Multi-Diseases Classification Including SARS-COV-1 : A Constructive Cascade Network Employing Progressive RMSprop with ResBlock using Temperature Information.

Xuan Feng

Research School of Computer Science, Australian National University, ACT, Australia
U6234544@anu.edu.au

Abstract. Cascade Correlation (CasCor) is a kind of Cascade network. It could improve the performance of a vanilla Neuron Network and automatically determine the net structure. However, due to the frozen weights connected to the hidden neurons in the CasCor algorithm, it may make the early hidden neurons to be poor feature detectors, and it is not allowed to fine-tune them. To deal with this problem, we use a Cascade Network Algorithm Employing Progressive RPROP (Casper) to modify CasCor, which allow us to fine-tune the hidden neurons. And to further the model's ability to capture new feature information and learning latent variables, we add a ResBlock layer to the model as an encoder to learn the latent features. In this study, our aim is to use our model to determine the optimal network structures and make the multi-disease classification including SARS-COV-1 given only the temperatures of the patient. If we only have the temperature information, the performances of vanilla neuron networks are not good enough due to the lack of features. Thanks to the hidden neurons installed by CasPer and the help of ResBlock, our model could capture latent feature information to improve performance. From the experiment statistics, we could find that the performance is improved. The validation accuracy could achieve 100.0% after several epochs and the validation loss is only about 0.00005 while the loss of simple CasCor is about 0.05.

Keywords: Classification, Neural Network, Feed Forward, Constructive Cascade Neural Network, CasPer, ResBlock

1 Introduction

Covid-2019 has spread all over the world and now been defined as a global pandemic. Recent studies [3] have found that COVID-19 not only has devastating effects on human lung tissues, but also attacks vital organs such as the heart, blood vessels, kidneys, gastrointestinal tract, eyes, and brain, causing very serious consequences. Many of us have been affected by this highly contagious disease since it is hard to determine that whether the individual is infected without some detailed medical test such as nucleic acid amplification tests [1]. Thus, it is necessary to figure out an efficient way for the patients and doctors to classify whether they are infected or not. Since Covid-2019 is like sars-cov which is named in 2003, I choose the dataset [2] which contains the fuzzy values of the temperature of the patients to determine whether one is got sars-cov by only focusing on his temperature. Thus, the problem in this paper is to predict the correct disease to the patient given his temperatures at different times.

Traditional fully connected feedforward neural networks have been shown that they have reasonable performances in such kind of non-linear classification tasks, but they may fail due to the difficulty of determining the optimal network structure. It is hard to determine the optimal network structures, and an unsuitable model will result in suboptimal problems like underfitting and overfitting, which will directly affect the prediction of the classification. To generate a neural network with proper structure and accurate classification, some constructive cascade algorithms such as Cascade Correlation (CasCor) and Cascade Network Algorithm Employing Progressive RPROP (CasPer) are introduced [5, 9].

The Cascade Correlation algorithm [5] is a kind of constructive cascade algorithm. It starts with a minimal fully connected network, whose input layer directly connected with the output layer. And every time the model is stuck in local minima, a hidden neuron is added to the network and connected to all previous hidden and input neurons. Before we add the new neuron, we maximize the correlation between the activation of the hidden neurons and the residual error of the network, these hidden neurons could be treated as feature extractors, which could allow the classification to fully make use of the features of the dataset. And after the hidden neurons are added, all the input weights to them are frozen.

A Cascade Network Algorithm Employing Progressive RPROP (Casper) [9] has been shown to be a powerful method for training neural networks. Similar to the CasCor algorithm, it installs one hidden neuron to the network every time the model is in local minima to form a cascade architecture. The difference to the CasCor is that CasPer do not use the correlation measure or weight freezing. Instead, RPROP is used to train the whole network each time a hidden neuron is added. We divide the weights in the model into three sets with different learning rates. Once a new neuron is added, the set and learning rates will be reset.

From the previous works, we could find that CasPer could generate a network with less hidden neurons and providing better performance. The reason is that CasPer do not use either the correlation measure between the new neuron and the output or the technique of freezing the weights connected to the hidden neurons.

A problem faced by constructive cascade algorithms such as CasPer is that the number of hidden neurons that are installed is potentially unlimited. The hidden neurons could be treated as feature extractors. To solve the problem, a ResBlock layer is introduced. It could be treated as an encoder, which could help the model to learn the latent variables in order to improve the generalization and the efficiency of the feature extracting.

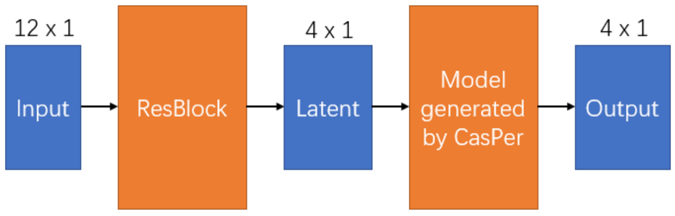
CasPer and ResBlock has been applied in this paper to achieve a proper network structure with good generalization to predict certain diseases given the temperature information in one day of the patients, that is the input is a vector containing the temperature information, and the output could be used to reveal what kind of disease it is likely to be.

2 Method

2.1 Network Topology

In this paper, we combine the ResBlock technique in ResNet [12] with CasPer. The model structure is shown in Fig. 1. The model could be divided into two modules. The first is a ResBlock and the second one is a model which would be extended by CasPer in the training process. The input is a vector containing the temperature information of a patient, and the output is the label of the patient. And 0 means normal, 1 means HighBP, 2 means pneumonia, 3 means SARS. The ResBlock is an encoder here, which allow the model to learn the latent variables from the input. The model generated by CasPer starts from a minimal network, and every new neuron installed can be treated as a feature extractor.

Fig. 1 The Network Structure



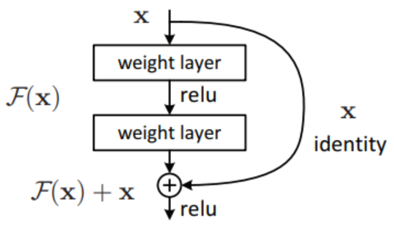
When it comes to the setting of the input, hidden and output neurons, we need to focus on the dataset used in this paper. The feature number of the input vector is 12, which tell us that the neurons in the input layer should be 12. The output of the ResBlock (also the input of the CasPer model) is related to the latent variables it should learn. By observing the dataset, we could find that the vector contains temperature information in 4 timestamps. This indicate that 4 is a suitable choice for the number of output neurons of the ResBlock. The number of classes in this multi-class classification is 4, so that the output layer consists of 4 neurons. And the output could be treated as the confidence of a class. Consider the output is [2, 1, -2, 0], it means that the confidence of this entry being class 0 is the greatest.

ResBlock

In ResBlock, we could suppose the input is a vector X, and the output is H(X). Instead of only having a weight layer one following another, we also have a skip connection. The skip connection is an identity mapping of the input X. And the weight layers could learn F(X), which is whatever we need to change about the input X. Thus, the ResBlock adds the learnt difference F(X) to the input X instead of directly learning the whole function H(X) which might be very complicated and hard to learn.

In the ResBlock introduced in the origin paper [12]. The skip connection uses the identity function since the input dimension is the same to the output dimension. While in our work, the ResBlock is used as an encoder, where the input dimension is larger than the output dimension. So, a linear function W is applied instead of an identity function. Thus, the output could be H(X) = F(X) + W*X.

Fig. 2 The ResBlock (from [12])

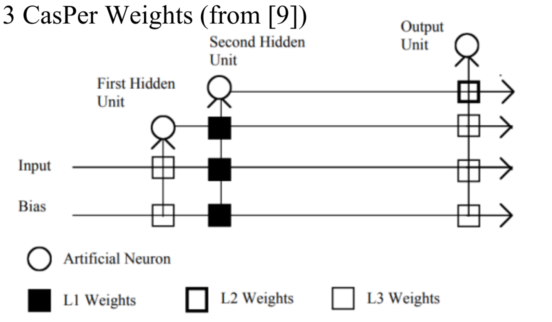

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

The ResBlock reuses the features and makes the training smoother and more effective. This could reduce the defects that the cascade network may generate a deep net with propagation delay. Since we learn the difference between the input and the output, the ResBlock will act as an identity function when the difference trends to be zero at the beginning. This could also help us to deal with the problem of overfitting.

CasPer

CasPer initializes with a minimal fully connected network consisting only of an input layer and an output layer. The input layer connected directly to the outputs. And every time the loss no longer decreases, a new hidden neuron will be added to the model. Then we will use a modified RPROP algorithm to continue to train the model.

Fig. 3 CasPer Weights (from [9])

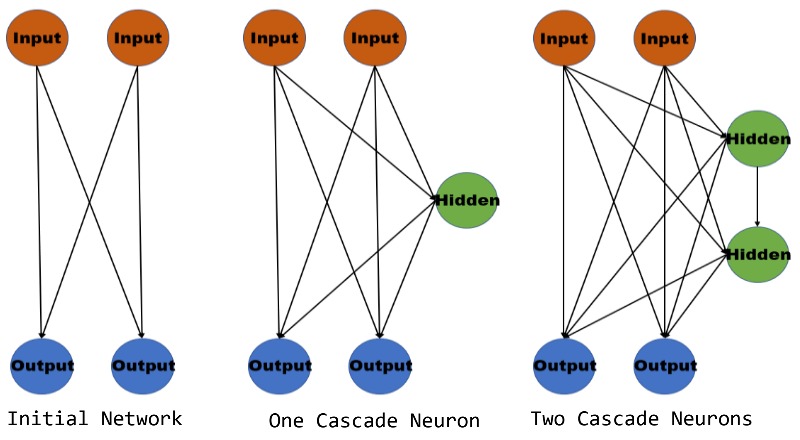


RPROP is a gradient descent algorithm. It uses separate adaptive learning rates for each weight. Each weight is assigned with an initial learning rate, and the learning rate is then adapted according to the sign of the loss gradient.

In Casper, the use of RPROP is modified. The model will be divided into three sets with different initial learning rates: L1, L2, L3 (shown in Fig. 3). The first set consists of all weights connecting to the new neuron from previous hidden and input neurons. The second set is made up of all weights connecting the output of the new neuron to the output neurons. The third set consists of the remaining weights, that is, all weights connected to, and coming from, the old hidden and input neurons. The values of L1, L2 and L3 are set such that L1 >> L2 > L3. The reason for this setting is similar to the idea of maximizing the correlation between the activation of the hidden neuron and the total error of the model in CasCor : the much higher value of L1 allows the new hidden neuron to learn the remaining network error. And having L2 larger than L3 allows the new neuron to reduce the network error, without too much interference from other weights.

Since there is no weight frozen in CasPer, the network is able to fine-tune an old weight in a slightly such as a previous L1 weight. This could improve the performance of an early hidden neuron when more neurons are installed. The model generated by CasPer is shown below in Fig.4.

Fig. 4 Model Generated by CasPer



Using cascade network structure allows us to train the model in a more efficient manner while the number of hidden units and depth of the neural network could be optimized automatically. Usually, when building a neural network, we will first determine the topology of the network, and then set other hyper-parameters such as the number of hidden units. CasPer starts from a minimal network, and automatically trains and adds hidden units, and finally forms a multi-layer structure. And from previous studies [6, 9], CasPer could improve the generalization of the feedforward neural networks comparing to CasCor we used in previous work.

The idea of setting different initial learning rates enables the network to make the new hidden unit a good feature extractor. And we could modify the feature extractor when the training continues to prevent it from damaging the generation.

In our work, we use RMSprop, an improved version of RPROP, since RPROP cannot be applied in mini-batch learning. The RMSprop algorithm no longer updates the learning step in isolation, but considers every previous gradient change.

2.2 Dataset

To train and evaluate our Cascade Network, we use the dataset that contains the temperature features with fuzzy signatures [2]. In this dataset, the temperature features are fuzzy values from 0 to 1. The data used in this paper contains the temperatures at 8 am, 12 pm, 4 pm and 8 pm. And each body temperature check has been divided into three fuzzy sets, that is “slight”, “moderate”, and “high”. Thus, the number of input features is 3*4=12. And the target classes are as follow : (0) High blood pressure, (1) Pneumonia, (2) SARS, and (3) Normal. So, the output layer is made up of 4 neurons, which leads to the minimal network which CasCor started with contains 12 input neurons and 4 output neurons. And the classification task is to recognize what kind of disease the patients have by the temperature information. The data in each column stands for the fuzzy signature of whether it is a "slight/medium/high" in the corresponding timestamp, and the data range from 0 to 1. It describes what extent the temperature at that time was "slight/medium/high". For example, if the data in “High 12 pm” is 0.9, it means the patient is likely to have a high temperature at 12 pm.

The dataset contains 1000 entries for each class, so there are 4000 data in total. And for the training and validating, the dataset is divided into a training set, which contains 75% of the data, and a validation set, which contains the remaining 25% data. In order to reduce the prediction bias of the model, the numbers of entries with different labels in the training set are well balanced, that is, there are 750 entries for each class in the training set.

When preparing the data, the data need no normalization because the data are fuzzy signatures which range from 0 to 1. And for each class, the corresponding label is 0 means normal, 1 means HighBP, 2 means pneumonia, 3 means SARS.

2.3 Training Detail

The cascade network was trained with PyTorch [7]. The training dataset containing 3000 entries was used to train the whole model with data shuffling, a batch size of 8 for model training. We used a RMSprop optimizer with momentum=0.9 and weight decay=0.00001. Since the proper model structure is determined by CasPer itself, we do not know how many epochs we need to get a satisfying model. Thus, the training epoch I set in the paper should be large enough, so that the cascade network could be fully trained by CasPer. The training stops when the loss of the network is lower than the given threshold. The threshold in the experiment is 0.1. Since momentum is set to the optimizer, the loss may fluctuate during the training, which may lead the CasPer to add some unnecessary hidden neurons. So, I force the network to train at least 5 epochs to allow it to be fully trained.

When setting the initial learning rates of L1, L2, L3, the value of L1, L2 and L3 are set as 0.2, 0.005 and 0.001 separately, which refers to the technique paper of CasPer [9]. From the work of Treadgold and Gedeon, these parameter values were found to be problem independent, and hence were treated as constants.

Cross-Entropy loss is used as the loss function in this experiment because it performs better in this problem, which is a multi-class classification. It is used in the training of the whole network.

Another hyper-parameter we need to control is the number of neurons in each layer. The feature number of the input vector is 12, which tell us that the neurons in the input layer should be 12. The output of the ResBlock (also the input of the CasPer model) is related to the latent variables it should learn. By observing the dataset, we could find that the vector contains temperature information in 4 timestamps. This indicate that 4 is a suitable choice for the number of output neurons of the ResBlock. The number of classes in this multi-class classification is 4, so that the output layer consists of 4 neurons.

3 Results and Discussion

We have used Cross-Entropy loss and the total accuracy to evaluate the overall performance of the classification model. The validation set has not been used for training or tuning models. And since the net topology is decided according to the number of cascade neurons. Thus, we should compare the performance of the networks with different numbers of hidden neurons. Since this validation set is balanced with the classes, the mean accuracy could reveal the performance well enough.

To show whether the hidden neurons installed by CasPer could give a feedforward network more power, we also compare the performance of the cascade network with a baseline net. In this case, the baseline is a vanilla fully connected network. And we compare our work with the previous work, which using CasCor.

3.1 Results

Model	Train Accuracy	Train Loss	Validation Accuracy	Validation Loss
Vanilla Fully Connected Network	93.33%	0.5357	90.1%	0.5047
Previous with 1 neuron installed by CasCor	99.13%	0.2722	99.3%	0.3046
Previous with 2 neurons installed by CasCor	99.63%	0.1812	99.3%	0.2397
Ours with 0 neuron installed by Casper	99.76%	0.0006	**100%**	0.00052
Ours with 1 neuron installed by Casper	**99.9%**	**0.00045**	**100%**	**0.000275**

Table 1 : The Average Performance of Different Network

Table 1 and Fig.3 states the training results and the validation results which are the average of 10 runs. And the data in bold is the best result. And there is no evaluation in the dataset [2].

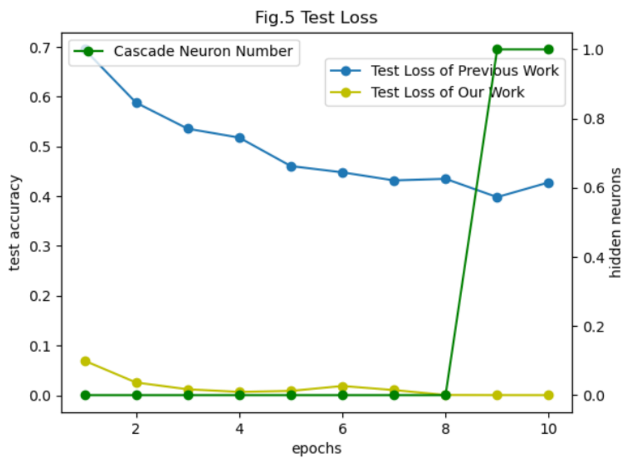
3.2 Discussions

From Table 1, we could find that with the increment of the number of cascade neuron, the loss of the fully trained network will reduce. This means the cascade neurons could indeed help the net to escape from the local minima, which could improve the generalization of the model.

Even though the validation accuracy of the vanilla fully connected network is around 90%, which is satisfying, the validation accuracy of the network with cascade neurons could be further improved. The models using the cascade algorithm could improve the validation accuracy to about 99%.

Although the loss of cascade network could be reduced by adding new cascade neurons or applying ResBlock, the accuracy has reached optimality. This is because the data in this problem is not complicated enough, thus our shallow network could work well enough.

From Table 1, we could also find that with the help of ResBlock, the networks perform much better than before. And our work with 1 neuron installed by Casper performs the best. We could find that using ResBlock, we could get a good result with a small number of hidden neurons.



From Fig.5, we could find that the training loss and test loss decrease with the epochs increases although there are some fluctuations, which may be caused by the momentum of the optimizer which allow it to escape from a local minimum. And installing a new hidden neuron into the network, the loss will decrease in most situations. The reason is that the new hidden neuron installed by CasPer could be used as a feature extractor thanks to the modified training method.

While we could also find that if we apply ResBlock, the initial loss is much less than without ResBlock. When hidden neuron in the cascade architecture is 0, CasPer and CasCor are not applied yet. This means ResBlock itself could greatly improve the performance and the generation. This is because the ResBlock is acting an encoder which allows the model to learn and represent the latent variables to help the model to learn the dataset and make a better prediction.

4 Conclusion and Future Work

The main objective of this study is to explore how ResBlock and CasPer could improve the performance of a feedforward network. From the experiment and the comparison, we could find that the ResBlock produces better structure and result with better generalization than a normal multi-layer fully connected network. With the help of ResBlock, the training process would be more efficient since we are learning the difference instead of the entire function. And the ResBlock is acting an encoder which learn and represent the latent variables, thus improving the performance and generation.

On the other hand, we could find that CasPer can be used to determine the optimal network structures with good generation when doing the sars-cov prediction given the temperatures of the patient. If we only have the temperature information, the performances of vanilla neural networks are not good enough due to the lack of features. Thus, the hidden neurons installed by CasPer can be treated as extra feature extractors efficient.

There are several future directions for this problem. When doing the training process and the validating process, we could find that the result is good enough if the number of hidden neurons is 1. To further investigate the constructive cascade algorithm, a larger dataset is necessary. And since the dataset we used now is well processed, we have no idea about whether our model is robust enough to the noises.

From works by Treadgold and Gedeon [6], though CasPer is successful to improve the performance of a feedforward network, the number of hidden neurons that are installed is potentially unlimited. A further difficulty faced by the cascade architecture is that the number of weights added per hidden neuron is exponential. We could build some other cascade architecture such as Higher Order Neuron (HON) [6] to solve the problems.

References

1. Nucleic Acid Amplification Tests (NAATs). (n.d.). Retrieved April 24, 2021, from https://www.cdc.gov/coronavirus/2019-ncov/lab/naats.html
2. Mendis, B. S., Gedeon, T. D., & Koczy, L. T. (2005). Investigation of aggregation in fuzzy signatures, in Proceedings, 3rd International Conference on Computational Intelligence, Robotics and Autonomous Systems, Singapore.
3. Jocelyn Kaiser Catherine Matacic Meredith Wadman, Jennifer CouzinFrankel. How does coronavirus kill? clinicians trace a ferocious rampage through the body, from brain to toes. Science, 38(8):1885–1898, 2020.
4. Parekh, R., Yang, J., Honavar, V.: Constructive neural-network learning algorithms for pattern classification. In: IEEE Transactions on Neural Networks, vol. 11, no. 2, pp. 436- 451, March 2000.
5. lman, S., Liebiere, C.: The Cascade-Correlation Learning Architecture. Technical report, School of Computer Science, Carnegie Mellon University (1990)
6. Treadgold, N.K., Gedeon, T.D.: Exploring Architecture Variations in Constructive Cascade Networks. In: IEEE Int. Jt. Conf. on Neural Networks, pp. 343–348 (1998)
7. Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. Pytorch. Computer software. Vers. 0. 3, 1, 2017
8. Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. December 2014
9. Treadgold N.K., Gedeon T.D. (1997) A cascade network algorithm employing Progressive RPROP. In: Mira J., Moreno-Díaz R., Cabestany J. (eds) Biological and Artificial Computation: From Neuroscience to Technology. IWANN 1997
10. Kwok, T., and Yeung, D. (1993) Experimental Analysis of Input Weight Freezing in Constructive Neural Networks. In Proc. 1993 IEEE Int. Conf. Neural Networks. pp. 511-516.
11. Riedmiller, M. and Braun, H. (1993) A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm. In: Ruspini, H., (Ed.) Proc. of the ICNN 93, San Francisco, pp. 586-591.
12. K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.