SUSTech Southern University of Science and Technology

ML4CV GROUP

# Group Discussion

November 30, 2020

# Compounding the Performance Improvements of Assembled Techniques in a Convolutional Neural Network

Jungkyu Lee, Taeryun Won, Kiho Hong

Clova Vision, NAVER Corp.

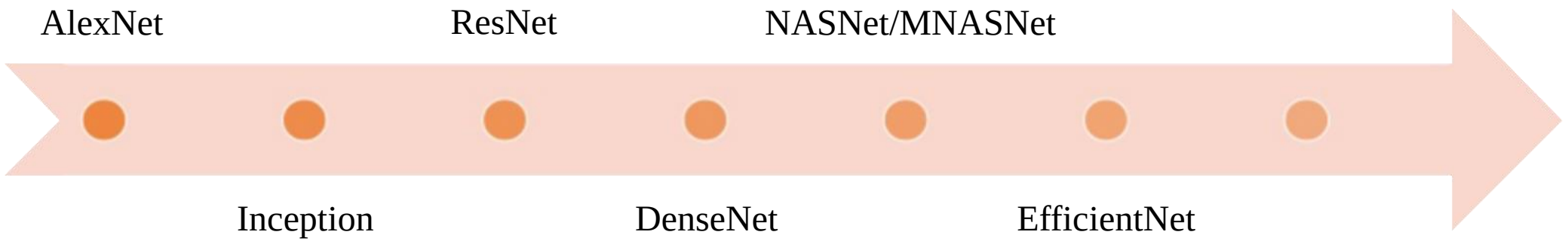
{jungkyu.lee, lory.tail, kiho.hong}@navercorp.com

# 1. Introduction

Lu Dong

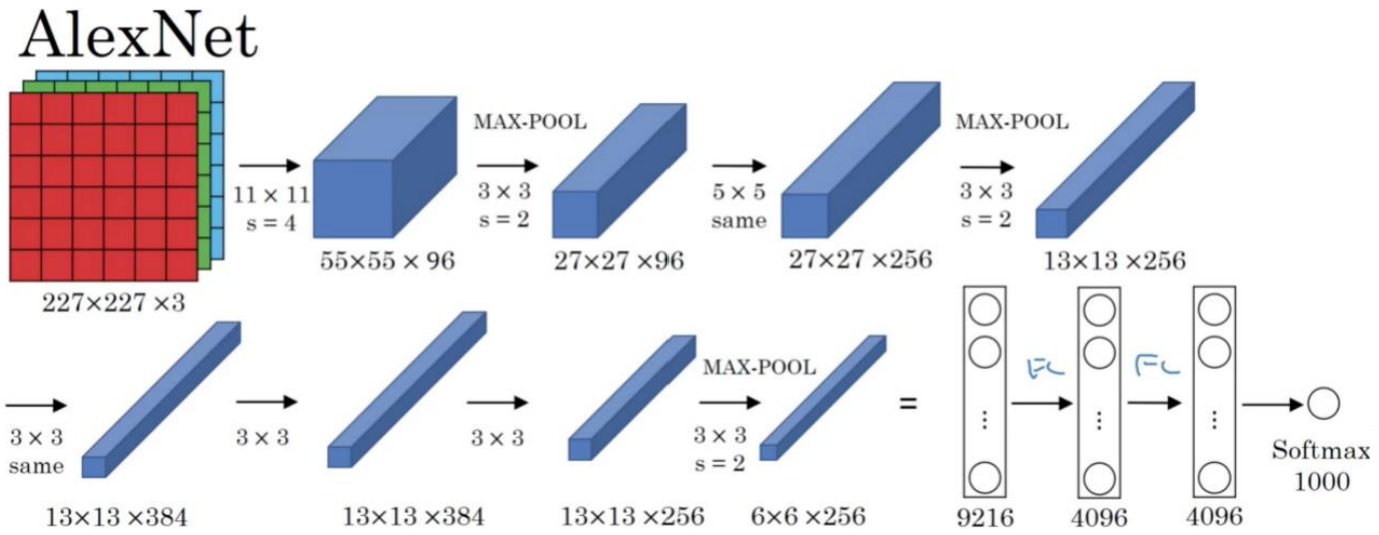
# *Introduction: background*

- **Target**: improve the performance of CNNs in image classification
- **One approach**: design new network architectures

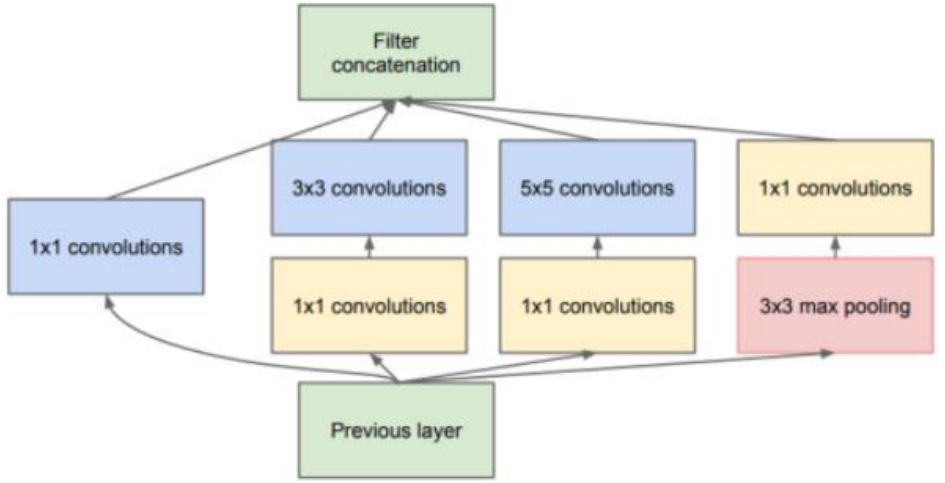



- **Another approach**: “Tricks”

# Introduction: AlexNet Inception




First deep CNN architecture

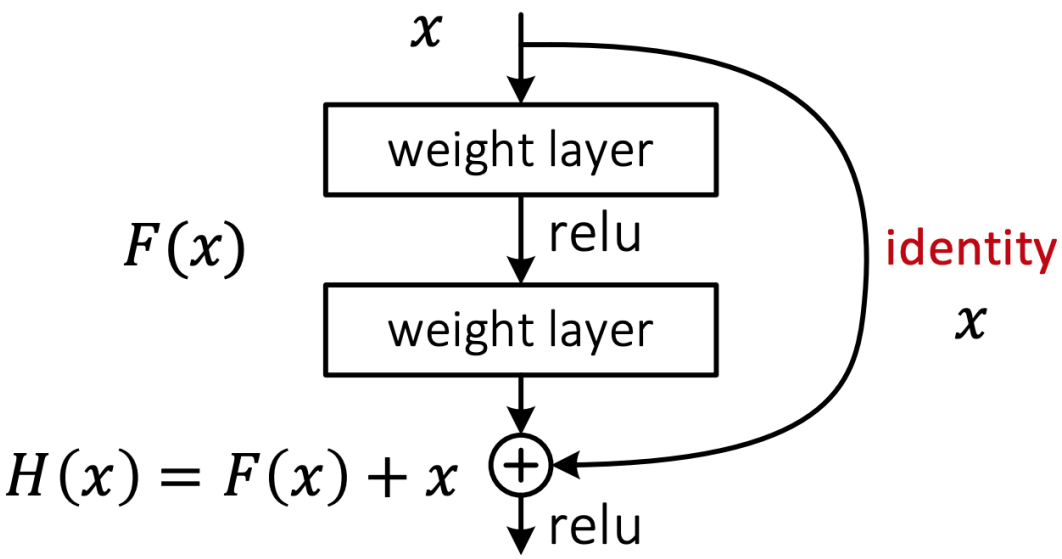



Inception Module

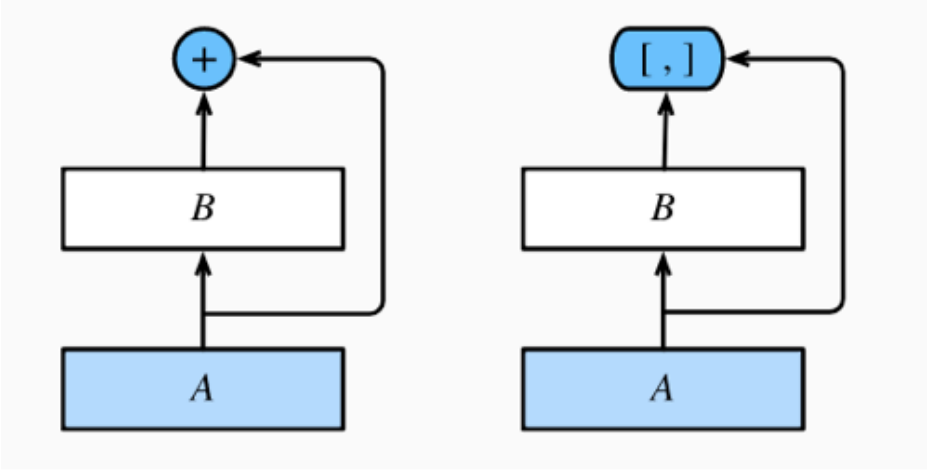
Convolution layers of different kernel sizes

# *Introduction: ResNet DenseNet*

- Residual net



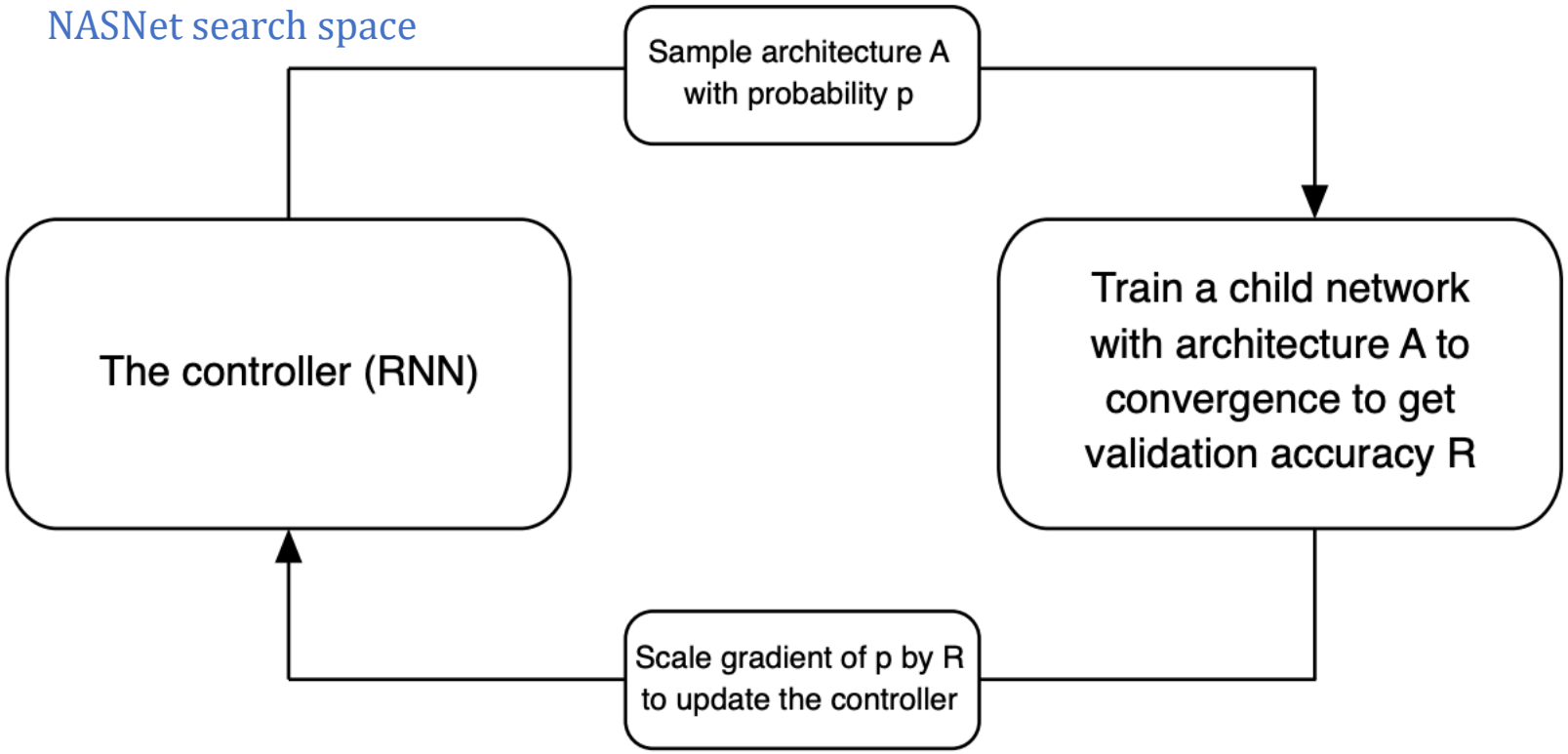

skip connection




ResNet(left) DenseNet(right)

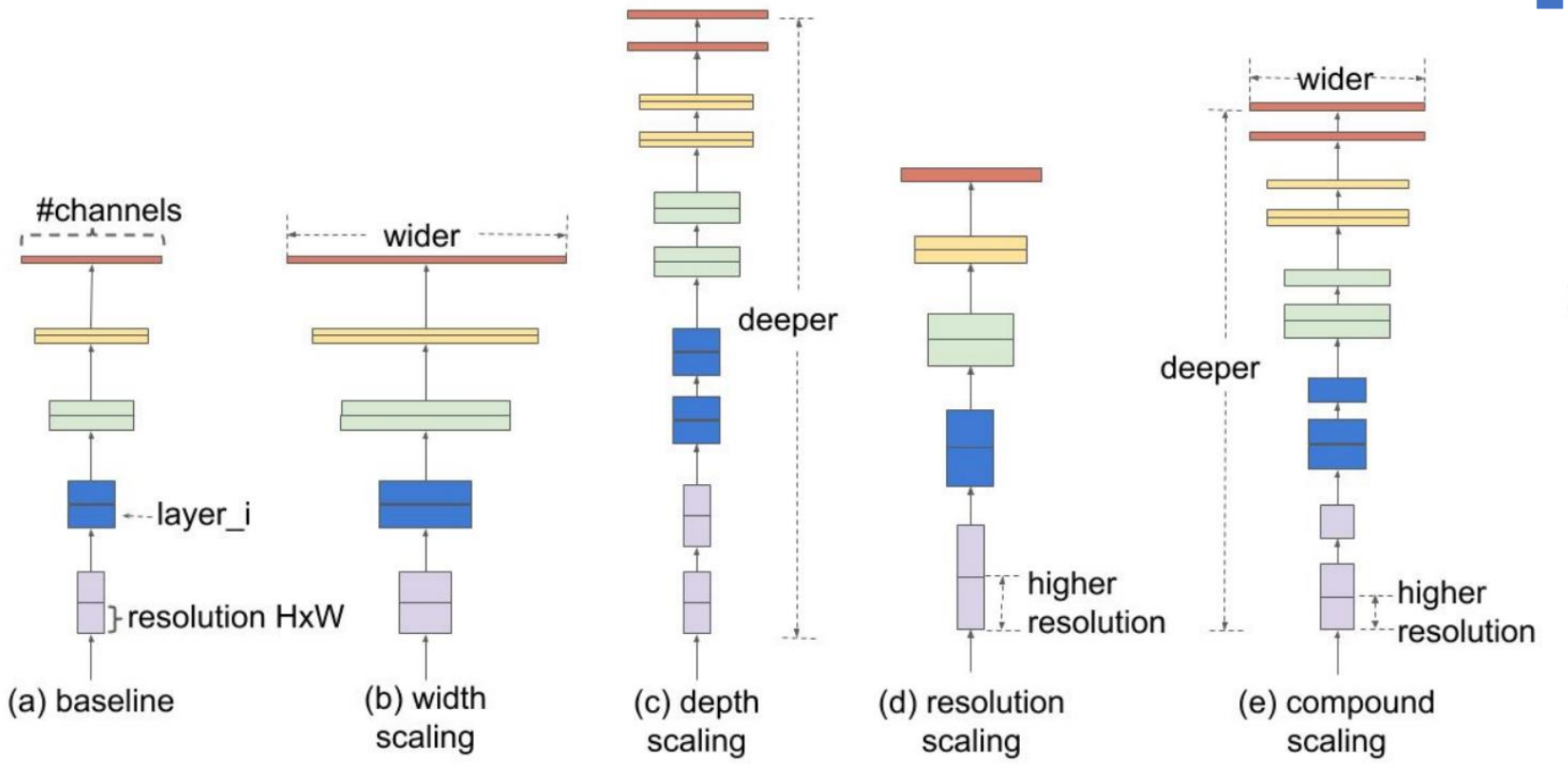
dense feature connections

# *Introduction: NASNet/MNASNet*




Network design was automatically decided to create models

# Introduction: EfficientNet




- Compound Scaling

$$\text{depth: } d = \alpha^\phi$$
$$\text{width: } w = \beta^\phi$$
$$\text{resolution: } r = \gamma^\phi$$
$$\text{s.t. } \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$$
$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

**Depth**: number of layers

**Width**: number of channels

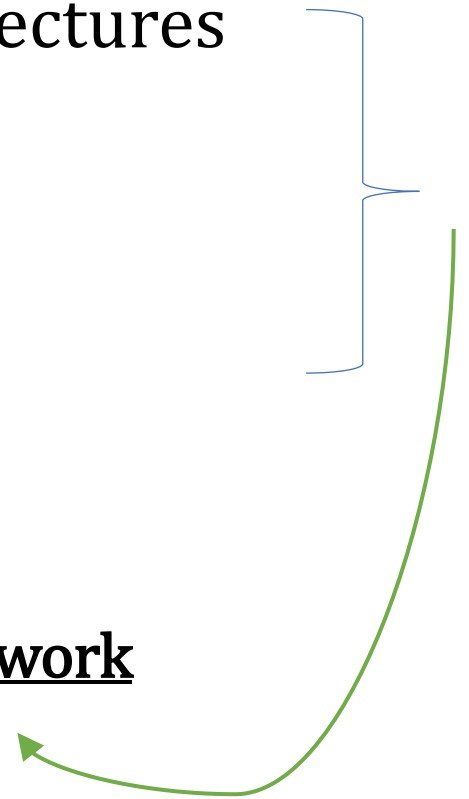
**Resolution**: input size

An efficient network by balancing the resolution, height, and width of the network

# *Introduction: our approach*

- One approach: design new network architectures
- **Another approaches**: “Tricks”
  - Data preprocessing
  - Learning rate decay
  - Parameter initialization

- **Our approach:**
  - Assembling these techniques into **a single network**



# *Introduction: contribution summary*

- Organize the existing CNN-related techniques:
  - Network tweaks → SENet
    - Modify the CNN architectures to be more efficient
  - Regularization:
    - AutoAugment
    - Mixup
    - Dropout/DropBlock
    - Use data augmentation to prevent overfitting

# *Introduction: contribution summary*

- Provide detailed experimental results:

| Model | Top-1 | mCE | mFR | Throughput |
|---|---|---|---|---|
| EfficientNet B4 [32] + AutoAugment [4] | 83.0 | 60.7 | - | 95 |
| EfficientNet B6 [32] + AutoAugment [4] | 84.2 | 60.6 | - | 28 |
| EfficientNet B7 [32] + AutoAugment [4] | 84.5 | 59.4 | - | 16 |
| ResNet-50 [8] (baseline) | 76.3 | 76.0 | 57.7 | 536 |
| Assemble-ResNet-50 (ours) | 82.8 | 48.9 | 32.3 | 312 |
| Assemble-ResNet-152 (ours) | 84.2 | 43.3 | 29.3 | 143 |

- Performance indicators:
  - Top-1 accuracy
  - mCE (mean corruption error)
  - mFR (mean flip rate)
  - Throughput

# 2. Preliminaries

Zhiqiang Wang

# Preliminaries——Training Procedure

Model: ResNet
Dataset: ImageNet ILSVRC-2012 dataset (1.3M training images and 1,000 classes)
Machine: a single machine with 8 Nvidia Tesla P40 GPUs; CUDA 10 platform and cuDNN 7.6

- **Preprocessing**

  **Traininig**: aspect ratio from 3/4 to 4/3
  area size from 5% to 100%
  resized as 224 * 224
  flipped horizontally prob: 0.5

  **Validation:** hold the aspect ratio
  resize the shorter size to 256
  center crop the image to 224*224

- **Hyperparameter**

  Batch sizes: 1024 Initial learning rate: 0.4
  Weight decay: 0.0001. Default epochs: 120
  Optimizer: SGD with momentum 0.9

- **Learning rate warmup**

  Linearly increase the learning rate from 0 to the initial learning rate at warm-up periods set to first 5 epochs.

- **Zero γ**

  Initialize γ = 0 for all batch-norm layers that sit at the end of a residual block. It is easier to train by creating an effect that shrinks the entire layer at initial the stage.???

- **Mixed-precision floating point**

  Only use mixed-precision floating point in the training phase because mixed-precision accelerates the overall training speed if the GPU supports it.
  Only acceleration, no improvement in accuracy.

- **Cosine learning rate decay**

  The cosine decay starts at a low rate from the begining of training, and then drops to a large rate in the middle and again at a small rate in the end.

# Preliminaries——Evaluation Metrics

Three metrics used to measure the performance of the model

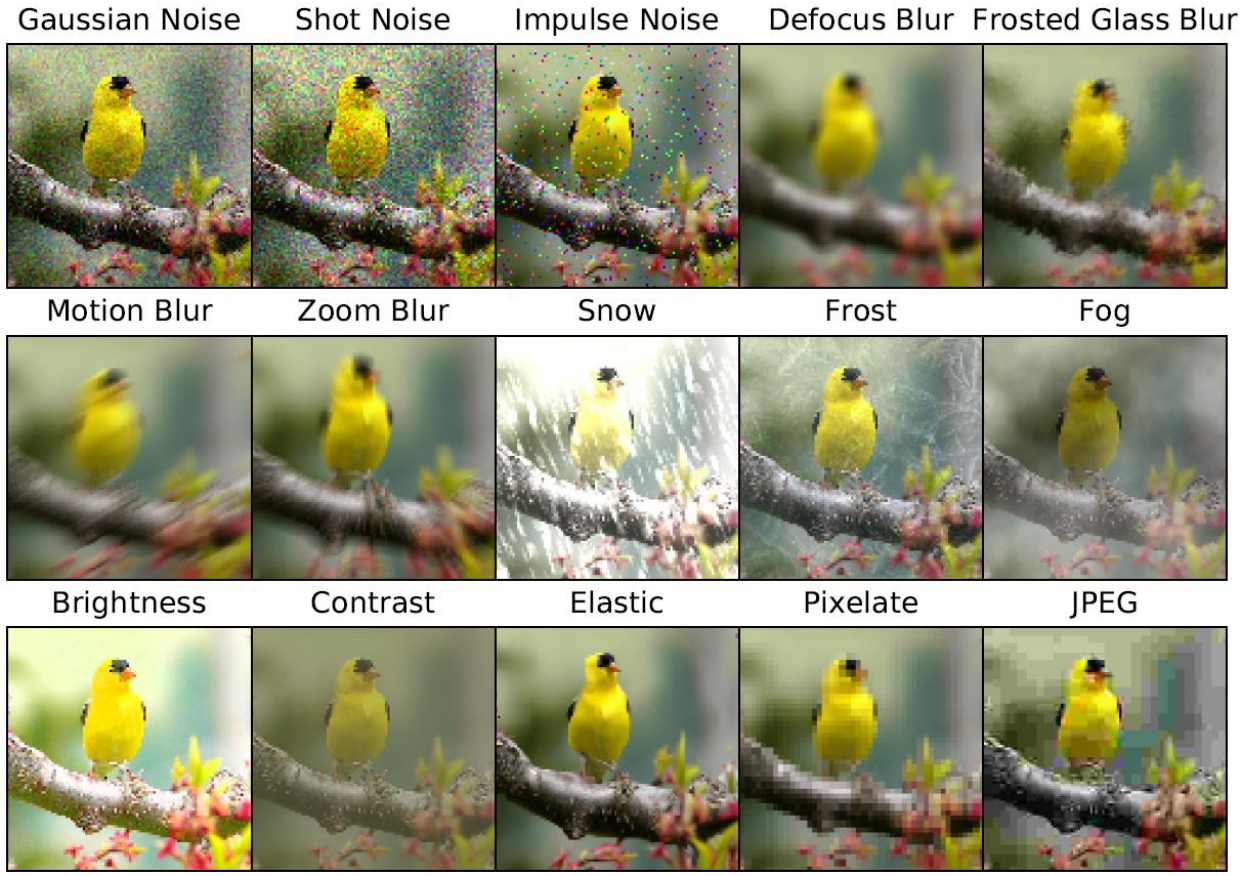
- **Top-1** classification accuracy on ImageNet ILSVRC-2012 validation dataset.
- **Throughput** defined as how many images are processed per second on the GPU device. We measured inference throughput for an Nvidia P40 1 GPU.
- **Mean Corruption Error (mCE)** measure the performance of classification model on corrupted images proposed by Hendrycks et al. [1]

[1] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. arXiv preprint arXiv:1903.12261, 2019.

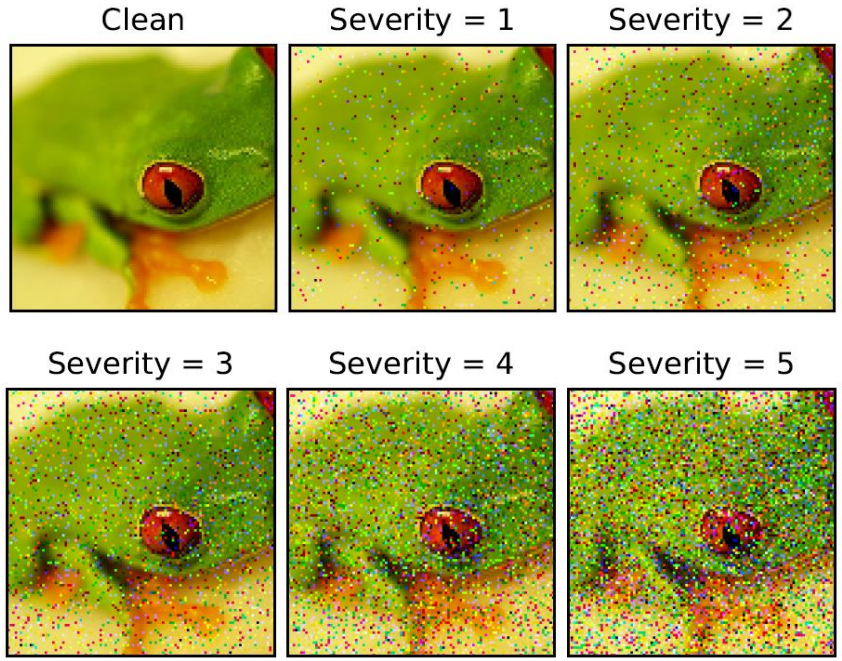
# ImageNet-C Benchmark

**ImageNet-C**

generated by applying a set of 75 common visual corruptions to validation images of ImageNet.




15 diverse corruotion types
5 different severity levels



# 3.1 Network Tweaks

Speaker: Zhu Liu

# ResNet-D

| layer name | output size | 50-layer |
|---|---|---|
| conv1 | 112×112 | 7×7, 64, stride 2 |
| conv2_x | 56×56 | 3×3 max pool, stride 2<br>[1×1, 64<br>3×3, 64<br>1×1, 256] ×3 |
| conv3_x | 28×28 | [1×1, 128<br>3×3, 128<br>1×1, 512] ×4 |
| conv4_x | 14×14 | [1×1, 256<br>3×3, 256<br>1×1, 1024] ×6 |
| conv5_x | 7×7 | [1×1, 512<br>3×3, 512<br>1×1, 2048] ×3 |
| | 1×1 | erage pool, 1000-d fc, soft |
| FLOPs | | $3.8\times10^9$ |

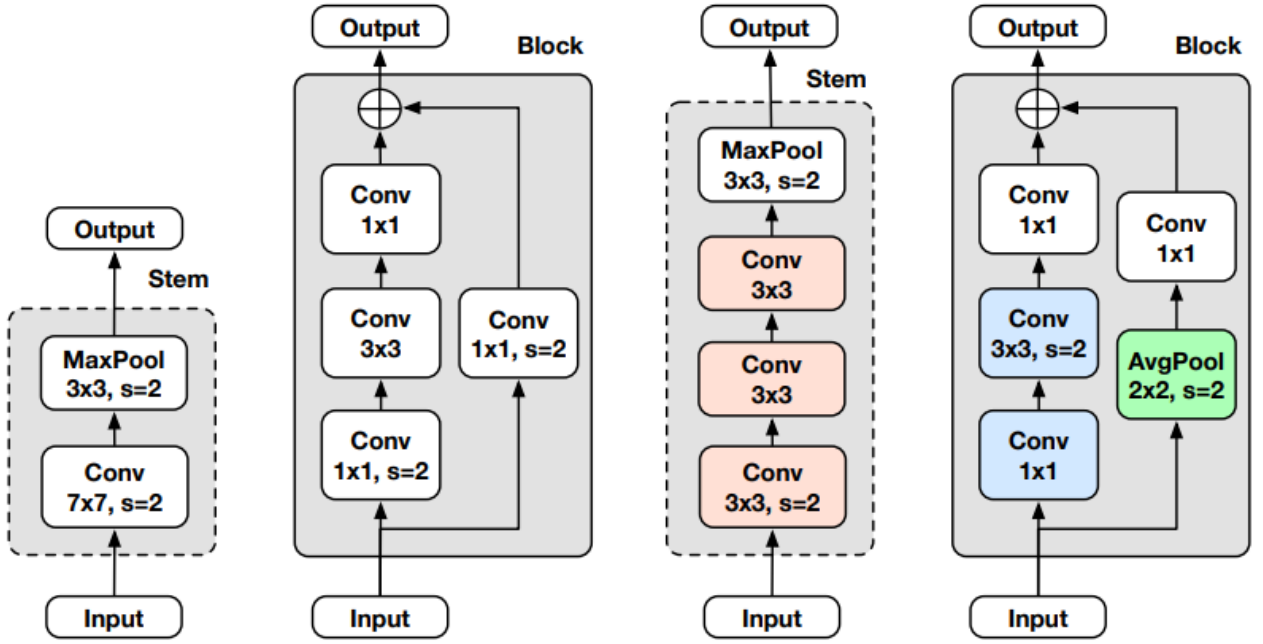



Figure 2. Comparison of changes between ResNet and ResNet-D.

Blue: The stride sizes of the first two convolutions have been switched.
Green: A 2*2 average pooling layer has been added with a stride of 2 before the convolution.
Red: A large 7*7 convolution has been replaced with three smaller 3*3 convolutions in Stem layer.

# Channel Attention (SE, SK)

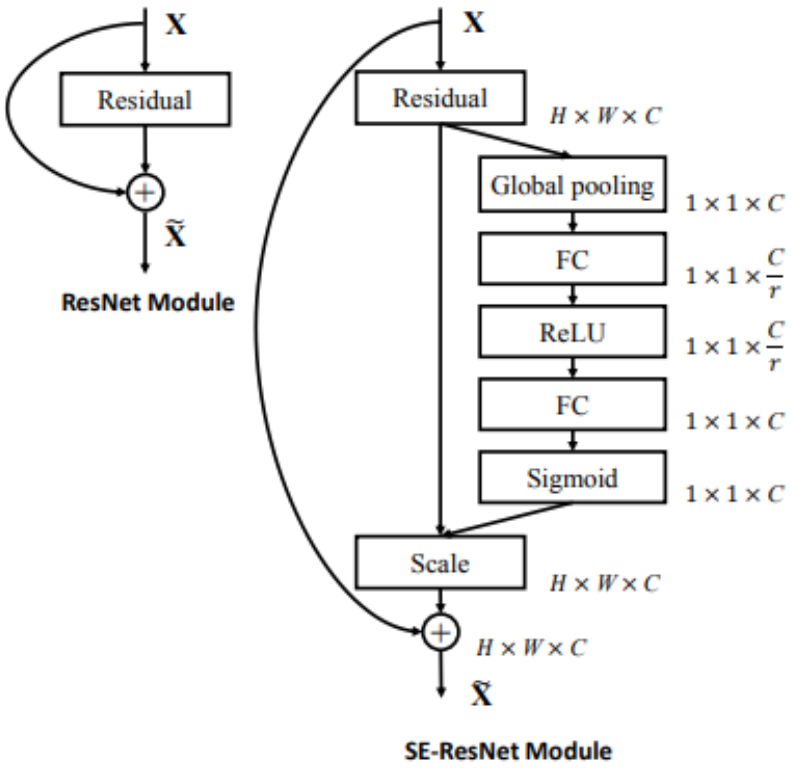



Fig. 3. The schema of the original Residual module (left) and the SE-ResNet module (right).

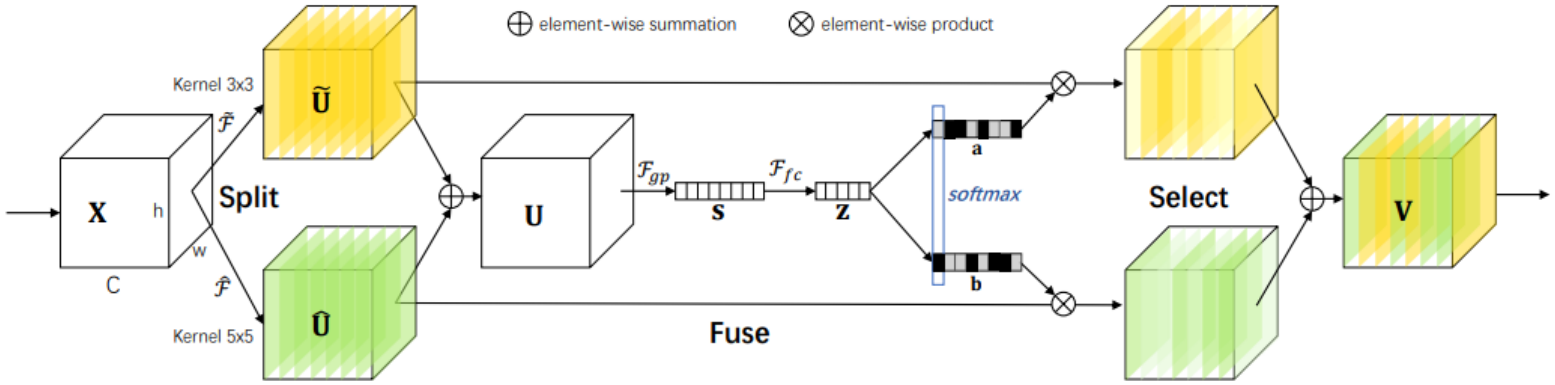



Figure 1. Selective Kernel Convolution.

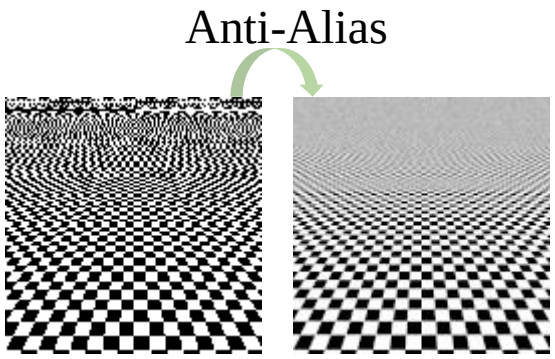
“ SK is used inspired by the fact that the receptive sizes of neurons in the human visual cortex are different from each other.”

# Channel Attention (SE, SK)

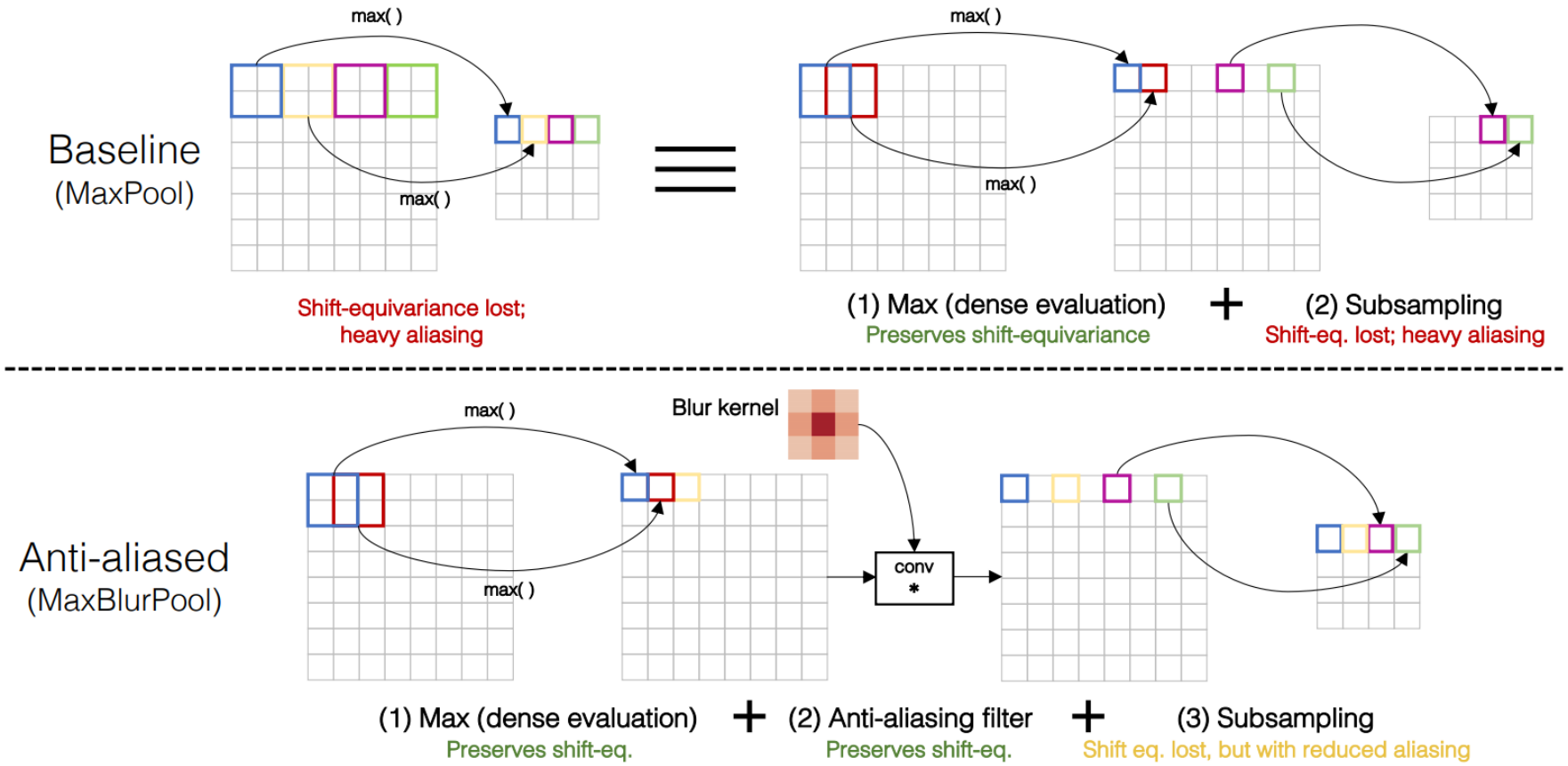
| Exp No. | Model | SK Configuration | SK $r$ | top-1 | throughput |
|---|---|---|---|---|---|
| C0 | R50 (baseline) | - | - | 76.30 | 536 |
| C1 | R50+SE | - | - | 77.40 | 466 |
| C2 | R50+SK | 3x3 + 5x5 | 2 | 78.00 | 326 |
| C3 | R50+SK† | 3x3, 2x-channel | 2 | 77.92 | 382 |
| C4 | R50+SK | 3x3, 2x-channel | 16 | 77.57 | 402 |
| C5 | R50+SK+SE | 3x3, 2x-channel | 2 | 77.50 | 345 |

Table 2. Result of channel attention with different configurations. R50 is a simple notation for ResNet-50. In these experiments, the learning rate starts from 0.4 and is decayed by 0.1 at 30, 60 and 90 epochs and is trained for 120 epochs. $r$ is the reduction ratio of SK in the *Fuse* operation.

# Anti-Alias Downsampling (AA)




- To improve the shift-equivariance of deep networks.
- AA can be applied to max-pooling, projection-conv, and strided-conv of ResNet.

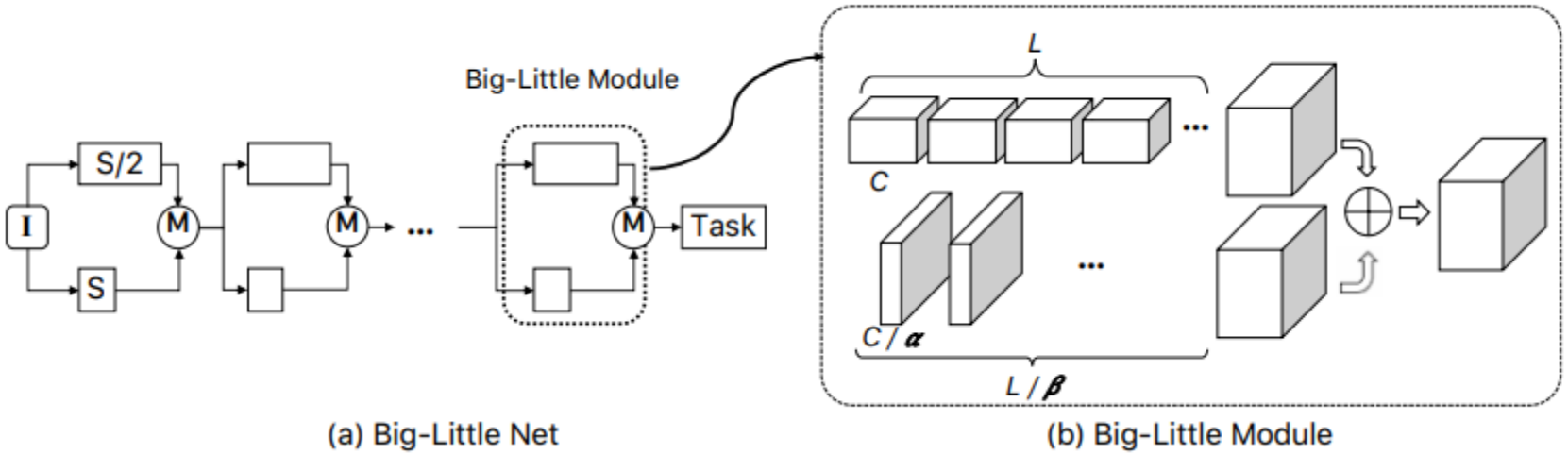


# Anti-Alias Downsampling (AA)

| Exp No. | Filter Size | Max Pooling | Pro-jection | Strided Conv | top-1 | throughput |
|---|---|---|---|---|---|---|
| A0 | - | X | X | X | 76.30 | 536 |
| A1 | 5 | O | O | O | 76.81 | 422 |
| A2 | 3 | O | O | O | 76.83 | 456 |
| A3 | 3 | O | X | O | 76.84 | 483 |
| A4 | 3 | X | X | O | 76.67 | 519 |

Table 3. Results for downsampling with anti-aliasing. The performance of the model was tested according to a range of applications for downsampling with anti-aliasing. All experiments were tested based on ResNet-50. We applied downsampling with anti-aliasing only to strided-conv based on the result of the experiment. In these experiments, the learning rate starts from 0.4 and is decayed by 0.1 at 30, 60 and 90 epochs and is trained for 120 epochs.

# Big Little Network (BL)




- Different resolutions while aiming at reducing computational cost and increasing accuracy.

# Overall

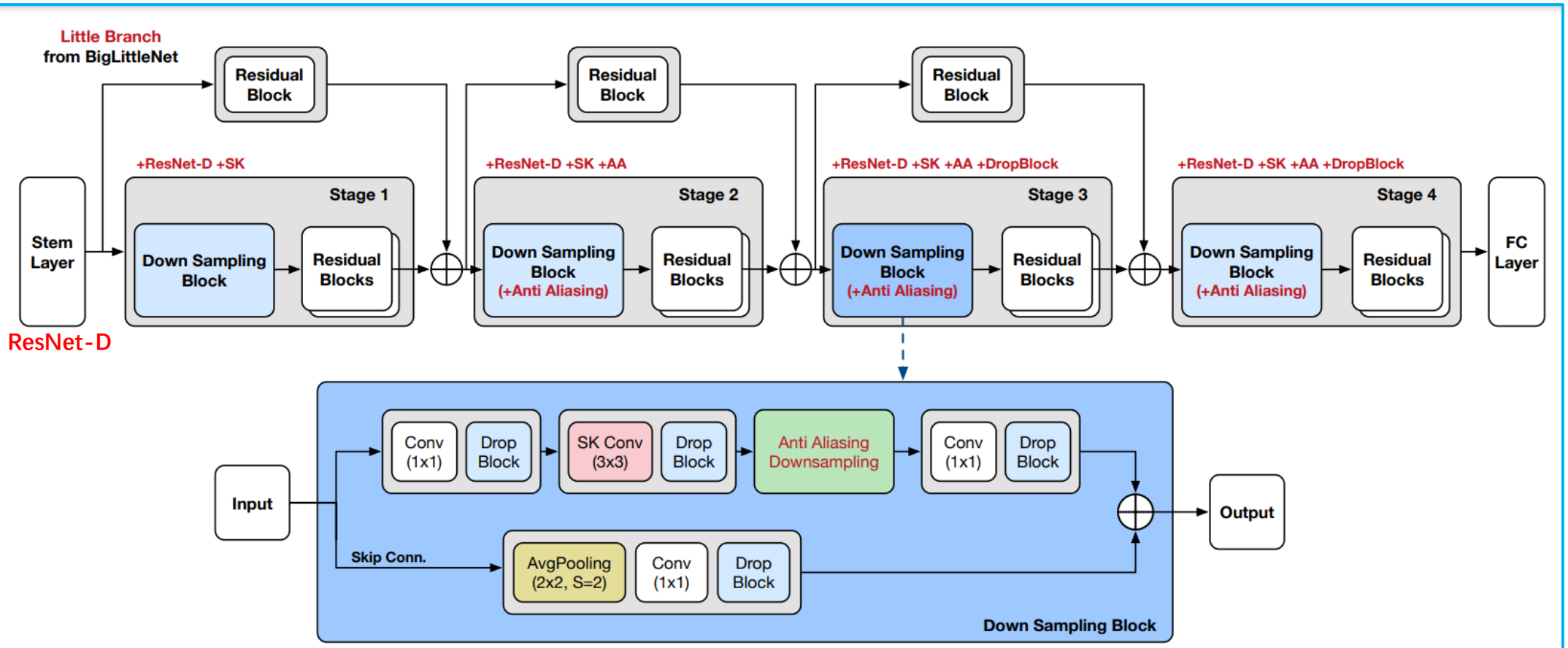



**Figure 1. Assembling techniques into ResNet.** We apply network tweaks such as ResNet-D, SK, Anti-alias, DropBlock, and BigLittleNet to vanilla ResNet. In more detail, ResNet-D and SK are applied to all blocks in all stages. Downsampling with anti-aliasing is only applied to the downsampling block from Stage 2 to Stage 4. DropBlock is applied to all blocks in Stage 3 and Stage 4. Little-Branch from BigLittleNet uses one residual block with smaller width.

# Reference


- Zhang, Richard. "Making convolutional networks shift-invariant again." (ICML 2019). Anti-Alias Downsampling
- Chen, Chun-Fu, et al. "Big-little net: An efficient multi-scale feature representation for visual and speech recognition." *arXiv preprint arXiv:1807.03848* (2018). Big-little net
- He, Tong, et al. "Bag of tricks for image classification with convolutional neural networks." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019. ResNet-D
- Hu, Jie, Li Shen, and Gang Sun. "Squeeze-and-excitation networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018. SE
- Li, Xiang, et al. "Selective kernel networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2019. SK

# 3.2 Regularization

by Qiushi Lin

# AutoAugument

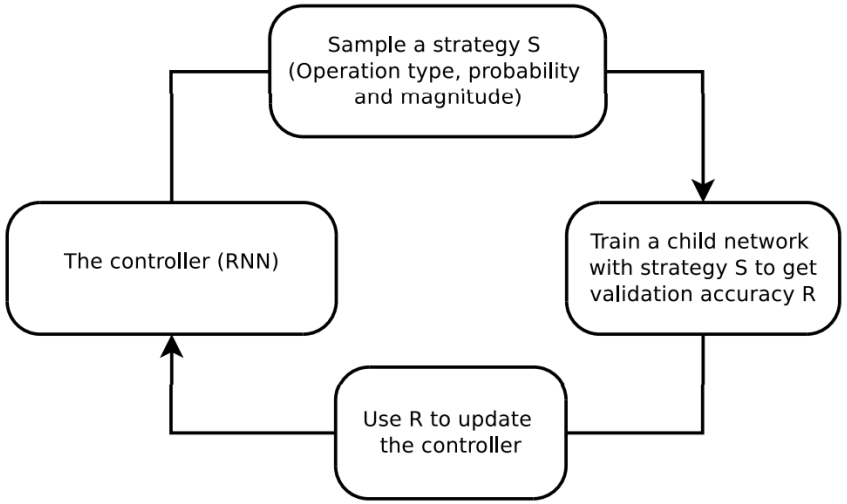



Figure 1. Overview of our framework of using a search method (e.g., Reinforcement Learning) to search for better data augmentation policies. A controller RNN predicts an augmentation policy from the search space. A child network with a fixed architecture is trained to convergence achieving accuracy R. The reward R will be used with the policy gradient method to update the controller so that it can generate better policies over time.

## Search Space

- a policy consists of 5 sub-policies with each sub-policy consisting of two image operations to be applied in sequence
- each operation is also associated with two hyperparameters
    1. the probability of applying the operation
    2. the magnitude of the operation
- operations selected from a popular Python image library, PIL

# Mixup

$$\tilde{x} = \lambda x_i + (1-\lambda)x_j, \qquad \text{where } x_i, x_j \text{ are raw input vectors}$$
$$\tilde{y} = \lambda y_i + (1-\lambda)y_j, \qquad \text{where } y_i, y_j \text{ are one-hot label encodings}$$

Two types of implementation

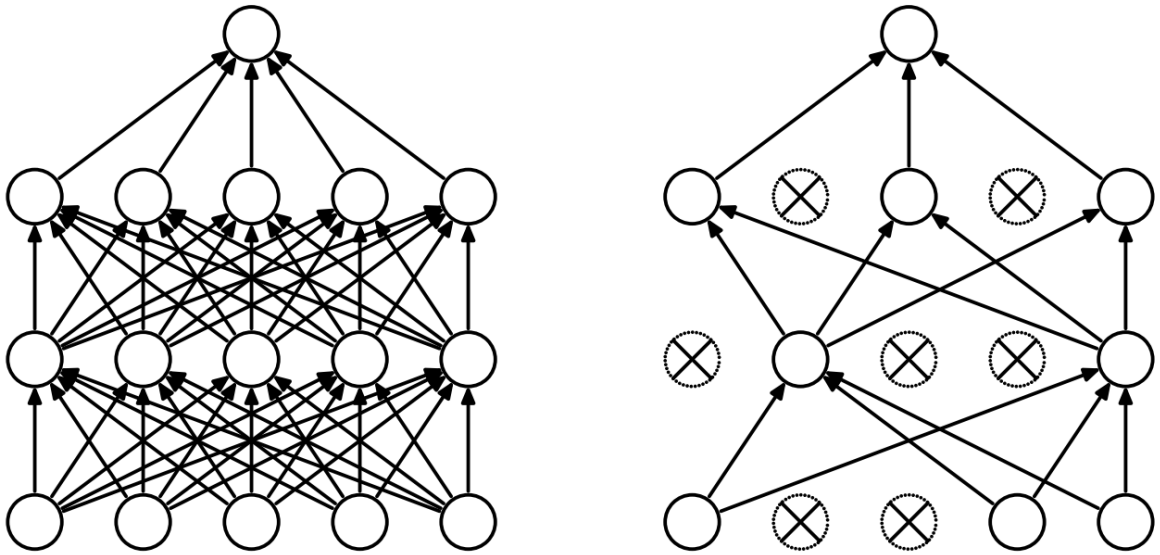
1. use two mini batches to create a mixed mini batch, which is is suggested in the original paper

2. use a single mini batch to create the mixed mini batch by mixing the mini batch with a shuffled clone of itself

| Model | Configuration | top-1 |
|---|---|---|
| R50D (E2) | LS | 77.37 |
| R50D | + Mixup (type=2) | 78.85 |
| R50D (E3) | + Mixup (type=1) | **79.10** |

Table 4. Result of the change of Mixup implementation type. We choose type=1 for the next experiment. E2 and E3 correspond to the experiment numbers given in Table 7.

# DropBlock



(a) Standard Neural Net (b) After applying dropout.

Figure 1: Dropout Neural Net Model. **Left**: A standard neural net with 2 hidden layers. **Right**: An example of a thinned net produced by applying dropout to the network on the left. Crossed units have been dropped.

# Label Smoothing

suppress infinite output and prevent over-fitting

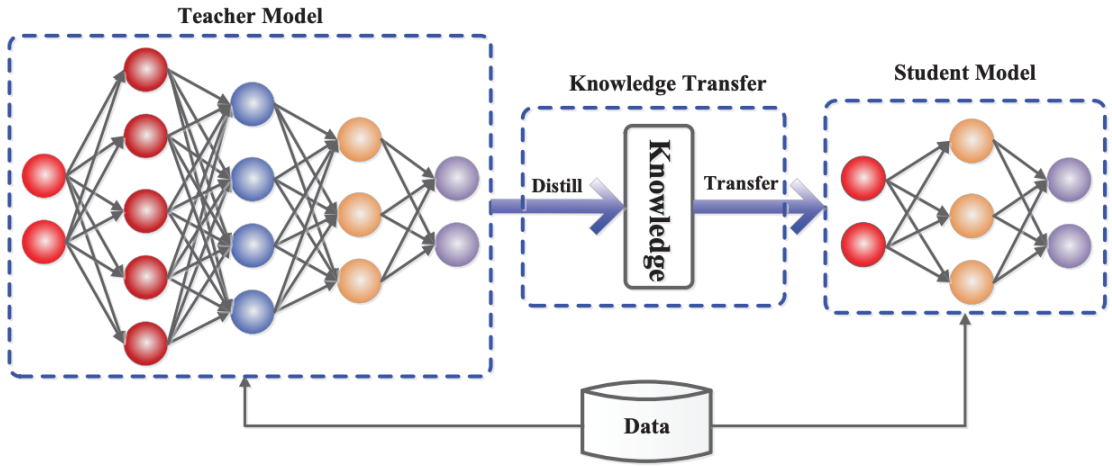
The entropy of this conditional distribution

$$H(p_\theta(\boldsymbol{y}|\boldsymbol{x})) = -\sum_i p_\theta(\boldsymbol{y}_i|\boldsymbol{x}) \log(p_\theta(\boldsymbol{y}_i|\boldsymbol{x})).$$

To penalize confident output distributions, we add the negative entropy to the negative log-likelihood during training

$$\mathcal{L}(\theta) = -\sum \log p_\theta(\boldsymbol{y}|\boldsymbol{x}) - \beta H(p_\theta(\boldsymbol{y}|\boldsymbol{x})),$$

# Knowledge Distillation




| Model | Configuration | top-1 |
|---|---|---|
| R50D+SK (E6) | LS+Mixup+DropBlock | 81.40 |
| R50D+SK | + KD (T=2) | 81.47 |
| R50D+SK | + KD (T=1.5) | 81.50 |
| R50D+SK (E7) | + KD (T=1) | **81.69** |

Table 5. Result of the change of KD temperature. On top of the E6 (described in Table 7), we apply KD by varying the temperature T to find the optimal T value. We choose T=1 for next experiment.

Speciality:

- T = 1, as the Mixup technique has already smoothed the teacher's signal
- use AmoebaNet-A as a teacher with 83.9% of ImageNet validation top-1 accuracy

# 4.1 Experiment result

Ablation study

Gengtiantian

Ablation experiments for assembling the individual network tweaks covered in Section 3.1 to find better model.

| Exp. No. | Model | Input Size | top-1 | throughput |
|---|---|---|---|---|
| M0 | R50 (baseline) | 224 | 76.87 | 536 |
| M1 | R50**D** | 224 | 77.37 | 493 |
| M2 | R50D+**SK** | 224 | 78.83 | 359 |
| M3 | R50D+SK+**BL** | **256** | 79.27 | 359 |
| M4 | R50D+SK+BL+**AA** | 256 | 79.39 | 312 |

Table 6. Performance comparison of stacking network tweaks. By stacking the ResNet-D, Selective Kernel (SK), BigLittleNet (BL) and downsampling with anti-aliasing (AA), we have steadily improved the ResNet-50 model with some inference throughput losses. The focus of each experiment is highlighted in boldface. The cosine learning rate decay is used in all experiments.

**The results show that:**

**1)** ResNet-D and SK can improve performance independently with little effect on each other.

2) To achieve higher accuracy while maintaining throughput similar to that of the R50D+SK, we apply BL using a 256x256 image resolution for inference, whereas we use 224x224 image resolution for training.

**3)** Applying AA just improves top-1 accuracy by 0.12%, because AA is a network structure designed for robustness to image distortion, the top-1 accuracy does not reliably determine the AA effect.

The impact of assembling the techniques described in Section 3.2. We stack the network tweaks and regularizations alternately to balance the performance effects.

| Exp. No. | Model | Regularization Configuration | Train Epoch | Input Size | top-1 | top-1 Δ | mCE | mCE Δ | throughput |
|---|---|---|---|---|---|---|---|---|---|
| | EfficientNet B0 [34] | Autoaug | - | 224 | 77.3 | - | 70.7 | - | 510 |
| | EfficientNet B1 [34] | Autoaug | - | 240 | 79.2 | - | 65.1 | - | 352 |
| | EfficientNet B2 [34] | Autoaug | - | 260 | 80.3 | - | 64.1 | - | 279 |
| | EfficientNet B3 [34] | Autoaug | - | 300 | 81.7 | - | 62.9 | - | 182 |
| | EfficientNet B4 [34] | Autoaug | - | 380 | 83.0 | - | 60.7 | - | 95 |
| | EfficientNet B5 [34] | Autoaug | - | 456 | 83.7 | - | 62.3 | - | 49 |
| | EfficientNet B6 [34] | Autoaug | - | 528 | 84.2 | - | 60.6 | - | 28 |
| | EfficientNet B7 [34] | Autoaug | - | 600 | 84.5 | - | 59.4 | - | 16 |
| E0 | R50 (baseline) | | 120 | 224 | 76.87 | 0.00 | 75.55 | 0.00 | 536 |
| E1 | R50**D** | | 120 | 224 | 77.37 | 0.50 | 75.73 | -0.18 | 493 |
| E2 | R50D | **LS** | 120 | 224 | 78.35 | 0.98 | 74.27 | 1.46 | 493 |
| E3 | R50D | LS+**Mixup** | 200 | 224 | 79.10 | 0.75 | 68.19 | 6.08 | 493 |
| E4 | R50D+**SE** | LS+Mixup | 200 | 224 | 79.71 | 0.61 | 64.48 | 3.71 | 420 |
| E5 | R50D+SE | LS+Mixup+**DropBlock** | 270 | 224 | 80.40 | 0.69 | 62.64 | 1.84 | 420 |
| E6 | R50D+**SK** | LS+Mixup+DropBlock | 270 | 224 | 81.40 | 1.00 | 58.34 | 4.30 | 359 |
| E7 | R50D+SK | LS+Mixup+DropBlock+**KD** | 270 | 224 | 81.69 | 0.29 | 57.08 | 1.26 | 359 |
| E8 | R50D+SK | LS+Mixup+DropBlock+KD | **600** | 224 | 82.10 | 0.41 | 56.48 | 0.60 | 359 |
| E9 | R50D+**BL**+SK | LS+Mixup+DropBlock+KD | 600 | **256** | 82.44 | 0.34 | 55.20 | 1.28 | 359 |
| E10 | R50D+BL+SK+**AA** | LS+Mixup+DropBlock+KD | 600 | 256 | 82.69 | 0.25 | 54.12 | 1.08 | 312 |
| E11 | R50D+BL+SK+AA | LS+Mixup+DropBlock+KD+**Autoaug** | 600 | 256 | 82.78 | 0.09 | 48.89 | 4.14 | 312 |
| E12 | **R152D**+BL+SK+AA | LS+Mixup+DropBlock+**KD**† +Autoaug | 600 | 256 | 84.19 | 1.41 | 43.27 | 5.62 | 143 |

Table 7. Ablation study for assembling the network tweaks and regularization with ResNet-50 on ImageNet ILSVRC2012 dataset. The focus of each experiment is highlighted in boldface. The cosine learning rate decay is used in all experiments. The top-1 accuracy and mCE scores for EfficientNet are borrowed from the official code in [17] and [37] respectively. As with other experiments, the inference throughput measurements of EfficientNet were performed on a single NVIDIA Tesla P40 using official EfficientNet code [17]. For comparison, we also experiment with ResNet-152. KD† uses EfficientNet-B7 instead of AmoebaNet as a teacher model.

**1)** The performance improvement effect of mCE is greater than the improvement of accuracy (E2,3,5,7,11).

**2)** SE is similar to the result of the regularization techniques confirming that channel attention is also helpful for robustness to image distortion.

**3)** BL and AA show a high performance improvement not only on top-1, but also on mCE. (E9.E10).

# 4.2 Transfer Learning: FGVC

Minghui Chen

# Stacking network tweaks and regularization

| Exp. No. | Backbone Model | Backbone top-1 | Regularization | Food-101 top-1 | Food-101 mCE |
|---|---|---|---|---|---|
| F0 | R50 (baseline) | 76.87 | - | 86.99 | 61.50 |
| F1 | R50D | 77.37 | - | 87.63 | 62.12 |
| F2 | R50D+**SK** | 78.83 | - | 89.77 | 57.20 |
| F3 | R50D+SK+**BL** | 79.27 | - | 90.15 | 57.16 |
| F4 | R50D+SK+BL+**AA** | 79.39 | - | 90.37 | 56.66 |
| F5 | R50D+SK+BL+AA | 79.39 | **DropBlock** | 91.25 | 53.13 |
| F6 | R50D+SK+BL+AA | 79.39 | DropBlock+**Mixup** | 91.64 | 48.53 |
| F7 | R50D+SK+BL+AA | 79.39 | DropBlock+Mixup+**Autoaug** | 91.85 | 41.73 |
| F8 | R50D+SK+BL+AA | 79.39 | DropBlock+Mixup+Autoaug+**LS** | 91.76 | 41.40 |
| F9 | R50D+SK+BL+AA+***REG*** | 82.78 | - | 90.63 | 53.98 |
| F10 | R50D+SK+BL+AA+*REG* | 82.78 | **DropBlock** | 91.62 | 51.01 |
| F11 | R50D+SK+BL+AA+*REG* | 82.78 | DropBlock+**Mixup** | 92.11 | 45.73 |
| F12 | R50D+SK+BL+AA+*REG* | 82.78 | DropBlock+Mixup+**Autoaug** | 92.21 | 41.69 |
| F13 | R50D+SK+BL+AA+*REG* | 82.78 | DropBlock+Mixup+Autoaug+**LS** | 92.47 | 41.99 |

***REG*** in backbone is effective

***REG*** in fine-tuning narrows the mCE gap.

Table 8. Ablation study of transfer learning with the Food-101 dataset. *REG* means that regularization techniques "LS + Mixup + DropBlock + KD + Autoaug" are applied during backbone training. The Food-101 mCE is not normalized by AlexNet's errors. We use the augmentation policy which is found by Autoaug on CIFAR-10 in these experiments [4].

# Dataset introduction

**Food-101**: They introduce a challenging data set of 101 food categories, with 101,000 images. For each class, 250 manually reviewed test images are provided as well as 750 training images. All images were rescaled to have a maximum side length of 512 pixels.

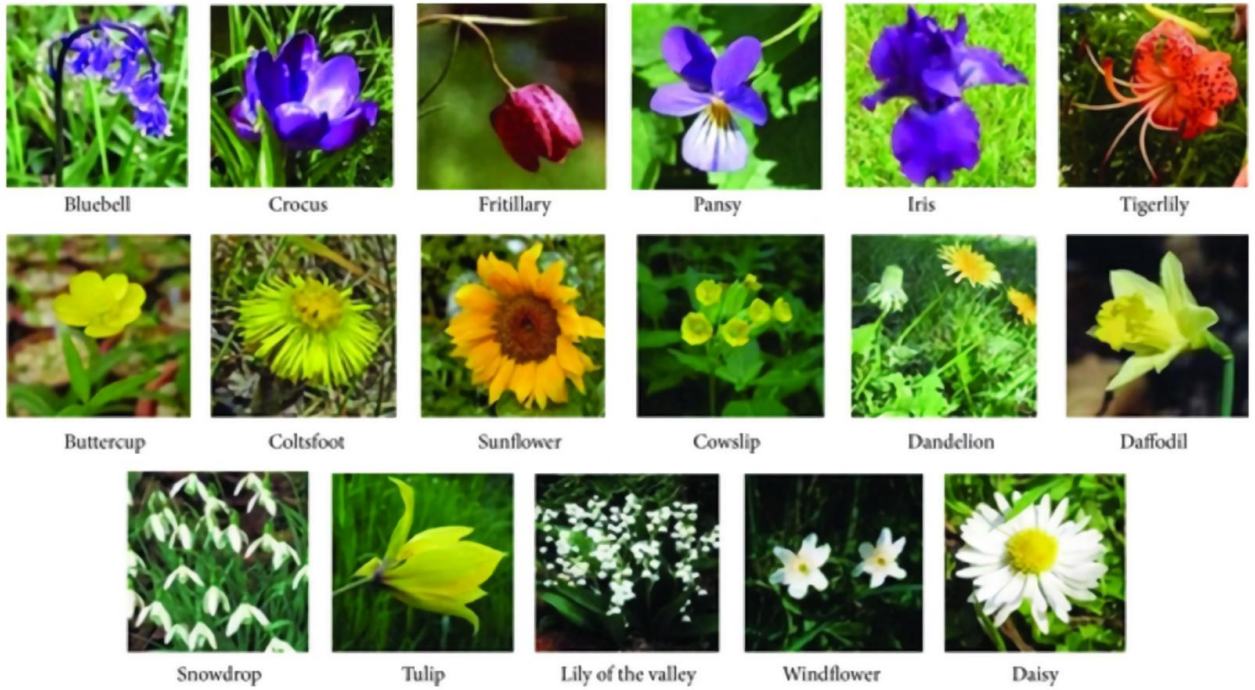
On purpose, the training images were not cleaned, and thus still contain some amount of noise. This comes mostly in the form of intense colors and sometimes wrong labels.

# Dataset introduction

**Stanford Cars**:  The Cars dataset contains 16,185 images of 196 classes of cars. The data is split into 8,144 training images and 8,041 testing images, where each class has been split roughly in a 50-50 split. Classes are typically at the level of Make, Model, Year, e.g. 2012 Tesla Model S or 2012 BMW M3 coupe.

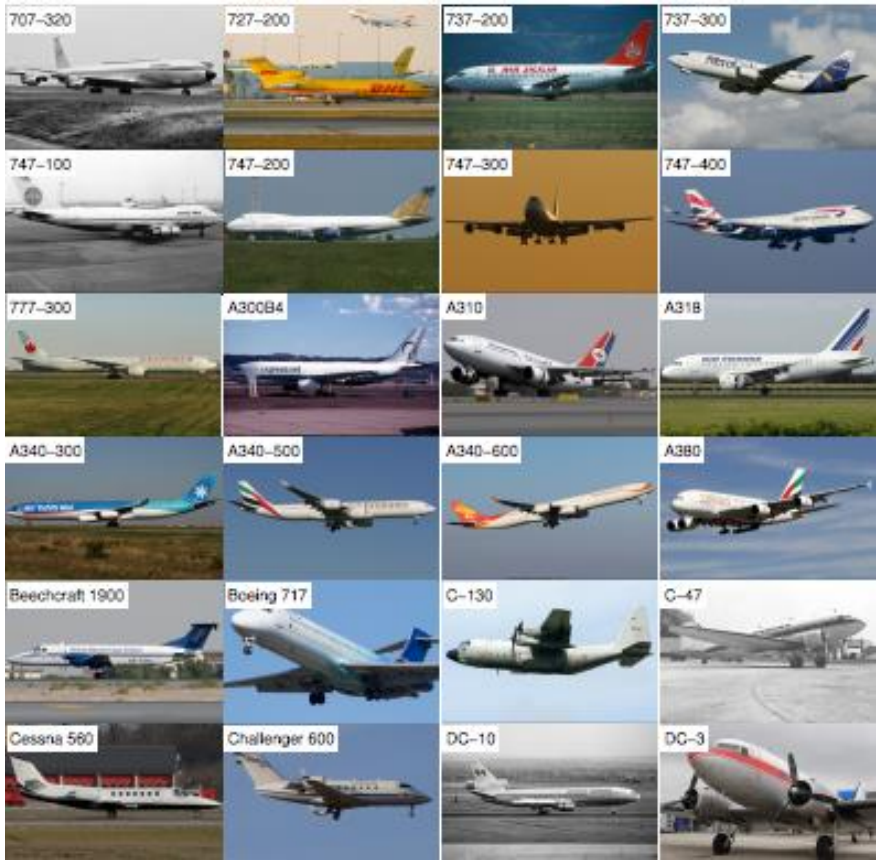
# Dataset introduction

**Oxford-Flowers**: They have created two flower datasets by gathering images from various websites, with some supplementary images from our own photographs. The first dataset is a smaller one consisting of 17 different flower categories, and the second dataset is much larger, consisting of 102 different categories of flowers common to the UK.



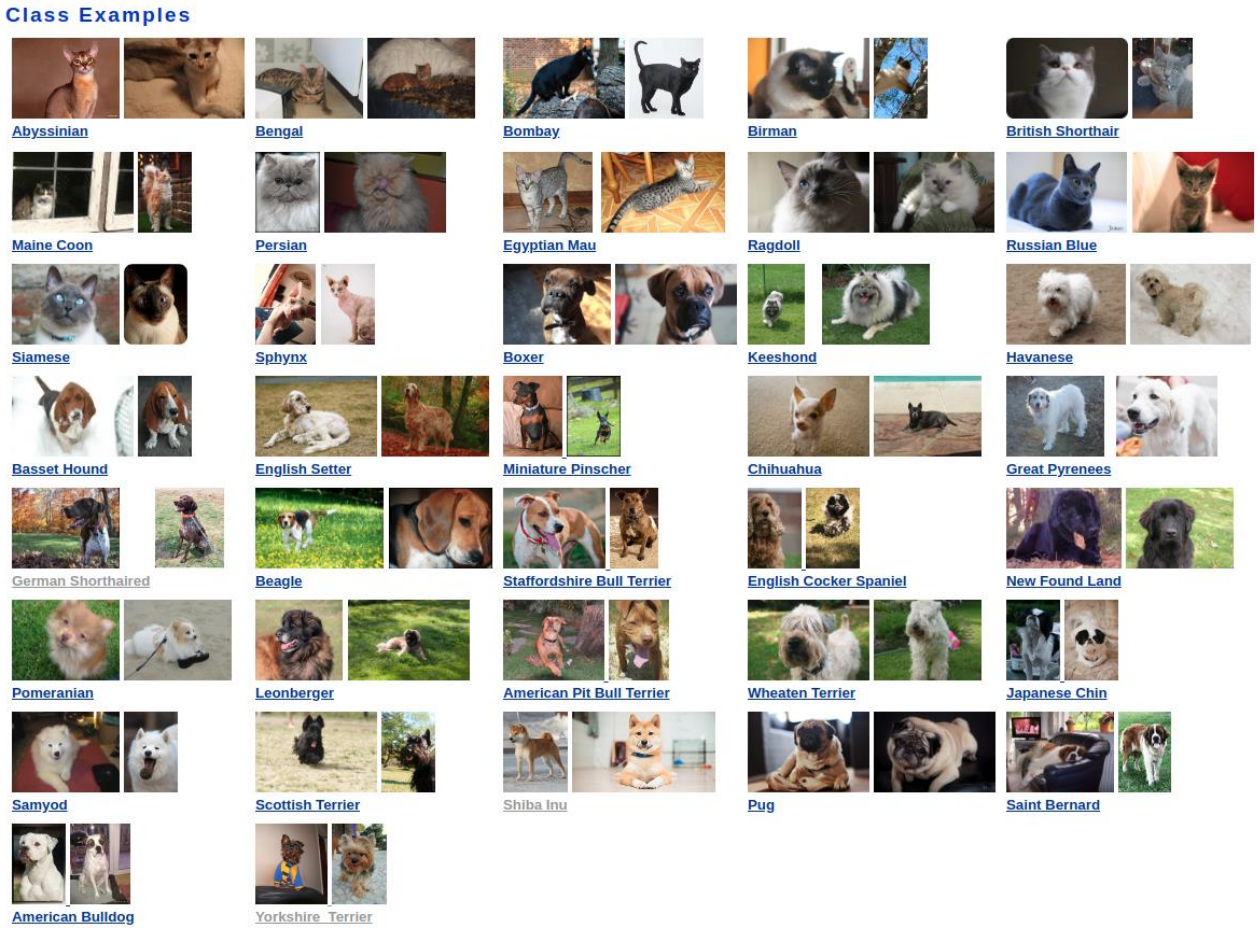
# Dataset introduction

**FGVC Aircraft**: Fine-Grained Visual Classification of Aircraft (FGVC-Aircraft) is a benchmark dataset for the fine grained visual categorization of aircraft. The dataset contains 10,200 images of aircraft, with 100 images for each of 102 different aircraft model variants, most of which are airplanes. The (main) aircraft in each image is annotated with a tight bounding box and a hierarchical airplane model label.



# Dataset introduction

**Oxford-IIIT Pets**: They have created a 37 category pet dataset with roughly 200 images for each class. The images have a large variations in scale, pose and lighting. All images have an associated ground truth annotation of breed, head ROI, and pixel level trimap segmentation.



# AmoebaNet

Regularized evolution for image classifier architecture search

AmoebaNet-B was obtained through platform-aware architecture search over a larger version of the NASNet space.

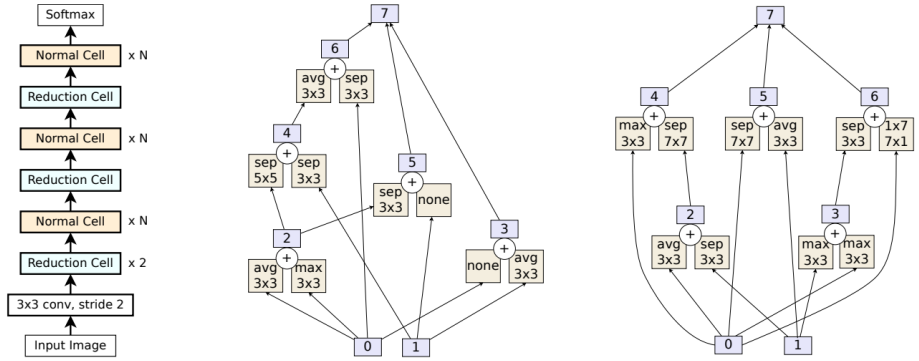



Figure 5: AmoebaNet-A architecture. The overall model [54] (LEFT) and the AmoebaNet-A normal cell (MIDDLE) and reduction cell (RIGHT).

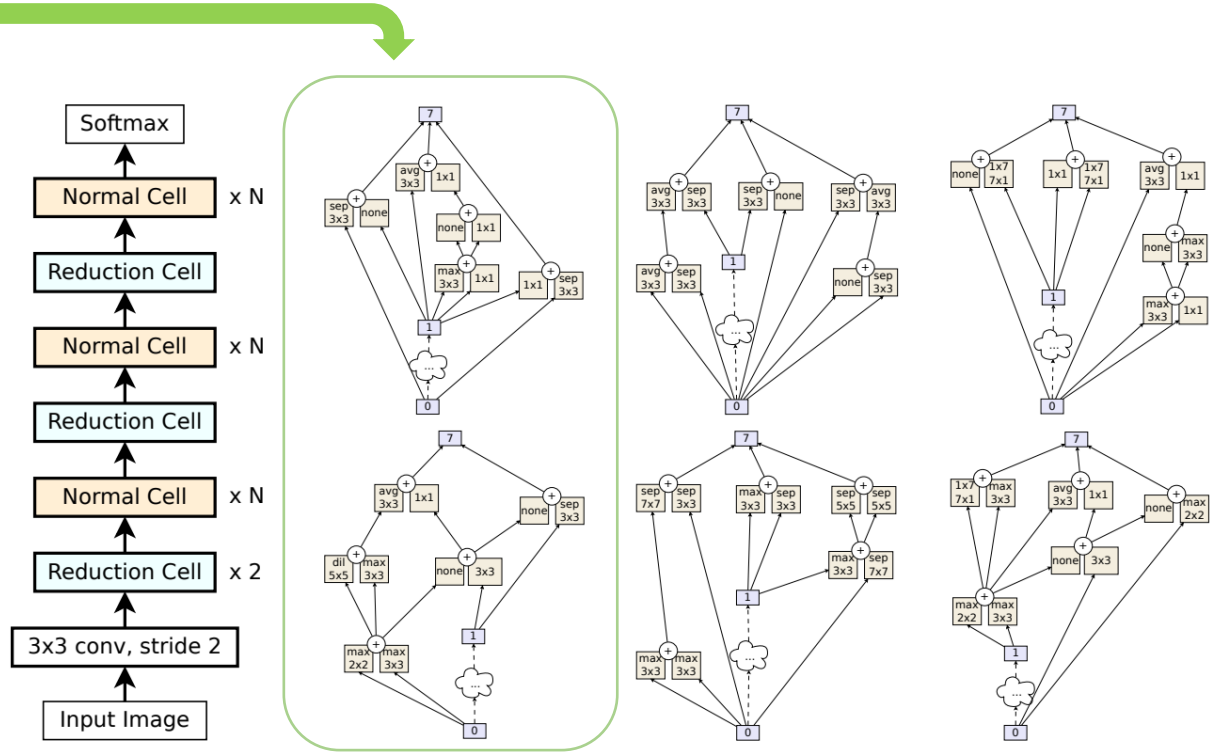



Figure D-1: Architectures of overall model and cells. From left to right: outline of the overall model [54] and diagrams for the cell architectures discovered by evolution: AmoebaNet-B, AmoebaNet-C, and AmoebaNet-D. The three normal cells are on the top row and the three reduction cells are on the bottom row. The labeled activations or hidden states correspond to the cell inputs ("0" and "1") and the cell output ("7").

# Transfer learning performance

| Dataset | The state-of-the-art Models | | ResNet-50 -tuned [18] | ResNet-50 | Assemble-ResNet-FGVC-50 |
|---|---|---|---|---|---|
| Food-101 | EfficientNet B7 [34] | 93.0 | 87.8 | 87.0 | 92.5 |
| Stanford Cars | EfficientNet B7 [34] | 94.7 | 91.7 | 89.1 | 94.4 |
| Oxford-Flowers | EfficientNet B7 [34] | 98.8 | 97.5 | 96.1 | 98.9 |
| FGVC Aircraft | EfficientNet B7 [34] | 92.9 | 86.6 | 78.8 | 92.4 |
| Oxford-IIIT Pets | AmoebaNet-B [16] | 95.9 | 91.5 | 92.5 | 94.3 |

Table 9. Transfer learning for FGVC task with our model and comparison with other state-of-the-art models. ImageNet-based transfer learning results are only compared with a single crop. Assemble-ResNet-FGVC-50 means that the final F13 model in Table 8. ResNet-50-tuned is a model in which the learning rate and weigh decay are tuned as in [18]. Unfortunately, Kornblith *et al*. [18] did not specify optimal hyperparameters for each datasets. We did not tune hyperparameters because we aim to compare between ResNet-50 and Assemble-ResNet-FGVC-50 rather than to achieve the state-of-the-art performance. Assemble-ResNet-FGVC-50 not only boosts the performance of ResNet-50 significantly, but also leads to performance comparable to heavyweight state-of-the-art models for all datasets.

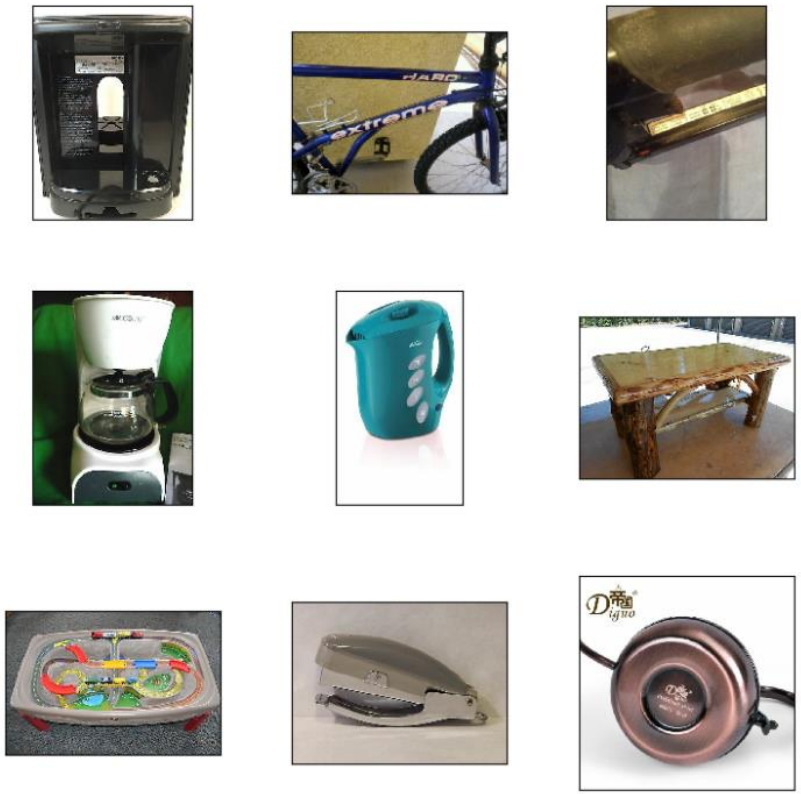
# 4.3 Transfer Learning: Image Retrieval

Jiang Xi

BL and AA did not work well on the SOP dataset.

DropBlock works well, but Autoaug do not improve the recall@1 performance

| Exp. No. | Backbone | Regularization | recall@1 |
|---|---|---|---|
| S0 | R50 (baseline) | | 82.9 |
| S1 | R50D | | 84.2 |
| S2 | R50D+SK | | 85.4 |
| S3 | R50D+SK+BL | | 85.2 |
| S4 | R50D+SK+BL+AA | | 85.1 |
| S5 | R50D+SK | DropBlock | 85.9 |
| S6 | R50D+SK | DropBlock+Autoaug | 83.7 |
| S7 | R50D+SK + *REG* | | 85.2 |
| S8 | R50D+SK + *REG* | DropBlock | 85.9 |
| S9 | R50D+SK + *REG* | DropBlock+Autoaug | 84.0 |




SOP collected Stanford Online Products dataset: 120k images of 23k classes of online products for metric learning.

# CUB200

Caltech-UCSD Birds 200 (CUB-200)  is an image dataset with photos of 200 bird species (mostly North American).
Warning: Images in this dataset overlap with images in ImageNet.

| Dataset | ResNet-50 | Assemble-ResNet-IR-50 |
|---|---|---|
| SOP | 82.9 | 85.9 |
| CUB200 | 75.9 | 80.3 |
| CARS196 | 92.9 | 96.1 |

IR uses a different regularization for each dataset.

| Dataset | Backbone | Regularization |
|---|---|---|
| SOP | R50D+SK+*REG* | DropBlock |
| CUB200 | R50D+SK+*REG* | DropBlock |
| CARS196 | R50D+SK+*REG* | DropBlock+LS+Autoaug |

Table 15. Model configuration for IR tasks. *REG* means "LS + Mixup + DropBlock + KD"