Leaf Classification Based on the Feature Discrimination with LSTM-RNN by Torch

Tingyu Pan

1ST ANU BIANNUAL BIO-INSPIRED COMPUTING STUDENT CONFERENCE,2018
Research School of Computer Science, Australian National University
U6259941@anu.edu.au

Abstract. The project of leaf image classification processing used the method of deep learning by training the processed dataset using Long Short Term Memory Recurrent Neural Network (LSTM-RNN). With 36 different leaf species as the class of 340 photographed leaf images translated into mathematical expressions as the attributes textually to constitute a pre-processed dataloader framework dataset. By using optimized methods as softmax function and bringing dropout layer to fix the overfitting problem, the classification results achieved an accuracy of approximately 92% for the normal feed-forward Neural Network. Theoretically the LSTM_RNN would improve the performance as the accuracy better than that.

Keywords: Classification; Neural Network; LSTM; RNN; Data Pre-processing.

1 Introduction

The real-world classification problem solving by machine has been a heated topic over the years. As the technology developed, diversity of classification expanded with image, voice, natural language, etc. The paper selected the plant leaf recognition problem to solve from the processed image information as numeric values by the work from Pauwels et al, which is more suitable and accurate for the supervised machine deep learning. A database of 36 different plant leaves with 340 instances were used with the sets of proposed variables as features. (Pauwels et al, 2009) [1]

With a massive amount of feature data, the paper chose deep learning technology as LSTM-RNN to learn the features of different leaves and gave a prediction of selected testing data. Moreover, as the amounts of attributes for input data of each instance could be considered as a sequence of feature value as a numerical sentence, the paper chose the method of recurrent neural network using the Backpropagation Through Time (BPTT) as an inspiration by the research of Gareth Oliver and Tom Gedeon [2] to learn the feature information to get a better performance. The purpose of the paper is to evaluate the accessibility of the NN to solve this plant recognition problem automatically with a high accuracy comparing to previous classification work. The network is trained from the pre-processed leaf dataset within a dataloader and partially tested from the same dataset which is selected randomly. Based on the Torch structure of Neural Network, the paper optimized the efficiency and performance with different optimizers and parameters. Also, using batch training to model the network make the complex hidden units easier to adjust the weights and bias values.

2 Materials and Methods

2.1 Geometric Leaf Features

This part of the paper will introduce how the unique visual geometric information of nature leaves were represented by a numeric dataset which can be expressed textually to analysis from the work of Pauwels et al by using mobile portal devices. Based on the natural features of the planets, the attributes of the leaves were divided in to two groups: shape and texture. Among the 15 attributes of the plants, shape features as eccentricity, aspect ratio and etc. were donated from the binary image parameter mathematically and textures from the 24-bit RGB images. The examples of leaf instances data tuple are showed as the figure below:

Eccentricity	Aspect Ratio	Elongation	Solidity	Stochastic Convexity	Isoperimetric Factor	Indentation Depth(Max)
0.72694	1.4742	0.32396	0.98535	1	0.83592	0.0046566
0.74173	1.5257	0.36116	0.98152	0.99825	0.79867	0.0052423
0.76722	1.5725	0.38998	0.97755	1	0.80813	0.0071573

Lobedness	Intensity(Ave)	Contrast(Ave)	Smoothness	Third Moment	Uniformity	Entropy	Class(Species)
0.0039465	0.04779	0.12795	0.016108	0.0052323	0.00027477	1.1756	1
0.0050016	0.02416	0.090476	0.0081195	0.002708	7.48E-05	0.69659	1
0.010121	0.011897	0.057445	0.0032891	0.00092068	3.79E-05	0.44348	1
0.0086068	0.01595	0.065491	0.0042707	0.0011544	6.63E-05	0.58785	1

Fig. 1. Examples: Geometric leaf Features

By using the geometric features, it established the plant discriminant analysis mechanism. The paper tried cut off some texture attributes which are not so geometric from training. The irrelevant attribute as the order number of the leaves were deleted. The results showed that the accuracy of the network was slightly dropped by 1 to 2 per cent. This showed no evidence about how much the analysis relied on the pure geometric information so we kept all attributes.

2.2 Artificial Neural Network

On the purpose of automatically unsupervised learning of a 340*15 dataset for classification of different 36 species leaf, which is not a big sampling database. The paper chose the feed-forward Neural Network to train and test the data to evaluate the prediction of classes based on Torch network using Python. Divided into three layers as usual ANN do, the parameters as hidden units amount and training epochs, etc. was firstly defined for the simple feed-forward NN training with input size fitting the attributes. To enhance the efficiency and speed of training we used the automatic method of PyTorch as the embedding layer of the ANN as DataFrameLoader to integrate the original geometric numeric numbers into torch tensors, which presented more characteristic features for the hidden units to learn and train. Fourteen attributes of the geometric leaf data except the class are used as the input tensors to be embedded in to the data loader. The class attributes marked as numbers in order are the targets of the network training outputs. For this paper, each class of leaves has 10-12 instances and we divide the original dataset randomly into 70% as the training dataset and the rest 30% as the test to evaluate the performance of ANN by calculating the classification accurate percentage from the testing data output. The evaluation of the accuracy is based on the table of confusion matrix, which plotted the output predicted class and actual labelled class as a matrix and calculate the percentage of the correct prediction with actual data with statistical methods.

Besides, during the process of network training, the paper applied the method of mini batch training as 10 instances each batch instead of the whole dataset. This method can effectively improve the efficiency of training with less influence from the noises than stochastic gradient descent and have a more stable convergence. The following formula shows the update rule of mini batch training:

$$\theta := \theta - \eta \sum_{i=1}^{k} \frac{\partial E}{\partial \theta} \tag{1}$$

Where k is the mini batch size, E is the error function and θ as the parameters of the NN.

2.3 Deep Learning-LSTM_RNN

The RNN from this topic is using cross entropy as loss function and softmax as optimizer in the hidden units to fit the classification problem. To realize the LSTM-RNN, we discussed about the recurrent model under the context of torch framework. The principle of recurrent neural network is simply connected the hidden units as the previous output using as the lateral hidden unit input. We explored several recurrent neural network architectures for multilabel classification of time series and choose the simplest way to pass over all inputs in chronological order.

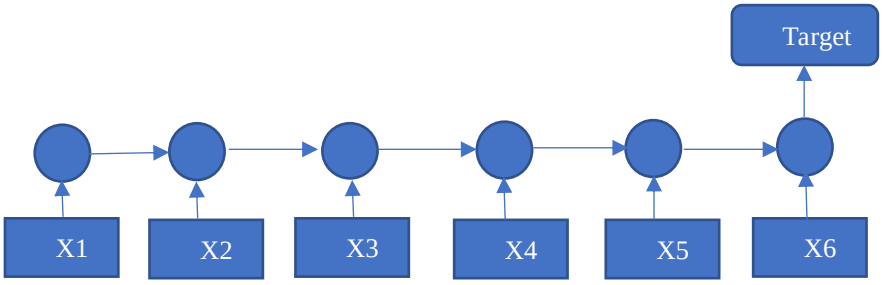

Fig. 2. RNN Model for multiple-label classification

The basic recurrent neural network architecture was indicated as the figure above. For each sequential row representing the leaf feature, we considered each attribute from this row as the input and the last attribute as the species as the output

target. The NN model only generated the output at the final step of the sequence. Therefore, the loss function only calculated at a single last sequence step and it was considered to be the average result of each label 'logged loss'.

Besides the part of RNN modelling, the paper also used the methods of embedding to integrate the input data for a better training. By adding the embedding layer to the torch model of RNN, each input was preprocessed as being mapped to a unique numerical vector in the vector space, which represent the characteristic features of the input more accurate and easy for hidden neuron to learn and train.

Due to the structure of the recurrent neural network, problem exists that it has long-distance temporary dependencies because of the accumulation of the weight matrix from step to step of the recurrent connection. The phenomenon of RNN is called the gradient vanishing. To fix this problem, we imported the Long Short Term Memory Layer (LSTM) consisted by multiple LSTM blocks assigned before each hidden neurons. Each block has three gates as the input, output and forget gates. The forget gates enable the LSTM unit to reset its state to learn continual tasks such as embedded Reber grammar.[3]

Except for all the methods mentioned above to optimize the performance of the accuracy of classification, a method of importing the regular method of deep learning as the dropout layer is used in this recurrent neural network based on the research by Vielzeuf,V et al[3]. The problem of deep learning RNN exists that the repeated learning could make the model over-learnt the training data as it can't distinguish test data other than the exact training data, which is called overfitting. Considering about how dropout method solve the overfitting problems with randomly limits of hidden units, the paper uses the conception of it to simulate the network reduction from the batch training to compare the network performance without reductions. Also, due to the sequential deep level training of the dataset, the dropout function could also be well solved the overfitting problem of the multi-label classification along with dropout.

The principle of dropout is simply generating a probability for the loss rate of the hidden units. By randomly picking hidden units refer to the amount of the proportion of loss rate, these hidden units are considered functionally 'closed' so that the update value would not be overlearning. Through this kind of network reduction, we are able to avoid the performance dropping problem of overfitting and abundant learning to improve the network efficiency.

3 Result and Discussion

The paper adjusts the parameters to lot as epoch times, hidden units amount, batch size, dropout rate, etc. to achieve a high performance, which is considered as the accuracy rate calculated from confusion matrix, to make a better classification of the leaves based on the discriminant features.

The realization part of the experiment was based on the pytorch framework, which can build the neural network by passing different parameters into it to adjust the performance of the model. We'll discuss about the functional performance from the results of both normal NN and LSTM-RNN.

Based on the character of the leaf dataset which has been processed in to text csv file, the paper chose exactly the amounts of leaf species and feature attributes as the output classes number and input size. To compare with the performance of similar research work in the field of plant recognition as Pedro F.B. Silva et al. did in 2013[3]. The leaf dataset is split the same as they did for the linear discriminant analysis with a randomized training dataset (70%) of the whole instances and the rest as the testing set. On the other hand, comparing to the 1000 iteral executions they used, the method here used as neural network set the epoch 1000 times as parameters but got a slight low performance with 85-87% accuracy compare to the LDA analysis with 87.3%. However, when epoch being reduced to 500 times, the network improves its accuracy to approximately 88-90%. This leads to the reduction for the hidden units with importing the dropout layer to fix the overfitting problem. After setting the dropout rate of the hidden layer as 50%, the paper raised the epoch back to 1000 times and achieved a better result of overall 92.5% accuracy, which is better than the misclassification rate of 13% with a more precise machine prediction about the leaf species based on the digital image data. In addition, a performance comparison between NN with dropout layer and without dropout layer is compared. From times of execution of the training and testing, it shows that the NN has a less misclassification rate around 2-4% when it has a dropout reduction function.

To enhance the feed-forward NN to a better training network, we change forward function of NN model to generate a recurrent structure of LSTM-RNN. The major realization process was showed as the codes below:

```
def init_hidden(self):

    return (Variable(torch.randn(1,batch_size,self.hidden_size)),
            Variable(torch.randn(1,batch_size,self.hidden_size)))

def forward(self, input):

    embeds = self.embedding(input)

    lstm_out, self.hidden = self.lstm(embeds.view(len(input),1,-1), self.hidden)

    output = self.fc(lstm_out.view(len(input),-1))

    out = F.log_softmax(output, dim = 1)

    return out
```

Fig. 3. LSTM-RNN realization codes

The conception of this structure was return the hidden units output results back to the next neuron in hidden layer as the inputs. So we paired up the output variable and next self.hidden result to control the sequential order. The final output as the last variable were processed by softmax function to generate the classification labels as result.

The deficiency of this model exists that the value type of the outputs varying influenced the process of training and had error to solve, which become a barrier to analyze the dropout rate influence for the further performance comparison with feed-forward neural network. Theoretically running with GPU by using .cuda() would fix the problem but it appeared that the edition currently used didn't support that. By changing the output tensor type in the training part with .long() or .float() also had a result of tensor type inconsistency of RNN.

Theoretically analyzing the results with assumptions, by using the LSTM-RNN for the leaf instances of each row in data file, the 15 attributes considered as a sequential sentence ought to have a better result compared to the overall 92% accuracy performance of feed-forward NN. However, with the problem of the datafile size not big enough for deep learning and overfitting problem, the enhancement of result is not supposed to be improved so much and need to be verified in the future work.

4 Conclusion and Future Works

The result of this paper shows that real world plant recognition could be solved automatically by machine networks from the geometric characteristic presentation with deep-learning. The accuracy of it is closed to an overall percentage of over 92% with a fundamental feed-forward Neural Network and to be improved with deep learning networks. The paper uses method to drop out random hidden units of network to fix the overfitting problem and promote the performance of network and LSTM to fix the temporary dependency problem to achieve a good result.

The possible disadvantage to this method of classification to be work out is that the prediction result of feed-forward neural network is still not stable enough. The reason may refer to the lack of some network reduction techniques as hidden unit pruning. For further work, it could use a more symmetrical method to reduce the excess units base on the characteristic weight adjustment within the hidden layer to gain a more stable performance. Moreover, the results show that considering more attributes with both shape and texture features has more accurate prediction of leaf classification in comparison with the LDA research. More features including color properties or feature correlation characteristic weight could be involved in the future work. Besides the basic data pruning issue to solve, the work to run the LSTM network well and make a detailed analysis is needed for future work. For instance, the influence from the sequential order as the leaves' features were queued in the sequence as a same order sentence for all leaf instance, I wondered that if change the order into a sparse sentence or based on the assumed influence of different features to arrange the sequence order would improve the performance. As the long-distance attributes could be not learnt well mostly, this work could be analyzed and discussed with the completeness of the major problem of LSTM fixed.

References

1. Pauwels, E.J., de Zeeuw, P.M., Ranguelova, E.: Computer-assisted tree taxonomy by automated image recognition. Eng. Appl. of AI 22(1), 26–31 (2009).
2. Oliver, G, Gedeon, T "Brain Computer Interfaces: A Recurrent Neural Network Approach", Australian National University (2010)
3. Vielzeuf, S. Pateux, and F. Jurie, "Temporal multimodal fusion for video emotion classification in the wild," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*. ACM, 2017, pp. 569–576.
4. Z. C. Lipton D. C. Kale C. Elkan R. C. Wetzel "Learning to diagnose with LSTM recurrent neural networks" CoRR vol. abs/1511.03677(2015)
5. Pedro F.B. Silva, Andre R.S. Marcal, Rubim M. Almeida da Silva, 'Evaluation of Features for Leaf Discrimination', Springer Lecture Notes in Computer Science, Vol. 7950, 197-204(2013).