Convolutional Neural Networks in Logarithmic Gradient Image Sensing

Qianyun Lu
Department of Electrical Engineering, Stanford University
San Francisco, United States
qylu@stanford.edu

***Abstract*—This paper explores the application of residual learning and proposes a minimal residual network (mResNet) in the classification of log-gradient (LG) images. ResNet-10, 18, and 34 are also implemented as comparison. All ResNets are trained and tested on four categories: person, car, bicycle, and bus. The training and test sets are generated based on Pascal VOC, consisting of 6433 and 2388 images, respectively. To further reduce the data volume, some modifications of datasets are used, including resolution, numbers of RGB layers, LG filters. Adversarial images are also generated to examine the robustness of the proposed mResNet. Because of the insufficient accuracy, this paper also performs the error analysis and discusses future work in details.**

Keywords— Classification, convolutional neural network (CNN), generative adversarial network (GAN), logarithmic-gradient, residual network (ResNet),

I. Introduction & Related Work

In image sensing, neural networks (NNs) have been applied for decades due to its good performance. Conventionally, NNs are implemented with cameras that measure static intensities. The intensity-based cameras, however, are sensitive to motion and noise, and thus brings unwanted issues. Fortunately, logarithmic-gradient (LG) cameras was proposed [1]. By the measuring static gradients instead of static intensities, LG cameras can effectively reject motion and reduce noises compared with conventional ones. Additionally, calculating log-gradient and quantizing results in a reduced requirements on the dynamic range of subsequent hardware, which means a potential cut on costs.

With the popularity of these cameras, various algorithms including NNs have been proposed and proven effective to improve the performance of sensing [2]. In spite of successful applications, new challenges keep emerging. Among many scenarios, the deployment at the edge has attracted extra attention. For edge devices, the requirements on power are quite stringent. Although deep NNs succeed in many scenarios, they cannot fulfill expectations in this case because of its energy-hungry nature. The research on this specific topic is far from sufficient. In other words, carefully designed and tuned, shallow NNs trained on log-gradient images are warranted. The primal challenge is the great loss of information. As shown in Fig. 1, only the some features of the picture are extracted and reserved after taking log-gradient. The subsequent quantization will perform further reduction on effective information.

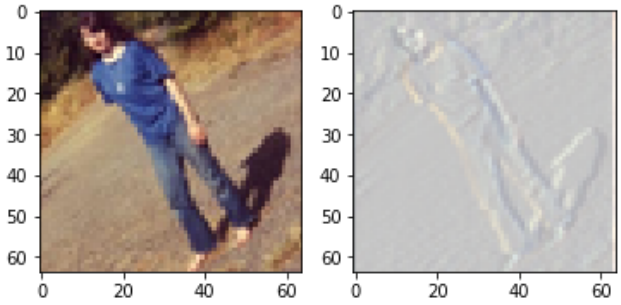


Fig. 1. RGB image and its log-gradient version.

Aiming at image sensing, this project explore the feasibility of residual networks (ResNets) in classification of LG images [3], and proposes a minimal ResNet (mResNet) as well as an adversarial network. Datasets of four categories: person, car, bicycle, and bus, are generated based on Pascal VOC. The ultimate goal of LG sensing is the reduction on data volume, which means a high dependence on the algorithms used to generate datasets. Therefore, this work also performs extra modification on datasets, e.g. down-sampling. Additionally, different numbers of RGB layers are also experimented to seek possible, extra reduction on image information. To summarize, ResNets are expected to correctly classify LG images in spite of the great loss of information. In this project, ResNets are trained from scratch to see if simple architectures are qualified.

II. Dataset and Features

In this section, details of datasets are discussed, including the generation and modification.

A. Pascal VOC

Since there are no log-gradient datasets available online, this project creates datasets based on Pascal VOC [4], which considers the prospect of autonomous driving. In preprocessing,

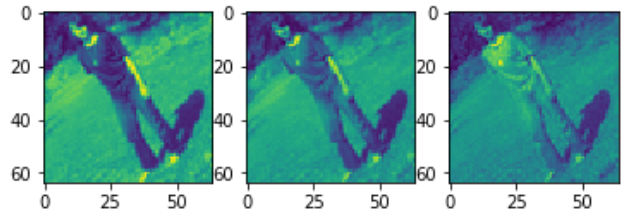


Fig. 2. R, G, B layers of an image.

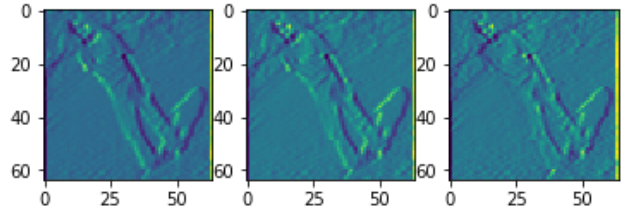


Fig. 3. R, G, B layers of an image after log-gradient.

images are cropped and resized. First, annotation files are used to segment original photos into smaller images that contain only one of the four categories: person, car, bicycle, and bus. Quantities of images are different and proportional to the frequency of them appearing in roads as listed in Table I.

B. LG Image Generation

Based on RGB pixels, the logarithmic-gradient of frames are then calculated by taking log of pixel values first:

$$P' = \log P \tag{1}$$

Here, P denotes the array of the pixel values in an image. Then, apply horizontal and vertical filters, as expressed by (2) and (3), respectively [2]. These two filters are similar to edge detectors in CNNs and extract edges as shown in Fig. 1.

$$h = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \tag{2}$$

$$v = [-1 \quad 0 \quad 1] \tag{3}$$

Take h as an example, G is the gradient array after filtering:

$$G = P' * h \tag{4}$$

G has the same dimension as P'. Next, the log-gradient images are exponentialized to make the features more prominent against background noise. Normalization and rescaling is involved in the subsequent steps, which is not introduced here.

C. Model Architecture

To avoid high bias in image classification, effective CNNs have complex architectures and are often very deep. However, deep NNs are hard to train. To resolve this issue, residual learning has been put forward which adds extra connections between certain layers and eases the training process [3]. The idea of adding shortcuts is efficient and has been adopted to develop new model architectures such as DenseNets [5]. This project utilizes the ResNet as the main architecture.

TABLE I. TRAINING AND TESTING DATASETS

	Pascal	**Image Distribution**				
	Datasets	*Person*	*Car*	*Bicycle*	*Bus*	***Total***
Train	VOC2012	2501	2447	817	668	6433
Test	VOC2007	1001	972	260	155	2388

TABLE II. 3-LAYER VS 1-LAYER LOG-GRADIENT DATASETS

		Dim. of Train Set	**Dim. of Test Set**
3-layer (R, G, B)	*Log-gradient*	(6433, *w*, *h*, 3)	(2388, *w*, *h*, 3)
	Original	(6433, *w*, *h*, 3)	(2388, *w*, *h*, 3)
1-layer (R only)	*Log-gradient*	(6433, *w*, *h*, 1)	(2388, *w*, *h*, 1)
	Original	(6433, *w*, *h*, 1)	(2388, *w*, *h*, 1)
2-layer (*h* +*v*)	*Log-gradient*	(6433, *w*, *h*, 2)	(2388, *w*, *h*, 2)

[a.] (*w*, *h*) = (64, 64) or (128, 128)

III. METHODS

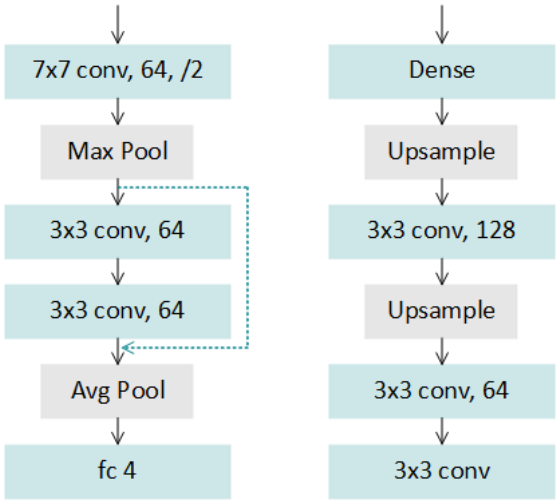
In this section, the modification of datasets and model architectures are discussed in details. This project is implemented on TensorFlow Keras.

A. Dataset Modification

Log-gradient is proposed especially for the reduction on the input dynamic range of hardware (e.g. image sensing chips) which can result in significant compression in data volume. In other words, LG is to reduce computational complexity because only the key features of images are extracted and reserved. Therefore, it is meaningful to explore possibilities to further reduce the data volume. In light of this, this paper tries reasonable modifications such as down-sampling and slicing. By down-sampling, the resolution is decreased from (128, 128) to (64, 64). By observation, color layers, i.e. R, G, and B, contains similar information, as shown in Fig. 2 and Fig. 3. Therefore, it is also worth trying only one layer of images. In Fig. 2, three layers of the image, R, G, and B, are plotted separately. Color layers after taking log-gradient are displayed in Fig. 3. Obviously, subplots are similar. In this project, R layer is taken and LG filter h is chosen as the main filter for experiments. Additionally, another dataset is created which combines the horizontal filter h with the vertical filter v, acting as a case of trading off between the accuracy and computational costs. Cases explored by this project is listed in Table II.

B. ResNets and Further Simplification

This project makes use of ResNet-50 from assignments [6]. ResNet-50, however, seems too deep to deploy at edge devices. Thus, ResNet-34, 18 and 10 are also implemented and further simplified into a minimal ResNet (mResNet). As shown in Fig.



(a) (b)

Fig. 4. (a) mResNet, (b) GAN.

TABLE III. TEST ACCURACY OF RESNETS

Datasets (image dim.)	**LG3** (*w*, *h*, 3)	**LG1-*vh*** (*w*, *h*, 2)	**LG1-*h*** (*w*, *h*, 1)
ResNet-50	40.67%	47.75%	40.93%
ResNet-34	75.45%	76.07%	71.59%
ResNet-18	61.12%	66.65%	56.59%
ResNet-10	80.10%	81.27%	77.62%
mResNet	**84.24%**	**82.69%**	**80.47%**

4(a), ResNet-10 is rather compact, but mResNet has a simpler architecture, making it much easier to deploy at the edge. The performance of these ResNets are discussed and compared in Section IV.

C. *Adversarial Network*

A generative adversarial network (GAN) for the proposed minimal ResNet is also implemented [7]. Similar to mResNet, this GAN has a simple architecture as shown in Fig. 4(b). The propose GAN can generate images of "non-person" that are classified by mResNet as "person", which poses a great threat to mResNet.

IV. EXPERIMENTAL RESULTS

In this part, experimental results for different parameters, different ResNets, and the visualization are presented. The primary metrics are accuracy and computational cost.

A. *Parameters*

Among different initializers, "he initializer" was chosen for experiments. Additionally, much time was devoted to find optimal batch size and epoch to train the model. It turns out that for deep NNs such as ResNets-34, a small epoch such as 30 is sufficient. Otherwise, it will end up overfitting. Effects of batch size and epochs on training time and accuracy are not orthogonal/independent.

Fig. 5. Original RGB "bicycle" and its adversial image.

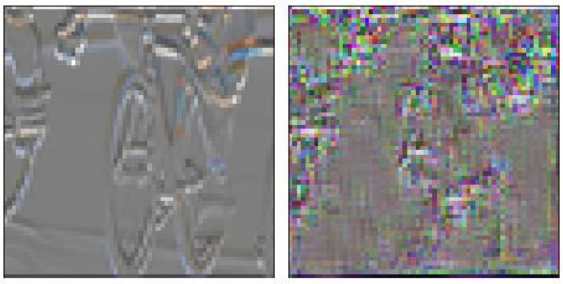


Fig. 6. LG3 "bicycle" and its adversial image.

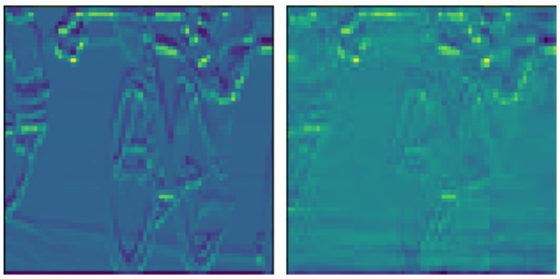


Fig. 7. LG1-*h* "bicycle" and its adversial image.

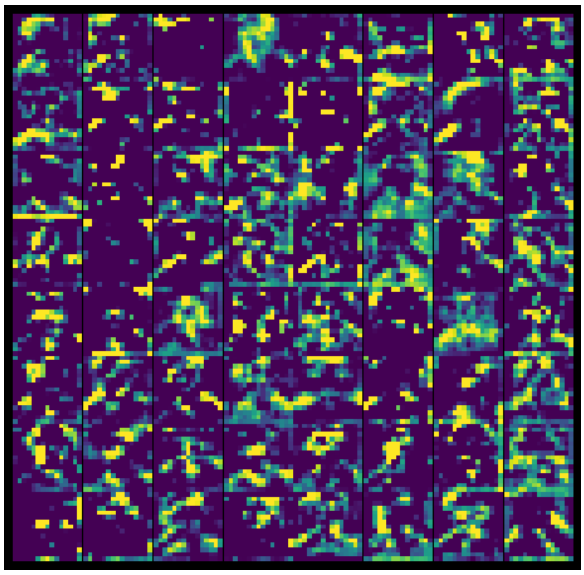
B. *Different ResNets*

First, ResNet-50 is tested as a baseline. As displayed in Table III, the proposed mResNet outperforms others on three LG datasets and are thus selected as the main architecture. It is worth mentioning that, mResNet is effective in this project because of the simple datasets which only contains four categories and uses low resolution images. If much more categories and high definition pictures are involved, mResNet will probably not qualify.

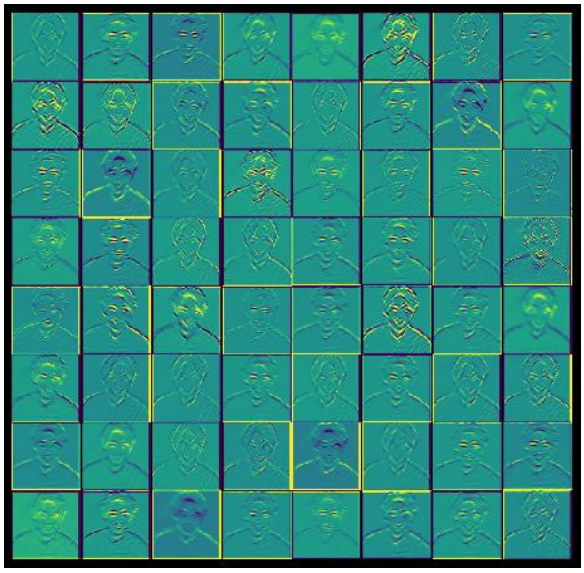
C. *Different Modifications of Datasets*

Table III lists the accuracy in three cases, respectively. "LG3" denotes the log-gradient 3-color-layer images; " LG1-*h*" represents 1-color-layer dataset based on filter *h*, while " LG1-*vh*" combines filters *v* and *h*. According to the results of mResNet, the accuracy decreases with the image dimensions, indicating the trade-off between model performance and computational complexity. However, it is very promising since the accuracy almost remains the same level.

(a)



(b)



(c)

Fig. 8. (a) Original & LG images, (b) activation map of an early layer, (c) activation map of a later layer.

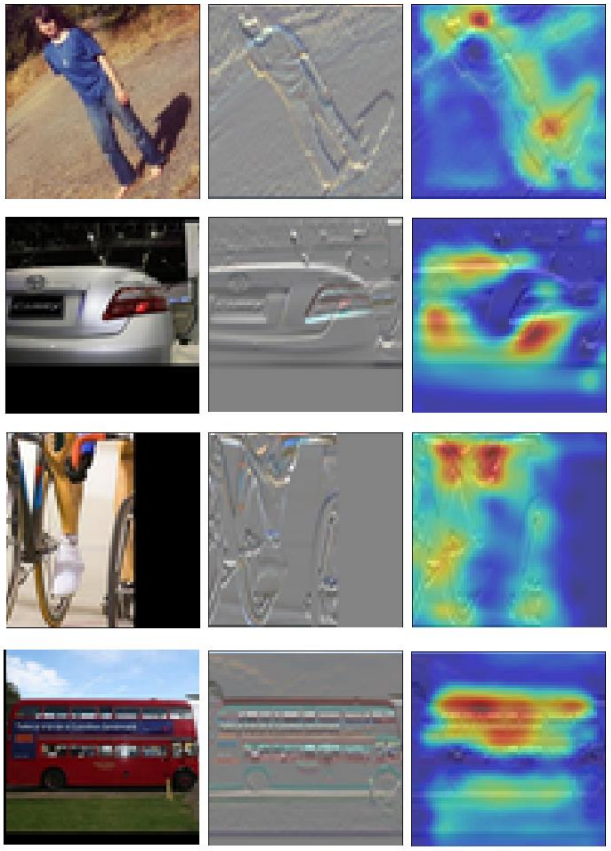


Fig. 9. Original images, LG images, and heatmaps of mResNet that is trained on LG datasets.

TABLE IV. ERROR DISTRIBUTION

Category	# of error	e in test set	e in each class
Person	70	2.93%	6.99%
Car	130	5.44%	13.37%
Bicycle	106	4.43%	40.77%
Bus	70	2.93%	45.16%
Total	376	15.75%	/

D. Adversarial Examples

Fig. 5, 6, and 7 show the examples of adversarial images of a bike which are classified as “car” instead.

E. Visualization

For better understanding of the model, activation maps and heatmaps are plotted [8][9].

a) Activation map: In NNs, early layers often learn simple features such as edges, and later layers can identify complex patterns like human faces in Fig. 8(c). In my case, what is detected in Fig. 8(b) is difficult to identify becuase much information is lost by taking log-gradient. However, it is still obvious that later layers are detecting human faces.

b) Heatmap: Fig. 9 shows the heatmap of four examples. For bicycles, it seems mResNet is dectecting the handle bar.

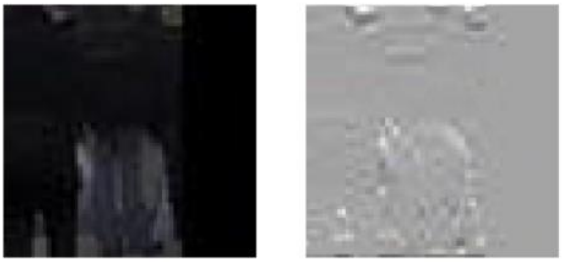


Fig. 10. An image that is hard to classify. ("Person")

Fig. 11. Black bar in an image.

F. Error Analysis

Table IV shows the distribution of misclassified examples in the test set. Although the quantities of errors in "person" and "bus" are the same, their error rates in each class are significantly different. According to Table IV, mResNet is not good at recognize bicycles and buses. Obviously, the accuracy is insufficient, the probable causes of which is discussed as follows.

a) Human error: Bayes error is this case is unknown, therefore the human error is considered for reference. Some images are difficult to classify even for human eyes, as shown in Fig. 10. Unfortunately, the quantity of these images is not ignorable.

b) Black bar: Since the images are cropped based on larger ones that contain multiple categories, it is inevitable that some of them are at the edge of the original image, as displayed in Fig. 11. In this case, the generated image is padded with black pixels. The quantity of these images is not ignorable and thus damages the learning of neurons.

c) Image format: Pascal VOC images are ".JPEG", which is a "damaged" version of RAW images: for example, the white balance and the color space are modified. Under this circumstance, significant amount of information is already lost and taking gradients will speed up the deteriation of accuracy. Moreover, little useful information is left after further downsampling and slicing. During experiments, some LG images turn out to be a grey block, no edges left.

V. Conclusion

In this project, mResNet is proposed to classify "person", "car", "bicycle", and "bus", which outperforms all other ResNets despite of its compactness. Additionally, an adversarial network is implemented which can successfully deceive the discriminator (mResNet). However, the project is far from perfect and much work remains to be done, as discussed in the following.

1) *Dataset:* Improving datasets is the primal task. First, the datasets is too small, and thus the NNs will usually end up overfitting. Second, the image quality is poor. As mentioned in Section IV-E, some original/LG images are hard for humans to classify. The resolution, i.e. dataset size, is limited by Colab and certainly needs improvements. Third , also mentioned in Section IV-E-c, the image format ".JPEG" is "damaged", it is necessary to switch to RAW images. (A dataset as small as 5, 000 RAW images will take up to hundreds of GB, which is very hard to train in Colab.)

2) *Transfer learning:*The only but nontrivial difference of this work from other CNNs is the distribution of input pixel values. In other words, it is reasonable to apply transfer learning here since low-level neurons might extract similar features. Note that, the last layers are usually removed in transfer learning; in this project, however, the first layers are the ones needs to be retrained or chopped off.

3) *Adversarial examples:* To confront the challenge posed by GANs, it is necessary to train mResNet on adversarial examples.

4) *Object detection:* Once the accuracy of classification is acceptable, the project can move on to the next stage: detection. Namely, try to detect "person", "car", "bicycle", and "bus" in an image.

Acknowledgment

This work is did as part of my research in Professor Boris Murmann's lab. I trained all models in Google Colab. Codes for cropping images are from Daniel Stanley, a student working on this topic previously. Those codes include clear statements inside the source file. Github repository of this project is: https://github.com/Qianyun-Lu/CS230_CNNs.

References

[1] J. Tumblin, A. Agrawal and R. Raskar, "Why I want a gradient camera," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San Diego, CA, USA, 2005, pp. 103-110 vol. 1.

[2] C. Young, A. Omid-Zohoor, P. Lajevardi and B. Murmann, "A data-compressive 1.5/2.75-bit log-gradient QVGA image sensor with multi-scale readout for always-on object detection," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 11, pp. 2932-2946, Nov. 2019.

[3] K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition," CoRR, abs/1512.03385, 2015.

[4] Pascal VOC: https://pjreddie.com/projects/pascal-voc-dataset-mirror/

[5] G. Huang, Z. Liu, L. v. d. Maaten and K. Q. Weinberger, "Densely connected convolutional networks," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 2017, pp. 2261-2269.

[6] Coursera, Convolutional Neural Networks: https://www.coursera.org/learn/convolutional-neural-networks?

[7] Keras-GAN: https://github.com/eriklindernoren/Keras-GAN

[8] Resnet visualization: https://github.com/m-peker/ResNet-Layers-Visualization

[9] Keras-Visualization: https://github.com/raghakot/keras-vis