also performs well especially on 'joy' and 'sadness' predictions though it seems to fail in categories such as 'anger' and 'shame'. Multinomial Naive Bayes (MNB), while having a decent overall F1-score, has difficulty distinguishing between certain emotions like 'disgust' from 'anger'. Lastly, KNN produces good results in predicting happiness but its high misclassification rates for instance in fear and guilt categories indicate the need for improvement. All in all BERTurk appears to be the strongest model when it comes to accurate prediction of emoticons in various emotional expressions. .

## 9.3 Achievements and Limitations

### 9.3.1 Achivements

1. High Accuracy: Thanks to its advanced structure and pre-training, BERT-based BERTurk has clearly stood out and is ready to be used and further perfected in emoji prediction tasks.

2. Effective Data Augmentation: Although the data produced adhered to the existing dataset, they did not deviate from the emoji labels, and the data it created allowed more successful results to be obtained compared to the existing dataset.

3. Vectorization Insights: The comparison between TF-IDF and CountVectorizer highlighted the importance of selecting appropriate vectorization techniques for better model performance.

### 9.3.2 Limitations

1. Model Variability: The large number of classes and the size of the data set challenges the discrimination power of models such as KNN and Naive Bayes.

2. Complexity of Emotions: As predicted, the increase in the number of classes affected the success of the models in general. However, all of them made a significant inference and gave good results.

## 9.4 Recommendations for Future Work

For those interested in advancing this field, the following suggestions are made:

1. **Further Data Augmentation:** To see how other data augmentation methods affect datasets and measure the success of machine learning algorithms run on them.
2. **Advanced Modeling Techniques:** Relatively more complex models, including transformers, can be preferred to increase performance
3. **Balanced Datasets:** Make sure classes and data are balanced so that class imbalance doesn't negatively impact the performance of models

As a result, this study gave ideas about how natural language processing methods and data generation methods can affect and improve the ability to predict emoji from Turkish texts. It provided the basis for comparative analysis for future research in this field.

# References


[1] M. O. Nusrat, Z. Habib, M. Alam, and S. A. Jamal, *Emoji prediction in tweets using bert*, 2023. arXiv: 2307.02054 [cs.CL].

[2] S. Jin and T. Pedersen, "Duluth UROP at SemEval-2018 task 2: Multilingual emoji prediction with ensemble learning and oversampling," in *Proceedings of the 12th International Workshop on Semantic Evaluation*, M. Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, and M. Carpuat, Eds., New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 482–485. DOI: 10.18653/v1/S18-1077. [Online]. Available: https://aclanthology.org/S18-1077.

[3] L. Zhou, Q. Xu, H. Suominen, and T. Gedeon, "EPUTION at SemEval-2018 task 2: Emoji prediction with user adaption," in *Proceedings of the 12th International Workshop on Semantic Evaluation*, M. Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, and M. Carpuat, Eds., New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 449–453. DOI: 10.18653/v1/S18-1071. [Online]. Available: https://aclanthology.org/S18-1071.

[4] J. Coster, R. G. van Dalen, and N. A. J. Stierman, "Hatching chick at SemEval-2018 task 2: Multilingual emoji prediction," in *Proceedings of the 12th International Workshop on Semantic Evaluation*, M. Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, and M. Carpuat, Eds., New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 445–448. DOI: 10.18653/v1/S18-1070. [Online]. Available: https://aclanthology.org/S18-1070.

[5] C. Baziotis, A. Nikolaos, A. Kolovou, G. Paraskevopoulos, N. Ellinas, and A. Potamianos, "NTUA-SLP at SemEval-2018 task 2: Predicting emojis using RNNs with context-aware attention," in *Proceedings of the 12th International Workshop on Semantic Evaluation*, M. Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, and M. Carpuat, Eds., New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 438–444. DOI: 10.18653/v1/S18-1069. [Online]. Available: https://aclanthology.org/S18-1069.

[6] L. Alexa, A. Lorenț, D. Gıfu, and D. Trandabăț, "The dabblers at SemEval-2018 task 2: Multilingual emoji prediction," in *Proceedings of the 12th International Workshop on Semantic Evaluation*, M. Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, and M. Carpuat, Eds., New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 405–409. DOI: 10.18653/v1/S18-1062. [Online]. Available: https://aclanthology.org/S18-1062.

[7] F. Barbieri *et al.*, “SemEval 2018 task 2: Multilingual emoji prediction,” in *Proceedings of the 12th International Workshop on Semantic Evaluation*, M. Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, and M. Carpuat, Eds., New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 24–33. DOI: 10.18653/v1/S18-1003. [Online]. Available: https://aclanthology.org/S18-1003.

[8] Ç. Çöltekin and T. Rama, “Tübingen-Oslo at SemEval-2018 task 2: SVMs perform better than RNNs in emoji prediction,” in *Proceedings of the 12th International Workshop on Semantic Evaluation*, M. Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, and M. Carpuat, Eds., New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 34–38. DOI: 10.18653/v1/S18-1004. [Online]. Available: https://aclanthology.org/S18-1004.

[9] İ. Sel and D. Hanbay, “Ön eğitimli dil modelleri kullanarak türkçe tweetlerden cinsiyet tespiti,” *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, vol. 33, no. 2, pp. 675–684, 2021. DOI: 10.35234/fumbd.929133.

[10] Z. "Wang and T. Pedersen, “"UMDSub at SemEval-2018 task 2: Multilingual emoji prediction multi-channel convolutional neural network on subword embedding",” in "*Proceedings of the 12th International Workshop on Semantic Evaluation*", M. "Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, and M. Carpuat, Eds., "New Orleans, Louisiana": "Association for Computational Linguistics", Jun. "2018", "395–399". DOI: "10.18653/v1/S18-1060". [Online]. Available: https://aclanthology.org/S18-1060.

[11] N. Wang, J. Wang, and X. Zhang, “YNU-HPCC at SemEval-2018 task 2: Multi-ensemble Bi-GRU model with attention mechanism for multilingual emoji prediction,” in *Proceedings of the 12th International Workshop on Semantic Evaluation*, M. Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, and M. Carpuat, Eds., New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 459–465. DOI: 10.18653/v1/S18-1073. [Online]. Available: https://aclanthology.org/S18-1073.

[12] E. AI, *Classification metrics - confusion matrix*, https://www.evidentlyai.com/, June 4, 2024. [Online]. Available: https://www.evidentlyai.com/classification-metrics/confusion-matrix.

[13] M. Publications, *Precision*, Manning LiveBook [Online]. Available from: https://livebook.manning.com/concept/nlp/precision. [Online]. Available: https://livebook.manning.com/concept/nlp/precision.

# Curriculum Vitae

## FIRST MEMBER

**Name-Surname:** Bedrettin Şamil Öztürk
**Birthdate and Place of Birth:** 04.10.1999, İstanbul
**E-mail:** samil.ozturk@std.yildiz.edu.tr
**Phone:** 0536 566 00 00
**Practical Training:** TÜBİTAK - Front End Developer Intern
Uyumsoft - ASP.NET Web Developer Intern

## Project System Informations

**System and Software:** Windows İşletim Sistemi, Python
**Required RAM:** 16GB
**Required Disk:** 256MB