RemNet
Remnant Convolutional Neural Network for Camera Model Identification

CONTENTS

01 Introduce

In designing CNNs for image forensic tasks, it has been common practice to use a preprocessing scheme to suppress the image contents and intensify the minute signatures

Conventional either fixed kernels or constraints has been used in some works

Methods reported so far suffer from their own **drawbacks** of using either fixed kernels or constraints

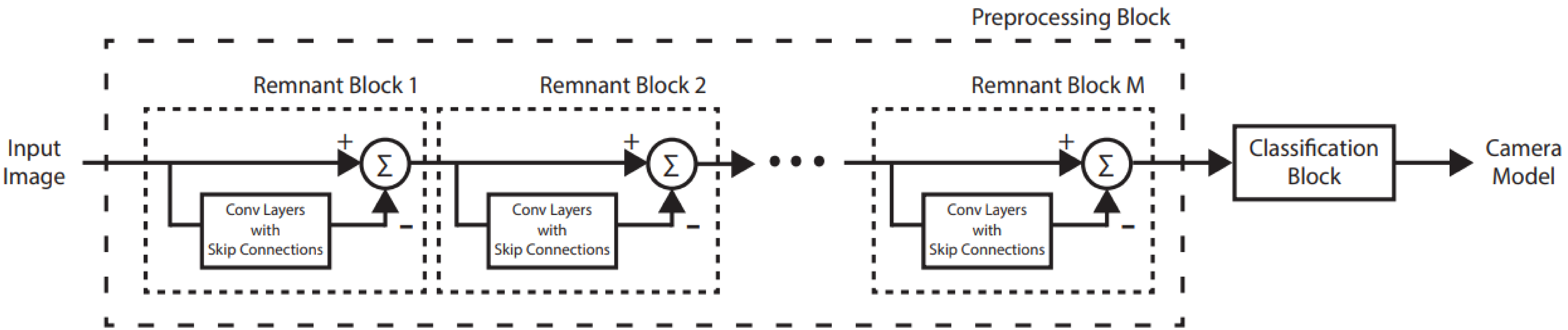
High-pass: loss of valuable camera model specific features
Median: may not serve the purpose optimally
Constrained: originally proposed for image manipulation detection

Introduce a preprocessing scheme that is completely data-driven but without any imposed constraints or fixed kernels

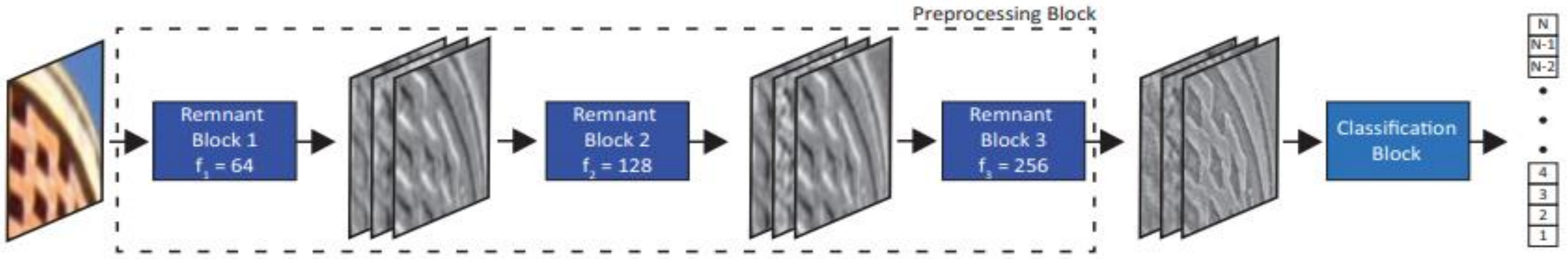
02 RemNet

- Consist of a data-driven **preprocessing block** and a **shallow classification block**

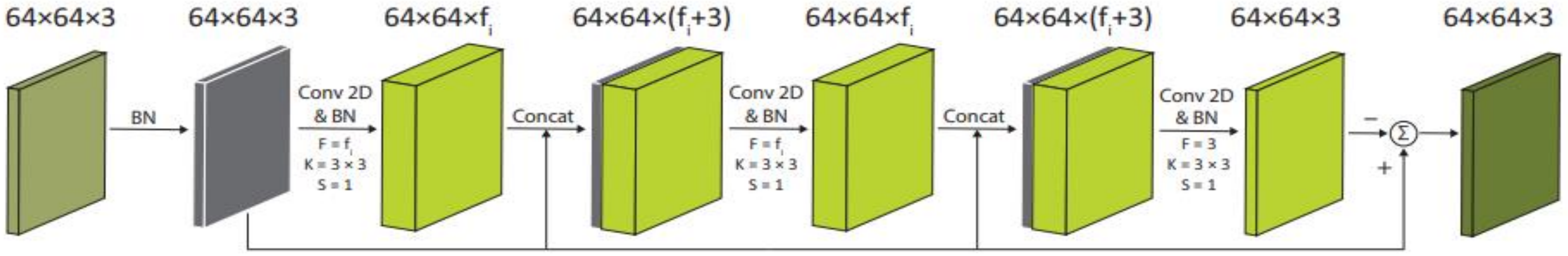
- Major constituent part of RemNet is the data-adaptive preprocessor that comprises of several remnant block



02 RemNet
Preprocessing Block



(a) Overall Architecture



(b) Architecture of the *i*-th Remnant Block

Expect to learn the required transformation that would disintegrate the undesired contents

→ Subsequent subtraction operation can suppress them

→ generate forensic feature enriched residue

02 RemNet
Preprocessing Block

Remnant Block

- Be designed as a linear preprocessor
- Multiple intra-block skip connections in remnant block
  → To preserve input information
  → To prevent gradient vanishing
- Pixel-wise subtraction operation
  → to generate the residue in a block
  → suppress undesired contents and generate forensic feature enriched residue

Subsequent blocks operate on the residue generated by the previous block
→ Information loss would gradually build up
→ Degradation of the model's performance

⇒ Skip Connection

02 RemNet
Preprocessing Block

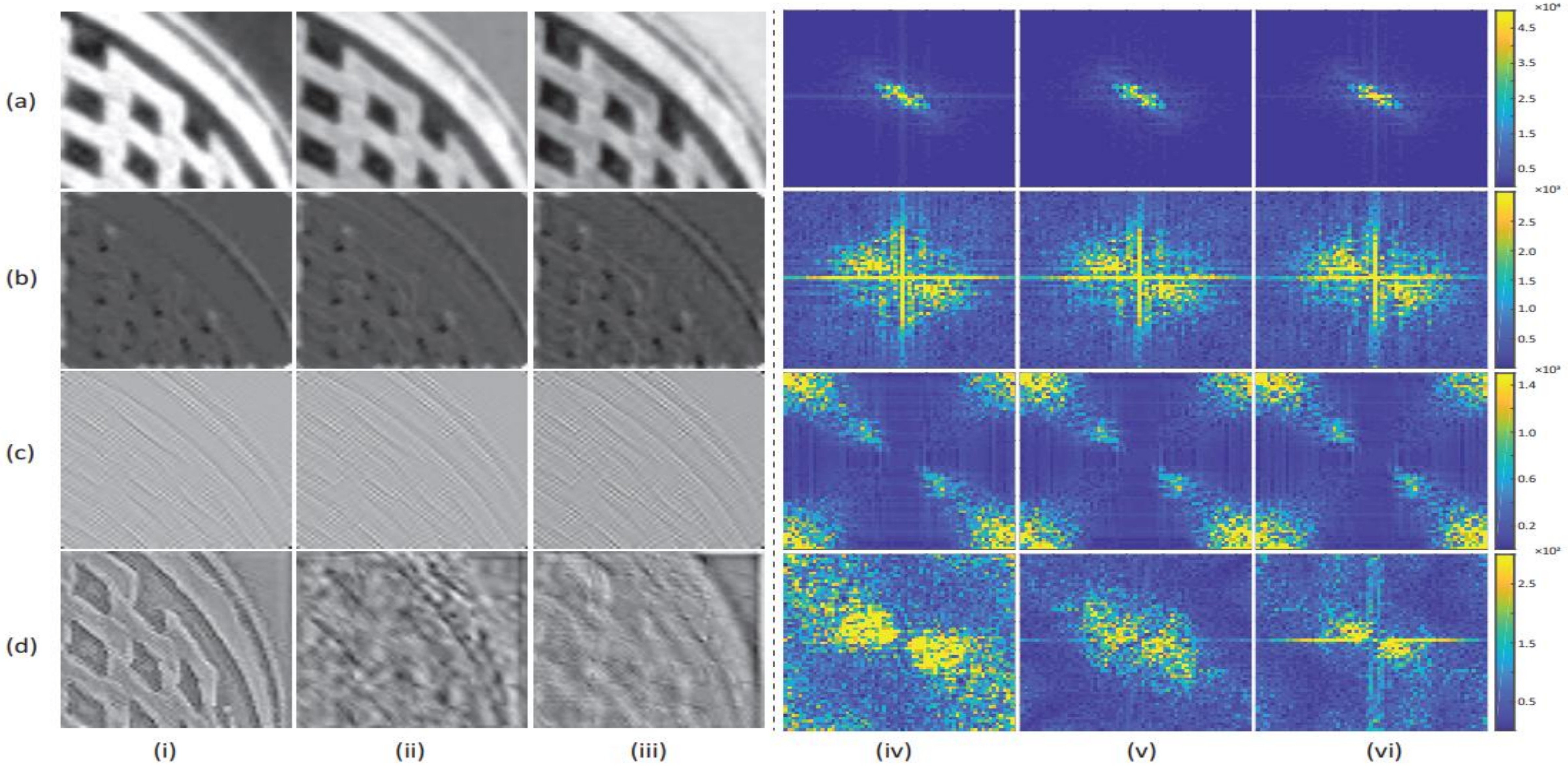
Comparison of outputs of various preprocessing schemes

Original image

Median filter

High pass filter

Output of the last remnant block



Different channel's outputs

Frequency responds

Apply the same frequency domain transformation on all the channels equally

02 RemNet

Classification Block

Extract higher-level camera model-specific features, reduce the dimensions of the feature vectors, and eventually generate a class probability of the source camera model of the input image

Table 1: Architecture of our proposed RemNet

Layers	Output Size	Kernels*
Preprocessing Block		
Remnant Block 1	64×64×3	$f_1 = 64$
Remnant Block 2	64×64×3	$f_2 = 128$
Remnant Block 3	64×64×3	$f_3 = 256$
Classification Block		
Conv 2D, BN, & PReLU	32×32×64	F = 64, K = 7×7, S = 2
Conv 2D, BN, & PReLU	16×16×128	F = 128, K = 5×5, S = 2
Conv 2D, BN, & PReLU	8×8×256	F = 256, K = 3×3, S = 2
Conv 2D, BN, & PReLU	4×4×512	F = 512, K = 2×2, S = 2
Average Pool	1×1×512	K = 4×4
Conv 2D	$1\times1\times N_{class}$	F = N_{class}, K = 1×1, S = 1
Softmax	N_{class}	–

* The letters F, K, and S represent the number of filters, their kernel size, and strides, respectively, in the corresponding convolution layers. The letter N_{class} represents the number of camera models.

- BN
  - Regularization and faster convergence
- PReLU
  - Do not put any constraint on the feature generation
- Average-pooling
  - Keep the number of parameters lower
  - Less prone to overfitting

02 RemNet

Loss function and training

Preprocessing block

$$\mathbf{y}_{\mathbf{p_i}} = \mathbf{x_i} - H\left(\mathbf{x_i}, \mathbf{W}_{\mathbf{p_i}}\right),$$

Classification block

$$\mathbf{y_c} = G\left(\mathbf{y}_{\mathbf{p_M}}, \mathbf{W_c}\right),$$

Multiclass catergorical crossentropy loss

$$L = \sum_{k=1}^{N_{class}} y_{c_i}^{*(k)} \log\left(y_{c_i}^{(k)}\right),$$

Update the weights of both the preprocessing block and the classifier block of the network.

03 Experiments
Result on Dresden Dataset

Dataset: Dresden Dataset

Choose only those camera models which have more than one device
→**18 models**

→Train: 7938, validation : 1353, test:540

Serial No.	Camera Model	No. of Images	No. of Devices	
			Train and Val.	Test
1	Canon IXUS 70	363	2	1
2	Casio EX-Z150	692	4	1
3	FujiFilm FinePix J50	385	2	1
4	Kodak M1063	1698	4	1
5	Nikon Coolpix S710	695	4	1
6	Nikon D200	373	1	1
7	Nikon D70 Nikon D70S	373	1 1	1 1
8	Olympus μ1050SW	782	4	1
9	Panasonic DMC-FZ50	564	2	1
10	Pentax Optio A40	405	3	1
11	Praktica DCZ 5.9	766	4	1
12	Ricoh Capilo GX100	559	4	1
13	Rollei RCP-7325XS	377	2	1
14	Samsung L74wide	441	2	1
15	Samsung NV15	412	2	1
16	Sony DSC-H50	253	1	1
17	Sony DSC-T77	492	3	1
18	Sony DSC-W170	201	1	1
	Total	9831		

03 Experiments

Result on Dresden Dataset

Splitting policy (83scenes)

- we selected evaluation photos ($\mathbb{D}_E$) from 11 scenes and a single instance per model.
- we selected training photos ($\mathbb{D}_T$) from 62 different scenes and instances.
- we selected validation photos ($\mathbb{D}_V$) from the remaining 10 scenes and instances used for training.

✓ Neural network does not overfit on the training data

✓ Testing accuracy is not biased by device specific features or the natural content of the scenes

Extract 256x256 sized cluster of pixels

$$Q(\mathcal{P}) = \frac{1}{3} \sum_{c \in [R,G,B]} \left[\alpha \cdot \beta \cdot (\mu_c - \mu_c^2\right) + \left(1 - \alpha\right) \cdot (1 - e^{\gamma \sigma_c})]$$

Compute the quality value of a cluster

a: 0.7, B:4, r:in(0.01)

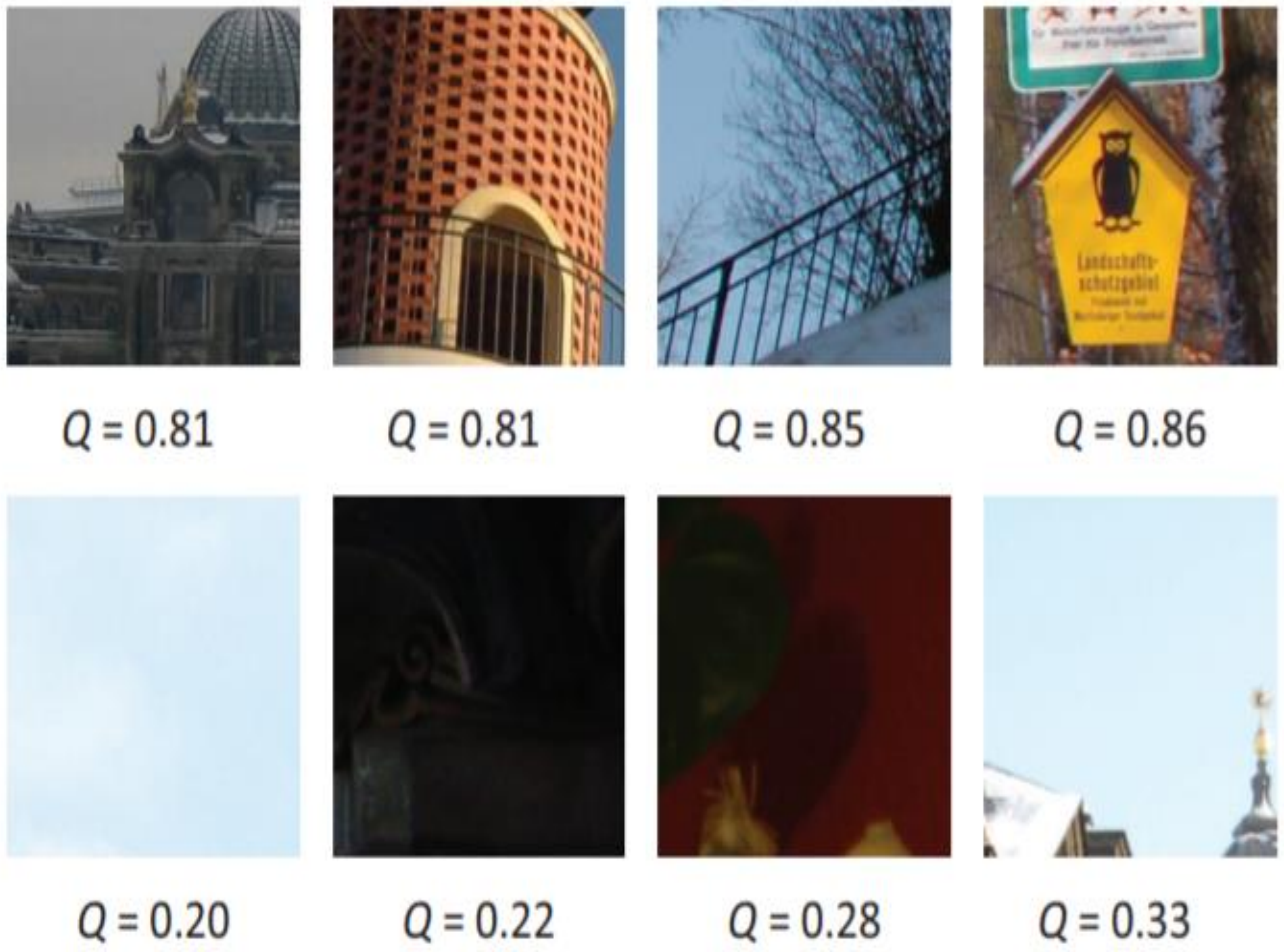
03 Experiments
Result on Dresden Dataset

Others – 99.32%
Smooth – 0.63%
Saturated – 0.03%

Selection strategy is almost identical to choosing the 'others' category patches

$$I_i \in \begin{cases} Subset1 = T_1(m, v) \\ Subset2 = T_2(m, v) \\ Subset3 = T_3(m, v) \end{cases} \tag{4}$$

$$\begin{cases} T_1(m, v) & m \in [0, 5] \bigcup [250, 255], v \in [0, 25] \\ T_2(m, v) & m \in [0, 5] \bigcup [250, 255], v \in [25, 50] \; || \\ & m \in (5, 250), v \in [0, 50] \\ T_3(m, v) & others \end{cases} \tag{5}$$



03 Experiments

Result on Dresden Dataset

Select a patch of size 64x64 randomly from a cluster of 256x256 in each epoch

- More data
- Multiple predictions for a given image and averaging over all of those predictions → may more accurate classification
- Prevents network from learning dominant spatial features of the image affixed directly to its contents
- Some of the rich quality clusters of 256x256 may contain a few bad patches of 64x64
  - → learn to extract features from saturated regions as well

Hyper parameter

- Kernel are initialized randomly with uniform distribution
- Loss function: categorical cross-entropy
- Optimizer : Adam(decay rate = 0.9, 0.999)
- Batch : 64
- Learning rate : 10e-3 (Validation loss does not decrease in two epochs → decrease factor: 0.5)
- Learning rate is 10e-7 → training stop
- Maximum train : 50epochs

03 Experiments

Result on Dresden Dataset

Test(unaltered)

1. Select N number of rich quality clusters of size 256x256
2. Average of the predictions on all non-overlapping patches of size 64x64
3. Majority voting

$$\text{Accuracy} = \frac{N_{corr}}{N_{tot}} \times 100\%,$$

Design Choice	Accuracy (%)
Remnant Blocks + Classifier (ReLU)	96.48
Remnant Blocks with Activation (PReLU) + Classifier (PReLU)	96.67
Remnant Blocks + Classifier (PReLU)	**97.03**

Method	Accuracy (%)
Bayar and Stamm [21]	95.56
Yang et al. [23]	94.81
Bondi et al. [24]	90.55
ResNet [32]	92.40
DenseNet [44]	93.33
Proposed Method	**97.03**

03 Experiments

Result on Dresden Dataset

Data Augmented(Train, Validation)

- JPEG compression : 70%, 80%, 90%
- Rescaling : 0.5, 0.8, 1.5, 2.0
- Gamma Correction : 0.8, 1.2

Method	Accuracy (%)
Bayar and Stamm [21]	93.89
Yang et al. [23]	95.19
Bondi et al. [24]	92.59
ResNet [32]	95.18
DenseNet [44]	95.05
Proposed Method	**97.59**

Data Augmented(Test)

- JPEG compression : 95%, 90%, 85%, 80%
- Rescaling : 0.8, 0.9, 1.1, 1.2
- Gamma Correction : 0.5, 0.75, 1.25, 1.5

Manipulation	Gamma Correction				JPEG Compression				Rescale			
Factor	0.5	0.75	1.25	1.5	95	90	85	80	0.8	0.9	1.1	1.2
Bayar and Stamm [21]	93.52	94.44	94.44	94.63	92.59	94.81	88.15	85.74	88.15	87.04	64.44	59.07
Yang et al. [23]	94.26	95.37	95.00	92.78	94.07	94.07	92.59	92.59	94.26	92.59	90.93	90.56
Bondi et al. [24]	85.92	91.85	89.07	92.03	84.07	85.92	91.48	90.74	92.56	92.77	91.48	89.44
ResNet. [32]	91.85	95.18	92.77	94.81	93.88	94.82	95.55	95.00	95.18	95.18	95.00	95.18
DenseNet. [44]	91.66	95.18	92.03	94.62	92.77	92.96	94.26	94.81	95.00	94.81	94.44	94.26
Proposed Method	96.11	97.22	96.11	95.56	97.59	94.81	92.59	92.78	95.00	93.33	92.04	92.41

03 Experiments

Result on Dresden Dataset

Improvements using Remnant block (train O , test X)

Method	Trained on Unaltered Train Set		Trained on Augmented Train Set	
	without remnant blocks	with remnant blocks	without remnant blocks	with remnant blocks
Bondi et al. [24]	90.55	95.92	92.59	96.29
ResNet [32]	92.40	96.85	95.18	98.33
DenseNet [44]	93.33	96.29	95.01	98.14
Proposed Classifier	93.31	97.03	95.74	97.59

Improvements using Remnant block (train O , test O)

Manipulation	Gamma Correction				JPEG Compression				Resize Scale			
Factor	0.5	0.75	1.25	1.5	95	90	85	80	0.8	0.9	1.1	1.2
Bondi et al. [24]	85.92	91.85	89.07	92.03	84.07	85.92	91.48	90.74	92.56	92.77	91.48	89.44
Remnant-Bondi et al.	94.07	95.74	95.37	95.92	88.88	89.07	93.52	92.22	91.66	91.85	90.00	88.14
ResNet. [32]	91.85	95.18	92.77	94.81	93.88	94.82	95.55	95.00	95.18	95.18	95.00	95.18
Remnant-ResNet	98.33	98.33	97.59	97.59	93.33	93.33	95.18	95.92	95.37	95.18	92.40	95.00
DenseNet. [44]	91.66	95.18	92.03	94.62	92.77	92.96	94.26	94.81	95.00	94.81	94.44	94.26
Remnant-DenseNet.	96.85	97.59	97.96	97.59	93.70	93.88	94.81	95.92	95.37	94.81	93.52	95.18

03 Experiments

Result on IEEE Signal Processing Cup 2018 Dataset

- Split into train and validation data by a 3:1 ratio
- **test dataset** is provided separately, which includes 2640 images of size 512, among which 1320 are unaltered, and the rest are augmented.
- Set the number of class 10

$$\text{Accuracy} = 0.7 \times (\text{Accuracy of Unaltered Images}) + 0.3 \times (\text{Accuracy of Manipulated Images})$$

Use for training and validation only

Table 9: IEEE SP Cup 2018 data and Flickr data

Camera Model	No. of Images	
	SP Cup Data	Flickr Data
HTC-1-M7	275	746
iPhone-4s	275	499
iPhone-6	275	548
LG-Nexus-5x	275	405
Motorola-Droid-Maxx	275	549
Motorola-Nexus-6	275	650
Motorola-X	275	344
Samsung-Note3	275	274
Samsung-Galaxy-S4	275	1137
Sony-NEX-7	275	557
Sub-Total	2750	5709
Grand-Total	8459	

03 Experiments

Result on IEEE Signal Processing Cup 2018 Dataset

Table 10: Accuracy of different methods on the IEEE SP Cup 2018 testing dataset

Method	Accuracy (%)
Bayar and Stamm [21]	90.97
Yang et al. [23]	94.83
Bondi et al. [24]	90.07
ResNet [32]	93.92
DenseNet [44]	93.70
Proposed Method	**95.11**

Effect of remnant blocks

Table 11: Comparative results of different models, in cascade with remnant blocks, tested on the IEEE SP Cup 2018 testing dataset

Method	Accuracy (%)
Remnant-Bondi et al.	92.15
Remnant-ResNet	93.98
Remnant-DenseNet	94.68