Exploration Between Image resolutions and Classification

Abstract

- Traditional **Convolutional Neural Network** (CNN) is separating an image from other by adjusting the weights of multiple nodes. In specific, **Maximum Pooling Arithmetic** is selecting the most triggered nodes to linearly distinguishing one image to another. For classification, a node implies the **pixels intensity** of the input. Thus it raises the hypothesis: **could a scaled lower resolution of an image might cause difficulties for the classifier to properly label to a proper class**. This paper and implementation is a proof of concept toward this hypothesis.

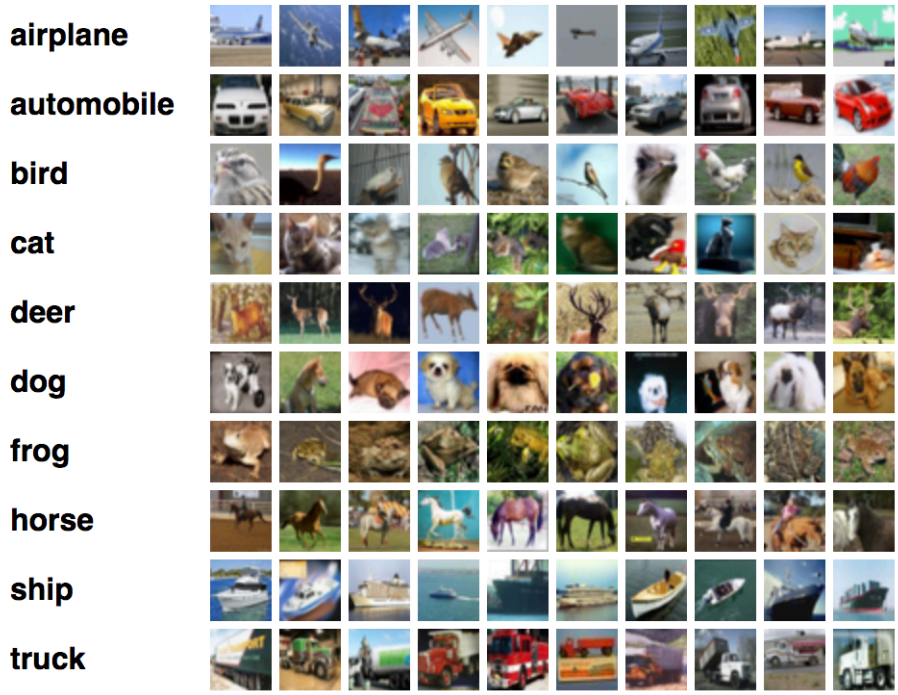
Project Overview

- MSRN stands for **Multi-Scale Residual Network**; it is a supervised learning model that learns to upscale the image from low resolution to a higher resolution in an arbitrary scale. This network serves a purpose of scaling an image to a given ratio by preserves the key features such as line and shape from a distorted image.
- The motivation for uses chose MSRN is that the model is multi-scaled. First of all, it is easy to reproduce the scaled SR output. Unlike most SR models which are sensitive to the subtle network architectural changes and highly rely on the network configuration. Secondly, it avoids inadequate features utilization. It enhances the performance not by blindly increasing the depth of the network. Instead, MSRN concatenates the layers and recomputed suitable pixel region to enhance resolution. Last but not least, it has good scalability. Therefore, this method is easy for us to observe the result and save the time of computation.
- Overall, there are three training steps. Firstly, the image is down-scaled and then up-scaled with the naive algorithm which results in a low-resolution image. Then feed it into classifier to predict the result. Secondly, the input is naive down-scale and feed into MSRN and trained. And the trained MSRN will be applied by generating SR images. SR images would then feed into classifier to train. By comparing the result of these classifiers, this paper can explore how much resolution improved on the same parameters classifier.
- Since the project consists of two parts, classification and super-resolution. Choosing a suitable dataset can lead concrete proof for the hypothesis.

Datasets

- The dataset has been trained is an annotated image dataset known as the **CIFAR-10 dataset** 2. It contains ten classes of images. As **figure 0.1** as shown, the dataset has

classes of airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The CIFAR-10 contains a total of 50,000 training images and 10,000 test images. Furthermore, the training sets contains exactly 5000 images for each class2.



-
  - ***figure 0.1 cifar 10 datasets*** *it smaple of the cifar 10 dataset*
- The motivation for choosing this dataset is that each object is clearly distinguishable by other class. It evades the concern of illumination, deformation, and occlusion of the image. For example, the important feature points that identify bird is clearly distinguishable than the key feature points of a truck. The learning curve for training such classifier is less computationally expensive than other specific class driven datasets.

- Therefore, this experiment was conducted on training an identical densenet 6 with two different of inputs, naive resized images and output images of MSRN. The evaluation metrics of the classifiers is the test error and test loss. In this context, feeding naive and super resolutions images can help this paper to conclude the impact of resolution on classification regarding test accuracy and test loss.

1. Approache Analysis

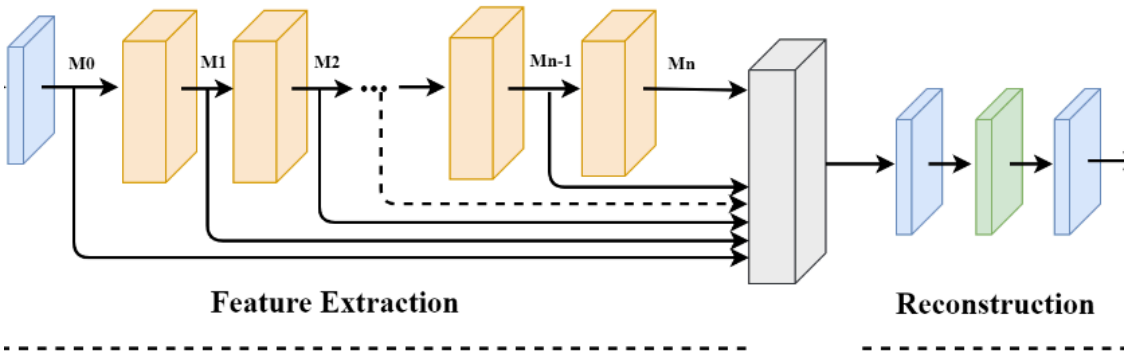
- All of the implementation and running instruction are in this repository

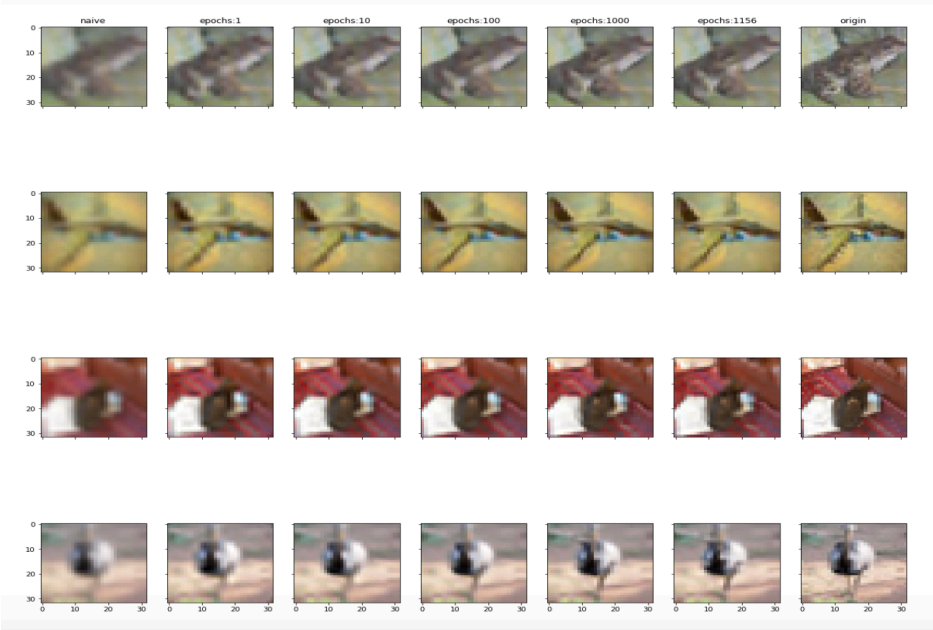
---

1.1 MSRN Scaled Image preprocessing

- The implementation of this approach can be looked up here

- MSRN is built by **Multi-Scale-Residual Block** (MSRB) which consists of multiple residual network blocks. MSRB performs lower resolution feature extraction on **y** channel from the input that is in color space **yCbCr**3. Then concatenated all of the filters and fed to a **Sub-Pixel Convolutional layer** to reconstruct a higher resolution image. Sub-pixeling Convolutional Neural Network is a network structure learning to upscale the lower resolution image to a higher resolution output by estimating ratio pixel arangement4. This enables MSRN to learn to generate a higher resolution output based on ground truth.



  - ***figure 1.0**, this show the architecture of MSRN it consists of n block of MSRB and reconstructed by a sub-pixel convolutional layer in reconstruction layer*
- The **blue block** in **figure 1.0** is 64 channels of Convolutional Layers and **orange block** is MSRB block. All of $M_1..M_n$ will be concatenated at the gray block to linearized by feeding into another **one kernel Convolutional** layer. This allows the network to learn the distinct pixel region from all of the previous filters. Finally, the output would feed into the **Sub-Pixel Convolutional Layer** to recreate the higher resolution output.
- 

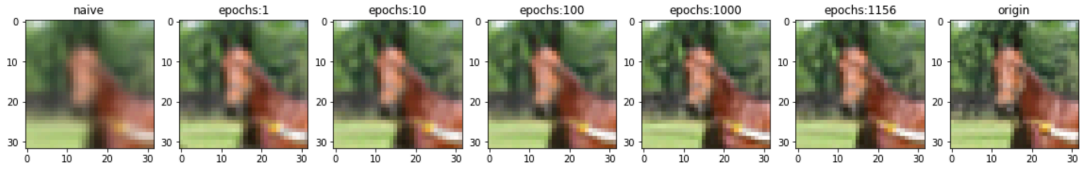
- *msrn result*

- ***figures 1.1**,image super resolution result by MSRN*

- The following function L is the loss function of training an MSRN. I_i^{LR} is y channel of **naive downscaled** images in $yCrCb$ colorspace. I_i^{HR} is y channel of the **original image** in $yCrCb$ and F_θ is the forward pass of the MSRN. The motivation for choosing origin image is that it is impossible to generate any image which has high resolution than origin input. Thus it would allow the network to generate output images that are infinite closes to origin image. The cost function can help MSRN to learn to evaluate how far are the features between pixel spaces which allow back propagation to derivate gradient to optimizes the output.

  - $L(F_\theta(I^{LR}, I^{HR})) = \sum_{i=0}^{n} ||F_\theta(I_i^{LR} - I_i^{HR})||_1 3$

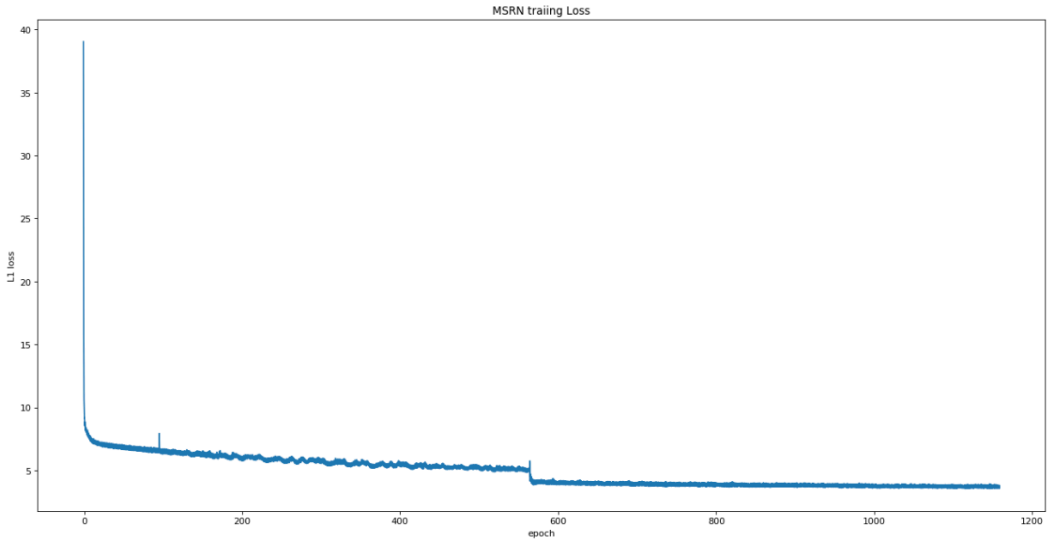
- With the back propagation, the network is able to adjust the weights to generate an image output that is closer to I^{HR}. Super resolution enables application of resizes and scaling of an image while preserving the key details of an object. In this experiment, MSRN is trained with 49,000 training images with 1,000 of test images. With `L1loss` (2) function, MSRN is able generated feature preserving output.



-
  - ***figures 1.1**, this show the learning steapness of **MSRN**. Left most image is traditional algorithm approaches. It show a horse image that generated by msrn in different epoches.*

- As the figure 1.2 as shown, the resolution of the output is slowly increased along with the epochs size. The output becomes smoother as the network was learned. It starts to improve the faces region of the horse from the input. This shows the outstanding performance of deep learning method in image template matching. It only requires one epoch to generate an image that can outperform traditional approach. However, as network is trained with more epochs, it becomes difficult for the network to learn.

- In **figure 1.2**, the downslope of `l1 loss` appears slower than previous epochs. It implies the network is struggling to learn due to backward gradients is too small for the model to update it. Another challenge is the network can't perform data normalization and whitening before training. For example, perform back propagation on a value between -1 to 1 is faster than 0 to 255. It takes 1000 epochs to deduces the loss to close to 5. As **figure 3.1** the images between epoch 100 and 1000 are not visually different.



-
- *naive approach*
  - ***figures 1.2**, this show the learning curve of **MSRN**. it plot against the epochs vs `L1 Loss`.*
- The network is trained with 64 batch size on `1080Ti GPU` for two days. Although the training procedure takes really long, it takes really fast for a network to predict during the demo. It is unquestionable that MSRN is challenging to train, exploring different scaling method can enhance the need for higher resolution. This paper will further explore a traditional preprocessing approach in terms of time complexity and implementation details.

1.2 Naive Resized Image preprocessing

- The implementation of this approach is here

- Naive often refer brute forces to solve the given problem. However, in this given approaches naive refer as `cv2.resize` API in openCV. The goal is applying a different geometric transformation to images like scaling by reprojecting image points on a different plane. It also preserves the key features points like ratios of distances between pixels.
- The algorithm started with input image X has shape of (h_x, w_x) and output y image is (h_y, w_y). First compute the scaling factor by computing $(\frac{h_y}{h_x}, \frac{w_y}{w_x})$ and mulitplied the identiy matrix I_n obtain the scaling matrix $M = \begin{bmatrix} \frac{h_y}{h_x} & 0 \\ 0 & \frac{w_y}{w_x} \end{bmatrix}$ then computed $f: X \rightarrow (Mx)I_{|X|} \rightarrow y$ to obtain image y' 5.
- In this approach, training and testing images will be processed by this algorithm twice. First downscaling the image by half and upscale once to origin sizes which helps the

input preserve the same size during classification. This allows this project to explore the result of classification with different scaling preprocessing.

-
  - ***figures 1.3****: visualizing the result and input of the image*
- With above mentioned, the algorithm is performing three matrix multiplications, so the running time complexity of this algorithm is $O(n^3)$.

1.3 Classifier

- The implementation details of the classifier is in here

- The classifier model was used in this project is **DenseNet**. It stands for **Densely Connected Convolutional Networks**. The motivation of this network building a performance driven classifier by constructing with a lot of convolutional layers 7.
- Two DenseNet was trained with different preprocessed images, one is super resolution and another is naive preprocessing. Both models are trained in 49000 training set and 1000 test set in 20 epochs. Classifying images are the key evaluation of previous two approaches. This section would discuss details of implementation in terms of evaluation metric, depth of the model and hyper parameters.
- Classification model often evaluated based on **error rate** and **loss**, in this experiment, the paper would describe the details of each metric and structure of the model.
- The error rate is evaluating the model with a test set based on the percentage that is predicting false positive and true negative. Assume $F(x_i) = y_i'$ is forward propagated function of DenseNet and the predicted label of x_i. Given the test set is set of images x_i with label y_i such that $1 \leq i \leq m$, the error rate is computed $\frac{1}{m}\sum_{i=1}^{m}(0, y_i' \neq y_i)$. Since the test set is not been backward propagated, it allows this experiment to evaluate the performance of the classifier.
- The loss of the DenseNet is Negative Log Likelihood loss (NLLloss) 8. This loss function is motivated by Bayes Theorem and Logistic regression to match non-linearly separable data in multi-dimensional data 8. Given batch size of N, the **NLLloss** is $l(x, y) = \sum_{i=1}^{N} - [log(p(x_i|y_i) - log(x|y_i')]$. This computes the associated probabilities that how close between prediction and label.

- The paper will further explore the result of the combination of all of the previous section. The following section will compare resolutions of each preprocessing methods and analyzing the result of by feeding the images into DenseNet in two of the formats.

2. Result & Analysis

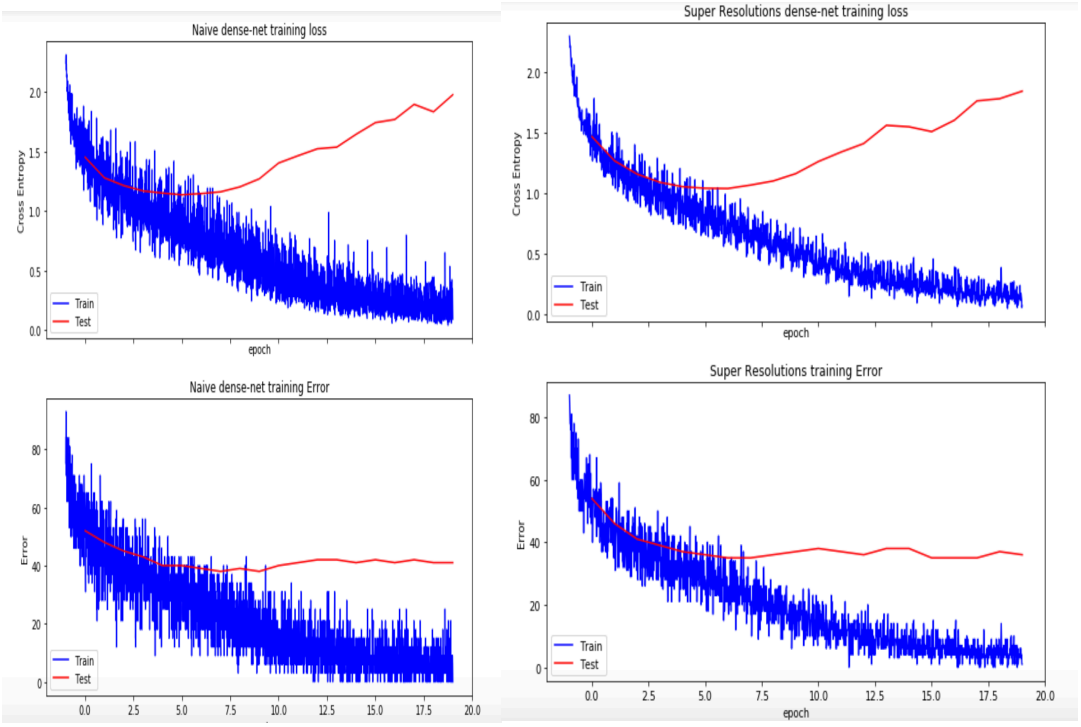
- The demo is in here

2.1 Resolution Comparison

- **Figure 2.3** shows the preprocessing of three test images in terms of the traditional algorithm approach, super resolution and origin image. In column `Naive`, the algorithm causes the input image to lose some important resolutions, including the legs and head of the deers. In comparison, the super resolution image appears more detailed and smoother toward the origin image.

  - ***figures 2.3**, comparison of resolutions between traditional approaches and super resolutions*

2.3 Classification



- – ***figure 2.4**, training loss and error rate of DenseNet in two approaches `naive` and `super resolutions`*

• Evaluation Metrics	• Naive Preprocessing	• Super Resolution
• minimum Error Rate	• 38%	• 35%
• minimum nllLoss	• 1.135	• 1.038

- – ***table 2.4** indicating the loss and error rate of the two approaches*

- In **figure 2.4**, the widening gap between loss indicates the naive preprocessing has higher variance than super resolution. This implies the network is more struggle to learn which are the key features that can distinguish between the inputs.

- Furthermore, the network appears overfitting, the test loss in epoch 5 starts to deviate from training loss. There might be a potential improvement by adding regularization and dropout layers in the classifier.

- As **table 2.4** shows, the test loss and error rate of super resolution preprocessing is lower than naive preprocessing. This concludes the importance of feature details for image classification, it motivates computer scientists to invent or increment the ability for the neural network to generate a higher resolution image.

3 Concolusion

- Through the previous section, it appears the network is able to learn faster from a higher resolution image than lower resolution. In conclusion, the resolution might impact the performance of a classifier in terms of prediction accuracy and training efficiency.

References

1. Etten Adam Van. Quantifying the Effects of Resolution on Image Classification Accuracy. Medium.com. retrieved from https://goo.gl/v2xa2T

2. Alex krizhesky, Vinod N, GEoffrey H. The CIFAR-10 dataset, Univeristy of Toronto. retreived from https://www.cs.toronto.edu/~kriz/cifar.html

3. Juncheng Li,FF , KM , GZ. Multi-scale Residual Network for Super Resolution. The European Conference on Computer Vision 2018. Retrieved from http://openaccess.thecvf.com/content_ECCV_2018/html/Juncheng_Li_Multi-scale_Residual_Network_ECCV_2018_paper.html

4. Wenzhe Shi, JC, FH, JT, AP. Real-time single image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. Twiter. Retrieve from https://arxiv.org/abs/1609.05158

5. *Hazewinkel, Michiel*, ed. (2001) [1994], *"Affine transformation"*, *Encyclopedia of Mathematics*, Springer Science+Business Media B.V. / Kluwer Academic Publishers, *ISBN 978-1-55608-010-4*
6. E. Pang. University of Toronto. An Efficient Implementation of Affine Transformation Using One-Dimensional FFT's. Retrieved from: https://ieeexplore.ieee.org/document/595392/

7. Gao Huang, Zhuang Liu, LM , KQ. Cornell University. Densely Connected Convolutional Networks. Retrieved from https://arxiv.org/abs/1608.06993

8. Negative Log Likelihood Ratio Loss for Deep Neural Network Classification retrieved from https://arxiv.org/pdf/1804.10690.pdf