An Innovative Application of the "Casper" Method: Optimizing the Structure of Deep Neural Networks

Yuliang Ma

Research School of Computer Science,
Australian National University,
u6462980@anu.edu.au

Abstract: Properly setting hyperparameters for "DNN" (Deep Neural Networks) to optimize its structure and performance has always been of great significance. "Casper" (Progressive RPROP Cascade Correlation) is an adaptive algorithm that can gradually build neural networks of the correct scale that matches the data set. Aiming to provide a new idea for building DNN, this article uses a database of the pupil diameter changes of 22 volunteers when they try to identify the authenticity of angry expressions and then designs an experiment to explore the effectiveness of an innovative composite network which combines both DNN and Casper. The results show that Casper is not suitable as an optimization algorithm for DNN. The method inspired by Casper will reduce the performance of the DNN by an average of 5.83%. The accuracy of the composite network is reduced by an average of 11.7% compared to the accuracy of the traditional DNN. This may be because that a few neurons that Casper added to the DNN earlier disrupted the balanced structure of the original DNN. And the weights in the DNN cannot be adjusted well. This problem may be solved by introducing more complex optimizer settings and connection mechanisms.

Keywords: DNN; Deep learning; Casper; Neural network structure

1. Introduction

1.1 Statement of Problem: Hyperparameter Setting for DNN

It is widely recognized that DNN, "LSTM" (Long Short-Term Memory) and "CNN" (Convolutional Neural Networks) are currently the most successful machine learning model structures used in data science and artificial intelligence fields. The latter two are both developed on the basis of DNN and therefore share the same principles. Therefore, it is of great significance to study how to improve the performance of DNN.

DNN is evolved from a multi-layer fully connected perceptron (i.e., neural network or “NN”) with a structure of three or more layers. To overcome the vanishing gradient phenomenon in NN, DNN usually use activation functions such as "ReLU" and "maxout" instead of "sigmoid". Ideally, if the hyperparameters such as the number of hidden layers and hidden neurons are reasonably set to fit the data set, DNN can be regarded as a black box that can simulate any unknown functions. However, in practice, it is often not easy to find parameters to build the good structure.

1.2 Potential Solution: the Casper Method

Casper is an improved version of “Cascor” (Cascade Correlation) algorithm. They are both neural network construction methods that gradually add neurons to the network while training. These methods contrast with the traditional method of directly designing the network structure before training. The intention of the approach is to make the neurons that are added earlier to the network learn as many data patterns as possible, so that the neurons that are added later can focus on reducing the loss that the previous neurons cannot further reduce. Therefore, the final model structure will match the complexity of the data (Kwok & Yeung, 1997).

The difference between Casper and Cascor is mainly reflected in how to add neurons to the network. Cascor “freezes” the weights of all the neurons added before, and then the training process will be divided into two stages: first train the weight of this newly added neuron to maximize the correlation between its output and the remaining loss. Then train the weights of all neurons connected to the output units to minimize the overall loss (Fahlman & Lebiere, 1990). Although Cascor has been proven to perform well (Fahlman & Lebiere, 1990), it may result in too many redundant units being added and therefore causes the model being too large because it always has some "frozen" weights. (Kwok & Yeung, 1993). And its performance on some regression and classification problems is sometimes not satisfactory (Hwang, You, Lay & Jou, 1996). Differently, Casper uses a gradient descent algorithm called RPROP as the optimizer (Treadgold & Gedeon, 2006). RPROP allows different units in the neural network to use different learning rates (Riedmiller & Braun, 2002). The basic idea is that when a new neuron is about to be added, the weights associated with it will be given the maximum learning rate. The earlier the neuron is added, the lower the learning rate it is given. Usually, the entire neural network will be divided into three regions for 3 different learning rates. During training, RPROP will calculate all weights in a manner like back propagation. Since there is no "frozen" unit in Casper, it is more flexible. Therefore, its performance in some tests, such as "Two Spirals Benchmark", is better than Cascor (Treadgold & Gedeon, 2006).

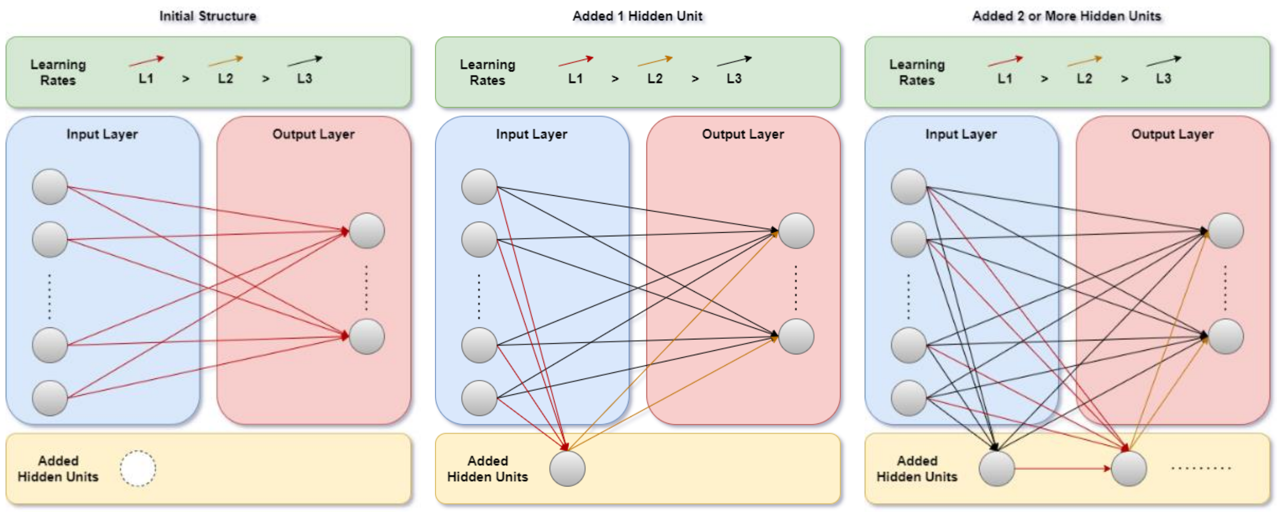

Fig. 1. The traditional Casper algorithm uses a simple fully connected NN as the initialization structure. Hidden neurons are gradually added to the network with different learning rates while training.

Usually, as shown in the picture above, the Casper network is initialized with a simple fully connected NN with no hidden neuron and is used independently to build the model from scratch. Since Casper can gradually increase the number of neurons to reduce the loss of previous training, this paper proposes the following question: Can Casper be used to further optimize a trained DNN and obtain a composite network to improve the performance? To answer it, we can use a trained DNN instead of a simple NN as the initial structure of Casper (As picture shown below), and then check whether Casper can further increase the accuracy that DNN cannot further improve.

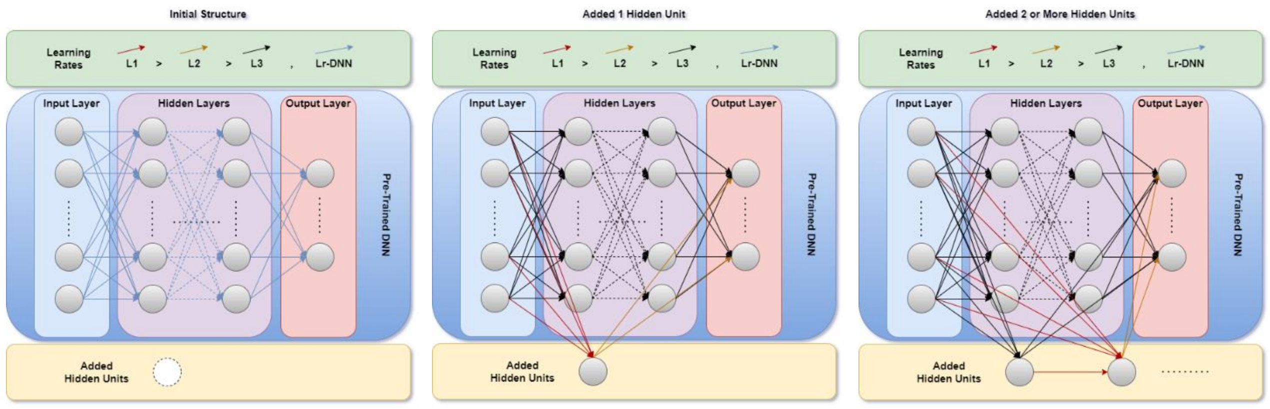

Fig. 2. This paper proposes a new type of composite network structure, that is, first train a DNN, and then use it as the initialization structure of the Casper algorithm. It is expected that the neurons added to the network using the Casper algorithm can further improve the performance of the DNN.

There are two possible results of the experiment:

- Increased accuracy: This will introduce a new reliable network structure to the industry.
- Decreased accuracy: This will show that Casper's mechanism of constructing networks is not compatible with DNN and the hypothesis fails.

The data set used in this paper comes from a short paper from the Australian National University (Chen, Gedeon, Hossain & Caldwell, 2017). It is pointed out in this article that changes in human pupils are more suitable to be used as indicators than changes in voices for judging whether people are angry or pretending to be angry. The prediction accuracy of the model built reached 95%, while the accuracy of human judgment for the same data set was only 60% (Chen, Gedeon, Hossain & Caldwell, 2017). This paper chooses the same data set because the data is not extremely patterned like the "Two Spirals" (Fahlman & Lebiere, 1990). Using highly patterned data will give Casper advantages, making the comparation with DNN less meaningful.

2. Method

2.1 Data Pre-processing

The "Anger" data set describes the pupil changes of 22 students watching 20 different videos. Each video has a label to indicate whether the angry expression in it is real or fake. This data set records the indicators of pupil diameter change of all volunteers watching every frame. After discarding invalid values and compressing the data set with a set of average methods, we will get a data set with 400 rows and 9 columns, where each row represents the data recorded when a single volunteer watching a particular video. The columns are shown in the following table:

Table 1. The content of the data set

Column Index and Name		Meanings
1.	(Empty)	Number of 20 valid volunteers (O0 - O20)
2.	Video	Number of the video (T1 - T10, F1 - F10)
3.	Mean	Mean of the pupil diameters
4.	Std	Standard deviation of pupil diameters
5.	Diff1	Changes in the diameter of the left pupil after watching the video
6.	Diff2	Changes in the diameter of the right pupil after watching the video
7.	PCAd1	Principal component of the left pupil data
8.	PCAd2	Principal component of the right pupil data
9.	Label	Label of the video (Genuine, Posed)

Data pre-processing process is as follows:

First, since the volunteer number is not directly related to the main issue, and there is a definite correspondence between the video number and the label, this paper deletes these two columns. Therefore, we will get 6 features with a label column.

Second, 0 - 1 encoding is used to transform the label.

Third, unlike the traditional method of using the "Sklearn" package to randomly divide the data, this article uses a special method for training, verification and test set division. This paper makes the proportions of true and false labels are each half in these three sets. Particularly, in the training set, make the labels appear alternately, thus constructing a balanced training set. This method has been proved that it can improve accuracy because the model will not be biased due to the uneven distribution of the data set.

2.2 Model Design and Hyperparameter Setting

As mentioned in the introduction, the model used in this experiment needs to be trained in two phases:

- Stage 1: Train a DNN model
- Stage 2: Use the DNN as initialization, and then use the Casper algorithm to continue adding neurons to the DNN.

The settings of the model in the two stages are shown in the following two tables:

Table 2. The settings in the first stage

Hyperparameter & Other Setting	Value	Reason
Number of hidden layers	3	Resulted from repeated experiments, using a smaller value to improve efficiency and avoid over-fitting and local minima.
Number of neurons in each hidden layer	120, 120, 4	The values of the first two layers are obtained by repeated experiments. The value of the third layer is set to 4 because this paper expects the model to be able to combine "Diff1" and "Diff2", "PCAd1" and "PCAd2".
Learning rate	0.2	Resulted from repeated experiments.
Number of epochs	2000	Resulted from repeated experiments.
Activation function	LeakyRelu	Resulted from repeated experiments and to avoid vanishing gradient
Loss function	CrossEntropy	The experiment is related to classification.
Optimizer	Adam	Resulted from repeated experiments.

Table 3. The settings in the second stage

Hyperparameter & Other Setting	Value	Reason
Optimizer	RMSprop, momentum = 0.1	It allows different parts of the network to use different learning rates. Add momentum to avoid local minimum.
Learning rates	L1: 0.2, L2: L1 / 4, L3: L2 / 50, Learning rates of the DNN part: L3	The first three learning rates are commonly used. Using L3 as the learning rate of the DNN part can make the change of the DNN small enough but not frozen.
Maximum number of hidden neurons allowed to be added	k = 15	The k value is obtained by verification. As shown in the Fig. 3., verification is performed every time a new neuron is about to be added. The verification loss is relatively stable before adding 50 neurons, and then may fluctuate sharply due to overfitting. As shown in the figure Fig. 4., the changes in the verification loss obtained before adding 15 neurons are relatively satisfactory.
Interval between checkpoints used to decide whether to add new neurons	15 + p * n, where p = 5 and n is the number of hidden neurons added	It is a commonly used value.
Maximum number of epochs	5000	Resulted from repeated experiments.
Batch size	80	Resulted from repeated experiments.

Threshold of the training loss drop	0.0015	Resulted from repeated experiments. If the training loss drops to a positive value and is less than this value, add a unit.
Maximum number of checkpoints that the model is allowed to stay at the same network state with the same number of hidden neurons	30	This is to prevent Casper from being trapped in a local minimum. If this happens, the training loss will continue to change between positive and negative values. When this happens more than 30 times, add a neuron.
Save point of the model	The 15th in previous 100 states.	When training stops due to the number of hidden neurons reaching the upper limit, the model may be trapped in the local minimum. At this point, the output state is not optimal, so this paper built a cache that can save the previous 100 network states and output the 15th network as the result when the training stops.
Activation function	LeakyRelu	Resulted from repeated experiments and to avoid vanishing gradient
Loss function	CrossEntropy	The experiment is related to classification.

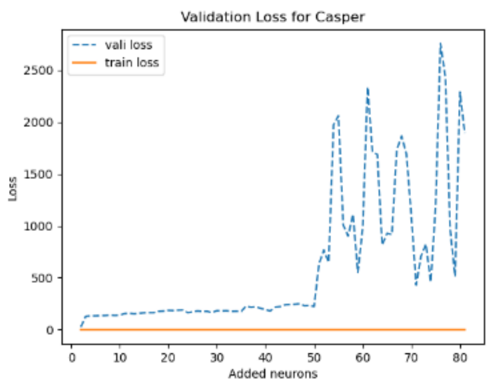

Fig. 3. Set the upper limit of the hidden neuron amount to 100 and preform validation each time a new neuron is about to be added. The verification loss changes smoothly when the number of neurons is less than 15.

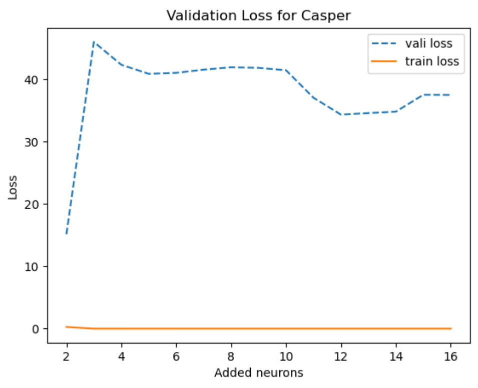

Fig. 4. Set the upper limit of the hidden neuron amount to 15 and preform validation each time a new neuron is about to be added. Then a typical curve is obtained as above. The verification loss shows a downward trend.

In addition to the above settings, this paper carefully specified the data sets used in the two stages. As mentioned in the data pre-processing section, this paper first divided the original data set into training set, validation set and test set. The ratio is set to 3:1:1. In the training set, 66.67% is used for DNN training, and the remaining 33.33% is used for subsequent Casper training. This paper assumes that this setting can best utilize the strengths of the algorithm.

To better evaluate the effectiveness of the DNN-Casper composite network, this paper uses two comparison methods:

- First, compare the accuracy of the composite network with the accuracy of the DNN model obtained in the first stage of training. This can indicate whether Casper in the second stage optimizes the DNN obtained in the first stage.
- Second, use 100% of the training set to train an another DNN and compare the accuracy of the composite network with the accuracy of this new DNN. This can explain whether this composite network is more effective. (Please note that after experimentation, the parameters of the new DNN mentioned here are set to be consistent with the DNN in the first stage.)

3. Results and Discussion

After training and testing the composite network and the DNN using the complete data set 50 times, the results are shown in the figure below:

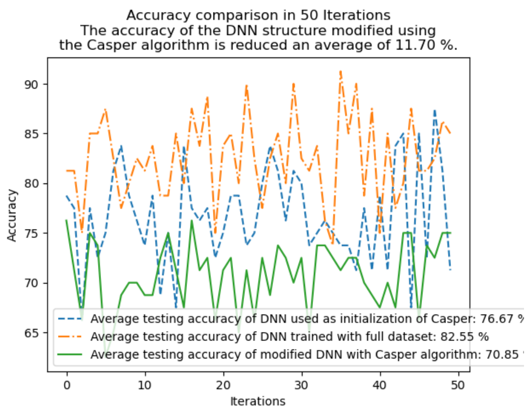

Fig. 5. Run the script 50 times and record the testing accuracy of the DNN in the first stage of the composite network, the DNN using the complete training set and the final composite network. Then plot a line graph.

The average accuracy of the conventional DNN model trained with the complete training data set reached 82.55%. The average accuracy of the composite network is only 70.85%, which is 11.7% lower than the former. This shows that the hypothesis in this article has failed, and Casper is not suitable as a tool to further optimize DNN.

Furthermore, we can find that in the first stage, the average accuracy of the DNN trained using part of the training data is 76.67%. It is unsurprising that this value is lower than the accuracy of the DNN obtained by using the complete database because the training uses less data. But after being modified using Casper in the second stage, this value further dropped by 5.83% to 70.85%. In the record of the 50 runs, there were only 9 times that the accuracy of the second stage was not lower than that of the first stage. This shows that Casper even reduces the performance of the original DNN.

When looking into the accuracy curve, we can see the mechanism of this process. Fig. 6. shows a typical change of verification accuracy along the number of neurons added in the second stage of a training. When Casper's first neuron was added to the trained DNN, the verification accuracy experienced a dramatic drop. Contrary to the hypothesis proposed earlier in this article, the new neuron construct a new channel between the input layer and the output layer, thereby destroying the balanced structure of the original DNN, rather than further optimizing it. Then as more neurons were gradually added to the network, the accuracy slowly increased, but it never returned to the original level because the weights of the original DNN have also been changed with the learning rate of “L3”. This shows that the Casper algorithm and the DNN structure are incompatible, and the two cannot be simply combined.

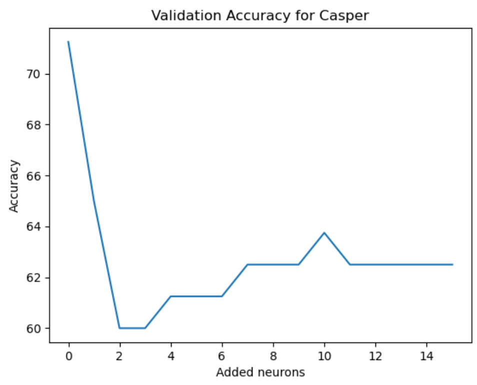

Fig. 6. Using a trained DNN as the initialization structure of the Casper algorithm, 15 hidden neurons are gradually added to the network. Record the validation accuracy before adding neurons each time. The accuracy dropped sharply when the first neuron was added, and then it increased, but never returned to the original level.

4. Conclusion and Future Work

Starting from the problem of selecting hyperparameters for DNN, this paper proposed a composite network structure using the idea borrowed from the Casper algorithm. This article designed an experiment using the "Anger" data set to test the performance. The results show that this new type of network is not very effective. Compared with traditional DNN, its accuracy is about 11.7% lower. The Casper algorithm will reduce the accuracy of a trained DNN by 5.83% on average. The reason may be that some neurons added by the Casper algorithm earlier will destroy the original balanced structure of DNN.

There are many points that can be improved in this new network structure. For example, in this article, the neurons added using the Caper algorithm are only connected to the input layer and the output layer. The hidden layers were optimized using the "L3" learning rate. This may result in decreased accuracy failed to recover or even surpass the previously level in the second stage. Maybe we can connect the newly added neurons to the hidden layers and use an even smaller learning rate "L4" to solve this problem. But in this situation, we must pay attention to the problems of overfitting and local minimum caused by adding too many hidden neurons.

As for the scenario about the data set, the accuracy of the best performing DNN in this article reached 82.55%, but there is still a gap between the 95% claimed by the authors of the source database. (Chen, Gedeon, Hossain & Caldwell, 2017) This may be because that this paper has used too many averaging methods in data processing, resulting in excessive data compression. We can improve accuracy by improving the processing method. In addition, if we do not discard the volunteer number column during data preprocessing, then for each video to be judged, we use the model to get 20 labels corresponding to the 20 volunteers instead, and then let these labels vote to get the result. This approach should be able to provide higher accuracy.

References

1. Chen, L., Gedeon, T., Hossain, M., & Caldwell, S. (2017). Are you really angry?. Proceedings Of The 29Th Australian Conference On Computer-Human Interaction. doi: 10.1145/3152771.3156147
2. Fahlman, S., & Lebiere, C. (1990). The Cascade-Correlation Learning Architecture. Advances In Neural Information Processing, II(1990), 524-532.
3. Hwang, J., You, S., Lay, S., & Jou, I. (1996). The cascade-correlation learning: a projection pursuit learning perspective. IEEE Transactions On Neural Networks, 7(2), 278-289. doi: 10.1109/72.485631
4. Kwok, T., & Yeung, D. (1997). Constructive algorithms for structure learning in feedforward neural networks for regression problems. IEEE Transactions On Neural Networks, 8(3), 630-645. doi: 10.1109/72.572102
5. Riedmiller, M., & Braun, H. (2002). A direct adaptive method for faster backpropagation learning: the RPROP algorithm. IEEE International Conference On Neural Networks. doi: 10.1109/icnn.1993.298623

6. Treadgold, N., & Gedeon, T. (2006). A Cascade network algorithm employing Progressive RPROP. Lecture Notes In Computer Science, (1240:733-742). doi: 10.1007/BFb0032532