Emojis For Sentiment Analysis

Muhammet Ali ŞEN, Muhammet Kayra BULUT
Bilgisayar Mühendisliği Bölümü
Yıldız Teknik Üniversitesi, 34220 Istanbul, Türkiye
{ali. sen, kayra. bulut}@std. yildiz. edu. tr

***Özetçe* —Bu projenin amacı doğal dil işleme aşamalarından anlamsal analiz olarak bir duygu analizi çalışmasıdır. Twitter platformu üzerinden toplanan ve tek emoji içeren veri seti üzerinden çeşitli vektörizasyon, örnekleme yöntemlerle makine öğrenmesi ve derin öğrenme yöntemleri karşılaştırılarak emoji tahmin skorları başarım oranları araştırılmıştır. Sonuç olarak verilen cümlenin emojisini tahmin etmektir.**

Anahtar Kelimeler—Doğal Dil İşleme (NLP) , Makine Öğrenmesi (ML) , Naive Bayes, Destek Vektör Makineleri (SVM) , K-En Yakın Komşu (KNN) , Derin Öğrenme (DL) , Kısa-Uzun Süreli Bellek (LSTM) , Emoji Tahmini, Yapay Sinir Ağları, Duygu Analizi.

***Abstract*—In summary, the project is using tweets containing a single emoji as a dataset to perform sentiment analysis, which is a stage of natural language processing. It will be comparing different methods of vectorization and sampling, using both machine learning and deep learning techniques, to evaluate the success rates of emoji prediction. The ultimate goal is to predict the appropriate emoji for a given sentence.**

Keywords—Natural Language Processing (NLP), Machine Learnig(ML), Emoji Prediction, Sentiment Analysis

I. Introduction

Here, we will provide a general overview of the project.

In recent years, social media has become an integral part of our daily lives, providing us with a platform to connect with others and share our thoughts, feelings, and opinions on various topics. There are numerous social networking platforms such as Twitter, Facebook, Instagram, and WhatsApp that have been created to facilitate communication and connection between individuals. Despite their differences in terms of features and capabilities, these platforms all share one common aspect: the use of emojis. Emojis, which include everything from facial expressions to animals, objects, and places, are widely used to communicate simple ideas and to enhance the emotions and feelings conveyed in our messages. However, the meaning of emojis is not always the same, and their use can vary depending on the context and culture. As a result, processing emojis remains a significant challenge for researchers in the field of natural language processing (NLP). In order to improve the effectiveness of NLP models in interpreting and generating emojis, researchers are constantly working on developing new techniques and algorithms that can better understand the nuances of this visual form of communication. [1].

Furthermore, emojis also play a crucial role in understanding the sentiment and emotion behind text-based communication on social media platforms. Emojis can convey a wide range of emotions, from happiness and excitement to sadness and anger, and can also be used to add sarcasm or irony to a message. Understanding the sentiment and emotion behind text-based communication is crucial in various applications such as sentiment analysis, customer service, and marketing. However, emojis can also introduce ambiguity and subjectivity, making it more difficult to determine the sentiment and emotion behind a message.

In addition to sentiment analysis, emojis also play a significant role in other NLP tasks such as text classification, machine translation, and text generation. Emojis can be used to add context, enhance the expressiveness of text, and make the text more engaging. However, the use of emojis also makes these tasks more challenging, as the meaning and context of emojis can vary widely.

Overall, emojis have become an essential part of communication on social media platforms, and their use has brought new challenges for NLP researchers. While understanding and processing emojis can be difficult, research in this area continues to evolve, and new techniques and algorithms are being developed to better understand the nuances of this visual form of communication.

II. Related Work

Here we will talk about related studies.

Social network analysis, which has been the subject of many scientific studies in recent years, is one of the most interesting topics for NLP researchers. Many studies have been conducted in this area. The Semeval data has been used as the data set for many of these studies on sentiment analysis or emoji prediction [2].

According to the article written by the team that collected the data set, SVM has been emphasized to make more accurate inferences compared to other CNN and LSTM methods [3].

The study by Cagri Coltekin and Taraka Rama, working on the same data set, was titled "SVMs perform better than RNNs at Emoji Prediction" and concluded that SVM makes more accurate calculations than RNN methods [4]. Another study on the same data set found that Naive Bayes produced better results than RNN algorithms [5].

In addition to these, there are also studies that emphasize that MNB results produce significantly better results compared to complex deep learning systems [6] and that a series of pre-studies are needed for RNN methods

to achieve higher success and that complex layered deep learning architectures are modeled [7].

Our project results have been compared to the results of many projects using the same data set, and similar results have been obtained. Although it was surprising that the success of a DL method like LSTM was lower than the success of an ML method like SVM, the fact that it achieved similar results to all projects motivated us to investigate which methods can be used to achieve more successful results in this study.

III. Data Preprocessing

Here, we will discuss the data processing steps.

A. Data Set

Our data set is the Semeval 2018 Task-2 [1] set, which is commonly used as a data set in scientific studies, especially in NLP studies.

As can be seen, the data is not synthetic and consists of natural, non-artificial data. It has been quite difficult to work with data that is difficult for people who are not familiar with the target audience of the tweets to understand or analyze, and correctly transfer this data to our model. The cleaned version of the data set is shown in Figure 1.

TEMİZLENMİŞ	TEMİZLENMEMİŞ
stop drinkin saint amp second,7	Can't stop drinkin' about you @ Saint & Second,7
ought ummmmm favorite season,0	"@OneRepublic Ought. Ummmmm favorite season,0
happy b day mickey,0	#mickeymouse #88thbirthday #vivaorlando #waltdisneyworld Happy B-Day Mickey!!!!!! You!"i...,0
farmers branch historical park,17	s e a s o n s g r e e t i n g s @ Farmers Branch Historical Park,17
myfavwww great lakes mall,13	myfavwww @ Great Lakes Mall,13
baaaaaaack fayetteville arkansas,6	we're baaaaaaack @ Fayetteville, Arkansas,6
w e bmw store,10	/// W ! D E /// : @user #RVGXOR #bmtrsoloyeyu #m3 #f80 #Conduktco @ The BMW Store,10
say point ahhh lol day e u,2	All you can say at this point is "Ahhh" lol #CubanCoffee #ELlibre #cafecito / Day 3: S.E U.S...,2
bllllllliard moonshine pipe company,1	Bllllllliard #copper #prohibition #moonshine #comingsoon @ Moonshine Pipe Company,1
looooove making new friends justicecailey gulf shores alabama,9	I looooove making new friends justice_cailey @ Gulf Shores, Alabama,9
ooooweeeeeeeeee bramlage coliseum,4	Ooooweeeeeeeeee!!!!!! #Kobe #MambaOut #KStateMBB #NikeBasketball @ Bramlage Coliseum,4
boom boom boom boom boomthe coolest party dmv tuesday,4	BOOM BOOM BOOM BOOM BOOMThe coolest party in DMV on a Tuesday...,4
b r u n e e blow dry haircut,1	B R U N E T T E #Repost @user Blow Dry: #tinatobar Haircut by me as...,1
anddddd,18	Anddddd they're off! #SUPERHEROSK_____ ** .@Partnr4StrngFam#BestOfGainesville...,18

Figure 1 Cleaned - Uncleaned Example

When we look at the data set, the results obtained are not very reliable because the data is actually real-world data. For example, there is the word "baaaack" in the data set, but it is known that the vector values (vectorization) or the numerical results (tokenization) of the words "back" and "baaaack" are different. However, the words are actually used in a similar sense in context. The same can be said for the words "looooooooveeee" and "love". In other examples, there are meaningless words such as "ough" or "ummmm". These words will have a negative impact on the results because they do not have much meaning. As can be seen in Figure 1. 3, some sentences have very few words left. This is also a situation that makes it difficult to extract meaning. Therefore, it will be quite difficult to interpret these data using ML and DL methods because the data set consists of natural and real data.

Our data set consists of 20 classes. These classes correspond to our emojis. When we examine the data set, although it consists of 20 classes, the class distribution is very imbalanced as shown in Figure 2. It is very difficult to model the training correctly on imbalanced data like this. To overcome this, a few over-sampling methods were tried.

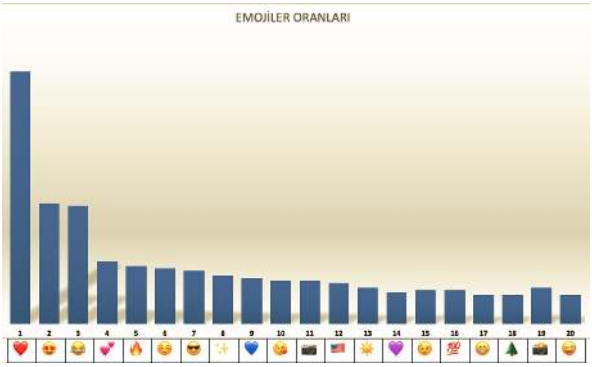

Figure 2 Emojis Ratios

B. Tweet Pre-Proccessing

First, we removed hashtags and "@" symbols with their corresponding words using the tweet preprocessor. These elements may confuse the meaning in the prediction of emojis, so they should not be present in our data set. Second, we removed stopwords such as "I, we, can, do. . . " since our data set is in English. In the third step, we removed extra spaces, meaningless single letters, and repeated words or sentences, and our data set was cleaned. After the cleaning process, as seen in Figure 2, there may still be words that do not convey much meaning.

IV. Model

Here, we will provide information about the model.

A. Over Sampling

"Imbalanced data" is a common issue in machine learning where the distribution of classes in the dataset is uneven. In our dataset, we observed that some classes are overrepresented while others are underrepresented. This can make it difficult to accurately model the data using machine learning algorithms. To address this issue, we tried several oversampling techniques such as Random Over Sampling (ROS) and Synthetic Minority Over-sampling Technique (SMOTE) to balance the dataset. These techniques help to balance the distribution of classes in the dataset, which can improve the performance of machine learning algorithms.

Random Over Sampling is a technique used to balance an imbalanced dataset by generating synthetic data for the minority class. In this technique, the algorithm randomly selects a minority class sample and replicates it in the dataset to balance the class distribution.

SMOTE (Synthetic Minority Over-sampling Technique) is also a technique used to balance an imbalanced dataset by generating synthetic data for the minority class. It works by selecting a minority class sample and finding its nearest neighbors. The algorithm then creates synthetic samples by linearly combining these samples.

The main difference between Random Over Sampling and SMOTE is that Random Over Sampling replicates the minority class samples randomly, while SMOTE creates synthetic samples by linearly combining the minority class

samples with their nearest neighbors. SMOTE is generally considered to be a better technique because it creates more diverse synthetic samples, whereas Random Over Sampling can create repetitive samples [8].

All of our studies have compared the scores by applying normal, ROS, and SMOTE techniques one by one.

B. Vectorization

Vectorization refers to the process of converting data into numerical vectors that can be used in machine learning algorithms. This process is necessary because most machine learning algorithms cannot operate on data in its raw form, such as text or images. Instead, the data must be transformed into a numerical representation that the algorithms can understand. There are several ways to perform vectorization, including count vectorization (CV) , term frequency-inverse document frequency (TF-IDF) , and word embeddings.

TF-IDF (Term Frequency - Inverse Document Frequency) and Counter Vectorizer are two methods used to represent text data in numerical form, which can then be used as input for machine learning models.

The main difference between the two is that Counter Vectorizer simply counts the number of occurrences of each word in the document, while TF-IDF also takes into account the frequency of each word in the entire corpus of documents. This means that the resulting vectors for each document will have higher values for words that are more unique to that document, and lower values for words that are more common across the corpus.

TF-IDF is generally considered to be a better representation of the importance of each word in a document, as it down-weights words that are very common across the entire corpus. This can be useful when the goal is to identify the main topics or themes in a set of documents, as common words that do not contribute much to the meaning of the document will have lower weights. Counter Vectorizer, on the other hand, is often used when the goal is simply to identify the most common words in a set of documents [9].

In our work, we have compared the use of Counter Vectorizer (CV) and Term Frequency- Inverse Document Frequency (TF-IDF) to vectorize our data, both with and without sampling methods.

V. Machine Learning

Here, we will provide information about machine learning.

A. Multinominal Naive Bayes

Multinomial Naive Bayes (MNB) is a classification algorithm that is based on the Naive Bayes theorem, which states that the probability of an event is equal to the probability of each feature occurring independently. MNB is an extension of the standard Naive Bayes algorithm that is used when the features are discrete and the values that they can take are limited. It is commonly used in text classification tasks, where the features are the occurrences of certain words or phrases, and the values are the number of times they occur in a given document. MNB is known for being simple and fast to train, and it is often used as a baseline model in machine learning tasks [10].

In our study, we have shared the results of MNB both in normal and oversampled conditions. We have also carried out a comparative study using CV and TF-IDF.

B. Support Vector Machines

Support Vector Machines (SVM) is a type of supervised learning algorithm that can be used for classification or regression tasks. It works by finding the hyperplane in a high-dimensional space that maximally separates the data points of different classes. SVM tries to maximize the margin between the two classes, meaning that the distance between the hyperplane and the nearest data points of each class should be as large as possible. In the case of non-linear boundaries, SVM can use kernels to transform the input data into a higher-dimensional space where the hyperplane can be found [11].

In the context of Support Vector Machines (SVMs) , the kernel is a function that takes in two input data points and returns a single number, indicating how similar or dissimilar the data points are. The kernel function is used to transform the data into a higher-dimensional space, so that it can be separated by a hyperplane. Some common examples of kernel functions include the linear kernel, polynomial kernel, and radial basis function (RBF) kernel. The choice of kernel function can have a significant impact on the performance of an SVM model.

here are several types of kernels that can be used with Support Vector Machines (SVM) . Some of the most common ones are:

- Linear kernel: This kernel is used when the data is linearly separable. It is the simplest kernel and is used as a baseline for comparison.
- Polynomial kernel: This kernel can be used when the data is not linearly separable. It transforms the input data into higher-dimensional space and allows for the data to be separated by a polynomial function.
- Radial basis function (RBF) kernel: This kernel is a popular choice for non-linear classification problems. It uses the distance between points in the input space to transform the data into higher-dimensional space.
- Sigmoid kernel: This kernel is similar to the RBF kernel, but it uses a sigmoid function instead of a radial basis function to transform the data into higher-dimensional space.

In our study, we compared these methods comparatively.

C. K-Nearest Neighbour

K-Nearest Neighbour (KNN) is a simple and effective method for classification and regression. It works by finding the k number of nearest neighbors of a given data point

and assigns the class label or value based on the majority of the class labels or values of the k nearest neighbors. For example, if k=3 and two of the three nearest neighbors have class label A and one has class label B, then the given data point will be assigned class label A. KNN is known for its simplicity and effectiveness and can work well on small and high-dimensional datasets. It is also known for being sensitive to the scale of the data, so it is often recommended to scale the data before applying KNN. [12] We obtained the best result with k=16, so we selected k=16 and carried out our work and compared it with other methods.

VI. Deep Learning

Deep learning is a subset of machine learning that is inspired by the structure and function of the brain, specifically the neural networks that make up the brain. It involves training artificial neural networks on a large dataset, allowing the network to learn and make intelligent decisions on its own. Deep learning algorithms have been successful in a wide range of applications, such as image and speech recognition, natural language processing, and even playing games like chess and Go [13].

A neural network is a type of machine learning algorithm modeled after the structure and function of the human brain. It is composed of multiple layers of interconnected "neurons, " which process and transmit information. Each neuron receives input from other neurons and uses that input to decide whether or not to transmit a signal to other neurons in the next layer. Neural networks can be trained to perform a variety of tasks by adjusting the strengths of the connections between neurons, known as "weights. " They are particularly well-suited to tasks that involve pattern recognition, such as image and speech recognition, and natural language processing [14].

Recurrent Neural Networks (RNNs) are a type of neural network that allows for processing sequential data. They are called "recurrent" because they perform the same operation for every element in the input sequence, with the output of one element being used as input for the next element. This allows RNNs to have a "memory" of past elements in the sequence, which can be useful for tasks such as language translation or predicting stock prices. RNNs can be implemented using various types of units, such as Long Short-Term Memory (LSTM) units or Gated Recurrent Units (GRUs) [14] .

A. Long Short Term Memory

Long Short Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) specifically designed to prevent the vanishing gradient problem. This problem occurs in traditional RNNs when trying to model long term dependencies, and makes it difficult for the network to learn. LSTM addresses this issue by introducing three gate mechanisms: an input gate, an output gate and a forget gate. These gates allow the network to selectively choose which information to keep or discard from the previous time steps, and help the network to better retain long term dependencies [15].

To prevent overfitting, we used the EarlyStop parameter. We also designed a 3-layer LSTM architecture for our model. We trained the model using the training data and tested it using the test data.

VII. Conclusion

Here, we will provide information about the conclusion.

A. Normal Dataset

As can see Table 1 and Table 2, generally, when f-1 scores are compared, SVM method has achieved the highest performance. Then comes the MNB method, and then the KNN method. Although MNB method gets %0 performance for most emojis, it gets high performance for emojis with high numbers, so its overall performance is higher than KNN method. The emoji number correspondences are shown in Table 3.

Table 1 Counter Vectorizer F-1 Scores with Normal Dataset

NORMAL	**SVM (%)**	**M. Naive Bayes (%)**	**KNN (%)**
0	39	39	31
1	14	18	5
2	38	39	23
3	2	3	0
4	34	32	14
5	0	1	3
6	3	2	2
7	13	3	2
8	5	3	1
9	0	0	1
10	4	4	0
11	44	31	14
12	35	22	12
13	1	1	0
14	0	0	0
15	2	1	0
16	0	0	0
17	60	30	26
18	2	1	1
19	0	0	0

Table 2 TF-IDF F-1 Scores with Normal Dataset

NORMAL	**SVM (%)**	**M. Naive Bayes (%)**	**KNN (%)**
0	39	37	33
1	12	4	5
2	38	32	18
3	2	0	0
4	29	8	2
5	1	0	2
6	7	0	4
7	15	0	6
8	7	0	1
9	0	0	0
10	2	0	0
11	34	3	13
12	33	0	5
13	2	0	0
14	0	0	0
15	1	0	0
16	0	0	0
17	59	0	13
18	1	0	1
19	0	0	0

With TF-IDF Figure 4, when comparing f-1 scores, the SVM method has achieved the highest performance by a

narrow margin. The MNB method comes next, followed by the KNN method. When looking at these methods, all of them have relatively high performances in similar emojis, and similarly show relatively low performance in similar emojis.

Table 3 Emoji İndex

B. Random Over Sampling

Table 4 Counter Vectorizer F-1 Scores with Normal Dataset

RANDOM OS	**SVM (%)**	**M. Naive Bayes (%)**	**KNN (%)**
0	38	17	7
1	22	17	9
2	38	34	22
3	9	15	5
4	37	37	14
5	7	10	9
6	9	11	3
7	13	16	7
8	13	14	11
9	8	13	8
10	11	14	6
11	41	35	21
12	32	31	16
13	3	9	5
14	1	7	10
15	11	14	5
16	3	6	4
17	62	51	35
18	7	12	4
19	0	2	3

With CV, when comparing the f-1 scores, the highest performance was achieved by the SVM method. Then comes the MNB method and then the KNN method. When looking at these methods, all methods showed relatively high performances in similar emojis. Similarly, they also show relatively low performances in similar emojis. In Table 4, it is clearly seen that the highest performance among the ML algorithms is in SVM. The emojis with the highest performance for ML methods, in order, are 17 (Christmas Tree) , 11 (United States) , 0 (Red Heart) , 2 (Face With Tears of Joy) , 4 (Fire) and 12 (Sun) . When looking at these emojis, the number 17 can be more specifically described with more specific words than the other emojis. Therefore, the highest emoji prediction performance is this emoji. When considering this emoji, the difference in performance between SVM and M. Naive Bayes is quite low compared to KNN method. When looking at the number 11 emoji, again, words found in the same context as the flag are relatively more specific. Here, the performance difference between the ML methods has decreased. The number 0 emoji is actually the emoji that is found most frequently in the data set, and its performance has a significant effect on the overall f-1 score. Here, SVM method has achieved a very high performance compared to other methods. M. Naive Bayes method is also relatively successful compared to KNN method. At the same time, one of the least successful emojis for KNN method is the number 0 emoji. When we come to the number 2 emoji, the gap in performance between the ML methods narrows. The main reason for the success of the methods in this emoji is that the number of emojis close to the number 2 emoji is low in the data set and the emojis close to the number 2 emoji are relatively few in the data set. At the same time, except for the number 17 emoji, the number 2 emoji can be said to be the emoji where KNN method has the most success in prediction. When we come to the number 4 emoji, the results of the SVM and MNB methods, the most successful methods, go back and forth. KNN method is still relatively unsuccessful. Again, the reason for the success of the methods is that the number of emojis close to the number 4 emoji is low in the data set. When we come to the number 12 emoji, the success of the methods, especially SVM and MNB, increases. Finally, when we come to the number 7 emoji, which is the least successful emoji among all emojis, the performance of all methods is quite low.

Table 5 TF-IDF F-1 Scores with ROS Dataset

RANDOM OS	**SVM (%)**	**M. Naive Bayes (%)**	**KNN (%)**
0	38	11	15
1	19	17	6
2	43	33	16
3	8	12	6
4	32	30	12
5	3	9	5
6	6	12	8
7	13	18	8
8	10	12	6
9	6	13	4
10	3	8	4
11	46	27	16
12	35	30	14
13	5	9	2
14	1	7	2
15	5	11	6
16	0	8	5
17	64	42	26
18	9	13	5
19	1	4	2

With TF-IDF, according to Table 5, SVM has the highest performance among the ML algorithms, followed by MNB. When the best performing emojis for ML methods are compared, they are 17 (Chirtmas Tree) , 11 (United States) , 2 (Face With Tears of Joy) , 0 (Red Heart) , 12 (Sun) , and 4 (Fire) with the CV vectorization method in that order. When considering the emoji with the number 17, the difference in performance between SVM and MNB is greater compared to the difference between MNB and KNN. In emoji number 11, the difference in performance between ML methods is smaller. When considering emoji number 2, the gap in performance between SVM and MNB narrows. In the same way, if we do not count emoji number 17, KNN can be considered the most successful method in predicting emoji number 2. In emoji number 0, SVM performs much better than the other methods. On the other hand, KNN performs better than MNB. In emoji number 12, all methods perform well compared to their performance in other emojis. In emoji number 4, the performance of SVM and MNB is very close, while KNN performs much worse.

Table 6 Counter Vectorizer F-1 Scores with SMOTE Dataset

SMOTE	**SVM (%)**	**M. Naive Bayes (%)**	**KNN (%)**
0	36	39	0
1	7	22	13
2	16	40	21
3	3	11	10
4	11	36	20
5	1	3	6
6	1	8	7
7	4	19	13
8	3	12	10
9	4	7	6
10	3	12	8
11	13	40	30
12	9	30	22
13	0	3	3
14	0	3	5
15	3	13	11
16	0	4	7
17	35	57	46
18	4	11	6
19	0	1	4

C. SMOTE

With CV as can see Table 6. When comparing f-1 scores, the highest performance is achieved by the MNB method. Then comes the SVM method, and then the KNN method. While the MNB method has a clear advantage in the emojis with the most occurrences, the KNN method has a very low performance in the 0th emoji, which has the most occurrences.

Table 7 TF-IDF F-1 Scores with SMOTE Dataset

SMOTE	**SVM (%)**	**M. Naive Bayes (%)**	**KNN (%)**
0	39	20	0
1	14	16	10
2	38	32	20
3	3	12	6
4	30	30	16
5	1	6	10
6	5	9	9
7	13	20	12
8	5	12	11
9	2	12	8
10	2	9	5
11	47	31	18
12	29	30	20
13	1	9	7
14	1	8	1
15	6	15	7
16	1	7	4
17	60	42	30
18	4	14	8
19	1	3	4

Table 7 shows that, when compared with the f-1 scores, SVM achieved the highest performance. MNB followed, and then KNN. SVM achieved higher performance in the emojis with the most occurrences, while KNN achieved very low performance in the 0th emoji with the most occurrences using TF-IDF.

D. Accuracy Ratio

According to Table 8, the most successful method in terms of performance is the SVM machine learning algorithm. When comparing the machine learning methods, the most successful method in the normal data set is SVM with a small difference. In the normal data set, the results of the LSTM, M. Naive Bayes, and SVM methods are very close. Also, the emoji that each method is most successful in predicting is the 17th emoji. This is because the emoji can be expressed with very special words. While the SVM method gives close performances for ROS, SMOTE, and the normal data set, the MNB method gets very low results when looking at the ROS results. The KNN method gives very low results both for SMOTE and ROS compared to the normal case. After being trained with the ROS data set, the LSTM method gives lower performance compared to the normal data set. The reason for the general decline in performance after applying the ROS and SMOTE techniques is that the data set we tested has a large number of 0th emojis. When the naturally and heterogeneously distributed emoji ratios are artificially equalized, when the models are tested with the homogenized data set, the prediction rate is more homogeneous, so the models predict the 0th emoji, which has the highest rate, less. Therefore, the decrease in the number of predictions of the 0th emoji, which is the most common in the data set, directly pulled down the obtained results.

Table 8 Accuracy Results

RANDOM OS	**SVM (%)**	**M. Naive Bayes (%)**	**KNN (%)**
0	38	11	15
1	19	17	6
2	43	33	16
3	8	12	6
4	32	30	12
5	3	9	5
6	6	12	8
7	13	18	8
8	10	12	6
9	6	13	4
10	3	8	4
11	46	27	16
12	35	30	14
13	5	9	2
14	1	7	2
15	5	11	6
16	0	8	5
17	64	42	26
18	9	13	5
19	1	4	2

E. Comparing Vectorization Methods

According to Table 9, when we consider emoji 4, the performance of MNB method decreases significantly when using the TF-IDF vectorization method compared to the CV method. Similarly, both SVM and KNN methods also show a significant decrease in performance. In a situation where detecting emoji 4 is important, the CV vectorization method can be preferred. When we look at emoji 11, the performance increases by about %5 when using the SVM method with the TF-IDF vectorization method. However, there is a significant decrease in performance for the other methods. When we consider emoji 12, there is a slight increase in performance when using the SVM method with the TF-IDF vectorization method. The performance of MNB method is similar for both CV and TF-IDF vectorization

methods. The performance of KNN method also appears to be similar, but with a significantly lower performance. Similarly, when considering emoji 17, the combination of SVM and TF-IDF gives better performance while other methods give a higher performance with CV. Therefore, it can be said that using the SVM method with the TF-IDF vectorization method would be better when considering these four emojis. It can also be said that using the CV vectorization method with KNN and MNB methods would give better results.

Table 9 Comparing Vectorization

Emoji	4 Fire		11 United States		12 Sun		17 Chirtmas Tree	
Yöntem	CV	TF-IDF	CV	TF-IDF	CV	TF-IDF	CV	TF-IDF
SVM (%)	37	32	41	46	32	35	62	64
M. Naive Bayes (%)	37	30	35	27	31	30	51	42
KNN (%)	14	12	21	16	16	14	35	26

REFERENCES

[1] F. Barbieri, J. Camacho-Collados, F. Ronzano, L. Espinosa-Anke, M. Ballesteros, V. Basile, V. Patti, and H. Saggion, "SemEval-2018 Task 2: Multilingual Emoji Prediction," in *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018)*. New Orleans, LA, United States: Association for Computational Linguistics, 2018.

[2] F. Barbieri, J. Camacho-Collados, F. Ronzano, L. E. Anke, M. Ballesteros, V. Basile, V. Patti, and H. Saggion, "Semeval 2018 task 2: Multilingual emoji prediction," in *Proceedings of the 12th international workshop on semantic evaluation*, 2018, pp. 24–33.

[3] F. Barbieri, J. Camacho-Collados, F. Ronzano, L. Espinosa-Anke, M. Ballesteros, V. Basile, V. Patti, and H. Saggion, "SemEval 2018 task 2: Multilingual emoji prediction," in *Proceedings of the 12th International Workshop on Semantic Evaluation*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 24–33. [Online]. Available: https://aclanthology.org/S18-1003

[4] 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*, 2018, pp. 34–38.

[5] L. Alexa, A. B. Lorent, D. Gifu, and D. Trandabat, "The dabblers at semeval-2018 task 2: Multilingual emoji prediction," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, 2018, pp. 405–409.

[6] S. Jin and T. Pedersen, "Duluth urop at semeval-2018 task 2: Multilingual emoji prediction with ensemble learning and oversampling," *arXiv preprint arXiv:1805.10267*, 2018.

[7] C. Baziotis, N. Athanasiou, G. Paraskevopoulos, N. Ellinas, A. Kolovou, and A. Potamianos, "Ntua-slp at semeval-2018 task 2: Predicting emojis using rnns with context-aware attention," *arXiv preprint arXiv:1804.06657*, 2018.

[8] A. Rajendran, C. Zhang, and M. Abdul-Mageed, "Ubc-nlp at semeval-2019 task 6: Ensemble learning of offensive content with enhanced training data," *arXiv preprint arXiv:1906.03692*, 2019.

[9] T. P. Kumar and B. V. Vardhan, "Multimodal sentiment prediction based on the integration of text and emojis," *Journal of Optoelectronics Laser*, vol. 41, no. 4, pp. 489–499, 2022.

[10] R. Ranjan and P. Yadav, "Emoji prediction using lstm and naive bayes," in *TENCON 2021-2021 IEEE Region 10 Conference (TENCON)*. IEEE, 2021, pp. 284–288.

[11] W. Bourequat and H. Mourad, "Sentiment analysis approach for analyzing iphone release using support vector machine," *International Journal of Advances in Data and Information Systems*, vol. 2, no. 1, pp. 36–44, 2021.

[12] I. Irawaty, R. Andreswari, and D. Pramesti, "Development of youtube sentiment analysis application using k-nearest neighbors (nokia case study)," in *2020 3rd International Conference on Information and Communications Technology (ICOIACT)*. IEEE, 2020, pp. 39–44.

[13] V. D. P. Kollipara, V. H. Kollipara, and M. D. Prakash, "Emoji prediction from twitter data using deep learning approach," in *2021 Asian Conference on Innovation in Technology (ASIANCON)*. IEEE, 2021, pp. 1–6.

[14] A. C. Coman, G. Zara, Y. Nechaev, G. Barlacchi, and A. Moschitti, "Exploiting deep neural networks for tweet-based emoji prediction," in *International Workshop on Semantic Evaluation*, vol. 4, 2018, p. 1.

[15] M. Liu, "Emonlp at semeval-2018 task 2: English emoji prediction with gradient boosting regression tree method and bidirectional lstm," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, 2018, pp. 390–394.