Dynamic inference techniques for deep neural networks

Zihan Zhang[1,3] and Linze Shi[2]

[1]Institute of Computer and Information Science, Southwest University, Tiansheng street, Chongqing, China
[2]Xi'an Jiaotong University, No. 28 Xianning West Road, Beilin District, Xi'an City, China

[3]ZZHzzh010906@outlook.com

Abstract. With the development of deep neural network, dynamic inference techniques have attracted extensive attention. When the traditional static neural network is faced with complex samples, it will generate a lot of computational redundancy, resulting in a waste of computing resources. In order to solve this problem, on the basis of static network, experts use dynamic reasoning technology to improve the network, so that the network can calculate the samples. Therefore, the network structure has been fundamentally improved by performing operations such as layer skipping and leaving early. Compared with the traditional static networks, the improved model has greatly improved the speed and scale. At the same time, the accuracy of operation has also been greatly improved.

Keywords: dynamic network, adaptive reasoning.

1. Introduction

The structure and parameters of the traditional static neural network are consistent with the training phase, while the dynamic neural network can adaptively adjust its structure/parameters according to the input in the inference phase. This feature makes it stronger than the traditional static neural networks in the following aspects. Firstly, due to the optimized inference computations, the operation efficiency is remarkably improved. Secondly, because inference has the opportunity to quit early according to different images, it shows satisfactory versatility.

At present, the core idea of dynamic inference methodology is based on adaptive reasoning, and its essence is to establish the dynamic integration of multiple models and activate these models by judging the input. Before formally pricing the dynamic neural network models, I will first introduce several important structures of the adaptive network and its corresponding aspects. Dynamic network architecture can usually be realized from three aspects: dynamic depth, dynamic width and dynamic routing in the hyper network.

1.1. Dynamic depth

As the deep network consists of multiple network layers stacked on top of each other, it is a relatively fast method to selectively execute the content of different network layers for different inputs, which involves two processing mechanisms: early exit and jump layer.

1.2. Dynamic width

In terms of dynamic width, there are two methods commonly used: multi-expert hybrid system and dynamic channel pruning in CNN.

1.3. Dynamic routing

Dynamic routing means that the router can automatically establish its own routing table, and adjust it in time according to the change of the actual situation.

2. Dynamic inference techniques

Firstly, we confirm what adaptive dynamic network is. An adaptive dynamic network is a system that processes information and adjusts the network when necessary, which is used to explain a large amount of complex information. The essence of it is an artificial neural network with adaptive learning ability.

In the past 10 years, dynamic neural networks have been widely studied. At present, it mainly focused on three directions. The first is the sample adaptive dynamic network that depends on data architecture or parameters to deal with each sample. The second is the spatial adaptive dynamic network for adaptive calculation according to the spatial position of the image. The third one is the time dynamic network for adaptive inference of the continuous data (such as video or text) along the time dimension. The next part of this section will describe the details of these three dimension.

2.1. Sample-adaptive dynamic neural network

In order to deal with different samples with different levels of insight, the sample-based adaptive dynamic network is proposed . The existing state-of-the-art sample adaptive networks are proposed from two perspectives: the 1st one is by adjusting the structure of the network and allocating an appropriate amount of computations according to each sample, as a way to reduce the redundant computations of the "simple" samples , so as to improve the inference efficiency. On the premise of keeping computation volume constant, the network parameters are adjusted according to each sample to increase the operation efficiency with the minimum cost. Because different inputs have different computation requirements, it is natural to carry out dynamic inference according to the characteristics of each sample. Specifically, from the second point of view, the depth or width of the network can also be adjusted to achieve the same goal. The organization with versatile capacity saves repetitive calculations for standard examples, yet additionally keep their portrayal capacity while distinguishing non-standard examples.

2.1.1. Dynamic depth. As modern neural networks need to recognize more and more "hard" samples in depth, a direct solution to reduce redundant computations is to use dynamic network depth for inference, which can be realized by retreating in advance, that is, allowing "simple" samples to exit at the shallow layers without having to traverse the deeper layer of the network. Another method is called layer hopping which is supposed to selectively skip intermediate network layers conditionally upon each sample. Because deep neural network is a sequential execution model, it has a good interference effect in practical application.

Early exiting. The input samples are different, and the shallow network can identify some standard samples. Then, these specification samples should be output earlier than the deeper network.

For the info test x, the forward spread of the L-layer profundity network f can be communicated as follows.

$$y = F^{L}\circ F^{L-1}\circ\cdots\circ F^{1}(x) \tag{1}$$

Where F^L indicates the operation function of layer $1 \leq \ell_i \leq L$. on the contrary, early exit is allowed to terminate the inference process in the middle layer. For the ith input sample seat, forward propagation can be written as follows.

$$y_i = F^{l_i}\circ F^{l_i-1}\circ\cdots F^{1}(x_i), 1 \leq l \leq L \tag{2}$$

The most direct way to achieve early exit is to cascade multiple models and adaptively retrieve the prediction of the early network, so as not to activate the latter network.Because the models in the cascade structure are independent of each other. In this way, whenever it is chosen to criticism a "troublesome" example to the following organization, the entire induction process should be executed for as long as anyone can remember. Halfway classifiers. A more minimal plan is to remember moderate classifier for a spine organization, and if vital, early highlights can spread to the more profound layers (see Fig.1).
Layer skipping.The general idea is to skip performing all the deep layers after a certain classifier. More flexibly, the network depth can also be adjusted dynamically, strategically skipping the partners in the middle layer without placing additional classifiers.

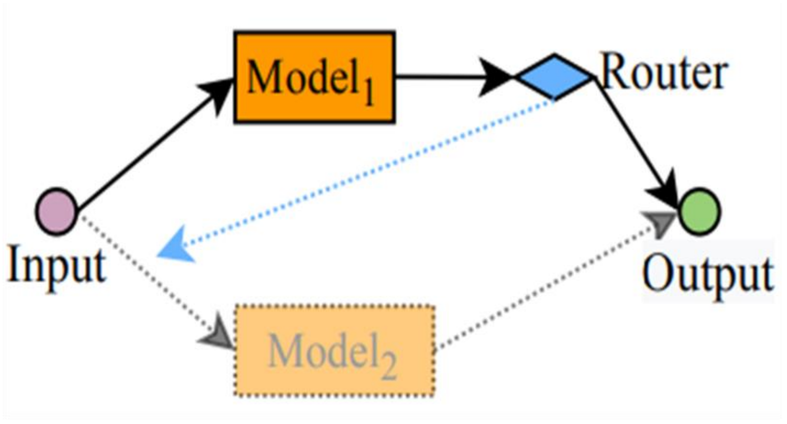

Figure 1. Cas cade of the model, whether there is a router decision in the dotted line part.

Adaptive time computations. When multiple layers are executed sequentially in a time step, a scalar called the pause score is accumulated. In the event that the score surpasses the limit, the secret condition of the RNN will be straightforwardly taken care of back to the subsequent stage.

Strobe function. In addition to comparing the calculated pause score with certain thresholds in the above method, due to the plug-and-play nature of the gating function, it is also a common choice for discrete decision making. By generating binary values based on intermediate features, the gating function can dynamically determine layer skipping or execution graphs(see Fig.2. Left).
Policy network. In addition to making sequential decisions based on intermediate features, another implementation is to use additional models to directly determine which layers in the network need to be executed based on a per instance basis. For example, block drop [34] builds a policy network, which takes each instance as input and generates binary gates for all layers in the pre-trained RESNET(see Fig.2. Right).

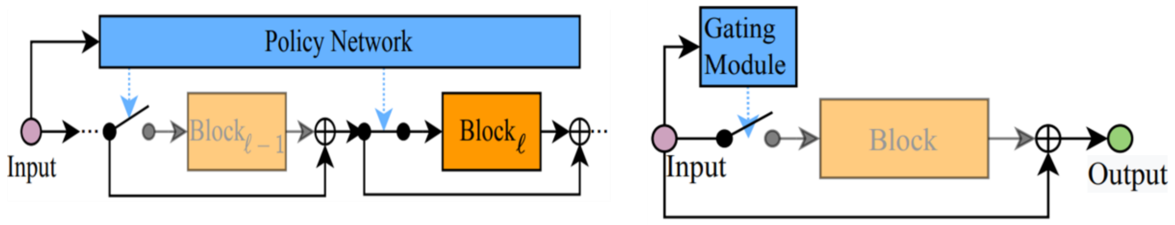

Figure 2. Layer skipping based on gating function.

*2.1.2. Dynamic width.*Another way to adapt the depth of network is to use dynamic width to perform inference: although each layer is still executing, its components (such as neurons, branches or channels) are selectively activated under input conditions. Therefore, this method can be regarded as a more fine-grained form of conditional computations.

Dynamic width of fully connected (FC) layer. How much calculation of the FC not entirely set in stone by its feedback and result aspects. It is generally believed that different neuron units represent different characteristics, so not all neuronal units need to be activated.[7,8,9,10]

Mixture of Experts (MoE). MOE can build the organization limit without extending the organization profundity [5], and that implies that numerous organization branches are inherent equal as specialists. ThFese specialists can perform specifically, and their results are intertwined with information related loads.

The traditional soft MOE method uses the re assigned weight to dynamically re scale the representations obtained from different experts(see Fig.3). However, all branches still need to be executed, and the amount of computations cannot be reduced during testing. In order to reduce the amount of computations and improve the inference efficiency, only a small number of hard gates with non-zero elements allow the model to allocate computations adaptively according to the input(see Fig.4).

Hard MOE has been implemented in different network structures. For example, HydraNet[11] replaced the convolution block in the last stage of CNN with multiple branches, and selectively executed these branches during testing. In the NLP task, the sparse gated MOE and the nearest switching converter [13] embed the hard MOE into the long-term and short-term memory [14] network and converter [6], respectively. In [16,11,13], only the branches corresponding to the Top-k element of the multivalued gate are activated, instead of using binary gates for selection as in [12].

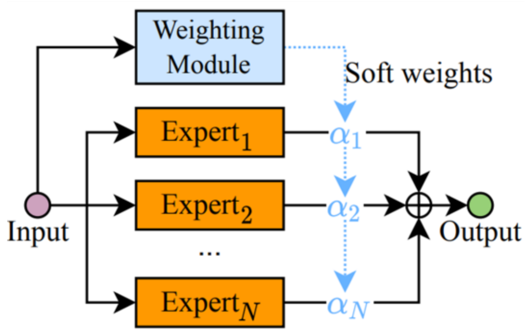

Figure 3. Soft weights for adaptive fusion.

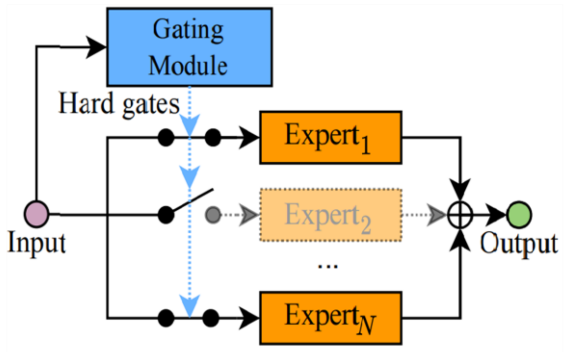

Figure 4. Selective execution of MoE branches.

Dynamic channel pruning in CNNs. Modern CNN usually has considerable redundancy in a large number of characteristic channels. Based on the consensus that the same channel may have different importance for different instances, the dynamic width of CNN can be realized by adjusting the number of channels at runtime. Compared with the static pruning method that permanently removes some filters, this dynamic pruning method selectively skips the computations of the channel in a data related manner. CNN’s capacity has not decreased, but the overall efficiency can be improved.

Dynamic channel pruning is a multi-stage model of dynamic depth, in which the later stage is conditionally performed according to early prediction. You can also build a multi-level network structure along the width and implement these networks step by step according to the needs. The aforementioned incremental activation paradigm determines the execution of later stages according to the previous output. Therefore, it is necessary to perform complete forward propagation for each stage, which may be suboptimal for reducing actual inference delay. Another popular solution is to determine the execution of each layer channel according to the gating function. Without auxiliary branching and computational redundancy, dynamic pruning can be carried out directly according to the feature activation value [15], and a regularization term is introduced into training to encourage the sparsity of intermediate features.

2.1.3. Dynamic routing. Most of the above methods adjust the depth and width to achieve the computing goal by activating the computing unit based on the input condition, while the computing diagram is adjusted by adjusting the dynamic path origin in the network. In order to achieve dynamic routing, there are usually routing nodes in the network responsible for allocating samples to different paths. Here we mainly discuss different networks and routing adjustments.

Multi branch structure.The simplest super network can be implemented by setting several candidate modules in each layer and dynamically selecting one of them. This is equivalent to making the probability distribution in equation 6 a hot state and can be regarded as a special form of hard MOE. The main difference is that only one branch is selected without any fusion operation. Various implementations have been explored.

Neural tree and tree structure network.Because the decision tree infers along a forward path that depends on input attributes, the combination of tree structure and neural network can enjoy the adaptive inference paradigm and representation ability of DNNS at the same time.

Hand designed super network. The supernetwork or NAS can be designed manually, and the routing policy of each instance is determined by the additional network or plug-in module[16,17].

2.1.4. Dynamic parameters. Dynamic architecture can adjust its inference diagram according to each instance and realize effective computation allocation. They usually have special architecture design and need specific requirements. Training strategy or careful super parameter adjustment. Another work is to use dynamic parameters to perform adaptive inference while keeping the architecture fixed. The existing technology has proved to be effective in improving the representation ability of the network, but the computing cost has increased slightly.

Generally speaking, parameter adaptation can be realized from three aspects:

Adjust the training parameters according to the input.

Generate network parameters directly from input.

Rescale features with soft attention.

Parameter adjustment.A typical method of parameter adaptation is to adjust the weight according to the input in the inference process. This implementation usually consumes very little computation to obtain adjustments.

Kernel shape adaptation.In addition to adaptively adjusting the weight, the convolution kernel can also be reshaped by adjusting the parameters to realize dynamic reception in the field. In this direction, when convolution is performed on each pixel, the deformable convolution samples pixels from adaptive positions in the feature map. Deformable kernel space weights are sampled to adapt to the effective receiver field while keeping the receiver field unchanged[18].

Weight prediction.Compared with the dynamic modification of model parameters, the weight prediction is more direct: it uses an independent model to directly generate instantaneous parameters Tu. This thought was first advanced in "Figuring out how to Control Fast-weight Memories: An option in contrast to dynamic repetitive networks",in which the weight forecast model and spine network model are feedforward networks. Ongoing work further stretches out this worldview to current organization designs and errands.

Dynamic sifting organization and super organization are two traditional strategies to understand the weight forecast of CNNs and RNNs separately. In particular, a channel age network is developed in DFN to create a channel for the convolution layer. Concerning handling consecutive information, the weight network of the fundamental RNN is anticipated by the more modest RNN at each time step contingent on input [19]. Simultaneously, task explicit data is likewise used to progressively foresee model boundaries.

Dynamic characteristics.The main effect or prediction parameters of inference with the adjusted method produce more dynamic and information features, thus improving the expressive ability of depth model. A more direct solution is to rescale the feature map using soft attention that depends on input. This dynamic features is easier to obtain, because the calculation chart needs to be slightly modified.Soft attention is widely used in various fields because of its simplicity and effectiveness. In addition, it can be easily combined with other methods.

2.2. Spatial adaptive dynamic network

In visual learning, it is observed that not all areas contribute similarly to the last forecast of CNN [23], which shows that spatial unique processing needs to lessen computational overt repetitiveness. As such, making the right calculations may just have to handle a little piece of pixels or locales, rather than the general calculations. In addition, taking low resolution as input is enough to produce good computations results [25], while static CNN receiving all inputs at the same resolution may produce a lot of redundancy.

Therefore, a spatial dynamic network is established to carry out adaptive inference for different spatial positions of the image. According to the granularity of dynamic computations, we further classify the related methods: 1) pixel level, in which each pixel in the feature will be adaptively processed. 2) Region level, where the model focuses only on strategically selected regions. 3) Resolution level, and each input image adopts adaptive resolution.

2.2.1. Pixel level dynamic networks. Ordinary spatial dynamic networks perform adaptive computation at the pixel level. At present, there are mainly two types of pixel-level dynamic networks. The first is the model with dynamic architecture, which can adjust the depth or width of each pixel when dealing with its characteristics. The second method is to use pixel-specific weights to perform convolution for the networks with dynamic parameters, so as to improve the flexibility of the feature representation.

Pixel level dynamic architecture.Based on the general understanding that the foreground pixels provide more information and higher computational requirements than the background pixels, some dynamic networks learn to adjust their architecture for each individual pixel. Existing work usually achieves this purpose by two manners. The first one is by utilizing sparse convolution, which convolutes only a subset of the sampled pixels. The second one is by additional refinement, which strategically allocates additional computations at certain spatial locations, for example, layers or channels.

Dynamic sparse convolution. In order to reduce unnecessary computations for the locations with less information, convolution can only be performed on strategically sampled pixels. The quality of the sampling feature location determines the accuracy and efficiency of the network to a great extent.

Dynamic additional refinement.Different from sampling a subset of pixels to perform convolutions, another series of work first compute the entire feature map that is relatively cheap for the pixels, and then adaptively activates the additional modules on some pixels for further refinement.

Dynamic pixel level parameters.The convolution weight after training can be dynamically adjusted through pixel-level attention. In addition to dynamic modification, weight prediction is also used to directly generate position specific convolution kernels.

Dynamic receive byte.Traditional convolution operations usually have cores of fixed shape and size, and the resulting unified receiving field across all layers may limit the recognition of objects with different shapes and sizes. In order to solve this problem, a series of work has learned the adaptive receiver fields with different feature positions. In addition, adaptive sampling can be carried out in kernel space instead of feature space to realize the adaptation[18]. However, the adaptive connection network [24] does not adjust the sampling position of the feature or kernel, but realizes self-transformation, local inference and global inference. The three branches of the output are fused with data related weighted summation. In addition to images, local and global information in non-European data, such as graphics, can also be aggregated adaptively.

Pixel level dynamic features. Applying soft attention in spatial direction to features can effectively increase the representation ability of the model [2,20,21,22]. Although it is equivalent to using dynamic weights to perform convolution, it is usually easier to achieve direct rescaling of features in practice.

Regional dynamic network.Pixel level dynamic networks need a specific implement for sparse computing, so they may face challenges in realizing real acceleration in hardware. Another method is to perform adaptive inference on the region or patch of the input image. At present, there are two mainstream methods: one is to perform parametric transformation on the region from the input feature map to make more accurate predictions [25， 36]. Another learning focus on the selected patches [4,26,27] is to improve the effectiveness and / or efficiency of the model.As shown in Fig.5, the region selection module generates transformation parameters or pays attention to the location of the region, and the subsequent network infers the input of transformation / clipping.

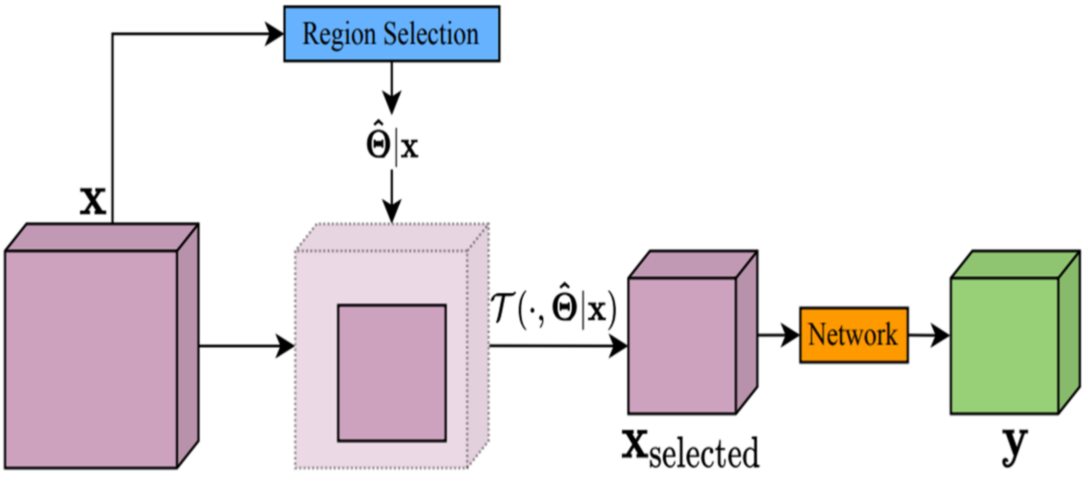
Region Selection
$\hat{\Theta}|\mathbf{x}$
$\mathbf{x}$
$\mathcal{T}(\cdot, \hat{\Theta}|\mathbf{x})$
Network
$\mathbf{x}_{\text{selected}}$
$\mathbf{y}$

Figure 5. Dynamic convolution on selected spatial locations.

Pixel transformation.The image can be dynamically transformed to eliminate some changes [25] to obtain better generalization ability, or exaggerate the significant region [36] to obtain effective visual attention. In addition, on the task that the model performance is sensitive to a small part of the region, the learning transformation is used to adaptively enlarge the significant region of the image.

Follow the selected patch. Inspired by the fact that information features may only correspond to certain areas of the image, a dynamic network with high attention is proposed to strategically select patches from the input to improve efficiency.

Hard attention of RNN.The most typical method of hard attention at the regional level is to formulate the classification task as a sequential decision-making process. In this direction, RNN is used to focus on only one patch at a time and make predictions iteratively [26,28]. At each step, the classifier RNN only sees a cropped patch and determines the next attention position until the last step is reached. By including early stop in the action space, the adaptive number of steps is further realized [28]. By allowing early exit, spatial and temporal adaptive inference can be realized [4, 28].

Strict attention.Rather than utilizing RNN to foresee the area of the district that the model should zero in on, class enactment planning (CAM) [23] is utilized to iteratively choose critical patches [29]. In every emphasis, choice is performed on the recently trimmed information, bringing about an ever-evolving refinement process, in which each scale takes the fix edited by the past scale as the info, and is answerable for learning the element portrayal for arrangement and the consideration guide of the following scale simultaneously.

2.2.2. Resolution level dynamic network. The research discussed above usually divides the feature map into different regions and processes them in an adaptive manner. One disadvantage of these methods is that sparse sampling or clipping operations may reduce actual efficiency. Traditional CNN mostly processes all inputs at the same resolution, resulting in considerable redundancy. Therefore, the resolution level dynamic network makes use of spatial redundancy from the angle of characteristic resolution rather than the importance of different positions. The prior art mainly includes down sampling

/ up sampling with adaptive scaling [30,31]and selectively activating subnetworks with different resolutions in multi-scale architecture [3,32].

Adaptive scaling.By performing down / up sampling with adaptive scaling, features with dynamic resolution can be generated. Then, the plug-in module is further used to predict the stride of the first convolution layer in each RESNET stage to generate features with dynamic resolution [31].

Dynamic resolution in multi-scale architecture.Another way to achieve dynamic resolution is to build multiple subnetworks in parallel [32] or cascade [3]. These sub organizations with various element goals are specifically enacted by the contribution during deduction. To stay away from repetitive figuring, hard choice is acknowledged through the goal versatile organization (ranet) [3], which permits each occurrence to restrictively actuate the sub organization to deal with the element portrayal from low to high goal.

3. Evaluation and discussion

We have introduced sample adaptive neural networks and spatial adaptive neural networks, which are not much different from previous neural networks. What are the advantages of dynamic neural networks compared with traditional static neural networks?

3.1. Speed up

First, the underlying structure, the router connects the various networks, dynamic routing can adapt to the adjustment of the network structure, while static routing is fixed. For example, in the case of sample dynamic networks, the sample dynamic networks usually processes different samples in a data-dependent manner, and because of the difference between the static route and the dynamic routes, the dynamic routes can be adjusted according to the input samples. By jumping layer, exit early etc, the redundancy of neural network computing is reduced so as to improve the efficiency of inference. Fig.6.compares this method with other networks to see the efficiency improvement.

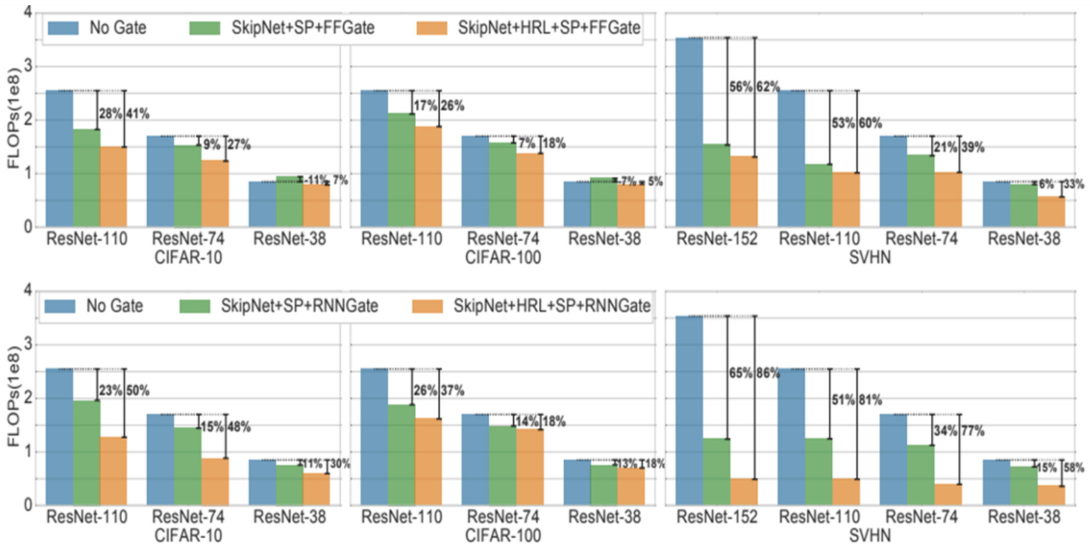

Figure 6. Computation reduction of SkipNet and SP and Skip NetHR and SP with feed-forward gates and recurrent gates while preserving the full network accuracy.

Using feed-forward and recursive doors reduces the amount of computation for Skip Net-SP and Skip Net-HRL-SP while maintaining complete network accuracy. The computations cost includes the computations of the gate. We were able to reduce the computational costs of the deepest models on THEFAR-10, 100 and SVHN data by 50%, 37% and 86%, respectively. Fine tuning with HRL can reduce computation by 10% or more than using SP only. Because feed-forward gate is more expensive, Skip Net with a loop gate usually saves more cost, which means that the calculation speed of Skip Net has been significantly improved.

Fig.7 (left) reduces the amount of computations by 12-30% by using RNN· Gates' skipping network, while maintaining complete network precision. Fig.7 (Right) Trade-offs between accuracy and cost under different α. When the α is small, the computations drops faster than the accuracy drops.

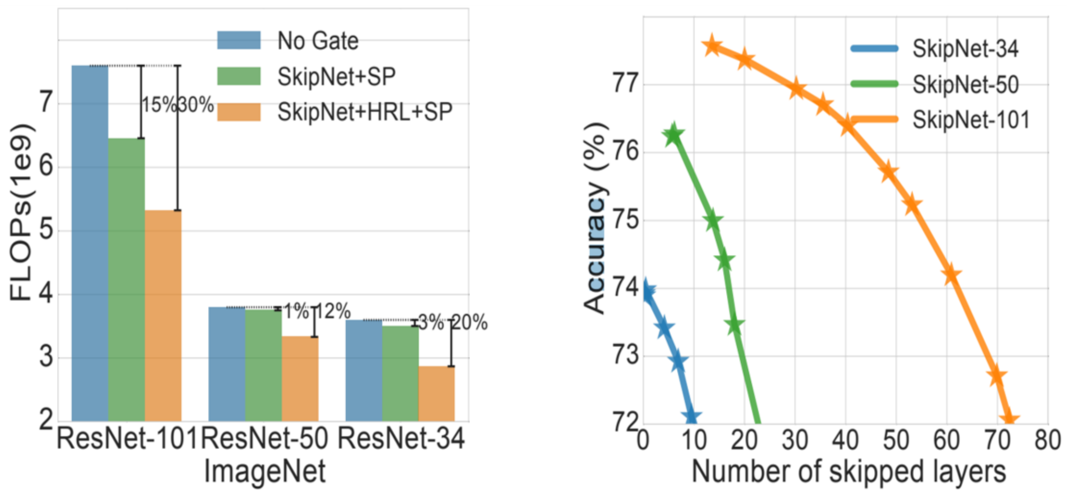

Figure 7. Computation Reduction. Accuracy - 1umber of skipped layers.

The above method is to adaptively change the network structure through router to improve the efficiency. There is another method in the sample neural network, that is, to adjust the network parameters to adapt to each sample, which is a typical method to adjust the weight according to the input of the interference process.

Table 1 shows the ImageNet validation accuracy and inference costs of the Cond Conv model under several baseline model architectures. Cond Conv improves the accuracy of all baseline architectures, but at the same time reduces the cost of inference by 10%, because only a small amount of computations are needed for weighting or forecasting.

Table 1. Imagenet verification accuracy and inference cost of cond conv model under baseline model architecture.

Parameter / Type	Baseline		CondConv	
	MADDs (×10^6)	Top -1(%)	MADDs (×10^6)	Top -1(%)
MobileNetV1(1.0x)	567	71.9	600	73.7
MobileNetV2(1.0x)	301	71.6	329	74.6
MnasNet-A1	312	74.9	325	76.2
ResNEt-50	4093	77.7	4213	78.6
EfficientNet-B0	391	77.2	413	78.3

3.2. Model size

As for the capacity of model, the previous explanation of sample neural network also clearly mentioned that since dynamic routing in the network precedes the reasoning network in path planning or weight prediction, it will inevitably increase the requirements of computation amount and the scale of the model [33], but I want to emphasize that the dynamic neural network increases the computation amount of dynamic routing before the network computing in order to reduce the redundancy of neural network calculation, so in Skip net and CondConv [13].

3.3. Accuracy

We've described the inference speed and model size of dynamic neural networks, and then we've compared the inference speeds of these optimized networks, and in the case of CondConv-EfficientNet-B0, the CondConv-EfficientNet-B0 model achieves the best performance of 78.3% accuracy when added up by 413M multiplication MixConv: Mixed Depthwise Convolutional Kernels(see Fig.8).

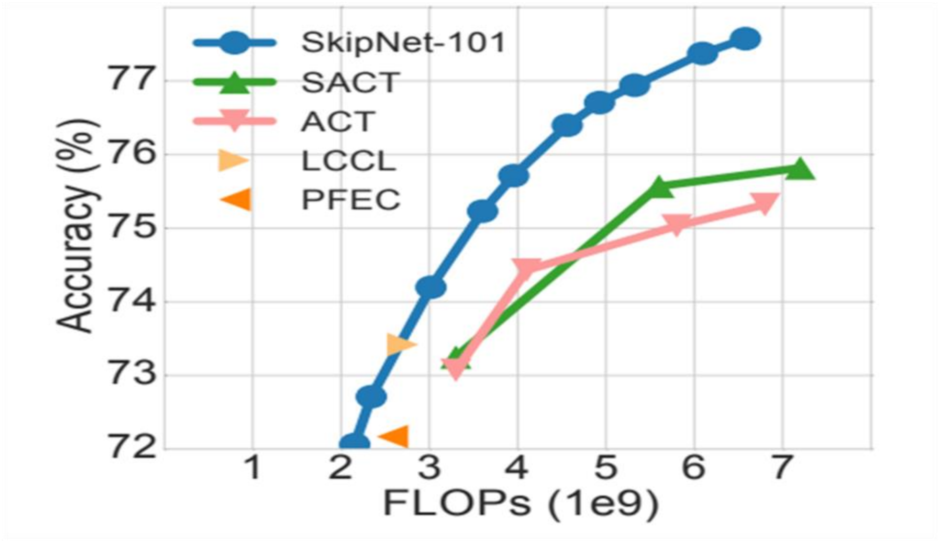

Figure 8. Comparison with others.

Back in Table 1, scale the CondConv-EfficientNet-B0 model to a depth multiplier of 1.1x to get the upgraded CondConv-EfficientNet-B0-depth. The CondConv-EfficientNet-B0 depth model requires only 614M multiplication addition to achieve 79.5% accuracy. When training with the same hyperparameter and regularized search space, the EfficientNet-B1 model scaled from the EfficientNet-B0 model using a composite coefficient achieves 79.2% accuracy in the multiplication operation of 700M.

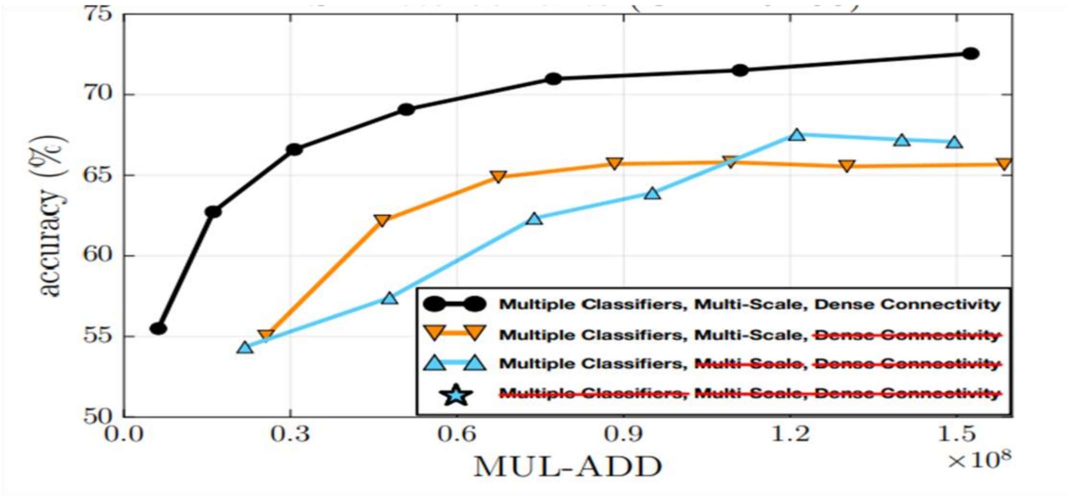

Figure 9. Ablation study.

In Fig. 9, we can also observe skipNet's comparison with the latest model. SkipNet has improved by about 3-5 percent over existing methods in both benchmarks, based on various trade-offs between computational cost and predictive accuracy.

3.4. Ablation study

Taking [35] as an example, the network is composed of three main components, namely multi-scale feature mapping, dense connectivity and intermediate classifier.

Start with an MSDNet with six intermediate classifiers and remove three main components at a time. Adjust the width of the network (the number of output channels per layer) to keep the computational cost of the entire network at about 3.0 × 108 times.

After removing all three components of MSDNet, a rule-like VGG convolution network is obtained. The figure shows the classification accuracy of all classifiers in the model. The following points can be observed.

Dense connection is very important to the performance of msDNet, and moving it can seriously affect overall accuracy (orange and black curves.

Removing multi-scale convolution only affects the accuracy of low-budget areas, consistent with our motivation that multi-scale design introduces identification features at an early stage.

In the end, STAR performs the function similar to MSDNet under a specific budget that completely matches the estimated cost, but it is not suitable for different budget constraints. CNN's final performance in a specific budget field is much better than that of the model without dense connection (orange curve). This indicates that the combination of dense connectivity and multi-classifiers is particularly important.

4. Conclusion

The above dynamic reasoning technology has made great progress and has been applied in many networks. However, when most networks make decisions, the results are effective, but some problems are not necessarily the best solution. For example, when making early exit decisions, the optimal scheme is overconfident and the threshold setting is also very sensitive. At the same time, we also notice that most of the existing networks have very clear directivity, so we can design a unified dynamic network to perfectly handle most of the problems we encounter, or as a backbone network.

References

[1] Y. Han, G. Huang, S. Song, L. Yang, H. Wang and Y. Wang, "Dynamic Neural Networks: A Survey," in IEEE Transactions on Pattern Analysis and Machine Intelligence.
[2] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In ECCV, 2018.
[3] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. Resolution Adaptive Networks for Efficient Inference. In CVPR, 2020.
[4] Yulin Wang, Kangchen Lv, Rui Huang, Shiji Song, Le Yang, and Gao Huang. Glance and focus: a dynamic approach to reducing spatial redundancy in image classification. In NeurIPS, 2020.
[5] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. Neural computation, 1991
[6] Tolga Bolukbasi, Joseph Wang, Ofer Dekel, and Venkatesh Saligrama. Adaptive neural networks for efficient inference. In ICML, 2017.
[7] Yoshua Bengio, Nicholas Leonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. arXiv preprint arXiv:1308.3432, 2013.
[8] Kyunghyun Cho and Yoshua Bengio. Exponentially increasing the capacity-to-computation ratio for conditional computation in deep learning. arXiv preprint arXiv:1406.7362, 2014.
[9] Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. Conditional computation in neural networks for faster models. ICLR Workshop, 2016.
[10] Andrew Davis and Itamar Arel. Low-rank approximations for conditional feedforward

computation in deep neural networks. arXiv preprint arXiv:1312.4461, 2013
[11] Ravi Teja Mullapudi, William R Mark, Noam Shazeer, and Kayvon Fatahalian. Hydranets: Specialized dynamic architectures for efficient inference. In CVPR, 2018.
[12] Shaofeng Cai, Yao Shu, and Wei Wang. Dynamic routing networks. In WACV, 2021.
[13] William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. arXiv e-prints, 2021
[14] Sepp Hochreiter and Jurgen Schmidhuber. Long short-term ¨memory. Neural computation, 1997
[15] Chuanjian Liu, Yunhe Wang, Kai Han, Chunjing Xu, and Chang Xu. Learning instance-wise sparsity for accelerating deep models. In IJCAI, 2019.
[16] Yanwei Li, Lin Song, Yukang Chen, Zeming Li, Xiangyu Zhang, Xingang Wang, and Jian Sun. Learning Dynamic Routing for Semantic Segmentation. In CVPR, 2020.
[17] An-Chieh Cheng, Chieh Hubert Lin, Da-Cheng Juan, Wei Wei, and Min Sun. Instanas: Instance-aware neural architecture search. In AAAI, 2020.
[18] Hang Gao, Xizhou Zhu, Stephen Lin, and Jifeng Dai. Deformable Kernels: Adapting Effective Receptive Fields for Object Deformation. In ICLR, 2019.
[19] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. In ICLR, 2016.
[20] Abhijit Guha Roy, Nassir Navab, and Christian Wachinger. Concurrent spatial and channel 'squeeze & excitation'in fully convolutional networks. In MICCAI, 2018.
[21] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In CVPR, 2017
[22] Shenlong Wang, Linjie Luo, Ning Zhang, and Li-Jia Li. Autoscaler: Scale-attention networks for visual correspondence. In BMVC, 2017
[23] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In CVPR, 2016.
[24] Guangrun Wang, Keze Wang, and Liang Lin. Adaptively connected neural networks. In CVPR, 2019.
[25] Max Jaderberg, Karen Simonyan, and Andrew Zisserman. Spatial transformer networks. In NeurIPS, 2015.
[26] Volodymyr Mnih, Nicolas Heess, and Alex Graves. Recurrent models of visual attention. In NeurIPS, 2014.
[27] Jianlong Fu, Heliang Zheng, and Tao Mei. Look closer to see better: Recurrent attention convolutional neural network for finegrained image recognition. In CVPR, 2017.
[28] Zhichao Li, Yi Yang, Xiao Liu, Feng Zhou, Shilei Wen, and Wei Xu. Dynamic computational time for visual attention. In ICCV Workshop, 2017.
[29] Amir Rosenfeld and Shimon Ullman. Visual concept recognition and localization via iterative introspection. In ACCV, 2016.
[30] Zekun Hao, Yu Liu, Hongwei Qin, Junjie Yan, Xiu Li, and Xiaolin Hu. Scale-aware face detection. In CVPR, 2017.
[31] Zerui Yang, Yuhui Xu, Wenrui Dai, and Hongkai Xiong. Dynamic-stride-net: deep convolutional neural network with dynamic stride. In SPIE Optoelectronic Imaging and Multimedia Technology, 2019.
[32] Huiyu Wang, Aniruddha Kembhavi, Ali Farhadi, Alan L. Yuille, and Mohammad Rastegari. Elastic: Improving cnns with dynamic scaling policies. In CVPR, 2019.
[33] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In ECCV, 201845
[34] Zuxuan Wu, Tushar Nagarajan, Abhishek Kumar, Steven Rennie, Larry S Davis, Kristen Grauman, and Rogerio Feris. Blockdrop: Dynamic inference paths in residual networks. In CVPR, 2018.
[35] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Weinberger. Multi-scale dense networks for resource efficient image classification. In ICLR, 2018.

[36] Adria Recasens, Petr Kellnhofer, Simon Stent, Wojciech Matusik, and Antonio Torralba. Learning to zoom: a saliency-based sampling layer for neural networks. In ECCV, 2018.