MULTI-LABEL CLASSIFICATION WITH NEURAL NETWORK

A Thesis

by

Sezin Ekşioğlu

Submitted to the
Graduate School of Sciences and Engineering
In Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in the
Department of Data Science

Özyeğin University
May 2022

MULTI-LABEL CLASSIFICATION WITH NEURAL NETWORK

Approved by:

Associate Professor Okan Örsan Özener,
Advisor
Department of Data Science
Özyeğin University

Assistant Professor Başak Altan Özener
Department of Business
Özyeğin University

Associate Professor Uğur Çelikyurt
Department of Business
Koç University

Date Approved: 25 May 2022

To my beloved family and friends...

ABSTRACT

Multi-label classification has huge importance for several applications, it is also a challenging research topic. It is a kind of supervised learning that contains binary targets. The distance between multilabel and binary classification is having more than one class in multilabel classification problems. Features can belong to one class or many classes. There exists a wide range of applications for multi-label prediction such as image labeling, text categorization, gene functionality. Even though features are classified in many classes, they may not always be properly classified. There are many ensemble methods for classification. However, most of the researchers have been concerned about better multi-label methods. Especially little ones focus on both efficiency of classifiers and pairwise relationships at the same time to implement better multi-label classification. In this paper, we worked on modified ensemble methods by getting benefits from k-Nearest Neighbors and neural network structure sequentially to address issues beneficially and to get better impacts from the multi-label classification. Publicly available datasets (yeast, emotion, scene, and birds) are performed to demonstrate the developed algorithm efficiency, and the technique is measured. Our algorithm outperforms benchmarks for each dataset with different metrics. The result of the algorithm is competitive with the state-of-the-art results. Especially, in the weighted average of false-positive minimization and false-negative minimization, the algorithm passes the benchmarks.

ÖZETÇE

Çok etiketli sınıflandırma, birçok uygulama için büyük öneme sahiptir, aynı zamanda zorlu bir araştırma konusudur. İkili hedefler içeren bir tür denetimli öğrenmedir. Çok etiketli sınıflandırma problemlerinde çok etiketli ve ikili sınıflandırma arasındaki mesafe, birden fazla sınıfa sahip olmaktır. Özellikler bir sınıfa veya birçok sınıfa ait olabilir. Görüntü etiketleme, metin kategorizasyonu, gen işlevselliği gibi çok etiketli tahmin için geniş bir uygulama yelpazesi bulunmaktadır. Özellikler birçok sınıfta sınıflandırılmış olsa da, her zaman uygun şekilde sınıflandırılamayabilirler. Sınıflandırma için birçok topluluk yöntemi vardır. Bununla birlikte, araştırmacıların çoğu, daha iyi çoklu etiket yöntemleri konusunda endişe duymaktadır. Özellikle bazıları, daha iyi çok etiketli sınıflandırma uygulamak için hem sınıflandırıcıların verimliliğine hem de ikili bağlara aynı anda odaklanır. Bu çalışmada, sorunları yararlı bir şekilde ele almak ve çok etiketli sınıflandırmadan daha iyi etkiler elde etmek için k - En Yakın Komşu algoritmasından ve yapay sinir ağı yapısından sırayla yararlanarak değiştirilmiş topluluk yöntemleri üzerinde çalıştık. Farklı veri setleri üzerinde algoritmamızı uyguladık. Algoritmamız, farklı metriklere sahip her veri kümesi için kıyaslamalardan daha iyi performans gösterir. Algoritmanın sonucu, en gelişmiş sonuçlarla rekabet edebilir ölçüdedir ve doğruluk açısından rakiplerini geçen sonuçlar elde edilmiştir.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Multi-label classification is a classification that involves predicting zero or more class labels. Multi-label classification problems include multiple label problems. In this classification, each sample can belong to one class or many classes at the same time. The methodology is used from direct marketing[1], image and text categorization[2], gene functionality [3], emotions, music categorization [4], and many other fields. For instance, in movie websites, there are many categorized movies according to the content of the movie. However, sometimes it is realized that even if the content of the movie is both romantic, comedy, and drama or something different, the categorization appears wrong. Similarly, in image annotation, the animal image can belong to "two-footed", "mammalian" and "two eared". Besides, scene images can be associated with "mountain", "sunset", and "beach" labels. As it is known, genes contain considerable diversity, and when we want to make a gene definition, this diversity is looked at for correct identification. The gene fragment may appear to belong to more than one class at the same time. For correct identification, correct classification is required. When text is desired to be categorized, there may be more than one topic with content and the content percentage of these topics can be close to each other. Thus, these samples are directly related to multi-label classification problems.

In recent years, binary classification problems have been studied widely rather than multi-label classification learning. However, multi-label classification has become a fundamental and hot topic and the number of studies is increasing day by day. The classification supplies benefit when an instance belongs to more than one label at the same time. Thus, the recognition or categorization gets easier. There are also some

challenges in this classification. Correlations between labels are the most fundamental point of the problems. If any of the two labels are correlated to each other, the categorization of samples gets hard and positioning instances can be complicated. Besides correlation issues, high dimensionality and imbalanced classes are also the main issues of multi-label classification.

Many algorithms have been developed to bring the solution for multi-label classification problems. Individual learners from heterogeneous or homogeneous models are combined to produce a common learner, which increases learning persistence by overcoming model overfitting and initialization sensitivity. [5]. Some ensemble [6] methodologies have been proposed as benchmarks; they apply bagging to achieve several classifiers and combine them according to the results of the voting. Particularly, the final result is selected according to the majority voting strategy by label ranking. Based on this, a new sample prediction is calculated by the average of the values of all the classifiers for each label instead of calculating the result based on weights for labels. These approaches do not consider the local pairwise correlation. There are also many traditional algorithms to solve multi-label classification, these are divided into two categories; problem transformation methodologies and algorithm adaptation methodologies. This first one transforms multi-label classification problems into many independent binary classification problems. It deals with the problem as multiclass problems. Hence, it can be applied directly. These algorithms are Binary Relevance (BR), Classifier Chains (CC), and Label Powerset (LP). The second one extends a specific learning algorithm. The algorithmic optimization approach involves modifying the existing single-label classification algorithm and making it suitable for processing multi-label instances [7]. These are, MLKNN, ML-DT and Rank-SVM. Scoring the results of the multi-label classification, some important metrics are used such as accuracy, hamming loss, F1 score, and AUC score. Accuracy measures the correctness of predicted labels. AUC area states are under the area of the ROC

curve which is the relational curve between true positive and false-positive rates. F1 score shows a harmonic means of precision and recall. In other words, it measures the consistency of the model. If the number of these metrics gets high, the model is successful. Hamming Loss takes into account the incorrectly predicted labels. It bases the prediction errors. When the number of hamming loss gets lower, the model predicts labels correctly.

In this paper, we solve this problem by getting benefits from the Neural Network structure due to the universal approximation feature of the strategy. Additionally, Neural Network can have many layers and the structure is extensible and flexible. It is possible to generate new hidden layers with various calculations. Publicly available yeast, birds, scene, and emotions are datasets are used. KNN and Neural Network structure is sequentially trained in our technique. We create two inputs for a neural network. One of them is the naturally trained dataset. The other one is created based on the k nearest distance of possible labels for train and test datasets. In fact, generally, the KNN (nearest k-neighbor algorithm) is a simple but effective method of learning machines to solve classification problems. Because the hypotheses are built locally and computation is deferred until the test data set is acquired, it is commonly referred to as an instance-based learning or lazy learning method. Based on the voting, a majority of its nearest neighboring instances rank an instance in the KNN-based method, and it belongs to the most common class among these neighbors.[8] In our algorithm, we used the distance calculation method to find potential labels. In other words, we get the diagonal symmetric of k nearest data points distances between train datasets also train and test datasets. We calculate the distances for both train and test ones, because of the calculated KNN features. When the calculations are applied for the training dataset, we can access the indexes of the closest points to each data point except itself. In this way, we derive KNN features from the label train dataset. Then, we get the mean value of the nearest points to find the optimal

one. In this way, KNN features are derived based on label train data points. Due to the float average results, we push the values to 0 or 1 based on an upper and a lower threshold. After completing the arrangement of inputs, we combined two of them. However, we do not put the second input into hidden layers. Instead of this, we add the modified input at the end of the layer structure. We test the performance of our algorithm and compare our benchmarks in terms of metrics. Finally, we observe that the results are competitive with the state-of-the-art results. However, the results especially perform seriously well on an f1 score basis, which is the harmonic mean of precision and sensitivity metrics.

CHAPTER II

PREVIOUS WORK

In this chapter, we reviewed benchmarks algorithms for multi-label classification. Bagging algorithms are Ensemble of Binary Relevance (EBR) [9], Ensemble of Label Powerset (ELP) [10], Ensemble Classifier Chain (ECC) [11] and The RAndom k-labEL (RAkEL) [12] are the most competitive algorithms. Ensemble Binary Relevance separates classifiers as sub-sample and behaves them as binary classification. However, while doing this computation, the algorithm does not care about the correlations between layers. Also, computational effectiveness is low due to the number of sub-samples.

The ELP model creates sub-groups using the bagging method and then combines the predictions of the classifiers with the majority voting method. It creates a different class dataset for each different tag combination. Therefore, it becomes difficult to correlate, complicates computation, and results in an unstable dataset. The RAkEL model selects k random labels and learns n label powerset classifiers, resulting in a balanced distribution. In ECC method, sub-groups are also created. After ECC subdivides the method, it considers the previous one's prediction when moving to the next classifier. that is, it calculates the next one based on the previous classifier. It is similar to a chain with its computational structure. There are different chains to different classifiers, and although these clusters are randomly chosen, performance may be adversely affected as the next prediction also takes into account the previous one. When a stacked-ensemble, Multi-label stacking (MLS) [13] algorithm is analyzed at benchmarks, MLS performs this calculation twice, treating each classifier as a binary classifier. Since the predictions made in the first step are

the precursors of the predictions made in the last step, Binary Relevance considers possible correlations to the contrary. This structure allows to diversify the model by providing a different attribute space for each classifier. Different classifier weights can be considered for different labels. The purpose of this structure is not to reduce the performance of possible unrelated values and classifiers and not to create noise. For this reason, classifiers in the MLS structure are called meta-level classifiers. In the Weighted ensemble classification structure, MLKNN [14] and AdaBoost [15] are the most popular examples. MLKNN is a weight-oriented model structure that calculates using distance and coefficient. This structure adapts the algorithm to perform multi-label classification directly, rather than transforming it into different subsets of problems. The multi-label version of kNN is represented by MLkNN. In the AdaBoost technique, the weight adjustment is as follows, if it is difficult to predict the train sets and labels, it modifies the classifiers by increasing the weights. If it's easy to predict, it uses the method of reducing weights, but because this algorithm is based on binary relevance, it ignores possible correlations. In the weighted classifier and stacked ensemble method [7], In this algorithm (MLWSE), the weight-oriented classifier method is applied using regularization. It is a method created by making use of classifier weights and binary label correlations at the same time.

Another study which is also related with multilabel classification focuses on the solution based on neural network structure. The algorithm which is called BP-ML works in the form of the global error functions. In other words, the algorithm generates a new error function in order to detect the dependencies between labels. It minimizes and maximizes the output values of neurons according to belongingness. At the same time, thresholds are arranged based on the input instances. [16] Studies similar to this algorithm structure focus more on thresholds in the neural network layer structure. Thus, they direct the incoming input to match the output value with thresholds. In addition, some studies focus on aggregation-based ensemble algorithms

and develop solutions to the multilabel classification problem. For example, ABC-based aggregation. This type of algorithm is useful for finding optimal meta-level classifier configurations.[17] Another group of studies focuses on loss functions. It is tried to get better results by introducing and addressing the modified loss functions to the NLP structure. By reweighting the BCE, the loss function is optimized and reweighting in such neurons is achieved with a lower error rate. [18] On the other hand, in studies where Advanced statistical methods that work like single index models are used, the interpretability of multilabel models and the analysis of the problem are facilitated by coefficients. In such studies, linearity in datasets is taken into account. In studies focusing on randomized sampling, instead of choosing a fixed number of samples in the datasets and making label predictions, k random samples are created and the probabilities are calculated by training these samples separately. But at this point, it may not always be possible to see a consistent result, since the samples are randomly generated. In algorithms that focus on tuples, the goal is to define labels correctly. There are possible labels in the Label set, and the samples are clustered as a tuple and given to the prediction. Tuples that receive high prediction results are stored in another sample set.

CHAPTER III

DATASETS

3.1 Dataset Description:

The datasets that we use in our analysis are public datasets. These are Emotions, Scene, Yeast, and Birds.

3.1.1 Emotion:

It is an English Twitter messages dataset with 6 labels. These are, there are 6 different emotions which are amazed - surprised, happy - pleased, relaxing - calm, quiet - still, sad - lonely, angry - aggressive. The instance number is 593. The dataset has labeled people's emotions. For example, from the comments entered on any social media platform, the emotion or feelings of that person can be defined. It is an example of a multi-label dataset because it is difficult to understand the emotion that people convey concerning the form of language they use. A sentence can contain more than one emotion at the same time, and it is very difficult to determine it. At the same time, it is not easy to classify according to the slang and coded words used in the sentence. [19] We split the dataset into train and test datasets to apply the machine learning structure. Train dataset, which is the dataset we train to enable the algorithm to learn has 474 samples and 72 features. After teaching the data to the algorithm, we reserved the test dataset to test whether it learns well or not. The test dataset has 119 samples and 72 features. The number of targets of labels is stated below in the table. When the dataset is compared with benchmarks, the best result seems in the "Multi-label classification with weighted classifier selection and stacked ensemble" paper [7] in terms of accuracy which predicts labels as exactly matched

a hamming loss which is the fraction of labels that are incorrectly predicted, and ranking loss which computes the average number of label pairs that are incorrectly ordered, f1 which measures the performance of the model. However, our algorithm boosts these metrics except the micro f1 score.

Table 1: Table of target numbers in emotion dataset

#	Label	Number of Target
1	**amazed.suprised**	173
2	**happy.pleased**	166
3	**relaxing.calm**	264
4	**quiet.still**	148
5	**sad.lonely**	168
6	**angry.aggresive**	189

3.1.2 Yeast:

The yeast dataset is protein-containing. It consists of interactions between proteins. It has 14 labels with 2417 instances Thanks to this dataset, interactions between proteins were detected and analyzed in detail. For the biology field, it is a dataset that is important to distinguish. Because there are many proteins and highly correlated proteins. The relationship between them must be discovered and the difference must be determined, in other words, they must be labeled correctly. [20] When the results of the algorithms are compared, the best evaluation metric results come from the "Multi-label classification with weighted classifier selection and stacked ensemble" paper. [7] However, our approach boosts the paper in terms of the performance metric f1, and micro f1 score.

Table 2: Table of target numbers in yeast dataset

#	Label	Number of Target
1	**Class1**	762
2	**Class2**	1038
3	**Class3**	983
4	**Class4**	862
5	**Class5**	722
6	**Class6**	597
7	**Class7**	428
8	**Class8**	480
9	**Class9**	178
10	**Class10**	253
11	**Class11**	289
12	**Class12**	1816
13	**Class13**	1799
14	**Class14**	34

3.1.3 Scene:

The dataset creates the views from the photos. It has 6 labels that are image based data (beach, sunset, fallfoliage, field, mountain and urban) with 2407 instances The problem here is that photos can make some images stand out more and the content of the photo can be misclassified [21]. Additionally, multiple images can be equally foregrounded. At this point, it is expected to belong to both classes. For example, in a photograph, both the mountain and the beach came to the fore. In this case, effective labeling is needed to accurately identify the photograph, otherwise, the image cannot be recognized properly. When the multi-label methodologies are compared, the best results are given by the "Multi-label classification with weighted classifier selection and stacked ensemble" paper.[7] in terms of accuracy and loss metrics. However, for the macro - F1 score, the classifier chain method returns the best result as 0.72. Our algorithm boosts the results both for f1 and macro f1 score as 0.73

Table 3: Table of target numbers in scene dataset

#	Label	Number of Target
1	**Beach**	427
2	**Sunset**	364
3	**FallFoliage**	397
4	**Field**	433
5	**Mountain**	533
6	**Urban**	431

3.1.4 Birds:

The dataset includes various bird species and it has 19 labels about the kind of birds with 645 instances. It is difficult to classify correctly because of the many varieties, due to their prominent features and possible similar features. For example, eye color, feather color, wing structure may be the same for a bird species, but even a slight difference in beak structure is enough to label it as a separate species. When the results of the multi-label classification algorithms are compared, in accuracy, the paper [7] gives the best effective result. On the other hand, Binary Relevance brings the most effective results for loss evaluation metrics. In terms of the performance of the model, Classifier Chain is a beneficial algorithm. Our algorithm gives the best result in terms of micro f1.

Table 4: Table of target numbers in birds dataset

#	Label	Number of Target
1	**Brown.Creeper**	14
2	**Pacific.Wren**	81
3	**Pacific.slope.Flycatcher**	46
4	**Red.breasted.Nuthatch**	9
5	**Dark.eyed.Junco**	20
6	**Olive.sided.Flycatcher**	14
7	**Hermit.Thrush**	47
8	**Chestnut.backed.Chickadee**	40
9	**Varied.Thrush**	61
10	**Hermit.Warbler**	53
11	**Swainson.s.Thrush**	103
12	**Hammond.s.Flycatcher**	28
13	**Western.Tanager**	33
14	**Black.headed.Grosbeak**	9
9	**Golden.Crowned.Kinglet**	37
10	**Warbling.Vireo**	17
11	**MacGillivray.s.Warble**	6
12	**Stellar.s.Jay**	10
13	**Common.Nighthawk**	26

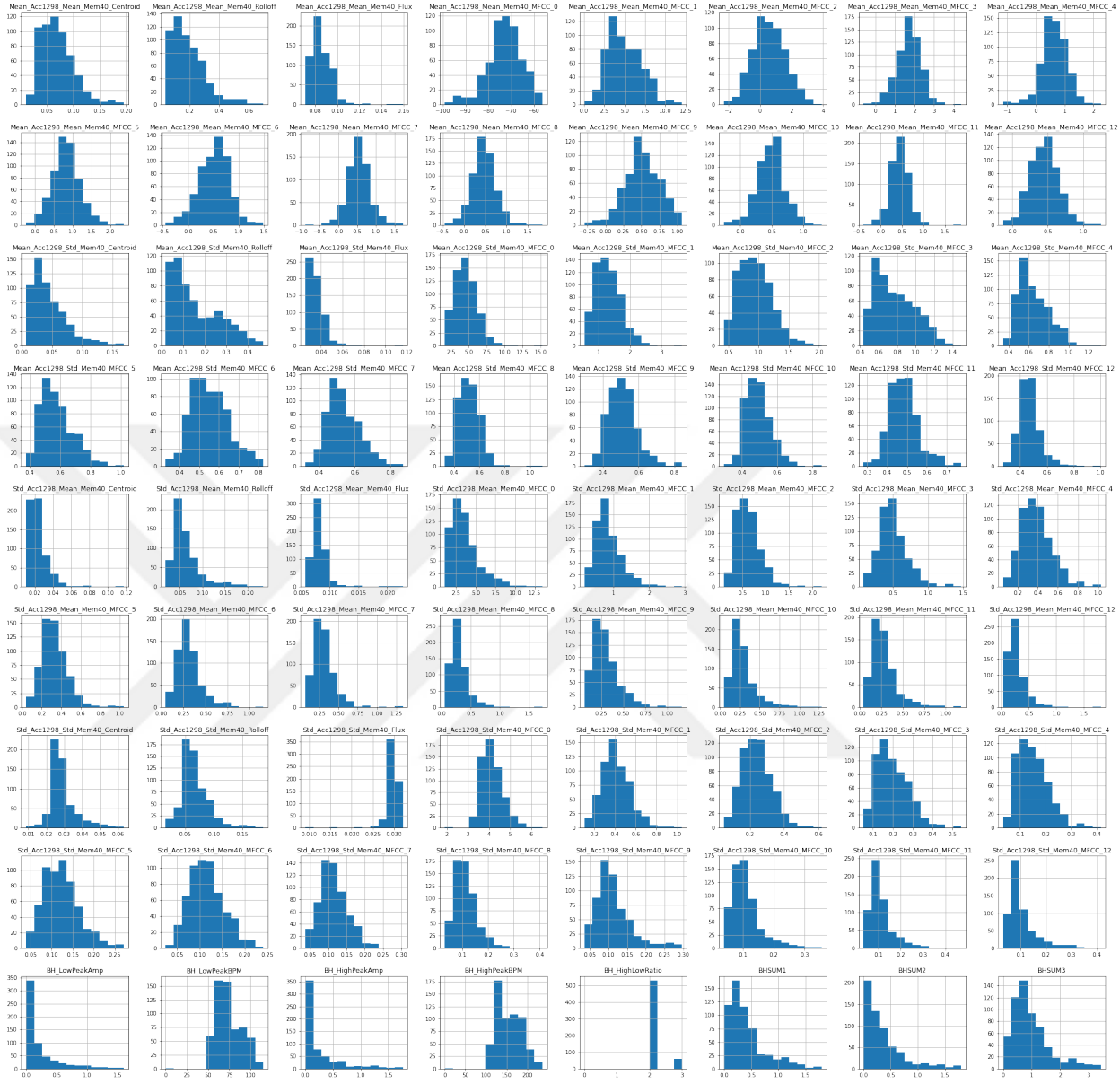

Figure 1: Histogram of emotion dataset features

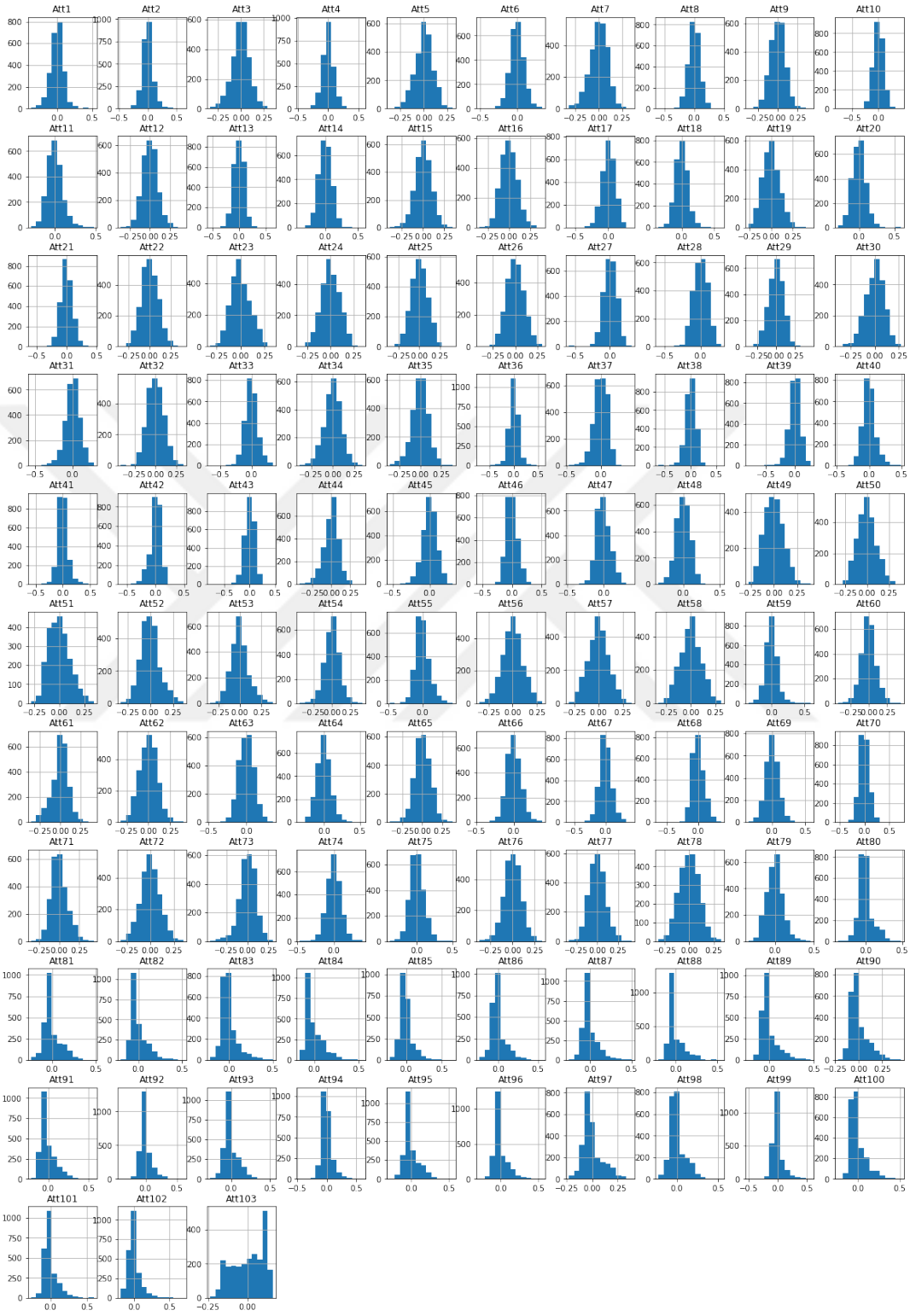

Figure 2: Histogram of yeast dataset features

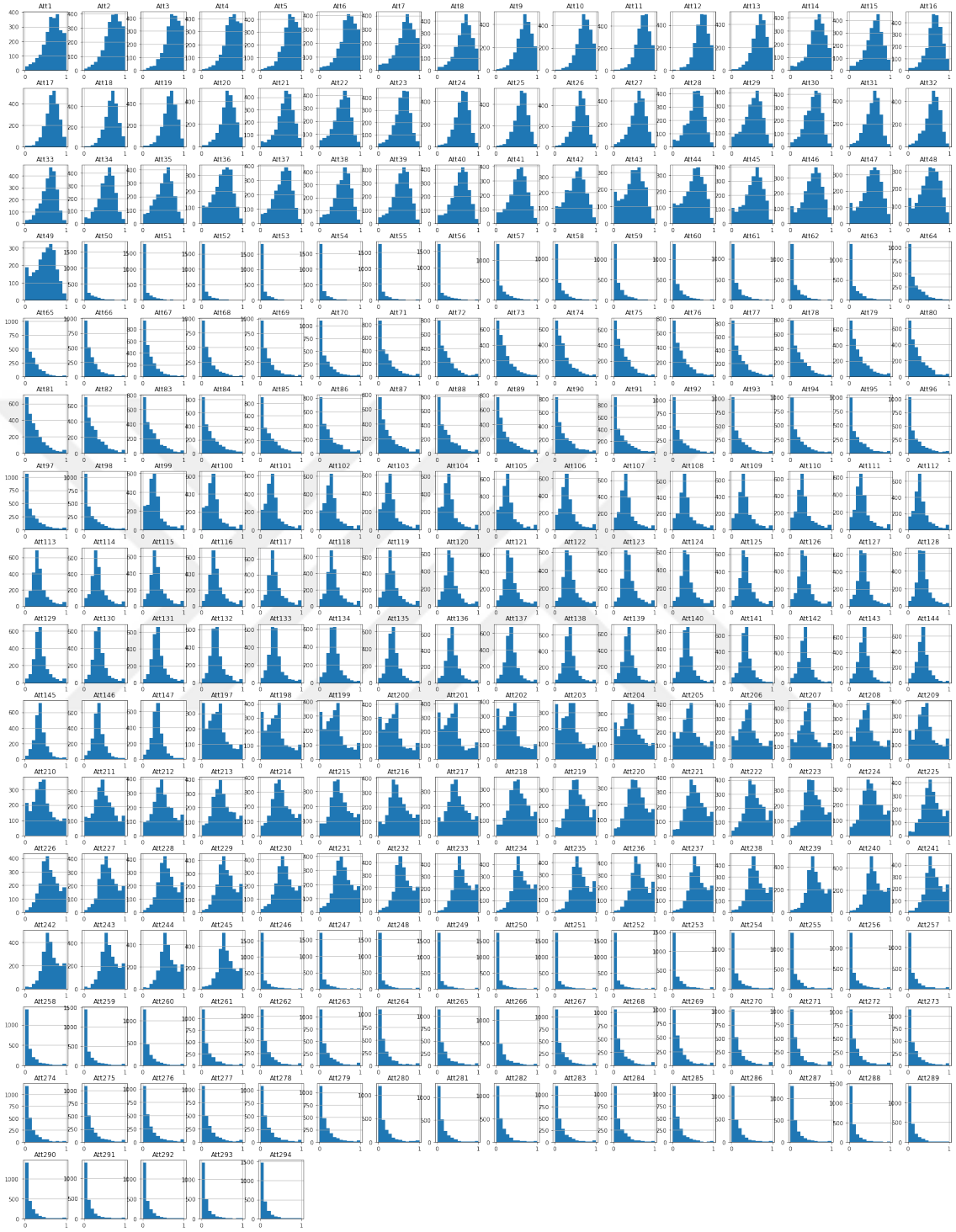

Figure 3: Histogram of scene dataset features

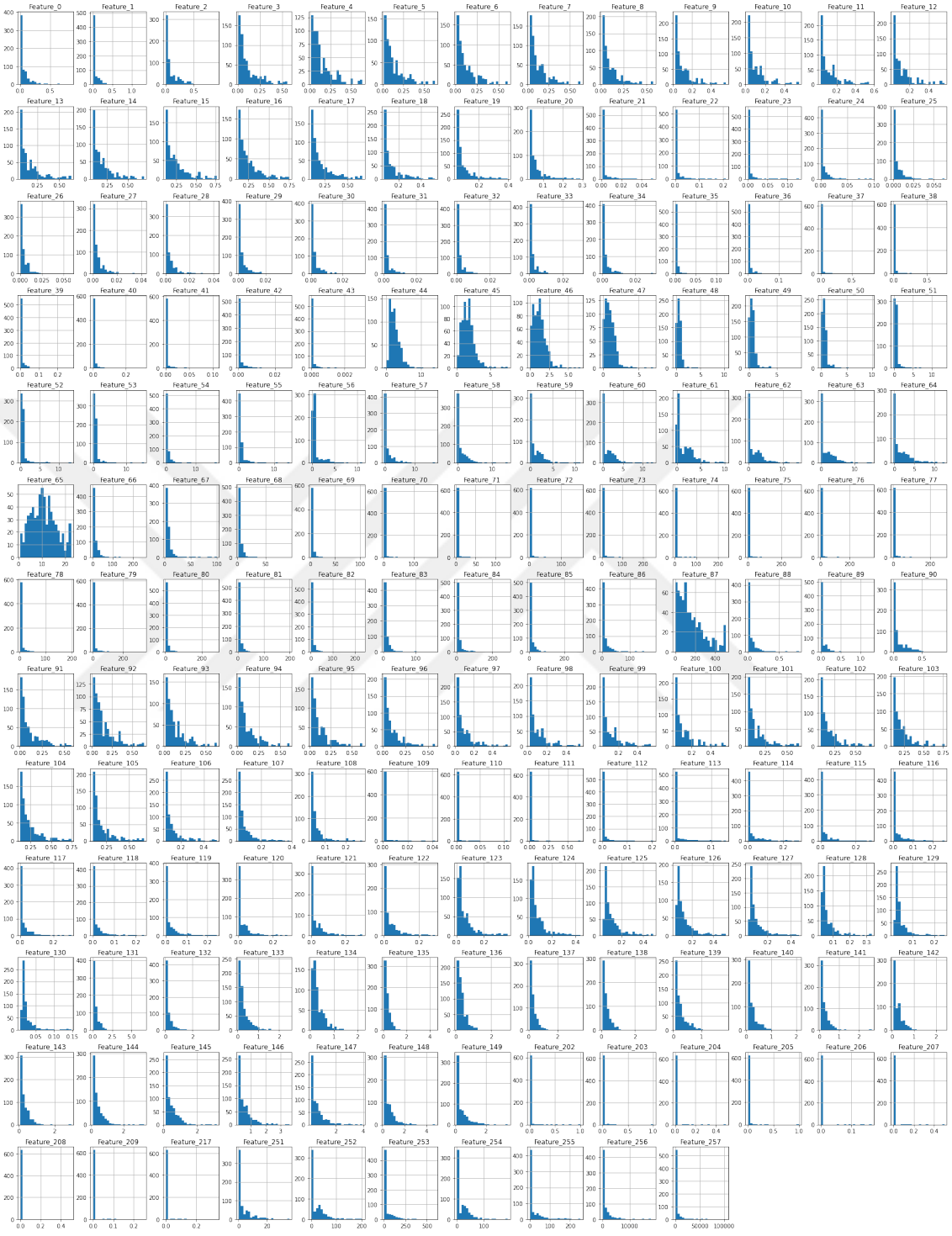

Figure 4: Histogram of birds dataset features

3.2 Background

There are several methodologies developed to handle multi-label classification problems. However, it is different from other classification problems because it deals with many labels and the main point is that labels cannot be correlated, and the sample can belong to one or more than one label at the same time. Some of the multi-label methods are BR, LP, CC, ML-KNN.

3.2.1 Multi-Label Classification Methods:

3.2.1.1 Binary Relevance (BR):

This method is under the problem transformation technique. A training dataset of multilabel problems that have L number of labels is divided into L number of single-label training datasets. In other words, the method works on multi-label classification problems as binary classification problems and it trains each label separately. At the end of the training process, union operation is applied to reach the final labels. However, the main challenge of this method is the correlation. the method cannot deal with the correlation circumstance between labels.[22] The formula of binary relevance training and classification parts are shown below. (t: number of labels, m:the model, T:binary training set, x: test sample)

Algorithm 1: *BR Algorithm*

1: Training part:
2: **for** each i=1 to t: **do**
3: train(m_i, T)
4: **end for**
5: Classification part:
6: **for** each i=1 to t: **do**
7: $t_i = m_i$ (x, t_i)
8: **end for**
9: return t_1 , t_2 ... t_t

3.2.1.2 LabelPowerset (LP):

The method is under the transformation technique. The algorithm works with a unique combination of labels. If the label is not combined any other label is called a single label and single labels are trained by a binary classifier. For combinations, the maximum probability compared to others is LP outputs. The method's benefit is that it deals with label correlation and instead of predicting each label one by one, it combines some of them. However, as long as the number of labels is high, the algorithm increases the computational complexity [22]. Particularly, according to this method, each combination of labels encountered on the samples is matched with a unique combination number and the multiclass classifier is trained using these numbers as classes. Thus, it causes computational complexity because the amount of resources required to run the algorithm is high, and it also needs more time and memory. The formulation of both training and classification part are stated below. (t: the number of unique combinations, m:binary classifier model, x:training set, P:probability of unique combination)

Algorithm 2: *LP Algorithm*

Training part:
for each i=1 to t: **do**
    train(m_i, x_i)
end for
Classification part:
for each i=1 to t: **do**
    $P_i = P_{ti}$
    if $P_i > P_i - 1$ **then**
        $T_j = T_i$
    end if
end for

3.2.1.3 Classifier Chain(CC):

The method is for multi-label classification problem transformation. It combines the Binary Relevance method's computing efficiency with a new algorithm. It learns each label as in the Binary Relevance. The classifiers are linked as chains. In other words, it behaves as a single label for each label and learns binary classification for each of them. The features which are given to each classifier are extended by binary values. These binary values indicate the previous labels. The code of both training and classifications parts are shown as below part. (L: number of labels, D: single label transformation and training, training part of(t,X), (t,X):training sample, C: chains of the model)

Algorithm 3: *CC Algorithm*

Training part:
for each i=1 to L: **do**
    D = do binary label training
end for
for (t, X) in D': **do**
    D = D U /t, l_1, l_2, ..., l_i
    Train C_i to predict BR of l_i
end for
D' = l_i as binary numbers
C_i = D'
Classification part:
for each i=1 to L: **do**
    X = X U C_i (x, l_1, ... l_i for each l_i
end for
return (t, X)

3.2.1.4 K-Nearest Neighbors Algorithm (KNN):

The algorithm predicts the class of the vector formed by the independent variables of the value to be predicted, based on the information in which class the nearest neighbors are dense. It finds the distance of the point to be estimated from other points by using various distance calculation functions and looks at its nearest neighbor as much as the k number given to the algorithm. These distance functions can be Euclidean, Manhattan, or Minkowski. Euclidean distance is the Pythagorean relation between two points when we consider a 2-dimensional space. That is, the bit is the square root of the sum of the square of the difference of two points on the x-axis on the x-y line and the square of the difference of two points on the y-axis. The Euclidean distance calculation is shown below.

$$d(Z,T) = \sqrt{[(x2-x1)^2 + (y2-y1)^2]} \tag{1}$$

In Manhattan distance, the distance is the sum of the absolute distances between the data points.

$$\sum_{n=1}^{k} |x_i - y_i| \tag{2}$$

Minkowski distance is a distance calculation method based on the "q" number of variables. It is similar to the Euclidean relation, we can see that when we substitute 2 for "q" in the formula below, we get the fully Euclidean relation.

3.2.1.5 Multi-Label K Nearest Neighbors(ML-KNN):

It is an algorithm that is under the adaptation approach. The method is based on the traditional K Nearest Neighbor algorithm(KNN). The difference from the KNN is the Bayesian approach[12]. The Bayesian theorem shows the relationship between conditional probabilities and marginal probabilities within the probability distribution for a random variable. As a first step, the algorithm selects the k-nearest neighbors, then each labeled sample in the neighbors' training set counts. With finally statistical analysis, the probability estimation is calculated and the final prediction of the label sets is supplied. Just like KNN, the Euclidean function is used as the distance function. Manhattan, Minkowski, and Hamming functions can also be used as an alternative to the Euclidean function. After the distance is calculated, the data points are ordered and the value is assigned to the appropriate class.

Figure 5: ML-KNN Structure

3.2.1.6 Neural Network:

It is a set of algorithms and similar structures of the human brain. The methodology interprets sensory data and the patterns that they define are numerical, images, sound, text, or time series. Neural Network both cluster and classify data. The structure has layers in order to transmit data. The layers are constructed from nodes and computations are applied in these nodes. Then it combines with coefficients and the data is transferred to the other label according to their weight importance. The structure of mathematical model of a neuron (perceptron) is shown below.

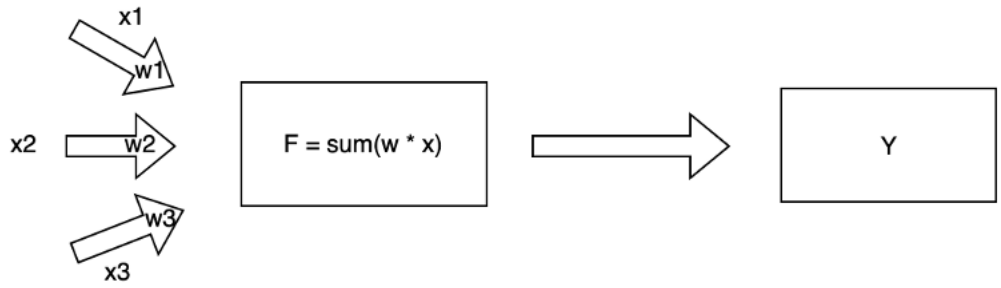

Figure 6: Neural Network Structure

Neural network supplies benefit to solve multilabel classification problems in terms of decomposition of issues into small groups and it enables to apply one for each technique. There are many optimization algorithms in the neural network in terms of time, speed and memory. This provides the flexibility to apply different algorithms to different datasets. These are "Gradient Descent," "Newton method", "Conjugate Gradient", "Quasi-Newton Method", " Levenberg-Marquardt algorithm". In fact, the main point that they all focus on is to increase the performance of the algorithm by minimizing the loss function, that is, the error rate. Gradient descent aims to reach the global minimum value by starting with randomly taken variables [23]. Newton method uses the second derivatives of the loss function to find better training directions [24]. Conjugate gradient is a learning style similar to gradient and newton. It performs the optimization with the Hessian matrix, which expresses the

local curvature of a multivariable function. Therefore, it provides fast conversion [25]. The Quasi-Newton method is a more complex version of the Newton method. Because it includes multiple operations as a calculation. Instead of the usual Hessian matrix calculation, it uses approximation and starts from the first derivative of the loss function. Therefore, it has to do a lot of iterations [26]. Levenberg-Marquardt is an optimization method that uses a sum of squared errors. This method gives faster results but does not work as memory efficient on large datasets [27]. In our approach, we selected MLP (Multi-Layer Perceptron) since we have different datasets with different behavior. The MLP model does matrix multiplication and addition sequentially. It has layers as seen in Figure 1. Every node has a value in vector space. The information flow takes place between the layers (with a forward orientation) with their weights, and eventually, the output is obtained. The method has universal approximation theory which enables us to compute functions under specific data. We selected Gradient optimization because we have several parameters also, combinations of them to find the optimal values and possible values can be correlated with each other. Hence, we find theta values that minimize the error we have.

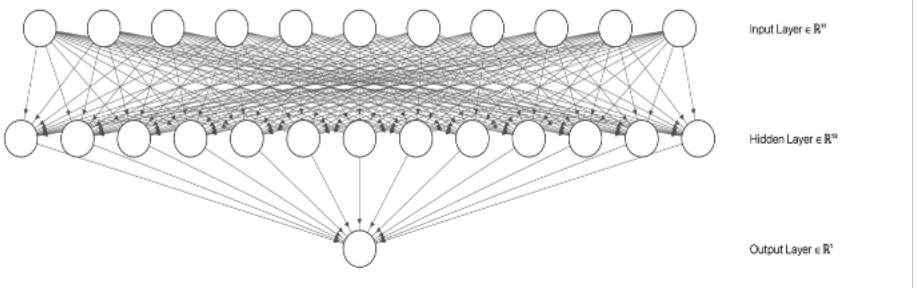

Figure 7: MLP Structure

CHAPTER IV

PROBLEM DEFINITION AND FORMULATION

Multilabel classification is a classification that involves multiple predicted labels. Instances can belong to one or many classes. The classification causes some problems in terms of positioning the labels for samples and for the past decades, multi-label classification problems have been growing enormously. There are many real-time examples of the issue such as movie categorization, gene functionality, text categorization, image definition, bioinformatics, and product categorization. According to the complexity of data, sometimes it is hard to define labels of the data and categorize them. In addition to this, labels can be correlated, they cannot be mutually exclusive from each other. Considering that the content of the movie includes drama, science fiction, and mystery. When a customer wants to search for a movie that contains drama and mystery, she/he goes to categories directly to find it. However, if the categorization is done incorrectly, most of the movies cannot be shown to the client. Although researchers are aware of the problem and try to solve it in an effective way, the exact solution is not found.

Besides, generally traditional algorithms that convert multiclass problems or consider binary classification problems for each label are applied to solve the problem. However, the algorithms could not focus on dimensionality, correlations between labels, and overfit circumstances. Another challenge of the multi-label classification problem is changes of the number of labels of the training dataset. Some datasets include a little number of labels. On the other hand, some datasets cover a large number of labels according to the whole dataset. At this point, algorithms that are selected or developed, cannot work effectively for each dataset. Thus, this paper not

only gives an insight into traditional algorithms for multi-label classification solution paradigms but also proposes a stacked ensemble solution by using kernel techniques on labels. As datasets, publicly available datasets are used. These are yeast, emotions, birds, and scenes.

4.1 Feature Engineering

Setting input features for the Neural Network method is the most crucial and challenging part in terms of achieving boosted results. We generate new features based on KNN, the symmetric euclidian distance between the train we use to teach the model, and the test datasets we set aside to test our algorithm. Normally, if the two-point distances are far from each other, they won't look that much alike. Calculation of the distance between data points can be applied by various methodologies such as hamming distance which calculates the distance between two binary vectors ([1,0,0] ; [0,1,0]), Euclidean distance calculates the distance based on the exact values of the vectors (float, integer), in the 2-D Euclidean distance calculation, the optimal length of the distances to the k nearest nodes is calculated. The nearest neighbor methods are greedy methods. That is, the choice that is possible and closest to the result is made. It simply suggests that when a choice has to be made, the choice that will bring the closest to the outcome is made. This is why they produce poor-quality lengths. If the parameter we call k is set to a large value, it increases the model complexity but makes the feature vector richer. Manhattan distance that computes the distance according to the sum of absolute differences between the vectors and Minkowski distance which is similar to the euclidean distance. Our datasets have a float and huge numbers. Hence, we select to use Euclidian distance. We calculate the distance between train data points themselves. In this way, k nearest data points indexes are achieved easily and KNN features are derived over the train label dataset.

In addition to the training dataset, we do not have the labels of the test data, so we use the matrix of the distances of the test data to the train data and calculate these features via train label data. After achieving the pairwise distances, we get the mean value of possible closest distances. Finally, we get the float numbers due to the average calculation of the distances. To convert the binary values, we push the numbers to 1 or 0 according to the high and low threshold values. At this point, we create arrays of possible effective values for both upper and lower thresholds. The upper values are 0.7, 0.8, 0.9, 1 and lower values are 0, 0.1, 0.2, 0.3. Also, we define possible k nearest points as a belief size array and try the combinations of the values. The size array includes from 5 to over 50. As the last step, we combine the binary-distance-based new features with the main data frame for each dataset separately and achieve the second input of the neural network.

4.2 Model Building

We examine this problem by getting benefits from KNN and neural network. We divide the problem into 2 parts.

The first one is getting pairwise distances of train and test data points in order to achieve mean values of k nearest distances. In other words, as a base step, we scale the train and test the dataset. Datasets may contain bad values and may be far from a normal distribution. We applied the scale method to all our datasets at the very beginning of the model to prevent possible data problems. We calculate the diagonal symmetric distance between trains and also test the dataset, then we get the average of the values of the k points closest to each data point in the feature train dataset. We calculate the distances with possible k numbers and as a final step of the input arrangement, due to the mean operation, result labels have float numbers, and we push these numbers 0 or 1 based on the possible upper and lower thresholds as we stated in the feature engineering part of the paper.

The second one is applying neural network structure with naturally trained datasets and KNN derived inputs. A multilayer perceptron (MLP) is a feedforward artificial neural network that produces a set of outputs from a set of inputs. An MLP is characterized by several layers of input nodes connected as a directed graph between the input and output layers. MLP uses backpropagation to train the network. The structure also enables to perform a learned transformation sequentially. In our algorithm, we use Gradient optimization due to large and possible correlated values. We put the naturally trained dataset into hidden layers with relu activation metric. The activation function is used to decide whether a neuron should be active or not.

We chose Relu (Rectified Linear Unit) which is a nonlinear function. If the features have minus values, the relu metric converts them from minus to 0. On the other hand, if the features have positive values, the metric transmits them directly. Instead of relu, we could use different activation functions such as ELU, Tanh, and Sigmoid. However, the main point of our dataset is having non-linear and correlated features. The function is less computationally expensive than Tanh and Sigmoid because it includes simpler mathematical operations. Besides, we do not want to have negative values because of the distance-focused algorithm. For instance, as long as we use linear, our model behaves as linear regression and non-linear relations between the values cannot be observed effectively. The below formula shows the "y" output with activation function by calculating the sum of the weights (w) of inputs (x) with adding bias (b) [28]. We generate 3 hidden layers by the first input.

$$y = Activation(\sum(ax + b)) \tag{3}$$

The final hidden layer is created by y train data which includes trained labels with relu activation. As a final output of the layers, we add the first input and the second input. In other words, we combined the two different generated inputs at the end of the hidden layer and gave them to the model as output. To compile the model, we specified some parameters such as adam optimizer, binary cross-entropy loss, and binary accuracy. Optimization methods are used to find the optimum value in the solution of nonlinear problems. Optimization algorithms such as stochastic gradient descent, adagrad, adadelta, adam, adamax are widely used in deep learning applications. There are differences between these algorithms in terms of success and speed.

The Adam optimization caches the momentum changes as well as the learning rates for each of the parameters [29]. Binary accuracy metric enables the algorithm to push 1 as long as the probability of a value is above the threshold; otherwise, push to 0. We use a model checkpoint to save a model and intervals of weights. With this approach, the model is loaded later to get the training from which state. Hence, we achieve the best weights to use in the model when we fit trained features and trained labels. In fact, we teach the model to match which label against which feature. As I mentioned before, the features consist of both known-derived labels and raw data, while labels belong to raw data labels. While doing this part of the model, we select epochs as 50. The model trains the data in chunks rather than all at once. After the first piece is trained, the weights are updated with backpropagation according to the success of the model. With the new dataset, the model is retrained, and the weights are updated. In summary, the most appropriate weights are tried to be given to the model at each step. These steps are called epochs. We set callbacks metric based on the best weight that we calculated by model checkpoint and finally we do this process with different validation splits. Validation split allows Keras to train and test split data entirely on its own. This prevents the model from overfitting,

that is, memorizing the dataset, and optimizing the model. By trying this process with different batch sizes, we find the most suitable parameters for our model. After training the model, we get the predictions of our test dataset. The brief code of the whole structure is shown below. (D: Datasets, C: value combinations of the parameter tunning, I: Iterative Stratification, X,Y: part of the dataset, ta: training part, te:test part, m:model, r:result, k: keys from the combinations, i:iterations)

Algorithm 4: *The structure of the algorithm*

```
for d in D do
    for c in C do
        k-fold = I(n,0)
        for t_a,t_e in k-fold.split(X,Y) do
            r=m(X[t_a], X[t_e], Y[t_a], Y[t_e], c)
        end for
    end for
end for
```

4.3 Evaluation Metrics

After completing the model, we evaluate the results of the models. There are several metrics that measure the effectiveness of models such as accuracy, AUC score, recall, precision, F1, hamming loss. In this work, accuracy is not the single metric that measures the effectiveness of the model. Also, due to using more than one dataset, we cannot remain limited to the same metric. Because some dataset is more imbalanced than others and the model could be biased. In other words, some data can dominate the dataset.

The metrics that we used in our datasets are accuracy, F1 score, and hamming loss. The accuracy metric measures the correctness of predicted values. In other words, accuracy is the ratio of correct predictions number to the total number of inputs. Thus, when the accuracy score is higher, the model effectiveness is higher.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

F1 score measures the test's accuracy. The metric is the result of the harmonic mean of precision and recall. In other words, the metrics imply how many samples the model classifies correctly and do not catch a significant number of samples. The metric gets values between 0 and 1. When the number gets closer to 1, that means, the score results in better performance.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{5}$$

Hamming Loss focuses on wrong, penalized labels prediction. The metric is the ratio of wrong labels to the total number of labels. In multi-label classification, the metric is calculated by the hamming distance between actual y and predicted y. Because of the focus on penalization, when the hamming loss score is lower, the effectiveness of the model gets higher performance.

$$HammingLoss = \frac{1}{m} \sum \frac{Y}{Q} \cap Z \tag{6}$$

CHAPTER V

RESULTS

For all datasets, we conducted feature engineering, achieved new inputs by KNN algorithm, and, built Neural Network model for them in Chapter 4. There are several algorithms for the multi-label classification problem and lots of methodologies are developed which are stated in Chapter 3, in the background part. Table1 and table2 show the analysis of the results of both our algorithm and benchmark algorithms. The best results are stated as bold characters and experiment results are shown based on the mean values of each algorithm in terms of Accuracy, Hamming Loss, Rankin Loss, F1, F1 macro, and F1 micro.

Table 5: Result table with benchmarks

Dataset	EBR	ECC	EPS	RAkEL	CDE	AB	MLS	MLWSEL1	MLWSEL21	KNN-NN
						Accuracy				
Emotions	0.517	0.532	0.522	0.422	0.524	0.028	0.422	0.806	0.807	0.812
Scene	0.605	0.659	0.642	0.534	0.538	0.000	0.534	0.917	0.915	0.909
Yeast	0.489	0.505	0.491	0.544	0.478	0.335	0.434	0.804	0.801	0.806
Birds	0.593	0.602	0.589	0.568	0.5888	0.456	0.568	0.949	0.955	0.956
						Hamming Loss				
Emotions	0.197	0.205	0.211	0.264	0.212	0.306	0.264	0.194	0.193	0.187
Scene	0.093	0.094	0.099	0.135	0.136	0.179	0.135	0.083	0.085	0.090
Yeast	0.205	0.210	0.212	0.248	0.228	0.232	0.249	0.197	0.199	0.193
Birds	0.042	0.043	0.046	0.051	0.047	0.053	0.051	0.051	0.045	0.043
						Ranking Loss				
Emotions	0.171	0.171	0.196	0.316	0.176	0.427	0.326	0.159	0.149	0.164
Scene	0.079	0.092	0.101	0.195	0.138	0.472	0.227	0.068	0.069	0.087
Yeast	0.185	0.191	0.202	0.336	0.219	0.363	0.316	0.171	0.168	0.171
Birds	0.098	0.111	0.140	0.199	0.134	0.229	0.168	0.120	0.110	0.134
						F1				
Emotions	0.597	0.612	0.615	0.509	0.608	0.037	0.509	0.639	0.614	0.689
Scene	0.620	0.675	0.655	0.573	0.573	0.000	0.573	0.708	0.672	0.730
Yeast	0.599	0.611	0.599	0.556	0.595	0.456	0.556	0.647	0.625	0.740
Birds	0.618	0.631	0.616	0.603	0.621	0.456	0.603	0.152	0.140	0.541
						F1 - Macro				
Emotions	0.639	0.641	0.631	0.551	0.635	0.038	0.551	0.608	0.584	0.670
Scene	0.709	0.728	0.707	0.634	0.629	0.000	0.634	0.700	0.665	0.731
Yeast	0.385	0.398	0.374	0.383	0.405	0.122	0.384	0.619	0.593	0.438
Birds	0.321	0.291	0.265	0.349	0.336	0.053	0.349	0.141	0.133	0.316
						F1 - Micro				
Emotions	0.662	0.663	0.654	0.564	0.654	0.063	0.564	0.664	0.658	0.656
Scene	0.705	0.718	0.700	0.624	0.617	0.000	0.624	0.750	0.733	0.725
Yeast	0.628	0.636	0.625	0.581	0.617	0.480	0.581	0.644	0.621	0.651
Birds	0.431	0.450	0.402	0.444	0.456	0.000	0.444	0.365	0.359	0.480

When the experiment results are analyzed, we can state that our algorithm mostly gives the best results. Especially in terms of accuracy and hamming loss metrics, our algorithm outperforms benchmarks.

Comparing with bagging algorithms which are EBR, ECC, EPS, RAkEL, and CDE, our algorithm outperforms according to the most evaluation metrics. Our algorithm focuses on the possible labels based on the pairwise distance and deals with the correlations that's why the algorithm outperforms the traditional bagging strategies.

If we analyze the stacked ensemble methods (MLS), we boost the results of the algorithm in such cases. The reason is that our algorithm treats each label with different importance and is sensitive to probability calculation with different behaviors. From the weighted ensemble algorithms side (AdaBoost), in order to be able to label correctly, we added our distance-oriented possible labels calculated with KNN to our features with feature engineering, and thus we finalized the probability of the label that the feature may belong to.

Comparing with the weighted ensemble method by regularizations (MLWSE-L1, MLWSE-L21)[7], we boost the result of them at some metrics. The first main reason is creating new features based on the pairwise distance and instead of training these features, we stacked at the end of the Neural Network structure. The second fundamental reason is that we use Neural Network due to the extensible layer features of the algorithm and it enables us to modify the layers. In other words, the ensemble strategy has universal approximation theory, and thanks to this feature, we understand the functionality of each dataset separately. Also, we design the whole structure as a sequence with multiple combinations of values to optimize the algorithm. Meanly, we generate calculation pairwise distances and NN structure as a sequence, additionally, giving multiple combinations of values returns us as the most optimized and outperformed evaluation metrics. The below figures show the comparison of each metric for

each dataset visually.

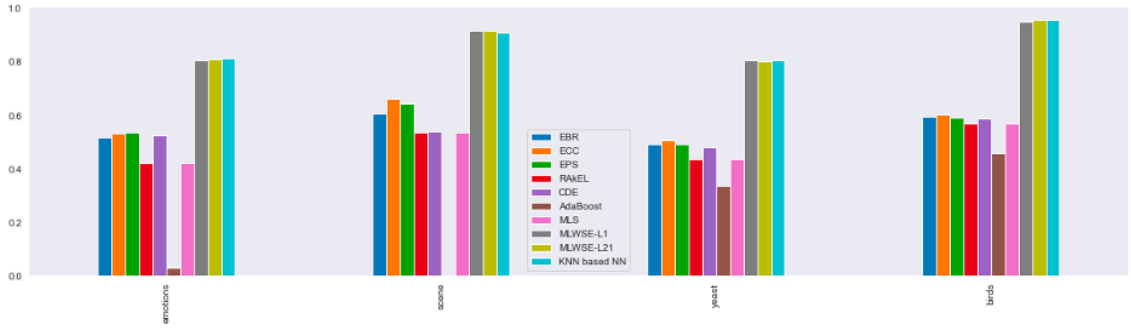

Figure 8: Accuracy histogram of algorithms

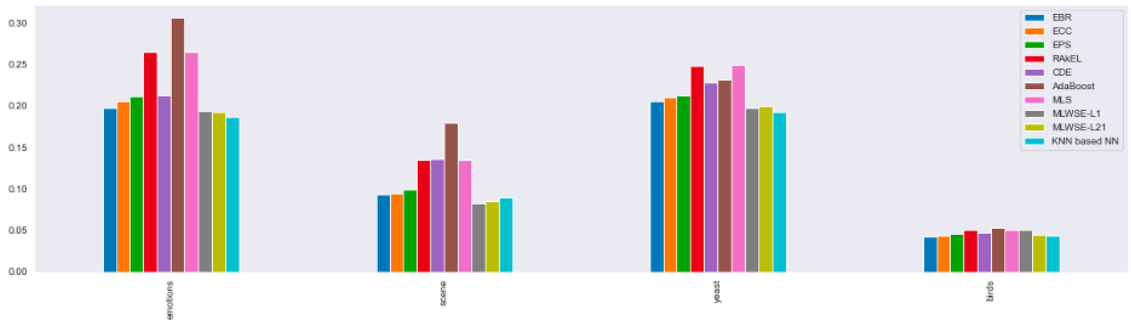

Figure 9: Hamming loss histogram of algorithms

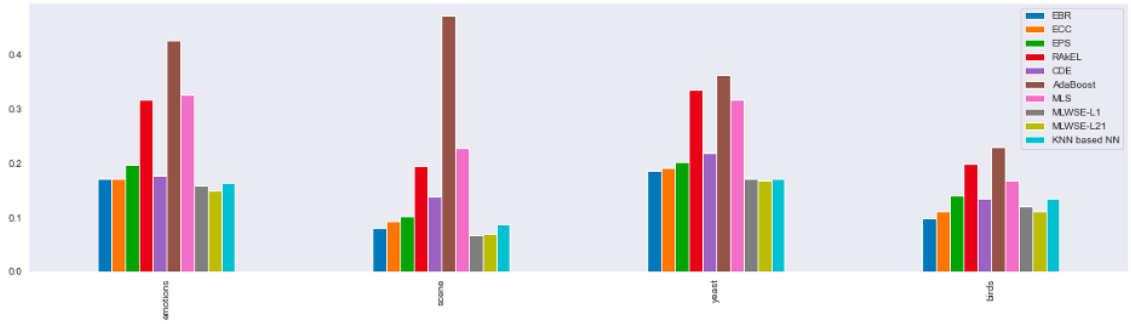

Figure 10: Ranking loss histogram of algorithms

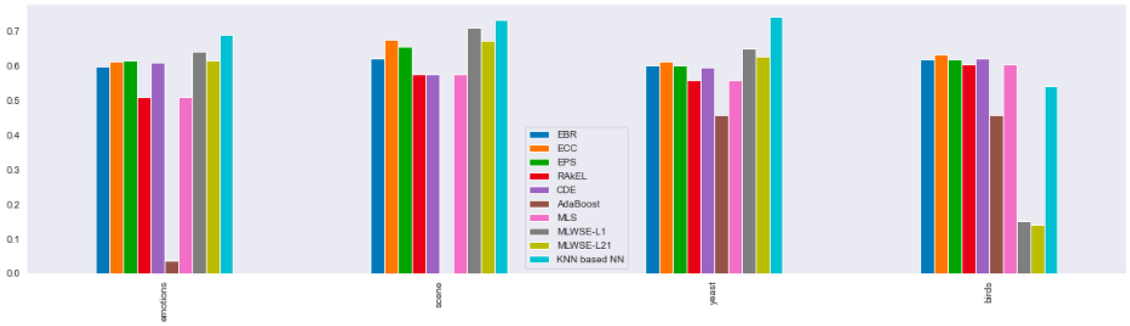

Figure 11: F1 score histogram of algorithms

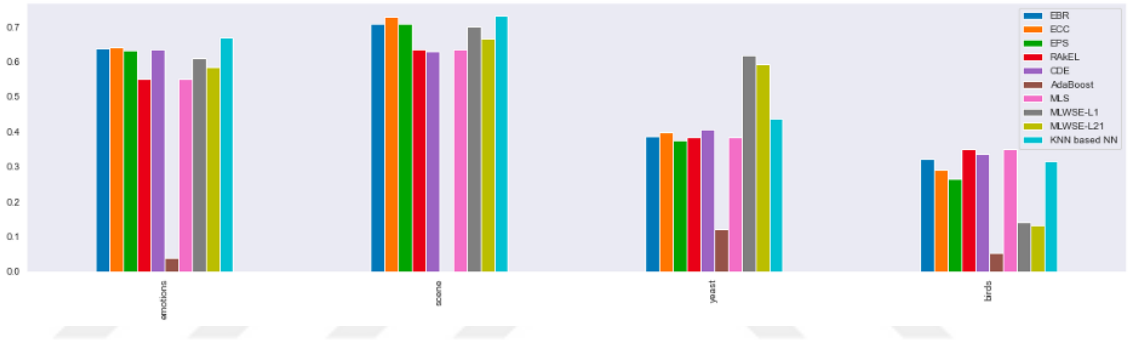

Figure 12: F1 macro score histogram of algorithms

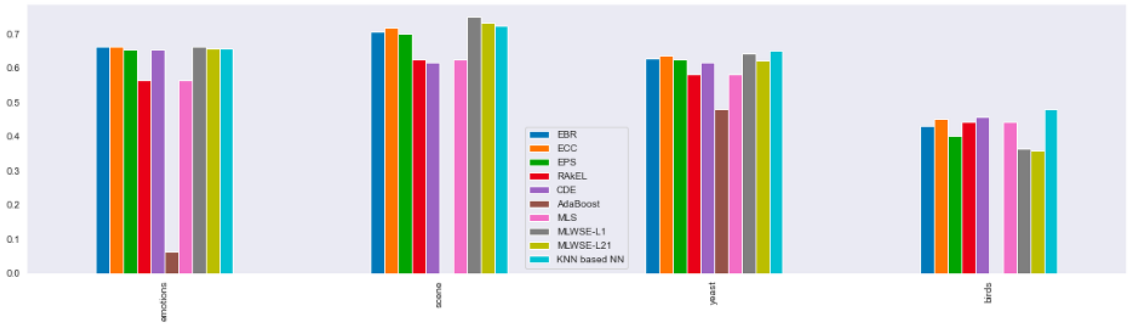

Figure 13: F1 micro score histogram of algorithms

According to the results, In the accuracy, the top 3 outperformed algorithms are MLWSE-L1, MLWSE-L21, and KNN based NN algorithms. Our algorithm boosts the weighted stacked ensemble method in emotions and birds datasets. In Hamming loss metric, our algorithm gives the best result except for the scene dataset. Ranking Loss, EBR outperforms in birds dataset and Regularization based algorithms give the effective results. When the performance of the models is compared; (F1, F1-Macro, and F1-Micro) our algorithm outperforms on emotions and scene datasets. However, ECC outperforms the birds dataset. When the F1 Macro and Micro evaluation metrics are deeply analyzed, the macro gives the boosted results in the emotions and the scene. On the other hand, micro returns as the outperformed results in the yeast and the birds datasets.

CHAPTER VI

CONCLUSION

In this paper, we developed a distance and neural structure-based algorithm for the multi-label classification problem. While performing the algorithm, we designed the distance calculation structure of the features to the labels as sequential with our original model structure. Our use of neural structure allowed us to observe the functional structure of the datasets well. In addition, this structure allowed us to add our binary labels, which we calculate based on distance and give as input to the main dataset, to our model in the last layer, without training them again. We ran the structure we created with the most optimal parameters in order to fine-tune the classification performance and algorithm optimization balance. When we compared our method with our benchmarks, we observed that we were competitive with state-of-the-art results, and even outperformed in some datasets and especially in model performance metrics.

REFERENCES

[1] Y. S. B. Zhang and W. N. Street, "Ensemble pruning via semi-definite programming.," *Journal of Machine Learning Research 7*, vol. 10, no. 24, pp. 1315–1338, 2006.

[2] Z. He, J. Wu, and P. Lv, "Multi-label text classification based on the label correlation mixture model," *Intelligent Data Analysis*, vol. 21, no. 23, pp. 1371–1392, 2017.

[3] R. E. S. Barutcuoglu, Zafer and O. G. Troyanskaya, "Hierarchical multilabel prediction of gene function," *Bioinformatics*, vol. 22, no. 7, pp. 830–836, 2006.

[4] C. Sanden and J. Z. Zhang, "Enhancing multi-label music genre classification through ensemble techniques," *In Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, vol. 12, no. 21, pp. 705–714, 2011.

[5] J. J. Y. Yang, "Adaptive bi-weighting toward automatic initialization and model selection for hmm-based hybrid meta-clustering ensembles," *IEEE Transactions on Cybernetics*, vol. 49, no. 5, p. 1657–1668, 2019.

[6] Z. Z. M.L. Zhang, "A review on multi-label learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 8, p. 1819–1837, 2014.

[7] Y. Y. Yuelong Xia a, Ke Chen b, "Multi-label classification with weighted classifier selection and stacked ensemble," vol. 557, no. 18, pp. 421–442, 2020.

[8] C. Sanden and J. Z. Zhang, "Enhancing multi-label music genre classification through ensemble techniques," *In Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, vol. 12, no. 21, pp. 705–714, 2011.

[9] K. C. J.M.Moyano, E.L. Gibaja, "Review of ensembles of multi-label classifiers: Models, experimental study and prospects, information," *Information Fusion*, vol. 44, no. 16, pp. 33–45, 2018.

[10] G. H. J. Read, B. Pfahringer, "Multi-label classification using ensembles of pruned sets, in: Proceedings," *IEEE International Conference on Data*, no. 15, pp. 995–1000, 2008.

[11] G. J.Read, B.Pfahringer, "Classifier chains for multi-label classification, machine learning," *Machine Learning*, vol. 85, no. 14, p. 333–359, 2011.

[12] I. V. G. Tsoumakas, I. Katakis, "Mining multi-label data, in: Data mining and knowledge discovery handbook," *Data Mining and Knowledge Discovery Handbook*, no. 13, p. 667–685, 2009.

[13] E. S. G. Tsoumakas, A. Dimou, "Correlation-based pruning of stacked binary relevance models for multi-label learning," *Proceedings of the 1st International Workshop on Learning from Multi-Label Data*, no. 12, p. 101–116, 2009.

[14] H. T. L Wang, H. Shen, "Weighted ensemble classification of multi-label data streams, in: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics," *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, vol. 10235, no. 11, p. 551–562, 2017.

[15] Y. S. R.E. Schapire, "Boostexter: a boosting-based system for text categorization, machine learning," *Machine Learning*, vol. 39, no. 11, p. 135–168, 2000.

[16] J. M. Rafał Grodzicki and L. Wang, "Improved multilabel classification with neural networks," vol. 5199, pp. 409–416, 2008.

[17] W. DING and S. Wu, "Abc-based stacking method for multilabel classification," *Turkish Journal of Electrical Engineering Computer Sciences*, vol. 27, pp. 4231–4245, 2019.

[18] Y. Huang, B. Giledereli, A. Köksal, A. Ozgur, and E. Ozkirimli, "Balancing methods for multi-label text classification with long-tailed class distribution," pp. 8153–8161, 2021.

[19] Y.-H. H. Elvis Saravia, Hsien-Chi Toby Liu, "Carer: Contextualized affect representations for emotion recognition," *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, no. 10, pp. 3687–3697, 2018.

[20] A. Clare, "Machine learning and data mining for yeast functional genomics," *IFAC Proceedings Volumes*, vol. 34, no. 5, 2003.

[21] J. K. V. K. F. Ranalli, "Scene classification with convolutional neural networks," no. 8, 2016.

[22] B. M. Brhanie, "Bekalu mullu brhanie, multi-label classification methods for image annotation," no. 7, 2016.

[23] N. E.M.Dogo, O.J.Afolabi and C.O.Aigbavboa, "A comparative analysis of gradient descent-based optimization algorithms on convolutional neural networks," *2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*, no. 6, pp. 92–99, 2018.

[24] T. M. Patrik KAMENCAY, Miroslav BENCO, "A new method for face recognition using convolutional neural network," *Advances in Electrical and Electronic Engineering*, vol. 15, no. 6, 2017.

[25] L. L. Bingjie Zhang, Tao Gao and Z. S. . J. Wang, "An improved conjugate gradient neural networks based on a generalized armijo search method," *Neural Information Processing*, no. 5, 2017.

[26] H. Ninomiya, "A novel quasi-newton-based optimization for neural network training incorporating nesterov's accelerated gradient," *Nonlinear Theory and Its Applications, IEICE*, vol. 8, no. 4, 2017.

[27] K. Bilski, Jarosław, "Algorithm for learning feedforwad neural networks," *Advances in Artificial Intelligence – IBERAMIA 2004*, no. 3, pp. 767–777, 2004.

[28] A. F. Agarap, "Abien fred , deep learning using rectified linear units," *ArXiv*, vol. abs/1803.08375, no. 2, 2018.

[29] Z.Zhang, "Improved adam optimizer for deep neural networks," *IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, pp. 1–2, 2018.

VITA

Sezin Ekşioğlu graduated from Istanbul Anatolian High School in 2014. She earned her B.Sc. degree in Computer Science from Özyeğin University in June 2019. She has been pursuing her M.Sc. degree in Data Science at Özyeğin University since September 2019, under the supervision of Dr. Okan Örsan Özener and Dr. Başak Altan Özener. Her research interests include ensemble classification strategies.