DOI: 10.21917/ijivp.2019.0298

CLUSTERING ALGORITHM NETWORKS TEST COST SENSITIVE FOR SPECIALIST DIVISIONS

M. Arvindhan[1], G.S. Pradeep Ghantasala[2] and N.V. Kousik[3]
[1,3]School of Computing Science and Engineering, Galgotias University, India
[2]Department of Computer Science and Engineering, Malla Reddy Institute of Technology and Science, India

Abstract

It has been proven that deeper Convolutional Neural Networks (CNN) can result in better accuracy in many problems, but this accuracy comes with a high computational cost. Also, input instances have not the same difficulty. As a solution for accuracy vs. computational cost dilemma, we introduce a new test-cost-sensitive method for convolution neural networks. This method trains a CNN with a set Based on the difficulty of the input instance, the expert branches decide to use a shallower part of the network or go deeper to the end. The expert branches learn to determine: is the current network prediction wrong and if the instance passed to deeper network layers it will generate the right output; if not, then the expert branches will stop the process of computation. The experimental results on the standard CIFAR-10 dataset indicate that in comparison with basic models, the proposed method can train models with lower test cost and competitive accuracy.

Keywords:

Test-Cost-Sensitive Learning, Deep Learning, CNN with Expert Branches, Instance-based Cost

1. INTRODUCTION

Neural networks with deep convolution have provided state-of-the-art results on different benchmarks [1] [2]. Many research on neural networks in the field of convolution has shown that deeper networks are more accurate. The state-of-the-art deep CNNs today have over one hundred layers and millions of parameters and weights [3]. This requires a great deal of computational power and time. For example, in every second, a cloud computing service should process too many requests, or mobile and embedded systems may not have enough power and hardware to run the network for their inputs. And that the networks' computational costs while preserving their accuracy during the inference is very necessary. If we look at the outputs of each network layer as a set, for example, a cloud computing platform may handle too many requests per second, or mobile and embedded devices may not have enough power and resources to operate the network for their inputs. It is therefore very important to reduce the computational costs of the networks while preserving their reliability during the inference. If we look at the outputs of each network layer as a set, different methods for reducing test-cost and compressing deep convolutional networks have been proposed. Compression methods attempt to reduce the number of network parameters, but these approaches do not necessarily result in faster networks; since most of a CNN's computation is related to operations that cannot be reduced by Compression of the network only. Some recent research focused on instance-based or input-dependent methods that use a set of models dynamically or use some parts of the models to generate the outcome for a particular instance [6]. As we know, even doubling the depth of the network will have a small effect on accuracy and not all input instances have the same difficulty.

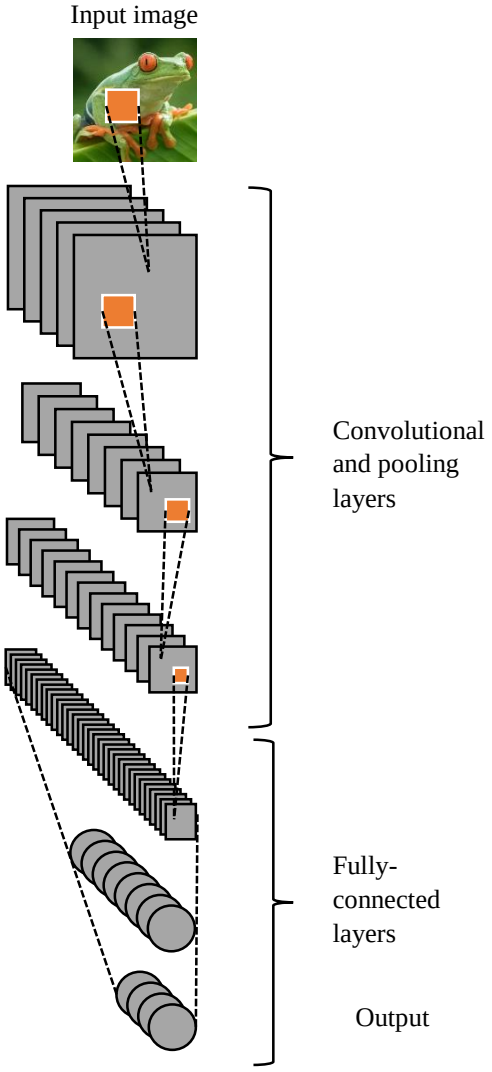

Fig.1. Illustration of deploying a typical CNN model on an input image

Along the line of dynamic and instance-based approaches, in this paper, we propose a new test-cost-sensitive method for deep convolutional networks which can learn to manage the available computational resources in the way that result in faster inference for many input instances. This method uses a set of middle output and expert branches in the convolutional network. When an instance is given to the network input, the computation is started layer by layer to the end of first middle output and expert branch. Deeper layers resulting in higher computational costs are only

used when the branch of experts shows the possibility of increasing the reliability of the performance and avoids unnecessary computational power consumption. This can reduce the overall cost of testing and retain an acceptable level of network reliability relative to the basic model. Standard data sets experiments show the advantages.

2. RELATED WORK

There are various types of costs during a machine learning process [7]. Since computational cost is a real challenge for deep neural networks, researches proposed different methods and approaches to solve it. In this section, we investigate the literature available in this field. These researches may do not use the test-cost-sensitive terminology but are relevant to the current research. The first class belongs to methods that train a new model or modify the trained models based on the original model [8]. Second category methods increase the speed of deep networks with advanced computational methods and use of hardware more effectively. Dynamic instance-based approaches are the third category of deep learning test-cost-sensitive methods that have resulted in effective solutions in recent years [6] and the proposed method belongs to this group. With some examples of research, we describe these approached in more detail.

3. MAKING A MODIFIED MODEL

This method changes an existing model or learn from scratch a new model to reduce the original model's complexity and computational operations. A new model is made from scratch to replicate the actions of the original model called the model of the instructor. The newly generated models are more compact, shallower in [10] and less filtered and fitter in [11] models. Network decomposition methods [12]-[14] is another group of model modification approaches that use estimation solutions. Filters are decomposed in these methods in a way that increases the network overall speed, but the performance of the initial network layers is still well calculated. Older network pruning methods [15] do not consider computational cost reduction as their goal, but model sparsification reduces its complexity, resulting indirectly in the faster network [16].

3.1 COMPLEX AND SMALL-LEVEL FORMS FOR COMPUTATION

These methods increase the speed of the deep network without altering the network structure, unlike the previous approach. One family of methods focuses on how to measure the performance of layers, specifically using Fast Fourier Transform (FFT) [9]. Another family aims to use the available hardware efficiently [17] [18] through low-level parallel computation, efficient memory use and low precision.

4. EVOLUTIONARY TECHNIQUES

All previous methods have a fixed behavior with all input instances and with an input-dependent system, they cannot assign computational resources. There was therefore a shortage of test-cost-sensitive approaches that only use computational recourse when it is needed based on the given instance and dynamically. Several approaches have been proposed in recent years based on this approach, which we call evolutionary methods. Adaptive methods can also be paired with two previous system classes and the benefits of both can be used. Network cascades are a major group of research in adaptive models. Such approaches train and use a collection of deep networks in a cascade style. We start with simple models with lower test cost and continue the process with more complex networks until a reasonable degree of confidence is reached for the performance produced. Models with heavier computations are therefore only used for more challenging input instances. Deep Decision Network proposed to identify objects in [19]. The approach recognizes example hardness and transfers more difficult objects in the cascade to subsequent models. The approach proposed for face detection in [20], called the cascade of convolutional neural networks. This works on image versions with different resolutions, excludes background regions in low-resolution stages, and moves to high-resolution assessments some difficult candidates. The authors of [6] suggested Deep Layer Cascade in a particular cascade style for the issue of semantic object segmentation. Unlike template cascades using a collection of models, layer cascade trains a single network with some internal branches that generate a degree of confidence for the object regions and stop the process for easier parts recognized in lower layers.

5. CNNS WITH EXPERT BRANCHES

We will describe the proposed approach in more detail in this paragraph. It is called as Expert Branches (CNN-EB) CNNs. First, we examine the relationship between computational cost in CNNs and classification test cost. Then in the third part of this chapter, we explain the details of the process and define it as an algorithm.

5.1 TEST-COST IN CNN

The word test-cost comes from medical diagnosis field and means that if we want to do any test on the patient to find the related values of that test, we should consider its cost. Based on this concept we define the test-cost in deep CNNs. The deep learning methods have two main properties: automatic learning of features, and a layered process of learning. The specifications of the learning process in deep CNNs mixed test-cost and computation cost concepts. That means in the process of feature extraction and learning in the layers of CNN, each layer gains values of a set of features (test-cost) by means of doing necessary computations (computation cost).

5.2 ARCHITECTURAL MODEL

The proposed deep CNN model consists of a common convolutional network and two types of augmented branches, including branches and expert branches of middle output (or classifier). They are paired together and work together on the CNN midpoints. The output divisions are extra output generators, which can identify the input instance tag in a category, for example. The expert branches look at the data from a different point of view; they decide to pass the input instance to higher network layers or find the current output produced by the paired output branch as the network's final output.

The process starts by getting input instance x and continues layer by layer to classifier branches O_i and expert branches E_i. If

we get to the last output branch or the check $\hat{d}_i$ node in the network decides to stops the process then $\hat{y}_i$ is considered as the final output of the network.

6. EXPERIMENTAL STUDY

6.1 THE DATASET

The CIFAR-10 dataset [22], one of the most commonly used data sets for image processing analysis, was used to test the methods. There are 60,000 objects in 10 different classes in this dataset. There are 6,000 images in each class. Approximately 85% of the images are used for training and the majority of the images are used for model research. The Table.1 shows the specifications of the CIFAR-10.

Table.1. Specifications of CIFAR-10 dataset used for evaluation of methods

Class	Train dataset	Test dataset
Airplanes	5,000	1,000
Birds	5,000	1,000
Cars	5,000	1,000
Cats	5,000	1,000
Deers'	5,000	1,000
Doges	5,000	1,000
Frogs	5,000	1,000
Horses	5,000	1,000
Ships	5,000	1,000
Trucks	5,000	1,000
Total	50,000	10,000

Three variants of the proposed expert branch method and two basic well-known inception v3 models are compared to specify the characteristics of the proposed method. In the "Auxiliary as Expert Branch" method, the output of the original auxiliary branch of inception v3 which we placed after module 5b, is used as the expert branch decisions by applying some thresholds. This method is very similar but we used the auxiliary branch output in a different way to determine the complete instance difficulty, not some parts of it. The "Auxiliary+5b as Expert Branch" is implemented by considering additional branch as CNN-EB's expert branch in addition to the 5b startup system.

The "Proposed Expert Branch" and "Auxiliary as Expert Branch" methods are almost above the imaginary line between two basic inception v3 methods. This indicates that the proposed CNN-EB method is successful in managing the use of computational resources by utilizing the shallower and deeper structure of the network for easier and harder instance, respectively.

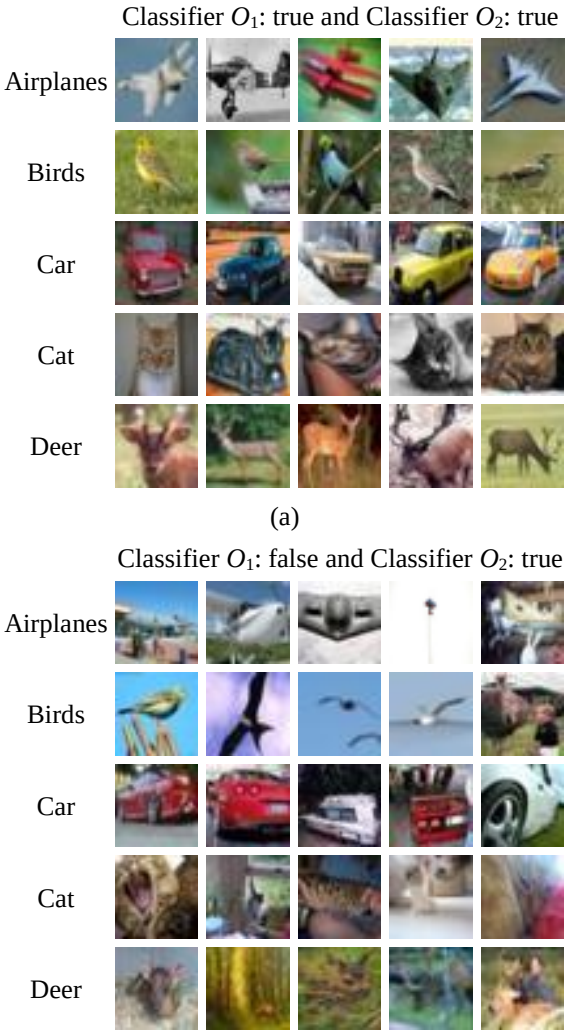

Fig.2. Sample results of output classifiers O_1 and O_2 for five CIFAR-10 classes. (a) Easier instances where both classifiers made true classification, and (b) Harder instances where only classifier O_2 made the true classification

The Fig.2 displays some test results of output classifiers O_1 and O_2 for five CIFAR-10 categories to provide a visual understanding of easy and hard images for classifiers. The left side shows simpler instances where artifacts within the images are visible, with a good position and angle that makes guessing the true tag for these instances easier for the shallower classifier. The right side contains more difficult cases in the image that contain parts of the objects, strange images, and multiple objects. For hard instances, only classifier O_2 that uses the deeper network structure will produce the true tag.

Table.2. Comparison of basic and proposed methods based on accuracy and time metrics

Method	Accuracy	Time (ms)	Accuracy Decrease	Time Saving
Inception v3 Auxiliary as Final Classifier	78%	185	-	-

Inception v3 Main as Final Classifier	83%	670	-	-
Auxiliary as Expert Branch	84%	500	1%	14%
Auxiliary + 5b as Expert Branch	81%	520	1%	9%
Proposed Expert Branch	85%	451	1%	21%
Auxiliary as Expert Branch	83%	435	2%	25%
Auxiliary + 5b as Expert Branch	83%	475	2%	19%
Proposed Expert Branch	83%	400	2%	31%

The Table.2 shows the performance of basic inception v3 methods and variants of the proposed expert branch method based on accuracy and time metrics. As we can see, 1% decrease in the accuracy of "Proposed Expert Branch" model in comparison with basic "Inception v3 Main as Final Classifier", results in 21% time and computational cost saving of the model, and 2% of accuracy decrease makes 31% time-saving. The "Proposed Expert Branch" can save more time than other proposed expert branch method variants with the same accuracy.

7. CONCLUSION

Deep convolutional neural networks test-cost is a challenging issue in real-world issues. In this paper, we introduce CNN-EB, which is a test-cost-sensitive CNN method that uses expert branches to determine input hardness the use of available computing resources is handled by using shallower network layers for easier instances and deeper layers for harder ones. To make deep models more efficient, the proposed method can be combined with other cost-sensitive CNN methods. The experimental results show that a small decrease in the accuracy of the proposed method in comparison with the basic models will result in significant time and computational resource saving. In order to better evaluate the proposed method, experiments on deeper models can be performed in future work and the efficiency of the proposed method can be investigated for these models.

REFERENCES

[1] S.P.S. Gurjar, S. Gupta and R. Srivastava, "Automatic Image Annotation Model using LSTM Approach", *Signal and Image Processing: An International Journal*, Vol. 8, No. 4, pp. 25-37, 2017.

[2] S. Maity, M. Abdel-Mottaleb and S.S. Asfour, "Multimodal Biometrics Recognition from Facial Video via Deep Learning", *Signal and Image Processing: An International Journal*, Vol. 8, No. 1, pp. 67-75, 2017.

[3] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition", *Proceedings of International Conference on Computer Vision*, pp. 1-9, 2015.

[4] D. Kadam, A.R. Madane, K. Kutty and B.S.V. Bonde, "Rain Streaks Elimination Using Image Processing Algorithms", *Signal and Image Processing: An International Journal*, Vol. 10, No. 3, pp. 21-32, 2019.

[5] A. Massaro, V. Vitti and A. Galiano, "Automatic Image Processing Engine Oriented on Quality Control of Electronic Boards", *Signal and Image Processing: An International Journal*, Vol. 9, No. 2, pp. 1-14, 2018.

[6] X. Li, Z. Liu, P. Luo, C. Change Loy and X. Tang, "Not All Pixels Are Equal: Difficulty-Aware Semantic Segmentation via Deep Layer Cascade", *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3193-3202, 2017.

[7] M. Naghibi, R. Anvari, A. Forghani and B. Minaei, "Cost-Sensitive Topical Data Acquisition from the Web", *International Journal of Data Mining and Knowledge Management Process*, Vol. 9, No. 3, pp. 39-56, 2019.

[8] A. Polyak and L. Wolf, "Channel-Level Acceleration of Deep Face Representations", *IEEE Access*, Vol. 3, pp. 2163-2175, 2015.

[9] A. Lavin and S. Gray, "Fast Algorithms for Convolutional Neural Networks", *Proceedings of IEEE International Conference on* Computer *Vision and Pattern Recognition*, pp. 4013-4021, 2016.

[10] L. J. Ba and R. Caruana, "Do Deep Nets Really Need to be Deep?", *Proceedings of International Conference on Advances in Neural Information Processing Systems*, pp. 2654-2662, 2014.

[11] A. Romero, N. Ballas, S.E. Kahou, A. Chassang, C. Gatta and Y. Bengio, "Fitnets: Hints for Thin Deep Nets", *Proceedings of International Conference on Neural and Evolutionary Computing*, pp. 782-789, 2014.

[12] X. Zhang, J. Zou, K. He and J. Sun, "Accelerating Very Deep Convolutional Networks for Classification and Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 38, No. 1, pp. 1943-1955, 2015.

[13] E.L. Denton, W. Zaremba, J. Bruna, Y. LeCun and R. Fergus, "Exploiting Linear Structure Within Convolutional Networks for Efficient Evaluation", *Proceedings of 27th International Conference on Neural Information Processing Systems*, pp. 1269-1277, 2014.

[14] M. Jaderberg, A. Vedaldi and A. Zisserman, "Speeding Up Convolutional Neural Networks with Low Rank Expansions", *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pp. 1-12, 2014.

[15] N. Strom, "Sparse Connection and Pruning in Large Dynamic Artificial Neural Networks", *Proceedings of 5th European Conference on Speech Communication and Technology*, pp. 1-4, 1997.

[16] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R.R. Salakhutdinov, "Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors", *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pp. 1-18, 2012.

[17] N. Vasilache, J. Johnson, M. Mathieu, S. Chintala, S. Piantino and Y. LeCun, "Fast Convolutional Nets with FBFFT: A GPU Performance Evaluation", *Proceedings of International Conference on Learning Representations*, pp. 1-17, 2014.

[18] M. Mathieu, M. Henaff and Y. LeCun, "Fast Training of Convolutional Networks through FFTs", *Proceedings of*

International Conference on Neural and Evolutionary Computing, pp. 1111-1119, 2013.

[19] V.N. Murthy, V. Singh, T. Chen, R. Manmatha and D. Comaniciu, "Deep Decision Network for Multi-Class Image Classification", *Proceedings of IEEE conference on Computer Vision and Pattern Recognition*, pp. 2240-2248, 2016.

[20] V. Vanhoucke, A. Senior and M.Z. Mao, "Improving the Speed of Neural Networks on CPUs", *Proceedings of International Workshop on Deep Learning and Unsupervised Feature Learning*, pp. 1-8, 2011.

[21] A. Toshev and C. Szegedy, "Deeppose: Human Pose Estimation via Deep Neural Networks", *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1653-1660, 2014.

[22] A. Krizhevsky and G. Hinton, "Learning Multiple Layers of Features from Tiny Images", Technical Report, University of Toronto, pp. 1-65, 2009.

[23] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision", *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818-2826, 2016.