

Appendix of the Paper: Enhancing Adaptive Deep Networks for Image Classification via Uncertainty-aware Decision Fusion

Xu Zhang
xuzhang22@m.fudan.edu.cn
School of Computer Science, Fudan
University, Shanghai, China

Zhipeng Xie
xiezp@fudan.edu.cn
School of Computer Science, Fudan
University, Shanghai, China

Haiyang Yu
hyyu20@fudan.edu.cn
School of Computer Science, Fudan
University, Shanghai, China

Qitong Wang*
qitong.wang@etu.u-paris.fr
Universite Paris Cite, Paris, France

Peng Wang
pengwang5@fudan.edu.cn
School of Computer Science, Fudan
University, Shanghai, China

Wei Wang
weiwang1@fudan.edu.cn
School of Computer Science, Fudan
University, Shanghai, China

A APPENDIX

A.1 Anytime prediction and budgeted Batch prediction on “E” structure of MSDNet and RANet

In the main text, we have analyzed the effectiveness of GCDM on MSDNet and RANet with “LG” structure. In this section, we further verify the effectiveness of GCDM on the MSDNet and RANet with “E” structure.

The results of *anytime prediction* setting are shown in Figure 1 (CIFAR10 and CIFAR100) and Figure 2 (ImageNet100 and ImageNet1000). The results of *budgeted batch prediction* setting are shown in Figure 3 (CIFAR10 and CIFAR100) and Figure 4 (ImageNet100 and ImageNet1000). The experimental conclusion was the same as what was drawn in the main text: GCDM consistently improves the performance of the original adaptive networks, whether in the “LG” or “E” structures

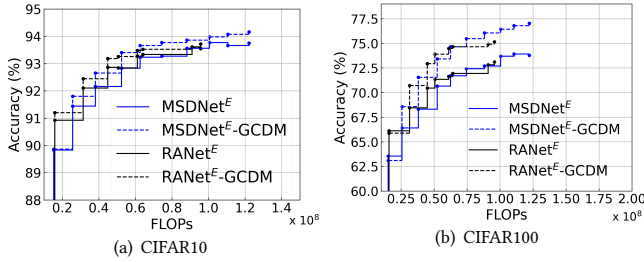


Figure 1: Accuracy (top-1) of anytime batch prediction on CIFAR10 and CIFAR100. With the same computational resources, existing methods equipped with the proposed GCDM can achieve better performance.

A.2 Diversity of early classifiers after regularization on CIFAR100

In the main text, we have shown the agreement measurement on 10 classifiers of MSDNet^E on ImageNet100 after regularized training. Here we additionally show the results on CIFAR100. As shown in Figure 6, values in bold denote that the corresponding classifiers obtain higher diversity after regularized training. It further proves

*Corresponding author.

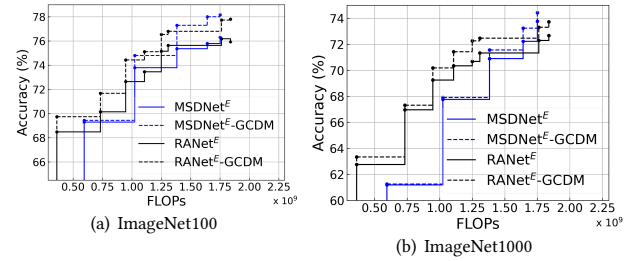


Figure 2: Accuracy (top-1) of anytime batch prediction on ImageNet100 and ImageNet1000.

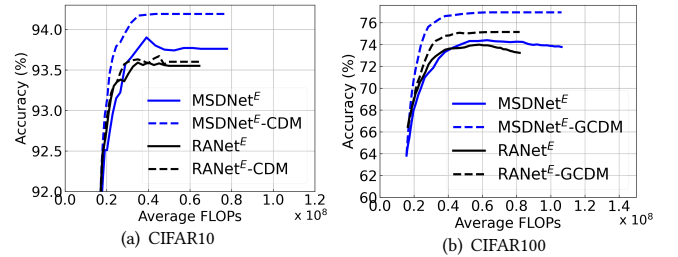


Figure 3: Accuracy (top-1) of budgeted batch prediction on CIFAR10 and CIFAR100.

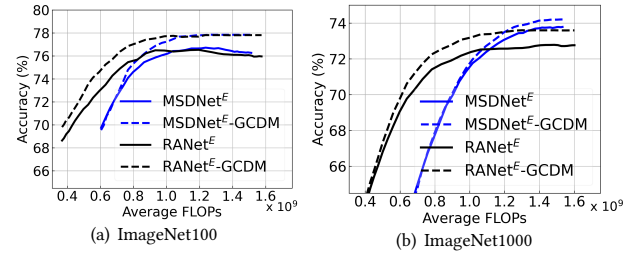


Figure 4: Accuracy (top-1) of budgeted batch prediction on ImageNet100 and ImageNet1000.

that regularized training won’t obviously harm the diversity of early classifiers and even can increase the diversity.

A.3 Using Stable Training Strategy (STS) on budgeted batch prediction

The conclusion is the same as that obtained in Section 4.3 and Figure 8 (b) in the main text. As shown in Figure 5, the proposed STS using both τ_1 and τ_2 during regularized training can improve the performance instability issues and outperform using τ_1 or τ_2 individually. Moreover, we observe that CDM can significantly improve the accuracy after regularized training, indicating CDM can work well with regularized training.

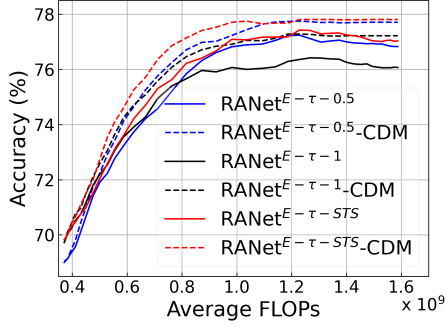


Figure 5: Results of *budgeted batch prediction* with proposed Stable Training Strategy (STS) on MiNi-ImageNet.

-	0	1	2	3	4	5	6	7	8	9
0	1.0	0.827	0.813	0.798	0.784	0.777	0.779	0.772	0.768	0.768
1	0.869	1.0	0.855	0.834	0.826	0.82	0.818	0.811	0.809	0.808
2	0.882	0.882	1.0	0.864	0.852	0.846	0.848	0.841	0.833	0.833
3	0.89	0.885	0.888	1.0	0.896	0.889	0.888	0.88	0.873	0.872
4	0.895	0.897	0.898	0.918	1.0	0.916	0.911	0.903	0.898	0.897
5	0.903	0.906	0.907	0.927	0.932	1.0	0.93	0.92	0.916	0.915
6	0.907	0.906	0.91	0.926	0.928	0.93	1.0	0.925	0.922	0.919
7	0.909	0.908	0.914	0.929	0.93	0.932	0.936	1.0	0.951	0.946
8	0.904	0.907	0.905	0.923	0.926	0.928	0.934	0.952	1.0	0.954
9	0.909	0.91	0.909	0.925	0.928	0.931	0.934	0.95	0.958	1.0

Figure 6: Agreement measurement on 10 classifiers of MSDNet on Cifar100 with regularization (G^+). Lower value (higher diversity) is better and bolded values denote decreasing after regularization.

A.4 Datasets

First, the CIFAR-10 and CIFAR-100 datasets are used in our experiment, which contains 32×32 RGB natural images and corresponds to 10 and 100 classes, respectively.

Second, the ImageNet100 dataset contains 100 classes and 60000 images and we split it into a training set (50000 images) and a testing set (10000 images).

Hence, the above three datasets both contain 50,000 training and 10,000 testing images. We hold out 5,000 images in the training

set as a validation set for selecting the model and searching the confidence threshold for adaptive inference.

Third, the ImageNet1000 dataset contains 1.2 million images of 1,000 classes for training and 50,000 images for validation. We use the original validation set for testing and hold out 50000 images from the training set as a validation set for model selection and adaptive inference tasks. Besides, the image size of Mini-ImageNet used in this paper is as same as ImageNet. The above settings of datasets follow the source codes and paper of [6, 8, 12].

A.5 Qualitative analysis of CDM

The deepening of the CNN network results in an expansion of the receptive field of the convolutional kernel, consequently amplifying the overlapping area between these receptive fields [9, 10]. Hence, deeper CNN tends to extract deep features, in which the image information is compressed, including more coarse-grained information (*i.e.*, semantic information) about the integrity of the image. In contrast, shallower CNN tends to extract shallow features, which contain more fine-grained image information such as color, texture, edge, and corner information [1, 2, 7].

Here, to further explain the above analysis, we visualize samples accurately classified by the final classifier in the top row, and samples misclassified by the final classifier but accurately classified by the early classifier in the bottom row. The results are shown in Figure 7.

A.6 Structures of baseline models

The baseline structures follow the source codes of MSDNet (<https://github.com/gaohuang/MSDNet>) and RANet (<https://github.com/yangle15/RANet-pytorch>). Details are as follows:

ResNet^{MC} and **DenseNet^{MC}** for CIFAR datasets. The *ResNet^{MC}* has 62 layers, with 30 basic blocks and each block consisting of 2 Convolution layers. We train early-exit classifiers on the output of every 5 basic blocks and there are a total of 6 intermediate classifiers (plus the final classification layer). The *DenseNet^{MC}* has 56 layers with three dense blocks and each of them has 18 layers. The growth rate is set as 12. We train early-exit classifiers on the output of every 8 layers for the first 5 classifiers and 14 layers for the last classifier.

MSDNet^E and **MSDNet^{LG}** for CIFAR datasets. MSDNet^E has 10 classifiers and the distance between classifiers is equidistant. The span between two adjacent classifiers is 2. The number of features produced by the initial convolution layer is 16. The growth rate is 6. The bottleneck scales and the growth rate factors are all 1, 2, and 4. MSDNet^{LG} has 7 classifiers and the distance between classifiers is growing linearly. The feature scales are as same as MSDNet^E.

MSDNet^E and **MSDNet^{LG}** for Mini-ImageNet and ImageNet datasets. MSDNet^E has 5 classifiers and the number of features produced by the initial convolution layer is 64. The growth rate is 16. The bottleneck scales and the growth rate factors are all 1, 2, 4, and 4. The span between two adjacent classifiers is 4. MSDNet^{LG} has 6 classifiers and the feature scales are as same as MSDNet^E. The initial span between two adjacent classifiers is 1. More details can be seen in the source code on GitHub.

RANet^E and **RANet^{LG}** for CIFAR datasets. RANet^E has 8 classifiers and four sub-networks with 8, 6, 4, 2 *Conv* Blocks. The numbers

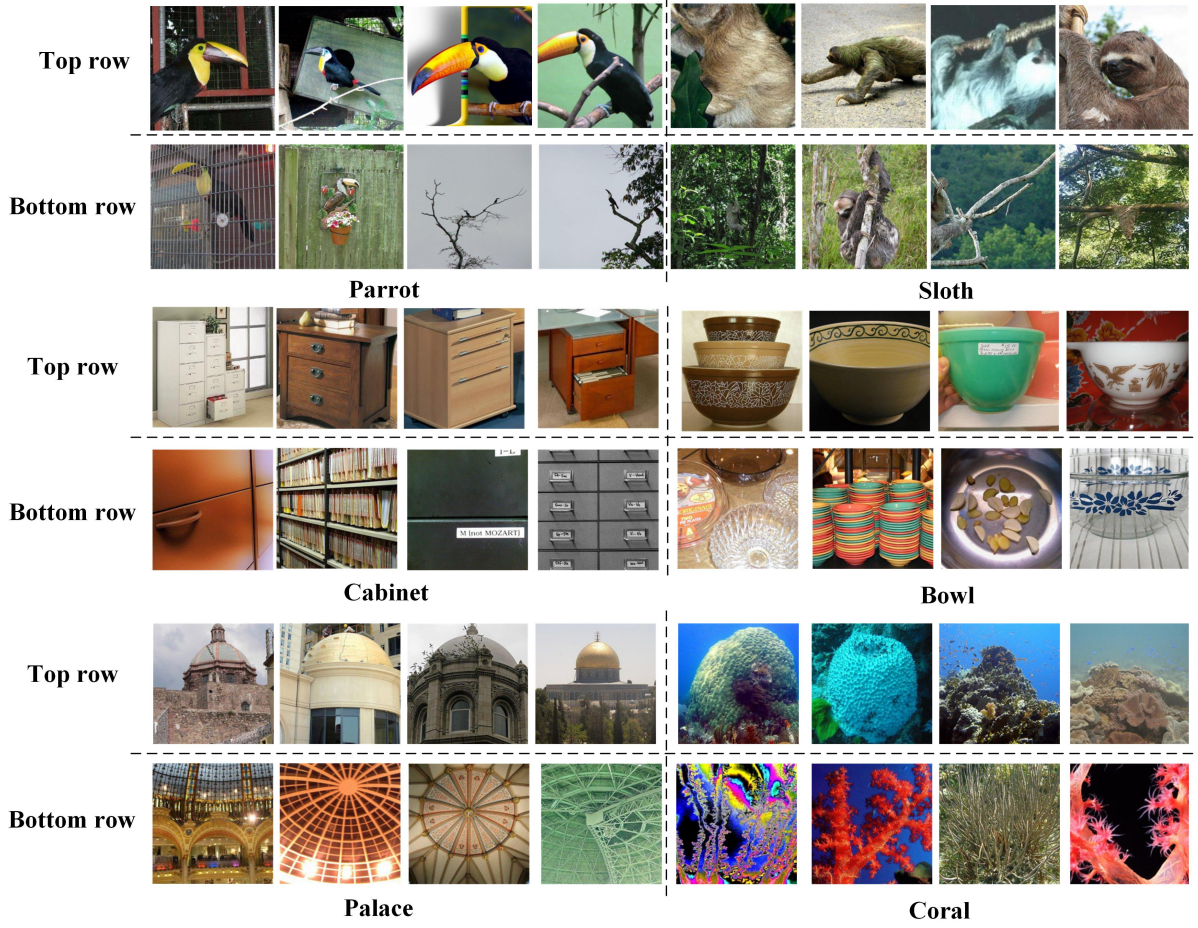


Figure 7: Qualitative analysis of Figure 1 (a) in the submitted Manuscript. The top row is samples that are correctly classified by the last classifier while the bottom row is correctly classified by early classifiers but wrongly classified by the last classifier.

of input channels and the growth rates are 16, 16, 32, 64, and 6, 6, 12, and 24, respectively. The number of layers in each *Conv* Block is set to 4. RANet^{LG} has 8 classifiers and four sub-networks with 8, 6, 4, 2 *Conv* Blocks. The numbers of input channels and the growth rates are 16, 32, 32, 64, and 6, 12, 12, and 24, respectively. The number of layers in a *Conv* Block is added 2 to the previous one, and the base number of layers is 2.

RANet^E and RANet^{LG} for Mini-ImageNet and ImageNet datasets. RANet^E has 8 classifiers and four sub-networks with 8, 6, 4, 2 *Conv* Blocks. The numbers of input channels and the growth rates are 64, 64, 128, 256, and 16, 16, 32, and 64, respectively. The number of layers in each *Conv* Block is set to 7. The architecture of RANet^{LG} is exactly the same as the RANet^E . However, the number of layers in a *Conv* Block is added to 3 to the previous one, and the base number of layers is 3.

We found that the final classifier performs poorly on samples (bottom row) that heavily rely on local texture, edge, and corner information. For instance, in samples of parrots, palaces, and corals, the final classifier exhibits the wrong classification on samples with distinct texture, edge, and corner features (bottom row). This is because the final classifier relies on deep features extracted from the

deepest CNN network for classification, which emphasizes overall high-level semantic information in images, but loses part of texture, edge, and corner details [1, 2, 7].

In contrast, the shallow features extracted from earlier CNN networks can classify these samples well. This is the reason for the observation in Figure 1: early classifiers perform better than the final classifier in certain classes. Hence, different classifiers have their own advantages and we can use the proposed uncertainty-aware attention mechanism-based fusion method to weight and integrate the decision information from $c - 1$ classifiers to enhance the performance of c -th classifier where $c \geq 2$ in CDM module.

A.7 Using CDM module under different prediction settings

In Figure 2 of the main text, we only show the difference between *anytime prediction* and *budgeted batch prediction* settings. Here we further show the version of the two prediction settings equipped with the CDM module in Figure 8(a) and Figure 8(b). Overall, for traditional prediction settings, *anytime prediction* or *budgeted batch prediction* all don't utilize the available $c - 1$ classifiers when inferring the c -th classifier. In contrast, in CDM module, we use the proposed

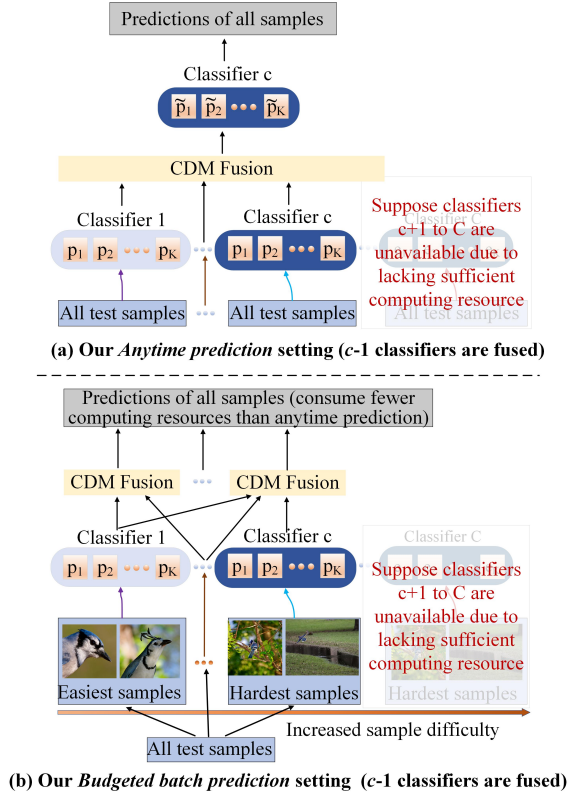


Figure 8: Illustration of prediction settings equipped with Uncertainty-aware Fusion based CDM module. CDM fuses the available $c-1$ classifiers when inferring the c -th classifier for performance improvement.

uncertainty-aware attention mechanism-based fusion method to weight and integrate the decision information from $c-1$ classifiers to enhance the performance of c -th classifier during inference.

A.8 Limitation and solution about CDM module

While the proposed Collaborative Decision Making (CDM) module performs well in most cases, it may occur that CDM fails in certain scenarios.

To address this, we can mitigate the potential adverse effects of CDM failures by utilizing the validation set¹. Specifically, for the c -th classifier ($c \geq 2$), we first apply CDM to fuse it with the previous $c-1$ classifiers on the validation set. If a performance drop occurs after fusion, we will retain the original classifier (no fusion) for actual testing, thereby minimizing the risk of performance degradation due to potential CDM failures. We will incorporate this discussion into the "Discussion about Limitations" section of the manuscript.

¹Note that the validation set is also used in *adaptive networks* for dynamic inference [12] and we are just utilizing it, with no need to reconstruct it.

A.9 Decision fusion methods

After obtaining decision outputs from different sub-models (classifiers), we need to fuse all decisions by using fusion methods to produce the final decision for classification. Traditional fusion methods include averaging fusion, weighted averaging fusion, voting fusion, and neural network fusion strategies [3, 4, 11]. Traditional methods don't take into account the uncertainty of classifiers, which may lead to unreliable fusion results. To this end, [5] proposes an EDL-based fusion strategy for multiview classification tasks. However, the potential issues of *fusion saturation* and *fusion unfairness* caused by the EDL theoretical framework have not been fully explored, which may lead to failure of the decision fusion and decreasing fusion performance.

REFERENCES

- [1] Hamed Habibi Aghdam, Elnaz Jahani Heravi, et al. 2017. Guide to convolutional neural networks. *New York, NY: Springer* 10, 978-973 (2017), 51.
- [2] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. 2017. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*. Ieee, 1–6.
- [3] Subhash C. Bagui. 2005. Combining Pattern Classifiers: Methods and Algorithms. *Technometrics* 47, 4 (2005), 517–518. <https://doi.org/10.1198/TECH.2005.S320>
- [4] Eric Bauer and Ron Kohavi. 1999. An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. *Mach. Learn.* 36, 1-2 (1999), 105–139. <https://doi.org/10.1023/A:1007515423169>
- [5] Zongbo Han, Changqing Zhang, Huazhu Fu, and Joey Tianyi Zhou. 2021. Trusted Multi-View Classification. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. <https://openreview.net/forum?id=OOsR8BzCn15>
- [6] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Multi-scale dense networks for resource efficient image classification. *arXiv preprint arXiv:1703.09844* (2017).
- [7] Nikhil Ketkar, Jojo Moolayil, Nikhil Ketkar, and Jojo Moolayil. 2021. Convolutional neural networks. *Deep Learning with Python: Learn Best Practices of Deep Learning Models with PyTorch* (2021), 197–242.
- [8] Hao Li, Hong Zhang, Xiaojuan Qi, Ruigang Yang, and Gao Huang. 2019. Improved techniques for training adaptive deep networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 1891–1900.
- [9] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. 2021. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems* 33, 12 (2021), 6999–7019.
- [10] Keiron O'shea and Ryan Nash. 2015. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458* (2015).
- [11] Omer Sagi and Lior Rokach. 2018. Ensemble learning: A survey. *WIREs Data Mining Knowl. Discov.* 8, 4 (2018). <https://doi.org/10.1002/WIDM.1249>
- [12] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. 2020. Resolution adaptive networks for efficient inference. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2369–2378.