Feature-Dependant Cross-Connections in Multi-Path Neural Networks

Dumindu Tissera*†, Kasun Vithanage†, Rukshan Wijesinghe*†, Kumara Kahatapitiya*,
Subha Fernando† and Ranga Rodrigo*
*Department of Electronic and Telecommunication Engineering, University of Moratuwa, Sri Lanka
†CODEGEN QBITS Lab, University of Moratuwa, Sri Lanka

***Abstract*—Learning a particular task from a dataset, samples in which originate from diverse contexts, is challenging, and usually addressed by deepening or widening standard neural networks. As opposed to conventional network widening, multi-path architectures restrict the quadratic increment of complexity to a linear scale. However, existing multi-column/path networks or model ensembling methods do not consider any feature-dependant allocation of parallel resources, and therefore, tend to learn redundant features. Given a layer in a multi-path network, if we restrict each path to learn a context-specific set of features and introduce a mechanism to intelligently allocate incoming feature maps to such paths, each path can specialize in a certain context, reducing the redundancy and improving the quality of extracted features. This eventually leads to better-optimized usage of parallel resources. To do this, we propose inserting feature-dependant cross-connections between parallel sets of feature maps in successive layers. The weights of these cross-connections are learned based on the input features of the particular layer. Our multi-path networks show improved image recognition accuracy at a similar complexity compared to conventional and state-of-the-art methods for deepening, widening and adaptive feature extracting, in both small and large scale datasets.**

I. Introduction

Learning a particular task in a dataset while handling the diversity among input samples, essentially requires the model to adapt to the context of the input. The naive approaches of network deepening or widening [1]–[4] tend to improve the performance as a result of the deep abstract feature extraction or the increased number of features extracted. Having multiple parallel paths/columns in each layer [5], [6], as opposed to conventional widening, prevents the quadratic increment of network complexity as a result of width enhancement. In model ensembling methods [1], [2], each model is expected to converge to slightly different local optima, giving a better combined performance. However, these approaches lack the adaptability to diverse contexts of input samples, and hence, any resource increment due to network deepening or widening is not well utilized and subjected to feature redundancy.

In this view, processing an input according to its context is important because, in such a setting, models can extract different sets of features with varying priorities. To this end, networks that adaptively extract information have been introduced to converge to better optimas in complex datasets, without merely increasing the number of parameters [8]–[12]. These approaches utilize additional parametric functions which process the inputs of the model or each layer to accommodate the differences in data points while supporting the learning of the main task. However, they are still limited in the sense of utilizing a single column.

(a) (b) (c)

Fig. 1: Three samples from the ImageNet [7] validation set. The First two images belong to the class hummingbird, whereas the third image is of an electric ray. In terms of the dominant color, the last two images are closer to each other. But if we consider abstract features such as the body patterns, pose or the type of animal, the first two images are closer to each other. Therefore, it is intuitive to learn different groups of features separately, in each layer of a network

The context of an input sample does not necessarily reflect the corresponding class but can provide more information, which may be present in multiple levels of abstractions. Figure 1 shows three samples (Image 1a, 1b and 1c) in the ILSVRC2012 [7] validation set. Image 1a and 1b show hummingbirds sitting on a leafy or a thorny branch, with grass or sky as backgrounds, respectively. Image 1c shows an electric ray immersed in water with a distinct body pattern and a pose that is different from the first two images. These contextual representations are distributed in multiple levels of a deep neural network. For instance, initial layers may capture the dominant color, the structure of edges and corners, whereas latter layers may capture more abstract information of the context such as the pose or even the class. Moreover, the image context which matters for the problem might be different from the visual contextual description provided by a human [13]. Simply put, the context of an input sample is distributed throughout multiple layers of a network.

In a parallel-path neural network, multiple groups of feature maps can be processed within a single layer. If each group contains a set of homogeneous feature maps that relates to a

distinct context, the filters extracting features on the particular group can be learned to be specialized to the respective context. This enables the network to be more efficient, having a smaller set of dedicated filters per group. Such a smart usage of resources leads to improved performance at a given complexity in contrast to having a single large set of filters per layer. To do such grouping of feature maps in each layer and to allocate inputs to the layer accordingly, we need an adaptive mechanism that routes between parallel sets of feature maps of two adjacent layers in a feature-dependant manner.

Since the contextual representation of an input sample is distributed along with the depth of a network, it is important to have such mechanisms to allocate resources per each layer or layer segment according to the level of context represented by the incoming feature maps. Hence, two samples which are allocated to similar paths in the initial layers might get different path allocations in later layers. For example, Image 1b and 1c share similar background color domains which are different from Image 1a. However, if body patterns and pose is considered, Image 1a and 1b share more similarities than Image 1c. Therefore in such a multipath network, Image 1b and 1c might be processed along similar paths in initial layers even though they represent two distinct classes. In last layers, Image 1a and 1b may get allocated to similar paths.

In this paper, we propose a multi-path architecture which consists of a novel adaptive cross-connection mechanism between parallel sets of feature maps in successive layers of the network. Our scheme enables the parallel paths of the network to be specialized in distinct contexts while having a soft routing between them to process the inputs through context-dependent pathways. The weight coefficients of the cross-connections are learned based on incoming sets of feature maps. We insert these cross-connections at selected locations along the depth of the network to perform a selective routing end-to-end. The outputs of such cross-connections act as inputs to the next layer or layer segment with dedicated families of filters per each parallel path. We summarize our main contributions of this paper as follows,

- We introduce a feature-dependant cross-connection mechanism between parallel paths of a multi-path network. The intuition here is to group homogeneous features of each layer into parallel paths and assign a soft routing between those paths, conditioned on the input. It allows the network to be more efficient, having paths specialized for different contexts and appropriate feature mixing between them.
- Our models surpass equivalent baselines and state-of-the-art counterparts for network deepening, widening and adaptive feature extraction with comparable complexity on image classification datasets.
- We empirically show that the resource allocation between parallel paths of a layer is based on its level of context abstraction. In fact, the gates at initial stages route simple contexts such as colors, and gates at latter stages route more complex contexts such as types of objects.

II. Background

Neural network deepening [1]–[3], [14]–[16] or widening [4], [5] shows promising improvements in complex datasets over conventional shallow neural networks [17], [18]. However, these methods do not effectively utilize the network capacity. Recent advances in adaptive learning, in which, the main task is supported by additional functions operating on input samples/feature maps of each layer, show a remarkable improvement over traditional deep or wide networks. These approaches either use additional learnable parametric functions to make the feature extraction of each layer sensitive to its input features [8], [9], [11], [19], [20], or use latent sub-networks which predict the parameters of the main network based on a bulk of input samples [10]. Another set of approaches use non-learnable computations between layers to regulate the flow of information dynamically [21], [22]. Among these, our method is closely related to the use of feature-dependant layer-wise supportive functions.

SENet [8] introduced an adaptive re-calibration of feature maps by weighting them channel-wise, based on coefficients learned through a non-linear function corresponding to the particular set of feature maps. This is commonly referred to as channel-wise attention. A feature-dependant support has also been used for adjusting the depth of a network based on the input. In particular, both ConvNet-AIG [11] and BlockDrop [12] introduced supportive algorithms to adaptively change the effective depth of Residual Networks [3] per input sample. ConvNet-AIG uses a non-linear regularized parametric function in each residual block to produce hard gates values which decide whether to keep or drop the corresponding residual block ,whereas BlockDrop takes this decision based on reinforcement learning. Highway networks [9], [23] use additional gates computed on features maps to pass information across layers, without subjecting to attenuation. However, these approaches are limited to using a single column, whereas we intend to use such feature-dependant support to allocate/route input features among parallel paths.

Approaches which use parallel convolution or residual operations in each layer [5], [6], [24], or model ensembling [1], [2] methods, do not allocate input samples or input feature maps to parallel operations in a data-dependent manner, and therefore, the parallel resource usage is not well optimized. The utility of parallel neural networks having each path conditioned on inputs pre-processed in different ways, have proven to produce better results [25], [26]. But these work do not introduce any connections between parallel neural networks, which is essential if we build a system to perform a soft assignment of paths in each layer, based on its level of context representation. To summarize, the utility of multi-path/column networks with feature-dependent resource allocation in each layer for learning a single task is not explored.

Multi-path networks with cross-connections between hidden layers that carry parallel tensors are widely used in the common multitasking domain [27], in which, multiple distinct tasks are performed on a given input sample. Cross-Stitch

networks [28] first proposed the use of cross-connections between parallel paths in selected layers of multi-path networks. However, the stitching coefficients which weights the cross-connections are independently learned as conventional weights, and therefore, fixed for the whole dataset after training. Both Sluice Networks [29] and NDDR-CNN [30] build upon the intuition of Cross-Stitch Networks.

Since the intention behind Cross-Stitch Networks is to learn a mix of task-specific and shared resources to learn multiple tasks per single input, the stitching coefficients being independently learned is sufficient. However, to build a multi-path network that performs path assignment based on the context of each hidden layer, so that the network can regulate the information flow end-to-end, these coefficients should be data dependant. Therefore, we introduce a set of gating coefficients, generated as functions of the corresponding input tensors to the cross-connection process. Our work can be interpreted as a fusion between the cross-stitching [28] and channel-wise attention [8], where the weighting coefficients of cross-connections are now learned by a set of non-linear, parametric, attention-like functions which operates conditioned on the input in contrast to being learned individually.

III. Adaptive Cross-Connections

The adaptive cross-connection algorithm which routes between two subsequent layers with parallel sets of feature maps, say input and output layers of cross-connections, is a two-step process. First, each input tensor, i.e., the input set of feature maps, is fed into a non-linear parametric function to calculate gate probabilities/coupling coefficients which couple the corresponding tensor with each tensor in the output layer. Next, each tensor in the output layer is constructed based on a summation over all the input tensors weighted by the corresponding coupling coefficients. The coupling coefficients calculated as functions of the input tensors makes this a data-dependent process.

Let's assume there are m tensors $[\mathbf{X}_{i=1\ldots m}]$ in the input layer and n tensors $[\mathbf{Y}_{j=1\ldots n}]$ in the output layer of a cross-connecting process. First, each tensor in the input layer predicts an n-dimensional vector $\mathbf{G}_i$, which contains n probabilities which correspond to the coupling coefficients between tensor i and each output tensor. $\mathbf{G}_i$ can be expressed as $[g_{i1}, \ldots, g_{in}]$, where g_{ij} corresponds to the scalar gate value between $\mathbf{X}_i$ and $\mathbf{Y}_j$.

The simplest way of calculating $\mathbf{G}_i$ is to directly perform a non-linear parametric computation on $\mathbf{X}_i$. However, this approach increases the number of parameters of this sub-network and the routing overhead, in turn. This increment is more evident if $\mathbf{X}_i$ is 3-dimensional. Therefore if $\mathbf{X}_i \in \mathbb{R}^{H\times W\times C}$, we feed $\mathbf{X}_i$ to a global average pooling layer to produce a $1\times1\times C$ descriptor with channel-wise means. Since each channel in a convolutional feature map represents information extracted by a specific kernel, global average pooling compresses input

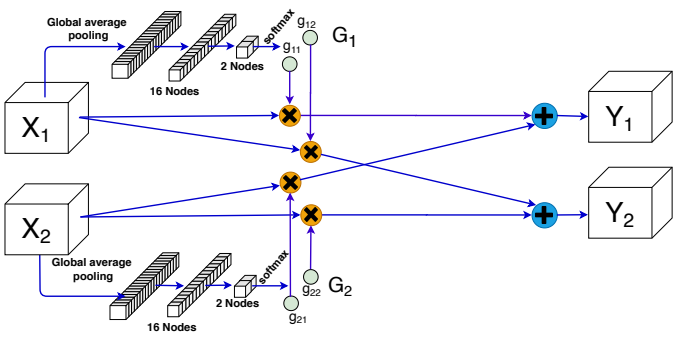


Fig. 2: Cross-Connections between two subsequent layers carrying 2 sets of feature maps each. The gating coefficients that weight the connections are learned based on incoming feature maps.

while retaining important information related to the context. Each element in the output of pooling $\mathbf{Z}_i$ is given by,

$$(z_i)_c = \frac{1}{H \times W} \sum_{a=1}^{H} \sum_{b=1}^{W} (x_i)_{a,b,c}.$$

$\mathbf{Z}_i$ is then fed into a non-linear computation to produce n latent relevance scores $\mathbf{A}_i$ ($[a_{i1}, \ldots, a_{in}]$). The non-linear function comprises of two fully-connected layers with 16 and n number of nodes repectively, separated by ReLU activation.

$$\mathbf{A}_i = \mathbf{W}_2(\mathrm{ReLU}(\mathbf{W}_1\mathbf{Z}_i))$$

We impose $\mathrm{softmax}$ activation on top of the n latent relevance scores $\mathbf{A}_i$ to calculate the gate probability vector $\mathbf{G}_i$.

$$\mathbf{G}_i = \mathrm{softmax}(\mathbf{A}_i), \quad i.e., \quad g_{ij} = \frac{\mathrm{e}^{a_{ij}}}{\sum_{k=1}^{n} \mathrm{e}^{a_{ik}}}.$$

This returns n softmax scores which represent the probabilities of $\mathbf{X}_i$ being routed into each output ($\mathbf{Y}_{j=1,\ldots,n}$).

To construct j^{th} output $\mathbf{Y}_j$, each input $\mathbf{X}_{i=1\ldots m}$ is weighted by the corresponding scalar gate value ($g_{i=1\ldots m,j}$), and summed over i.

$$\mathbf{Y}_j = \sum_{i=1}^{m} (g_{ij} \odot \mathbf{X}_i)$$

Here $\odot$ stands for the element-wise multiplication between g_{ij} and $\mathbf{X}_i$, which broadcasts the scalar g_{ij} to match the dimensions of $\mathbf{X}_i$.

We further show the cross-connection process below in matrix form for simple illustration of pixel-wise operations. Let $x_{i_{a,b,c}}$ be the pixel value at location (a, b, c) of X_i and $y_{j_{a,b,c}}$ be the pixel value at location (a, b, c) of Y_j. Let $\mathbf{G}$ be the $n \times m$ matrix with column i denoting n gate values computed from $\mathbf{X}_i$. The cross-connection between the two layers can be illustrated as,

$$\begin{bmatrix} y_{1_{a,b,c}} \\ \vdots \\ y_{n_{a,b,c}} \end{bmatrix} = \begin{bmatrix} g_{11} & \cdots & g_{m1} \\ \vdots & \ddots & \vdots \\ g_{1n} & \cdots & g_{mn} \end{bmatrix} \begin{bmatrix} x_{1_{a,b,c}} \\ \vdots \\ x_{m_{a,b,c}} \end{bmatrix}. \quad (1)$$

Algorithm 1 Routing between two adjacent layers with m input and n output sets of feature maps respectively.

Input:
$\mathbf{X}$: inputs $\{[\mathbf{X}_i \text{ for } i = 1, \ldots, m]\}$
Calculating gate values:
for $i = 1$ **to** m **do**
$\mathbf{Z}_i \leftarrow \text{global\_average\_pooling}(\mathbf{X}_i)$
$\mathbf{A}_i = [a_{i1}, \ldots, a_{in}] \leftarrow \mathbf{W}_2^i(\text{ReLU}(\mathbf{W}_1^i \boldsymbol{Z}_i))$
$\mathbf{G}_i = [g_{i1}, \ldots, g_{in}] \leftarrow \text{softmax}(\mathbf{A}_i)$
end for
Construction of outputs:
for $j = 1$ **to** n **do**
$\mathbf{Y}_j \leftarrow \sum_{i=1}^{m}(g_{ij} \odot \mathbf{X}_i)$
end for
Return:
$\mathbf{Y}$: outputs $\{[\mathbf{Y}_j \text{ for } j = 1, \ldots, n]\}$

Cross-Stitch Networks [28] use similar connections as in equation 1. However, their coupling coefficients are learned independently, whereas we produce them based on input feature maps. Learning the coefficients independently is sufficient to produce a fixed mixture of shared and task-specific resources for a given dataset, in a scenario where multiple tasks are performed on the same input. In contrast, our intuition is to specifically make these coefficients feature-dependant and let the model decide the mixture of context-specific and shared resources corresponding to a given input sample.

Figure 2 shows cross-connections between two adjacent layers, each carrying two parallel sets of feature maps. Algorithm 1 demonstrates the cross-connection algorithm. We insert the proposed adaptive cross-connection layers between a selected set of layers in parallel path networks. This allows the parallel paths in the layers without cross-connections to learn different context information independently. Since a cross-connection layer only calculates a path assignment and does not directly contribute towards learning features for the main task, adding such layers does not increase the depth of the model.

IV. Experiments

To validate the effectiveness of adaptive cross-connections, we conduct various experiments on both small and large scale image recognition datasets. In the ablation study conducted in CIFAR10 [31], we compare our multi-path networks with baselines and evaluate the effect of the number of parallel paths. Next, we compare our multi-path architectures with existing state-of-the-art (SOTA) adaptive feature extractors in CIFAR10 and CIFA100 [31]. We further evaluate our models in ImageNet/ILSVRC2012 [7] dataset to compare a two-path network with deeper and wider single path networks. We also use several visualization techniques on a multi-path CNN trained on a subset of ImageNet data to highlight the effect of adaptive path assignment.

TABLE I: Compared Nets: Cn - convolutional layer of n filters. Fn - fully connected layer of n nodes.

Network	Structure
BaseCNN	$C32$ $C32$ $C64$ $C64$ $C128$ $C128$ $F32$ $F32$ $F10$
WideCNN	$C64$ $C64$ $C128$ $C128$ $C256$ $C256$ $F32$ $F32$ $F10$
DeepCNN	$C32$ $C32$ $C64$ $C64$ $C128$ $C128$ $C128$ $C256$ $C256$ $C256$ $F32$ $F32$ $F10$
BaseCNN Ensemble	Ensemble of 3 BaseCNNs
All Ensemble	Ensemble of BaseCNN, WideCNN and DeepCNN
SENet	SENet [3] on BaseCNN and DeepCNN
Cr-Stitch2	Cross-stitch network [28] with 2 parallel BaseCNNs
BaseCNN-X	BaseCNN - X paths with adaptive cross-connections
ResNet-X	ResNet [3] - X paths with adaptive cross-connections

TABLE II: Ablation study in CIFAR10 - Classification errors (%). BaseCNN-X surpasses single path deeper and wider networks and even ensembles of them. They also show superior performance to both SENets and Cross-stitch networks.

Network	Params (M)	Error %
BaseCNN/WideCNN/DeepCNN	0.55/1.67/2.0	9.26/8.96/8.49
BaseCNN Ensemble	1.66	7.87
All Ensemble	4.27	6.9
SEBaseCNN/SEDeepCNN	0.58, 2.06	8.99,8.15
Cr-Stitch2	1.11	7.89
VGG16 [2]	14.9	6.98
BaseCNN-2	1.11	7.03
BaseCNN-3	1.67	**6.51**
BaseCNN-4	2.22	**6.55**

A. Ablation Study in CIFAR10

For the ablation study, the baselines are chosen as follows. BaseCNN is a standard convolutional neural network with 6 convolutional layers followed by 3 dense layers. DeepCNN is deeper with 10 convolutional layers. The total number of parameters in DeepCNN is still more than 3 times the parameters in BaseCNN. The WideCNN has a similar structure to BaseCNN but in each convolutional layer, the number of filters is doubled. We also evaluate the performance of an ensemble of 3 BaseCNNs, and an ensemble of a BaseCNN, DeepCNN and a WideCNN.

We design our multi-path CNNs such that a single path is analogous to the BaseCNN. BaseCNN-X stands for a BaseCNN like a multi-path net with X number of parallel paths. The feature-dependant cross-connections are first used to expand the input image to parallel RGB maps. More cross-connections are added after 2nd, 4th, 6th convolutional layers and after the 2nd dense layer. Finally, we average the final parallel dense layer predictions. In addition, we build SENets [8] on top of BaseCNN and DeepCNN with the SE operation added in each convolution. We also build a Cross-Stitch Network which comprises two BaseCNNs with non-adaptive stitching operations replacing the adaptive cross-connections. Table I briefly explains the compared structures.

In the ablation study, all the models are trained for 200

TABLE III: Classification error (%) comparison with SOTA methods. ResNet20-3 outperforms ResNet110. ResNet20-3/4 and ResNet32-3/4 show on-par or superior performance to existing adaptive architectures which are mostly based on ResNet110. BaseCNN-X architectures surpass CNN based adaptive image classifiers which have similar or more number of parameters

Network	Params (M)	CIFAR10	CIFAR100
ResNet20/110/164 [3]	0.27/1.7/1.7+	8.75/6.61/5.93	-/26.88/25.16
WRN-40-2 [4]	2.2	5.33	26.04
SEResNet110 [8]	1.7	5.21	**23.85**
BlockDrop [12]	1.7	6.4	26.3
ConvNet-AIG [11]	1.78	5.76	-
ConvNet-AIG all [11]	1.78	5.14	-
ResNet20-2 Ours	0.55	5.5	27.36
ResNet20-3 Ours	0.82	5.18	25.76
ResNet20-4 Ours	1.1	**4.96**	**24.81**
ResNet32-2 Ours	0.94	5.14	25.96
ResNet32-3 Ours	1.41	**4.96**	**24.51**
ResNet32-4 Ours	1.88	**4.59**	**23.52**
Sabour *et al*. [21]	8.2	10.6	-
Highway [9], [23]	2.3	7.54	-
HyperNetworks [10]	0.15	7.23	-
BaseCNN-2/3/4 Ours	1.11/1.67/2.22	**6.53/6.09/6.26**	-

epochs with a batch size of 128, and the learning rate is decayed by a factor of 10 after 80th and 150th epoch. Table II illustrates the performance of baselines and our multi-path nets. All BaseCNN-X nets comfortably surpass the BaseCNN and even conventional deeper (DeepCNN) or wider (WideCNN) networks. BaseCNN-3 shows superior performance to the ensemble of 3 BaseCNNs and even the ensemble of all nets. This shows the promising nature of our adaptive cross-connections and further validates that the improvement is not merely due to the widened nature of our models.

Our models also surpass SENets we built on top of BaseCNN and DeepCNN showing the superiority of feature dependent cross-connections of multi-path networks over deeper nets with the attentional recalibration of feature maps. In addition, BaseCNN-2 shows superior results to Cross-Stitch2 Net, confirming that adaptive cross-connections are more suitable for a parallel path network to learn from single-input-single-output distributions while adjusting the resource allocation per sample.

B. Comparison with SOTA for Small-Scale Datasets

In this section, we compare our multi-path networks with existing deep, wide and adaptive image classifiers in the literature. We adopt our cross-connections to multi-path ResNets [3] and modify the ResNet20 and ResNet32 variants with multiple paths. The initial convolution and pooling operation is shared which is followed by an adaptive one-to-many connector to expand to multiple parallel paths. We insert cross-connections after each set of residual blocks separated with strides and the final dense predictions are averaged. We report the performance of both ResNet and CNN variants of multi-path architectures in CIFAR10 and CIFAR100 datasets in Table III. All the models are trained for 350 epochs where we used a batch size of 64 for ResNet variants and 128 for CNN variants. The learning rate is decayed by a factor of 10 after 150 and 250 epochs. We use standard data augmentation of pixel shift and random horizontal flip.

Table III shows the performance of our models and existing approaches in CIFAR. Our multi-path ResNets which are based on ResNet20 and ResNet32 show superior performance to conventional ResNets. In particular ResNet20-3 (three paths) is sufficient to surpass the deeper ResNet110's performance in both datasets. All our ResNet variants except ResNet20-2 surpass the Wide ResNet variant (WRN-40-2) [4] which has 40 layers of depth, a width ratio of 2 and more complexity. Also, ResNet20-3/4 and ResNet32-3/4 architectures show on-par or superior performance to the adaptive networks such as SENet [8], ConvNet-AIG [11] and BlockDrop [12] which are based on deep ResNet110. ResNet20-3/4 and ResNet32-3 have significantly less number of parameters (0.82M, 1.1M, 1.41M) compared to ResNet110 and ResNet110 based adaptive networks (1.7M). The table also shows that the performances of CNN based adaptive image classifiers are surpassed by BaseCNN-X architectures.

TABLE IV: Single-crop and 10-crop validation error (%) in ILSVRC2012 dataset. ResNet18-2 with two paths comfortably outperforms ResNet18 and marginally outperforms ResNet34 and Wide ResNet18 with a width factor 1.5.

Network	Params(M)	Single-Crop		10-Crop	
		Top-1	Top-5	Top-1	Top-5
ResNet18 [4] [32]	11.7	30.4	10.93	28.22	9.42
ResNet34 [4] [3]	21.8	26.77	8.77	24.52	7.46
WRN-18-1.5 [4]	25.9	27.06	9.0		
ResNet18-2 (Ours)	23.4	**26.48**	**8.6**	**24.5**	**7.34**

C. ILSVRC2012 Dataset

ILSVRC2012/ImageNet Dataset [7] is a large-scale complex dataset with 1.3M training images and 50k validation images categorized into 1000 classes. To train on this dataset, we modify the ResNet18 variant [3] with two paths (ResNet18-2). The insertion of cross-connections follows a similar setup as described in Sec. IV-B for ResNets. We train our model for 110 epochs with a batch size of 256 in two GPUs. We use an SGD optimizer with a momentum of 0.9 and the learning rate is decayed by a factor of 10 once every 30 epochs. We augment images by re-scaling to 256$\times$ 256 and taking random crops of 224$\times$ 224, which are randomly flipped horizontally.

Table IV shows the top-1 and top-5 errors evaluated for both single-crop and 10-crop testing in ILSVRC2012 dataset. ResNet18-2 comfortably surpasses ResNet18, and shows a slightly superior performance to ResNet34 which shares a similar amount of parameters to our design. Also, ResNet18-2 marginally surpasses the Wide ResNet18 [4] with a width ratio

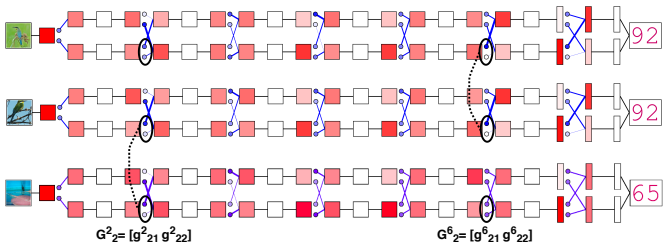


Fig. 3: Information flow of the input images in Figure 1. The relative activation strengths of the input and the output tensors of each cross-connection layer are indicated by the saturation of red color. The contrast of blue circles and the thickness of the lines indicate gate strengths between paths. The gating vector G_2^2 at an initial stage of the network, shows similar routing patterns for Image 1b and 1c, whereas the gating vector G_2^6 at a latter stage, shows almost exactly the same routing for Image 1a and 1b (Best viewed on screen).

of 1.5 (WRN-18-1.5). Due to the quadratic increment in the number of parameters with width enhancement, WRN-18-1.5 has more number of parameters than ResNet18-2.

V. Visualizing and Understanding

We use several visualization techniques to empirically validate the nature of the feature-dependency of our approach. Here, we train a two-path network based on VGG13 [2], namely VGG13-2, on a subset of a subset of ImageNet which consists of the first 100 classes. This subset contains 130k training images and 5000 validation images. Each path of the model is similar in structure to VGG13, but the number of convolutional filters in each layer is halved and the number of nodes in the dense layers is reduced to 256. The proposed cross-connections are added similarly to BaseCNN-2.

A. Information Flow through Cross-Connections

We show the adaptiveness of information flow through our model by indicating the relative activation strengths of the parallel input and output tensors of each cross-connection, and the corresponding gating strengths. To approximate the relative activation strength of each tensor, i.e., each path, we normalize its average activation w.r.t that of all tensors in a layer. The normalized activations are then mapped into red color intensities of the boxes representing the tensors in a cross-connection layer. We represent the coupling coefficients, i.e., gate values, as blue circles connected with straight lines. The contrast of circles and the thickness of lines are proportional to the respective gate values. We only use this visualisation scheme on the layers with cross-connections.

Figure 3 shows the route visualizations of VGG13-2 for the three input images in Figure 1, which verifies the nature of the feature-dependency of our model. In particular, pay attention to the two gate vectors G_2^2 ($[g_{21}^2\ g_{22}^2]$) and G_2^6 ($[g_{21}^6\ g_{22}^6]$) which correspond to either initial or latter stages of the network respectively. Here, G_i^l denotes the gate vector computed on input i in cross-connection layer l. Image 1a and 1b corresponding to the same class trigger different gating patterns in G_2^2, whereas Image 1b and 1c corresponding to different classes, show interestingly similar patterns (g_{21}^2 getting maximized). However, this behaviour is flipped in G_2^6, in which, Image 1a and Image 1b leads to equivalent gating patterns (g_{21}^6 getting maximized) which is distinguishably different from the Image 1c.

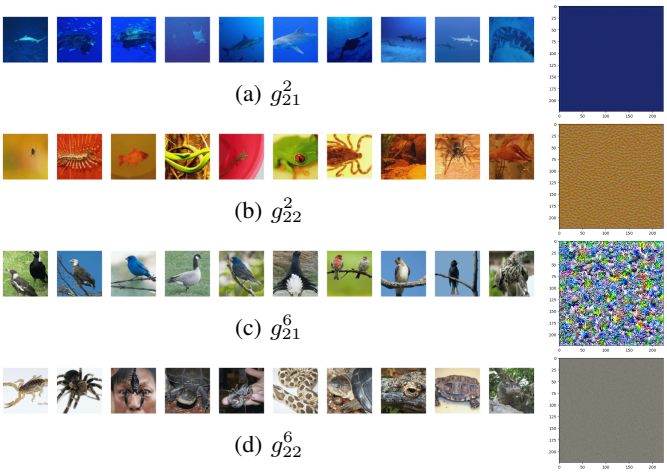
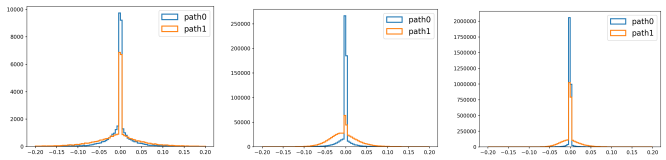


Fig. 4: Each row shows the top 10 images with the highest activations for a particular gate, followed by a synthesized image that maximizes the gate neuron. The gates in initial layers such as g_{21}^2 and g_{22}^2 get triggered by low-level patterns, whereas the gates in deeper layers such as g_{21}^6 and g_{21}^6 get triggered by more abstract patterns. In fact, here, g_{21}^2 and g_{22}^2 are triggered by colors blue or brown-green respectively, whereas g_{21}^6 and g_{22}^6 are triggered for bird poses/patterns or patchy/scaly appearances (Best viewed on screen).

B. Maximizing Gate Activations

To further understand the described behaviour in Section V-A, we present selected images from the validation set, which maximize each gate value among the pairs $[g_{21}^2,\ g_{22}^2]$ and $[g_{21}^6,\ g_{22}^6]$. If we consider each pair, maximizing one gate value causes the other gate value to be minimized due to the softmax activation. In other words, these are the contextual representations which trigger the two extremes of each gating vector. To further show the context which gives the maximum trigger at a particular gate, we freeze the trained network and synthesize an input which maximizes the corresponding gate neuron (before softmax). We add an L2 regularization to produce smooth images. This process is similar to class-specific image generation [33].

Figure 4 shows the results of this study, in which, each row shows the top 10 images corresponding to each gate, followed by the synthesized input. Since G_2^2 corresponds to initial stages of the network, its components g_{21}^2 and g_{22}^2 get triggered by low-level, i.e., less-abstract features. For instance, the synthesized image that gives the maximum trigger at g_{21}^2 can be identified as a uniformly distributed blue color



(a) layer4 (b) layer8 (c) layer11

Fig. 5: Weight histograms of parallel paths at selected layers. By grouping homogeneous feature maps to parallel paths, the set of filters in each path has become distinct from each other.

image whereas, at g_{22}^{2} it is a brown-green image, both with minimal pattern complexity. In contrast, the gating vector G_{2}^{6} corresponds to latter stages of the network, and therefore, the corresponding gates g_{21}^{6} and g_{22}^{6} get triggered for more abstract features. If we consider g_{21}^{6}, all 10 images are associated with bird poses. The synthesized image further shows complex bird-related possible patterns and colors. In contrast, among the images for g_{22}^{6}, we can see a high correlation with animals with a patchy or scaly appearance. However, the synthesized image here, shows a complex pattern that is hard to be described by the human eye.

Based on these visualizations, we can describe the behaviour of gating patterns presented in Section V-A. The blue backgrounds of Image 1b and 1c have been prominent causing them to maximally trigger g_{21}^{2}. However in latter stages of the network, g_{21}^{6} is triggered by Image 1a and 1b as they both correspond to birds with similar poses. This verifies that the context information which affects the gating mechanism has changed to more abstract features similar to the concept of a class. This further emphasizes that the image context which matters in the problem domain is distributed among multiple layers of the network, scaling from low-level to high-level depending on the depth. Therefore, having cross-connections at multiple depths enables gating, i.e., path allocation, based on the level of context abstraction represented at the corresponding depth.

C. Weight Histograms of Parallel Paths

One of our motivations for deploying a multi-path architecture is to group homogeneous feature maps into parallel paths, so that, each set of filters corresponding to a path can be dedicated to its context. This implies that the set of filters in each parallel path should learn distinct features, reducing the redundancy. To validate this claim, we plot weight histograms of parallel paths at selected layers. Figure 5 shows these weight histograms of parallel paths at feed-forward layers 4, 8 and 11. In these cases, we can see that the weight histograms of parallel paths have become distributed.

VI. Conclusion

In this paper, we explored the utility of multi-path neural networks as dynamically adaptive models to learn from complex datasets. In particular, we selectively introduced adaptive cross-connections between successive pairs of layers in a multi-path network, to cluster similar feature maps into parallel paths and learn a soft routing between them. The proposed cross-connections are weighted by a set of non-linear parametric coefficients produced based on incoming sets of feature maps to make the path selection process feature-dependent. Simply put, in the forward pass, an input image to this multipath network gets adaptively allocated among parallel paths in each layer, based on the nature of the feature maps in the corresponding layer. The experiments conducted on both small and large-scale image classification datasets show that such multi-path networks are capable of surpassing state-of-the-art adaptive image classifiers and conventional single path networks of increased width or depth which are of similar or even higher complexity. We further validate the nature of the feature dependency of our model and its ability to capture and adapt to the context of an input which is distributed among multiple layers along with the depth.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[4] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *British Machine Vision Conf.*, 2016, pp. 87.1–87.12.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.

[6] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1492–1500.

[7] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[8] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, June 2018, pp. 7132–7141.

[9] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," *arXiv preprint arXiv:1505.00387*, 2015.

[10] D. Ha, A. Dai, and Q. V. Le, "Hypernetworks," in *Proc. Int. Conf. Learn. Representations*, 2017.

[11] A. Veit and S. Belongie, "Convolutional networks with adaptive inference graphs," in *Eur. Conf. Comput. Vis.*, 2018, pp. 3–18.

[12] Z. Wu, T. Nagarajan, A. Kumar, S. Rennie, L. S. Davis, K. Grauman, and R. Feris, "Blockdrop: Dynamic inference paths in residual networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8817–8826.

[13] K. Kahatapitiya, D. Tissera, and R. Rodrigo, "Context-aware automatic occlusion removal," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 1895–1899.

[14] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Artificial intelligence and statistics*, 2015, pp. 562–570.

[15] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," in *Proc. Int. Conf. Learn. Representations*, 2015.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European conference on computer vision*. Springer, 2016, pp. 630–645.

[17] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, p. 533, 1986.

[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[19] J. Hu, L. Shen, S. Albanie, G. Sun, and A. Vedaldi, "Gather-excite: Exploiting feature context in convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 9401–9411.
[20] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "Eca-net: Efficient channel attention for deep convolutional neural networks," *arXiv preprint arXiv:1910.03151*, 2019.
[21] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Adv. in Neural Inf. Process. Syst.*, 2017, pp. 3856–3866.
[22] G. E. Hinton, S. Sabour, and N. Frosst, "Matrix capsules with EM routing," in *Proc. Int. Conf. Learn. Representations*, 2018.
[23] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Advances in neural information processing systems*, 2015, pp. 2377–2385.
[24] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-first AAAI conference on artificial intelligence*, 2017.
[25] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3642–3649.
[26] M. Wang, "Multi-path convolutional neural networks for complex image classification," *arXiv preprint arXiv:1506.04701*, 2015.
[27] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.
[28] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, June 2016, pp. 3994–4003.
[29] S. Ruder, J. Bingel, I. Augenstein, and A. Søgaard, "Latent multi-task architecture learning," in *Proc. of AAAI 2019*, February 2019, pp. 4822–4829.
[30] Y. Gao, J. Ma, M. Zhao, W. Liu, and A. L. Yuille, "Nddr-cnn: Layerwise feature fusing in multi-task cnns by neural discriminative dimensionality reduction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3205–3214.
[31] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.
[32] Facebook, "fb.resnet.torch." [Online]. Available: https://github.com/facebookarchive/fb.resnet.torch
[33] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.