SEP-Nets: Small and Effective Pattern Networks

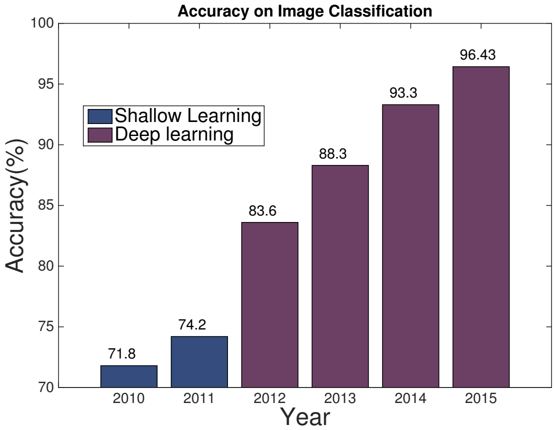
Zhe Li[†,‡]
Xiaoyu Wang[†], Xutao Lv[†] and Tianbao Yang[‡]

[†]Snap Research, [‡]The University of Iowa

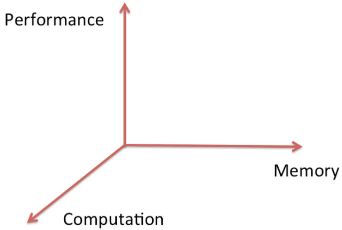
Thursday 24th August, 2017

Outline

The success of deep learning

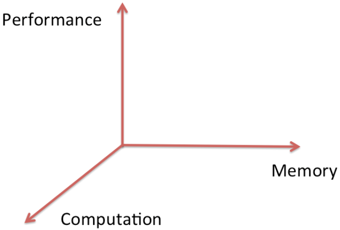


Three aspects of deep learning



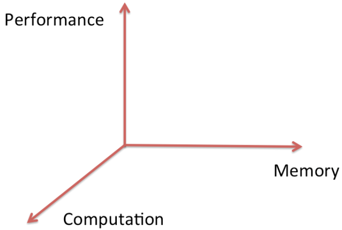
- Performance (Test accuracy): Almost Done

Three aspects of deep learning



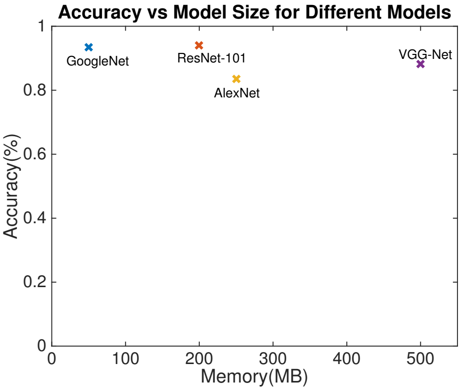
- Performance (Test accuracy): Almost Done
- Computation (Number of floating point operations): Not Yet

Three aspects of deep learning



- Performance (Test accuracy): Almost Done
- Computation (Number of floating point operations): Not Yet
- Memory (Number of parameters): Not Yet

Let's see performance and memory



What's wrong?

Not affordable for large neural network models

- Mobile device

Highly require small and effective neural networks

What's wrong?

Not affordable for large neural network models

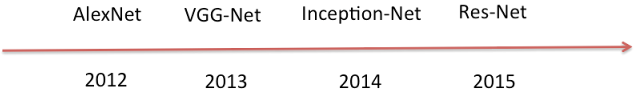
- Mobile device
- Embeded device

Highly require small and effective neural networks

Outline

Where to start?

Let's review the most successful nerual network structures:

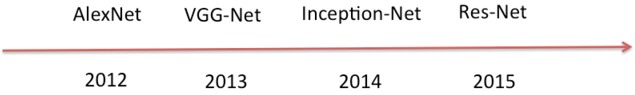


- Fully connected layers and convolution layers have most parameters in neural network models.

Focus on convolutional layers

Where to start?

Let's review the most successful nerual network structures:

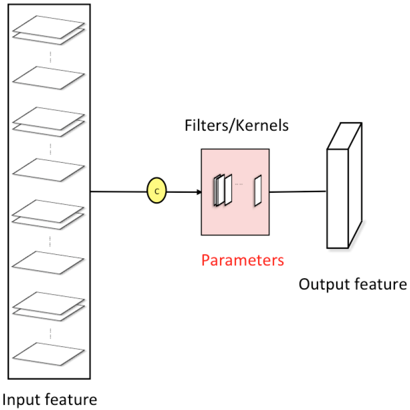


- Fully connected layers and convolution layers have most parameters in neural network models.
- Fully connected layers have been removed in modern deep CNN (Incpetion-Net, ResNets)

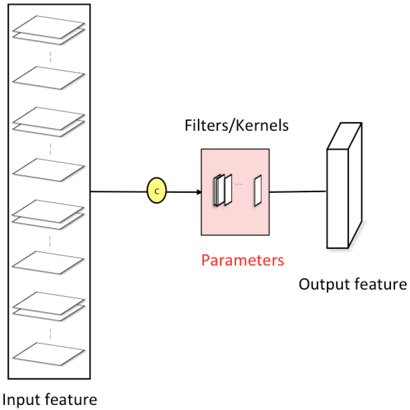
Focus on convolutional layers

Zoom in convolutional layers

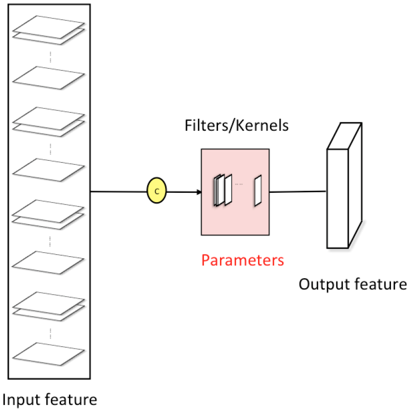
Zoom in convolutional layers



Zoom in convolutional layers

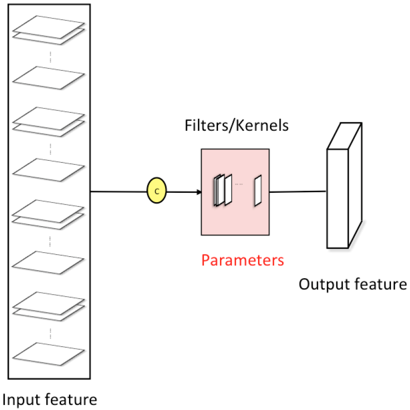


Zoom in convolutional layers



- $7 \times 7, 5 \times 5, 3 \times 3$ filters

Zoom in convolutional layers



- $7 \times 7, 5 \times 5, 3 \times 3$ filters
- 1×1 filters

Pattern Binarization

- $k \times k (k > 1)$ filters serve as spatial pattern extraction.

-0.0219	0.0408	-0.0547
-0.0855	0.0478	-0.0510
-0.0105	0.0924	-0.0126

-1	1	-1
-1	1	-1
-1	1	-1

A trained 3×3 filter from GoogleNet (Left) and its binarized version (right)

Pattern Binarization

- $k \times k (k > 1)$ filters serve as spatial pattern extraction.
- 1×1 filters serve as data transformation.

-0.0219	0.0408	-0.0547
-0.0855	0.0478	-0.0510
-0.0105	0.0924	-0.0126

-1	1	-1
-1	1	-1
-1	1	-1

A trained 3×3 filter from GoogleNet (Left) and its binarized version (right)

Pattern Binarization

- $k \times k (k > 1)$ filters serve as spatial pattern extraction.
- 1×1 filters serve as data transformation.
- Reduced number of parameters in model dramatically.

-0.0219	0.0408	-0.0547
-0.0855	0.0478	-0.0510
-0.0105	0.0924	-0.0126

$\Rightarrow$

-1	1	-1
-1	1	-1
-1	1	-1

A trained 3×3 filter from GoogleNet (Left) and its binarized version (right)

How to use Pattern Binarization?

Easily adopted to any successful networks structures such as GoogleNet, ResNet including the designed SEP-Nets as following procedure:

- Train a full neural network such as GoogleNet, ResNet and SEP-Net from scratch

How to use Pattern Binarization?

Easily adopted to any successful networks structures such as GoogleNet, ResNet including the designed SEP-Nets as following procedure:

- Train a full neural network such as GoogleNet, ResNet and SEP-Net from scratch
- Binarize $k \times k(k > 1)$ convolutional filters in the well-trained neural network model

How to use Pattern Binarization?

Easily adopted to any successful networks structures such as GoogleNet, ResNet including the designed SEP-Nets as following procedure:

- Train a full neural network such as GoogleNet, ResNet and SEP-Net from scratch
- Binarize $k \times k (k > 1)$ convolutional filters in the well-trained neural network model
- Fine-tune the scaling factors of all binarized $k \times k$ filters and the floating point representation of all 1×1 filters by back-propagation on the same dataset.

Outline

Pattern Resident Block

Pattern Resident Block

Pattern Resident Block

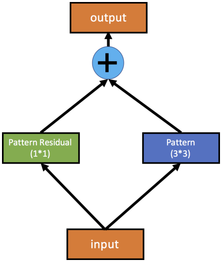
- Consists of 1×1 and $k \times k$ convolutions, which are executed in parallel and their feature map are added together.

Pattern Resident Block

- Consists of 1×1 and $k \times k$ convolutions, which are executed in parallel and their feature map are added together.
- Additive 1×1 convolutions act the residual between fully 3×3 filters maps and binarized 3×3 filtered maps.

Pattern Resident Block

- Consists of 1×1 and $k \times k$ convolutions, which are executed in parallel and their feature map are added together.
- Additive 1×1 convolutions act the residual between fully 3×3 filters maps and binarized 3×3 filtered maps.



SEP-Net Module

SEP-Net Module

SEP-Net Module

- 1×1 convolution layer: dimension reduction

SEP-Net Module

- 1×1 convolution layer: dimension reduction
- 2 PRB blocks with different output channels

SEP-Net Module

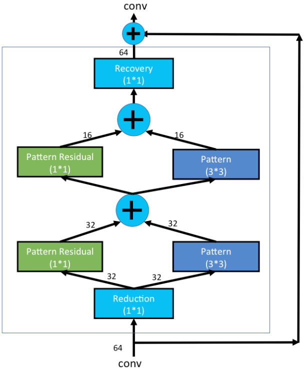
- 1×1 convolution layer: dimension reduction
- 2 PRB blocks with different output channels
- 1×1 convolution layer: dimension recovery

SEP-Net Module

- 1×1 convolution layer: dimension reduction
- 2 PRB blocks with different output channels
- 1×1 convolution layer: dimension recovery
- Skip connection

SEP-Net Module

- 1×1 convolution layer: dimension reduction
- 2 PRB blocks with different output channels
- 1×1 convolution layer: dimension recovery
- Skip connection



Group-wise Convolution

Group-wise Convolution

Group-wise Convolution

- Adopt group convolution to reduce the model size.

Group-wise Convolution

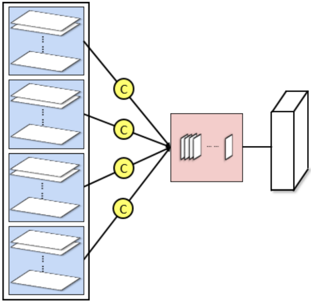
- Adopt group convolution to reduce the model size.
- Split the input features maps into N groups and apply convolution to each group.

Group-wise Convolution

- Adopt group convolution to reduce the model size.
- Split the input features maps into N groups and apply convolution to each group.
- Set group number as the number of input channels, it degenerates to depth-wise convolutions (Used in Google's MobileNets)

Group-wise Convolution

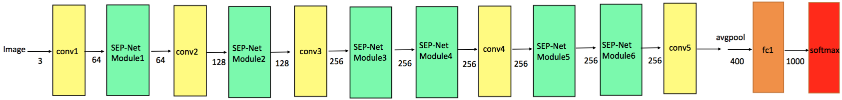
- Adopt group convolution to reduce the model size.
- Split the input features maps into N groups and apply convolution to each group.
- Set group number as the number of input channels, it degenerates to depth-wise convolutions (Used in Google's MobileNets)



Outline

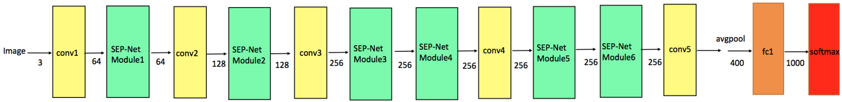
The Proposed SEP-Net structures

- Proposed two small SEP-Nets for mobile/embeded devices.



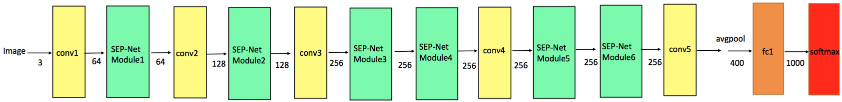
The Proposed SEP-Net structures

- Proposed two small SEP-Nets for mobile/embeded devices.
- One model has 1.3M parameters while the other 1.7M.



The Proposed SEP-Net structures

- Proposed two small SEP-Nets for mobile/embeded devices.
- One model has 1.3M parameters while the other 1.7M.
- Shared same following structure with slightly difference (group number, output dimension of the last convolution layer)



Outline

1. Motivation
2. The Proposed Method
3. The Ingredients for SEP-Nets
   - Pattern Resident Block
   - SEP-Net Module
   - Group-wise Convolution
4. The Proposed SEP-Nets
5. Experimental Results
6. Conclusion

Experimental Results

- Justify that pattern binarization can reduce number of parameters dramtically. (Small)

Experimental Results

- Justify that pattern binarization can reduce number of parameters dramtically. (Small)
  - CIFAR10 with ResNet-20, 34, 44, 50

Experimental Results

- Justify that pattern binarization can reduce number of parameters dramtically. (Small)
    - CIFAR10 with ResNet-20, 34, 44, 50
    - ImageNet with GoogleNet, Costomize-Inception-Net

Experimental Results

- Justify that pattern binarization can reduce number of parameters dramtically. (Small)
    - CIFAR10 with ResNet-20, 34, 44, 50
    - ImageNet with GoogleNet, Costomize-Inception-Net
- Justify that fine-tuning other parameters of the binraized network with fixed binarized patter could achieve comparable performance. (Effective)

Experimental Results

- Justify that pattern binarization can reduce number of parameters dramtically. (Small)
  - CIFAR10 with ResNet-20, 34, 44, 50
  - ImageNet with GoogleNet, Costomize-Inception-Net
- Justify that fine-tuning other parameters of the binraized network with fixed binarized patter could achieve comparable performance. (Effective)
  - same as the above setting.

Experimental Results

- Justify that pattern binarization can reduce number of parameters dramtically. (Small)
  - CIFAR10 with ResNet-20, 34, 44, 50
  - ImageNet with GoogleNet, Costomize-Inception-Net
- Justify that fine-tuning other parameters of the binraized network with fixed binarized patter could achieve comparable performance. (Effective)
  - same as the above setting.
- Show that the designed SEP-Net structures could achive better or comparable performance on ImageNet than using similar sized networks such as MobileNet. (Small & Effective)

Experimental Results–Training strategy

CIFAR10

- Preprocessed by Global Contrast Normalization and ZCA whitening.

Experimental Results–Training strategy

CIFAR10

- Preprocessed by Global Contrast Normalization and ZCA whitening.
- 32×32 crop is randomly sampled from the padded image.

Experimental Results–Training strategy

CIFAR10

- Preprocessed by Global Contrast Normalization and ZCA whitening.
- 32×32 crop is randomly sampled from the padded image.
- Initial learning rate is 0.1 and divided by 10 at iteration $32K$, $48K$.

Experimental Results–Training strategy

CIFAR10

- Preprocessed by Global Contrast Normalization and ZCA whitening.
- 32×32 crop is randomly sampled from the padded image.
- Initial learning rate is 0.1 and divided by 10 at iteration $32K$, $48K$.
- Maximum number of iteration is $64K$.

Experimental Results–Training strategy

CIFAR10

- Preprocessed by Global Contrast Normalization and ZCA whitening.
- 32×32 crop is randomly sampled from the padded image.
- Initial learning rate is 0.1 and divided by 10 at iteration $32K$, $48K$.
- Maximum number of iteration is $64K$.
- The momentum is 0.9 and the weight decay is 0.0001.

Experimental Results–Training strategy

CIFAR10

- Preprocessed by Global Contrast Normalization and ZCA whitening.
- 32×32 crop is randomly sampled from the padded image.
- Initial learning rate is 0.1 and divided by 10 at iteration $32K$, $48K$.
- Maximum number of iteration is $64K$.
- The momentum is 0.9 and the weight decay is 0.0001.
- Train on one GPU using mini-batch SGD with a batch size 256.

Experimental Results on Pattern Binarization

- Effective on CIFAR10

Model	Acc	**Ref**	**Full**	**BiPattern**	**Refined**
ResNet-20	Top-1	0.9125	0.9118	0.1546	0.8649
	Top-5	-	0.9974	0.5104	0.9941
ResNet-32	Top-1	0.9249	0.9276	0.2634	0.9021
	Top-5	-	0.9972	0.6932	0.9962
ResNet-44	Top-1	0.9283	0.9283	0.4825	0.9145
	Top-5	-	0.9982	0.8765	0.9965
ResNet-56	Top-1	0.9303	0.9375	0.5382	0.9302
	Top-5	-	0.9977	0.9574	0.9971

Experimental Results on Pattern Binarization

- Small on CIFAR10.

Model	**Full Network**	**Pattern Network**
ResNet-20	292K	55K
ResNet-32	487K	78K
ResNet-44	682K	100K
ResNet-56	876K	123K

Experimental Results on Pattern Binarization

- Small on CIFAR10.

Model	**Full Network**	**Pattern Network**
ResNet-20	292K	55K
ResNet-32	487K	78K
ResNet-44	682K	100K
ResNet-56	876K	123K

- Use one number to represent a binarized 3×3.

Experimental Results–Training strategy

ImageNet on GoogleNet

- Initial learning rate is 0.01 and follow a polynomial decay.

ImageNet on C-InceptionNet

Experimental Results–Training strategy

ImageNet on GoogleNet

- Initial learning rate is 0.01 and follow a polynomial decay.
- Maximum number of iteration is $600K$.

ImageNet on C-InceptionNet

Experimental Results–Training strategy

ImageNet on GoogleNet

- Initial learning rate is 0.01 and follow a polynomial decay.
- Maximum number of iteration is $600K$.
- The momentum is 0.9 and the weight decay is 0.0001.

ImageNet on C-InceptionNet

Experimental Results–Training strategy

ImageNet on GoogleNet

- Initial learning rate is 0.01 and follow a polynomial decay.
- Maximum number of iteration is 600K.
- The momentum is 0.9 and the weight decay is 0.0001.
- Train on one GPU using mini-batch SGD with a batch size 128.

ImageNet on C-InceptionNet

Experimental Results–Training strategy

ImageNet on GoogleNet

- Initial learning rate is 0.01 and follow a polynomial decay.
- Maximum number of iteration is $600K$.
- The momentum is 0.9 and the weight decay is 0.0001.
- Train on one GPU using mini-batch SGD with a batch size 128.

ImageNet on C-InceptionNet

- Initial learning rate is 0.1 and divided learning rate 10 time after every 24 epochs.

Experimental Results–Training strategy

ImageNet on GoogleNet

- Initial learning rate is 0.01 and follow a polynomial decay.
- Maximum number of iteration is $600K$.
- The momentum is 0.9 and the weight decay is 0.0001.
- Train on one GPU using mini-batch SGD with a batch size 128.

ImageNet on C-InceptionNet

- Initial learning rate is 0.1 and divided learning rate 10 time after every 24 epochs.
- Train total 90 epochs.

Experimental Results on Pattern Binarization

- Effective on ImageNet

Model	Acc	Ref	Full	BiPattern	Refined	Multicrop
GoogLeNet				1x1 pattern:		
				0.0013	0.6117	0.636
				0.0075	0.8395	0.856
				2-8 3x3 pattern:		
				0.3706	0.6797	0.6893
	Top-1	-	0.6865	0.6290	0.8827	0.8898
	Top-5	0.8993	0.8891	5x5 pattern:		
				0.5141	0.6917	0.6984
				0.7619	0.8904	0.8965
				3x3 & 5x5 pattern:		
				0.1428	0.6694	0.6812
				0.31738	0.8763	0.8844
C-InceptionNet	Top-1		0.648	0.0476	0.6400	0.6521
	Top-5		0.863	0.1464	0.8550	0.8626

Exprimental Results on Pattern Binarization

- Small on ImageNet

Model	Full Network	Pattern Network	
GoogLeNet	6.99M	3×3	4.43M
		5×5	6.43M
		3×3 and 5×5	3.87M
C-InceptionNet	5.10M		2.43M

Exprimental Results on Pattern Binarization

- Small on ImageNet

Model	Full Network	Pattern Network	
GoogLeNet	6.99M	3×3	4.43M
		5×5	6.43M
		3×3 and 5×5	3.87M
C-InceptionNet	5.10M		2.43M

- Use one number to represent a 3×3 or 5×5 kernel.

Experimental Results for the Designed SEP-Nets

- Small and Effective on the designed SEP-Nets

Model	Parameter Number	Size (bytes)	Top-1 Acc
MobileNet	1.3M	5.2MB	0.637
	2.6M	10.4MB	0.684
SEP-Net-R	1.3M (small)	5.2MB	0.658
	1.7M (large)	6.7MB	0.667
-	-	-	-
SqueezeNet	1.2M	4.8MB	0.604
MobileNet	1.3M	5.2MB	0.637
SEP-Net-R (small)	**1.3M**	**5.2MB**	**0.658**
SEP-Net-B (small)	1.1M	4.2MB	0.637
SEP-Net-BQ (small)	1.1M	1.3MB	0.635

SEP-Net-R: SEP-Net with raw valued weights
SEP-Net-B: SEP-Net with pattern binarization
SEP-Net-BQ: SEP=Net with pattern binarization and other weights quantized using linear quantization with 8 bits

Another Angle to Pattern Binarization

Analyze the effect of binarizing 1×1 filters and $k \times k$ filter from the view of quantization error:

- Let W denote an $c \times k \times k$ convolutional filter.

Another Angle to Pattern Binarization

Analyze the effect of binarizing 1×1 filters and $k \times k$ filter from the view of quantization error:

- Let W denote an $c \times k \times k$ convolutional filter.
- binarization seeks to approximate it by αB, where B is a binary filter with entries from $\{1, -1\}$ and α is a scaling factor.

Another Angle to Pattern Binarization

Analyze the effect of binarizing 1×1 filters and $k \times k$ filter from the view of quantization error:

- Let W denote an $c \times k \times k$ convolutional filter.
- binarization seeks to approximate it by αB, where B is a binary filter with entries from $\{1, -1\}$ and α is a scaling factor.
- From the viewpoint of minimizing the quantization error, α, B can be sought by solving the following problem:

$$\min_{\alpha \in \mathbb{R}, B \in \{1,-1\}^{c \times k \times k}} E(W, B, \alpha) \triangleq \|W - \alpha B\|_F^2 \qquad (1)$$

Another Angle to Pattern Binarization

Analyze the effect of binarizing 1×1 filters and $k \times k$ filters from the view of quantization error:

- The optimal B^* can be found by thresholding, i.e., $B^*_{i,j,l} = 1$ if $W_{i,j,l} \geq 0$ and $B^*_{i,j,l} = -1$ if $W_{i,j,l} < 0$.

Another Angle to Pattern Binarization

Analyze the effect of binarizing 1×1 filters and $k \times k$ filters from the view of quantization error:

- The optimal B^* can be found by thresholding, i.e., $B^*_{i,j,l} = 1$ if $W_{i,j,l} \geq 0$ and $B^*_{i,j,l} = -1$ if $W_{i,j,l} < 0$.
- The optimal α_* can be computed by $\alpha_* = \frac{\sum_{i,j,l} |W_{i,j,l}|}{c \times k \times k}$.

Another Angle to Pattern Binarization

Analyze the effect of binarizing 1×1 filters and $k \times k$ filters from the view of quantization error:

- The optimal B^* can be found by thresholding, i.e., $B^*_{i,j,l} = 1$ if $W_{i,j,l} \geq 0$ and $B^*_{i,j,l} = -1$ if $W_{i,j,l} < 0$.
- The optimal α_* can be computed by $\alpha_* = \frac{\sum_{i,j,l} |W_{i,j,l}|}{c \times k \times k}$.
- To quantitatively understand the effect of binarizing 1×1 filters and $k \times k$ filters, we compute the quantization error for all filters in the well-trained GoogleNet and obtain averaged quantization error for different filters:

1×1	3×3	5×5
0.0462	0.0029	0.0056

Outline

1. Motivation
2. The Proposed Method
3. The Ingredients for SEP-Nets
   - Pattern Resident Block
   - SEP-Net Module
   - Group-wise Convolution
4. The Proposed SEP-Nets
5. Experimental Results
6. Conclusion

Conclusion

- Proposed pattern binarization method.

Conclusion

- Proposed pattern binarization method.
- Designed a new pattern residual block.

Conclusion

- Proposed pattern binarization method.
- Designed a new pattern residual block.
- Designed a novel SEP-Net Module.

Conclusion

- Proposed pattern binarization method.
- Designed a new pattern residual block.
- Designed a novel SEP-Net Module.
- Proposed Small and Effective Pattern Networks.

Conclusion

- Proposed pattern binarization method.
- Designed a new pattern residual block.
- Designed a novel SEP-Net Module.
- Proposed Small and Effective Pattern Networks.
- Achieved the-state-of-art performance.

Future future following this work

- Train the pattern network with binarized $k \times k$ filters from scratch?

Future future following this work

- Train the pattern network with binarized $k \times k$ filters from scratch?
- Reduce computation cost (number of floating point operation)?

Question?