Use Bimodal Distribution Removal and Deep Neural Network in Letter Recognition Data Set to predict the 26 capital letters.

U6341832

Pengze Bian

Abstract

McCulloch, W. and Pitts, W. (1943) created a neural network computational model based on mathematics and an algorithm called threshold logic. Since then, more and more people have tried to use neural networks for deep learning, and various methods for improving neural networks have emerged in an endless stream. The aim for this report is trying to achieve a classification task to predict the 26 capital letters. In this report, there are two different neural networks have been built to predict the same attribute. One is the NN which is the basic neural network model and the other one is DNN (Deep Neural Network). That is mean one classifier need to be trained to obtain the prediction. In this report, pre-processing methods, Bimodal Distribution Removal will be introduced and implemented to predict a real dataset, Letter Recognition Dataset. Moreover, some techniques had contributed to the results. The accuracy of prediction has improved by these methods. Compare to the research (Pavlov, D., Popescul, A., Pennock, D. M., & Ungar, L. H. 2003), the best accuracy is 76.62 with the mixture of maxent models on 5 components. The best training accuracy is 71% and the best testing accuracy is 61% from this paper. There is still a big gap between the results from this report and the literature.

Keywords: Classification, 26 capital letters, Bimodal Distribution Removal

Introduction

In this research, Letter Recognition Data Set which has 20000 instances, 16 attributes and one target attribute is chose. The objective is to identify 26 capital letters in the

English alphabet. Each stimulus was converted into 16 primitive numerical attributes which were then scaled to fit into a range of integer values from 0 through 15. The training set is the first 16000 items and then set the test dataset for the remaining 4000. The reason for choosing this dataset is there are enough instances for classification task. 26 target outputs are more difficult for a classifier than two classes classification task, however, which can better detect the ability of neural network classification. Moreover, the dataset has converted attributes to numeric values. Thus, it will be efficient because few works should do in pre-process. One problem is that when reading the dataset csv field, there are no names for each column which may cause some problems which does not influence the result. The task is use the 16 attributes to get the expected letter. Through the neural network, the recognition of the handwritten letter is more accurate. First, loading the dataset and pre-processing the data. According to the different class, it might need to sample the data in a good method. The next step is predicting the result with the train data and test data. Several methods have been used to perform the analysis. Random sampling the samples in the test_size of 0.2. It should have pre-process, normalized data and standardized data, however, the result has no significant difference. Correlation between attributes, stratified sampling, cross validation to avoid the overfitting. After these methods, Bimodal Distribution Removal has been implemented to clean the outliers. Finally, different models have been built to predict the letters and several methods and values have been used to evaluate the results of models. All these methods will affect the result to some extent.

Method

The dataset has 20000 instances and 17 attributes. The target attribute is the first column. The methods will be introduced by the order of the code.

1. train_test_split method (contribute to NN)

The function train_test_split is going to get the random train data and test data. Before sampling these data, one thing should be considered is the different class may have different distribution in predicting. Thus, the dataset has been checked the numbers of different class have no difference generally. That is mean stratified sampling is unnecessary. Then the correlation between attributes has been shown [Figure 1]. The input features are preferably irrelevant. If the correlation of certain dimension inputs is too strong, then the weights that are connected to these input neurons in the network play a similar role. When training the network, the effort spent on adjusting the relationship between these weights is wasted.

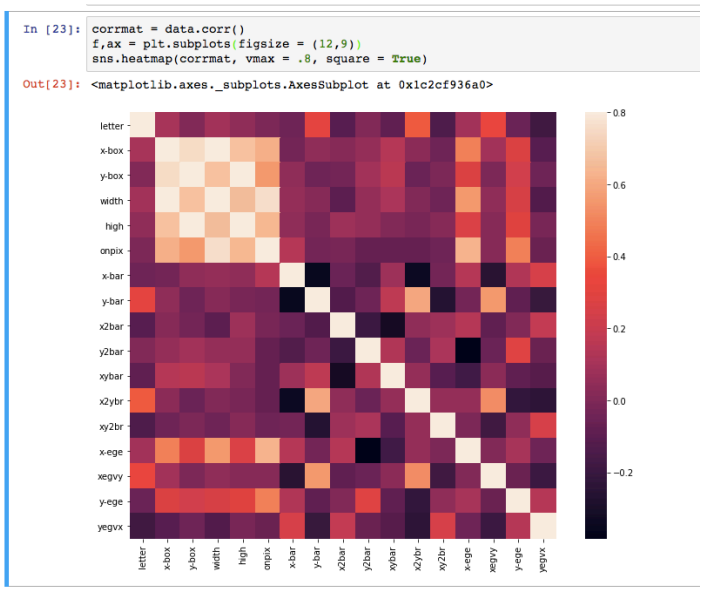


Figure 1 The correlation between attributes

2. Overfitting problem

Overfitting refers to making the hypothesis too strict to obtain a consistent assumption. Usually increase the amount of data and test sample set to avoid overfitting. Cross validation is a good method to avoid this problem. In this paper, the process of cross validation contains two steps. train_test_split gets the random train dataset and test dataset and then use logistic regression to get a score to evaluate [Figure 2].

```
lr = LogisticRegression(penalty='l1', C=0.1)
lr.fit(x_test_array, y_test_array)
print('Training accuracy:', lr.score(x_test_array, y_test_array))
```

```
Training accuracy: 0.71
```

Figure 2 Cross-Validation

The score shows the accuracy of this algorithm. According to this score, overfitting problem is not obvious.

3. Pre-process

(1) Normalize data

MinMaxScaler is a method to normalize data. This method performs a linear transformation of the initial data and maps a value V of attribute A to V` and V` ∈ [new_maxA, new_maxB] [Figure 3].

```
train_x=data[:,1:17]
train_y=data[:,0]

scaler = MinMaxScaler( )
scaler.fit(train_x)
scaler.data_max_
normorlize_x=scaler.transform(train_x)
```

Figure 3 Normalize data

However, the values of target attribute are in the range of 0 to 1 after normalizing data. All float values convert to long type after using Variable function which leads the values become inaccurate.

4. Bimodal Distribution Removal.

To get a good prediction, Bimodal distribution removal has been implemented [Figure 4]. It is one kind method for detecting the outlier in one dataset. The best advantage of this method is all the weaknesses of other outlier detection methods were addressed. This method is to generate frequency distributions of the errors for all patterns in the training set every 50 epochs in the training process. There is a standard of judgment is the normalized variance of errors over the training. When this variance is below 0.1, we need to calculate the mean error and

```
    for i in enumerate(trainloader,0):
        outputs = net(X)

        loss = criterion(outputs, Y)

        net.zero_grad()

        loss.mean().backward()

        optimizer.step()

        lossTensor = torch.cat((lossTensor, loss.data), 0)
```

```
# Apply BDR
if ((epoch % 50) == 0) & useBDR & (epoch != 0):
    error_df = pd.Series(lossTensor.numpy())
    subset_df = error_df[error_df > t_bias_mean[epoch]]
    threshold = subset_df.mean() + subset_df.std()
    remain_df = pd.DataFrame(data_train)[error_df <= threshold]
    remain_df = remain_df.reset_index(drop=True)
    remain_df = list(remain_df)
    trainloader = DA.DataLoader(remain_df, batch_size=batchSize, shuffle=False)
```

Figure 4 Bimodal Distribution Removal.

LossTensor is used to collect error in each epoch. Remain_df is the dataset after removing all patterns which error is bigger than the mean of subset plus the standard deviation. useBDR is a switch which controls BDR

take those patterns error greater than the mean error from the training set. Moreover, calculating the mean and standard deviation of this subset. We remove all patterns which error is bigger than the mean of subset plus the standard deviation which multiply by a number greater than 0 and less than 1 from the training set. Repeat the steps above until the normalized variance of errors over the whole training set is below 0.01.

5. NN (Neural Network with BP algorithm)

Define the numbers of inputs, hidden neurons, output neurons, learning rate and numbers of epochs. There are 16 attributes and the inputs should be 16. The task needs to predict 26 different letters and 26 outputs should be defined. The train dataset is 80% of the whole dataset which has about 16000 instances. Define a network layer that is not implemented in torch. nn to make the code more modular. This time it needs to expand our own torch.nn. Modules. For Neural Network, it is difficult to decide how many hidden units should have in this network. With too few hidden units, the model may not have enough flexibility to capture nonlinear features in the data (Tsaioun, k., Kates, S. A., 2011). The theory shows that too many variables can get better fitting results and even overfitting. At the same time, the theory shows that too many variables can be automatically eliminated. One function torch. utils. data. DataLoader helps you effectively iterate data. Each epoch's export data will be exported after randomly assigned.

6. DNN (Deep Neural Network with BP algorithm)

Deep Neural Network is a model of Deep learning. The neural network is based on the extension of the perceptron, and the DNN can be understood as a neural network with many hidden layers. The layers are fully connected, that is, any one of the neurons in the i layer must be connected to any one of the neurons in the i+1 layer. In this paper, DNN has been defined as one input layer, two hidden layers and one output layer. Input units is 16 because there are 16 attributes contribute to prediction.

128 units for first hidden layer and 64 units for second hidden layer. 26 output units for 26 different letters. The training set is 600 data and 200 data for test set. DNN train the data one by one and calculate the error for each data. That is why the training set is not 80% of the whole dataset (it will take long time for train).

7. The evaluation methods

(1) Accuracy.

In NN, the accuracy is 100* sum(correct) / total. In DNN, the accuracy is count / test_label. shape [0]. count is the number of correct predictions. Accuracy is the most direct criterion for evaluating this neural network.

(2) The error curves

For both networks, error curves have been plotted to show the trend of error with training times. Normally, the result is better if the error decreases a lot.

(3) Confusion matrix

The confusion matrix, also called the error matrix, is a standard format that represents the evaluation of precision and is represented by a matrix of n rows and n columns. Each column of the confusion matrix represents the prediction category. The total number of each column indicates the number of data predicted to be the category; each row represents the true attribution category of the data. The total number of data in each row indicates the data of the category. The number of instances. The values in each column indicate that the actual data is predicted to be the number of the class: as shown in the figure below [Figure 5].

One example: 43 in the first column of the first row indicates that 43 instances that actually belong to the first class are predicted to be the first class, empathy, and the 2 in the first row of the second row indicates that there are 2 instances that actually belong to the second category that were incorrectly predicted as the first category.

43	5	2
2	45	3
0	1	49

Figure 5

Results and Discussion

NN (Neural Network): All these processes have used train_test_split method and methods from Overfitting problems. (Table 1)

Input neurons	Hidden neurons	Output neurons	Learning rate	Number of epochs	Method	Confusion matrix	Accuracy (Training)	Accuracy (Testing)	The error curves
16	50	26	0.001	1000	normalized data		1.17%		
16	15	26	0.01	1000			12.73%		
16	15	26	0.01	2000			24.49%		
16	100	26	0.01	1000			41.77%		
16	100	26	0.01	2000			52.21%	50.2%	
16	33	26	0.01	2000	BDR, Dataloader	Figure 6	56.81%	54%	Figure 8
16	33	26	0.01	2000	Dataloader	Figure7	62.81%	61%	Figure 9

```
Confusion matrix for testing:

Columns 0 to 12
  133     2     0     0     0     0     0     0     0     0     0     0     2
    1   108     0     2     0     0     0     0     0     0     0     0     2
    0     0    84     2     5     2     2     0     1     0    15     2     4
    8    12     0   105     1     0     0     0     0     4     0     0     3
    0     6     9     0    77     0     2     0     2     0     2     0     0
    0    14     0     1     0    52     0     0     0     0     0     0     0
    4     7    40     3     3     0    43     0     0     0     3    24     9
    7     4     2    12     0     2     1     2     0     3    18     0     9
    1     6     0     1     0     0     0     0   109     3     0     2     0
    1     7     0     2     0     1     0     0    11   103     0     0     0
    4     1    15     3     4     0     1     0     0     0    44     6     1
    6     3     0     1     3     0     2     0     0     0     5   122     0
    3     0     0     0     0     0     0     0     0     0     0     0   149
    0     1     0     4     0     0     0     0     0     0     6     0     5
    6     0     0    22     0     0     1     1     0     0     3     2     3
    0     5     0     4     0    15     1     0     0     0     1     0     0
    5    12     0     3     0     0     4     0     0     0     3     0     2
    3    13     0     5     1     0     2     0     0     1     3     0     8
    7    29     1     0     0     8     0     0     5     4     0     3     0
    0     1     2     1     2     4     2     0     3     0     5     0     0
    0     0     0     1     0     0     5     0     0     0     1     0    15
    0     1     0     0     0     0     0     0     0     0     0     0     5
    0     0     0     0     0     0     0     0     0     0     0     0     8
    0     5     4     6     8     0     1     0    10     2     1     1     0
    0     2     0     1     0     4     0     0     0     0     0     0     1
    2     7     0     0     2     2     0     0     0     0     0     1     0
```

Figure 7 Confusion matrix

Figure 9 Error Curves

```
Confusion matrix for testing:

Columns 0 to 12
 131    4    0    0    0    0    0    0    1    4    0    1    4
   1  115    0    2    0    0    0    0    0    3    0    0    8
   0    0   75    2    3    4    5    0    1    1    3   19   13
  17   26    0   77    1    1    1    1    0   21    0    0    4
   0    6   20    0   40    1    1    0    4    2    2   23    1
   3   14    0    0    0   93    0    0    0    4    0    0    0
   7   13   47    1   19    0   17    0    0    0    0   26   14
  21    5    5    8    0    1    8    1    0    2   11    0   35
   0    6    0    0    0    0    0    0  113   13    0    3    0
  14   10    0    2    0    1    0    0   21   82    0    0    0
  12    2   46    0    9    0    1    0    0    0    4    1   25
   1    5    0    0    4    0    6    0    0    7    2  123    0
   6    0    0    0    0    0    0    0    0    0    0    0  156
   7    0    0    4    0    0    0    0    0    0    0    2   49
  17    2    0   65    1    0    2    0    0    1    0    9   14
   1    7    0    0    0   24    1    0    0    2    0    0    0
   7   23    0   11    0    0   11    0    0    1    0    3    4
  20   44    0    6    8    0    2    0    0    3    2    0   15
   9   49    0    0    0    8    0    0    0    9    0    3    0
   0    2    0    0    2   32    2    0    1    1    3    0    3
   0    0    1    0    0    2    2    0    0    0    0    0   23
   0    2    0    0    0    4    0    0    0    0    0    0    9
   0    0    0    0    0    0    0    0    0    0    0    0   13
   7    3    2    0    8    0    0    0   10    4    1    5    0
   0    3    0    0    0    6    0    0    0    0    0    0    4
   0   16    0    0    0    1    0    0    0    3    0    1    0
```

Figure 6 Confusion matrix

Figure 8 Error Curves

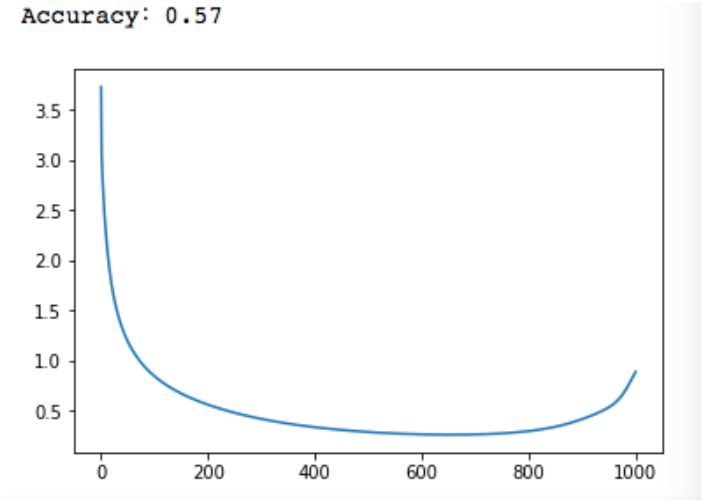
For NN model, analyze the last two rows in Table1. The function Dataloader helps you effectively iterate data which optimize the results. However, when BDR switch is True which means apply BDR method, the result has not improved. Meanwhile, the error curves show that the error from BDR is a bit higher than the normal training when the epoch is 2000. According to Slade, P. and Gedeon, T. D. (1993), Bimodal Distribution Removal is efficient when the dataset is known to be very noisy. If the dataset is clean and large, normal backpropagation well perform better. That is mean Letter Recognition Data Set is a very clean dataset, that is mean Bimodal Distribution Removal is not the efficient method to process the dataset. According to Pavlov, D., Popescul, A., Pennock, D. M., & Ungar, L. H. (2003), the accuracy of letter recognition is 76.62 when the mixture of maxent models on 5 components in Figure 10

Name		1	3	5	7	9	11	13	15
Letter recognition	A	72.42	74.65	76.62	76.47	76.45	76.35	76.07	—
	L	-1.069	-0.974	-0.862	-0.868	-0.840	-0.870	-0.914	—
	T	3004	11992	20800	29540	40801	45913	—	—
Yeast	A	51.67	54.00	53.33	50.00	50.67	52.00	55.67	54.00
	L	-1.264	-1.228	-1.248	-1.280	-1.256	-1.268	-1.237	-1.255
	T	33	76	128	188	259	310	376	434
MS Web	A	72.37	75.09	75.62	75.44	75.62	75.61	75.38	75.73
	L	-0.528	-0.504	-0.492	-0.491	-0.488	-0.487	-0.490	-0.485
	T	25	126	233	239	358	423	538	585
Vehicle	A	71.11	70.65	71.47	71.35	71.59	71.24	70.76	71.01
	L	-0.771	-0.767	-0.719	-0.736	-0.748	-0.754	-0.749	-0.743
	T	15	30	51	67	89	91	122	144
Vowel	A	43.03	49.89	52.12	49.29	51.31	52.02	49.39	50.10
	L	-1.665	-1.448	-1.465	-1.460	-1.471	-1.418	-1.451	-1.485
	T	9	49	80	83	105	116	113	240

Figure 10. Pavlov, D., Popescul, A., Pennock, D. M., & Ungar, L. H. (2003), the accuracy of letter recognition is 76.62 when the mixture of maxent models on 5 components.

Deep Learning Method: DNN (Deep Neural Network)

The training dataset is 600 data and test dataset is 200 data because the high accuracy of DNN comes at the cost of ultra-high computational complexity. The calculation engine in the usual sense, especially the GPU, is the foundation of the DNN. The platform for research of this paper is not good enough. The training dataset is 80% which has 16000 data, it will cost much time for DNN.



Compare to NN model, DNN preforms better according to the error curve.

Figure 11 600 data for training for DNN. Error Curve.

Conclusion and Future Work

The research uses different pre-process techniques and neural network models to obtain a better prediction. The main methods in this paper are Bimodal Distribution Removal and DNN model. Bimodal Distribution Removal has been used for Letter recognition dataset, however, the result is not improved because of the specialty of this dataset. Bimodal Distribution Removal is a good method which can optimize the prediction normally. DNN model need a good platform because the computational complexity. Overall, this paper has shown the effect of BDR. Compare to the traditional neural network (NN), Deep Learning model DNN has a better performance generally. In the future, A noisy dataset should be used to better prove the functions of different pre-process methods, especially Bimodal Distribution Removal. Moreover, apply the DNN model for the whole dataset should be done to get the final prediction to detect the advantages of DNN which compare to the traditional neural networks.

Reference

Letter Recognition Dataset:

David, J. S. (1991). Letter Recognition Data Set. UCI. Retrieved from: http://archive.ics.uci.edu/ml/datasets/Letter+Recognition

McCulloch, W. S., Pitts, Walter. (1943). A logical calculus of the ideas immanent innervous activity. The bulletin of mathematical biophysics. 5 (4): 115–133. ISSN 0007-4985. doi: 10.1007/BF02478259.

Pavlov, D., Popescul, A., Pennock, D. M., & Ungar, L. H. (2003). Mixtures of Conditional Maximum Entropy Models. ICML.

Slade, P. and Gedeon, T. D. (1993). Bimodal distribution removal. *New Trends in Neural Computation: International Workshop on Artifical Neural Networks.* pp 249-254. Retrieved from: https://link.springer.com/chapter/10.1007%2F3-540-56798-4_155

Tsaioun, k., Kates, S. A., (2011). ADMET for Medicinal Chemists: A Practical Guide. Retrieved from: https://books.google.com.au/books?isbn=0470922818