Application of Large-scale Multi-class Image Classification

CS689

Nan Zhang

Abstract: Image classification is a complex process which depends upon various factors. In this paper, I reviewed the current activity of image classification methodologies and techniques, including convolution neural network(CNN) and residual neural network (resNet) methods. In addition to the prospects of image classification, I discussed the present techniques and issues. The focus will be on newer classification techniques which were developed to enhance pre-existing classification techniques. Moreover, an experiment was designed with colleagues to apply different neural networks to an image classification competition, and our current result achieves 70% accuracy in Kaggle leader board.

Introduction:

In everyday life of the present, classification helps us in to make decisions [7]. In the future, classification and recognition technology will be a basic milestone for robotic development. The need for classification arises whenever an object is placed in a specific group or class, depending upon the attributes corresponding to that object. Classification problems were addressed in many industrial problems. Researchers have developed innovative methods for enhancing classification accuracy. Numerous images are produced every day, which makes classifying necessary so that it's easy and fast to access. During the classification, information processing helps categorize images into various groups—for example, stock market prediction, weather forecasting, bankruptcy prediction, medical diagnosis, speech recognition, character recognition, etc. In these areas, classification problems can be solved mathematically and in a non-linear function.

Basically, accurate identification of the features present in an image is the major objective of image classification. There are two kind of classification categories: supervised classification and unsupervised classifications. For supervised classification, we make use of a labeled, trained database along with human intervention. In the case of unsupervised classification, inferences are drawn from unlabeled data. Here we focus more on supervised classification, as there are many labeled datasets available, such as MINST and ImageNet. With these labeled images as references, software can be trained to classify other images.

Problems states in CS231N [9]:

Problems in current image classification include
Viewpoint variation: an object is photographed from different angle.
Scale variation: different objects may scale different.
Deformation: many objects can be deformed in extremely way.
Occlusion: the image only shows the portions of the objects.
Illumination conditions: computer may be confused about the extremely bright and dark condition of pixels.
Background clutter: the object itself may blend into the background, making it difficult for the classifier to identify the object.

Intra-class variation: many different types of objects, each with its own design.

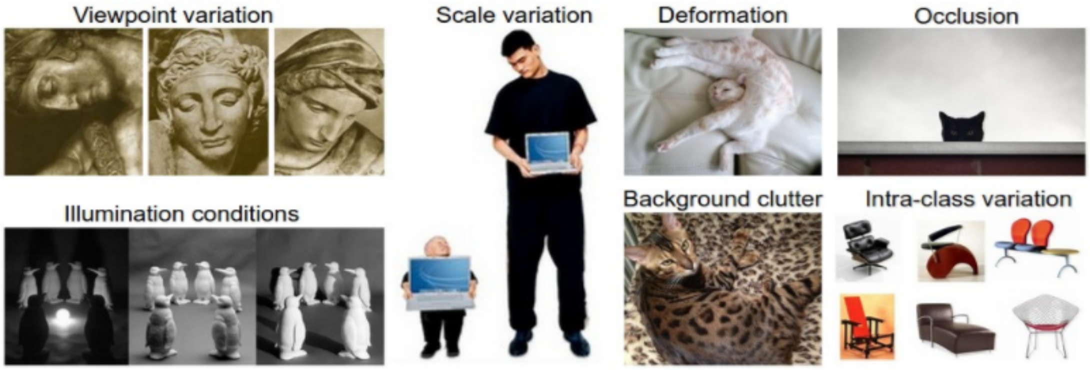


Figure 1: Variation Examples.

Research Objectives

In general, the research objective is to design an image classification model or neural network to classify different images. The model will be given many examples of each class. An algorithm will be designed to learn from these given examples the graphic appearance of each class. Figure 2 is an example of a dataset [9]:

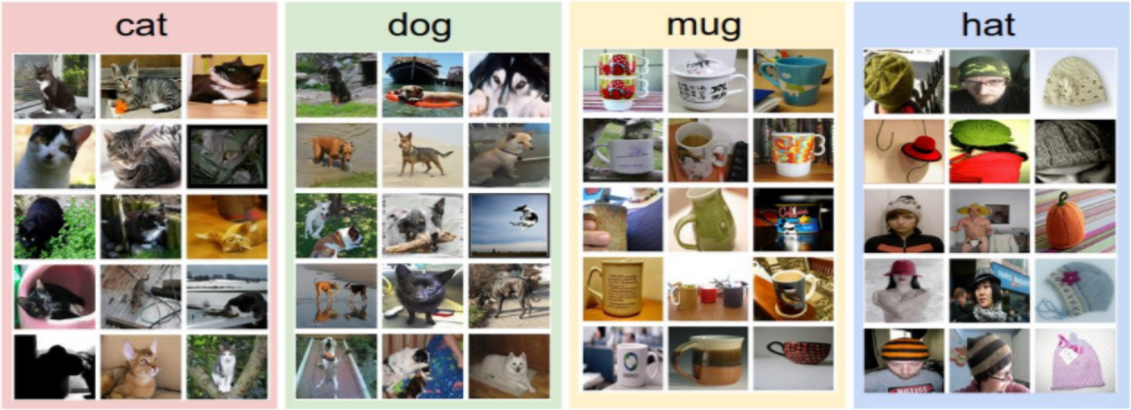


Figure 2: Example of dataset.

Since an image is an array of pixels in computer vision, the input of the classification model is a set of N images, with each of them an array of pixels. Each image will have a label that contains K different classes. The pipeline could be formalized as below [9]:

- Input: Input consists of a set of N images, each labeled with one of the K different classes. This data is also called a training set.
- Learning: The task is to use the training set to learn every class. This step is also called training a classifier, or learning a model.

- Evaluation: The end goal is to analyze the accuracy of the classifier by asking it to predict labels for a completely new set of images, then compare the correct labels of these images to the ones predicted by the model.

Literature Review

ImageNet Classification with Deep Convolutional Neural Networks [1]

1. Research Overview

This paper is the first one to introduce the deep Convolution Neural Networks(CNNs) [6] to the computer vision field. The researchers trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into 1000 different classes. The test data achieved top-1 error rates of 37.5%, and top-5 error rates of 17.0%, which is noticeably better than the previous work. The neural network contains 60 million parameters and 650, 000 neurons, with five convolutional layers. Some of them are followed by max-pooling layers, and three fully-connected layers that have a final 1000-way softmax output.

2. Design

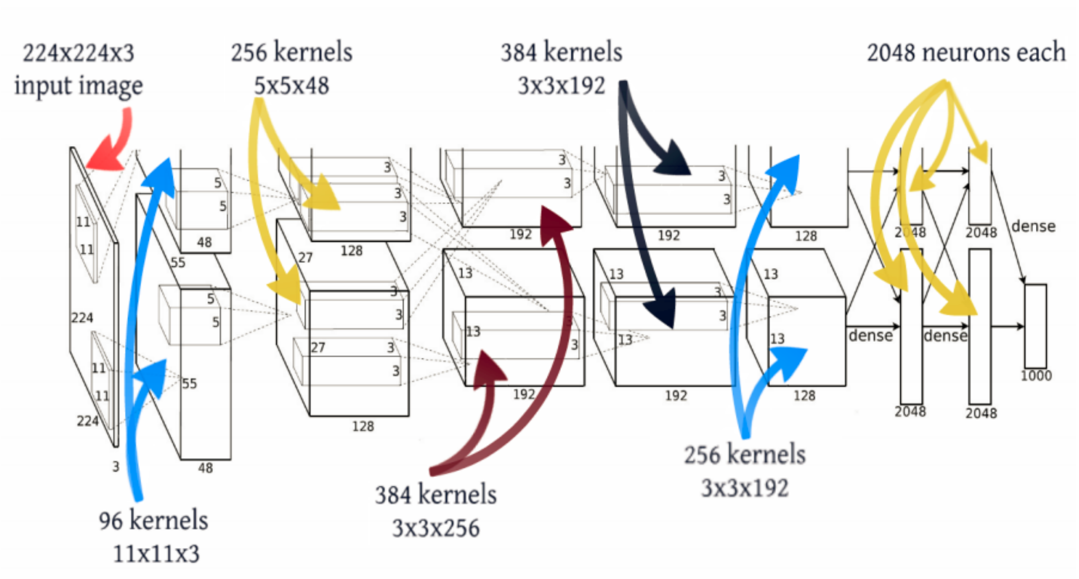
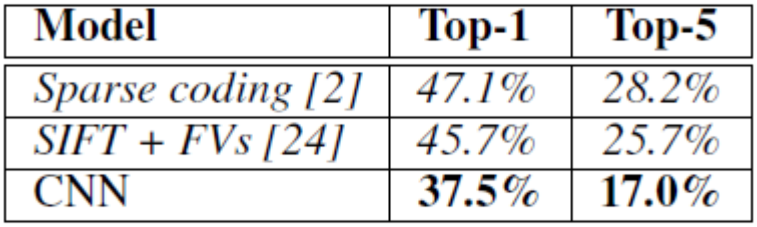


Figure 3: An illustration of the architecture of CNN.

Their network has eight layers with weights; the first five are convolutional and the remaining three are fully connected. The last fully connected layer is a 1000-way softmax that creates a distribution over a thousand class labels. The first convolutional layer uses 96 kernels of size 11×11×3 to filter the 224×224×3 input image. The stride is 4 pixels. The second convolutional layer takes the output of the first convolutional layer as input and uses 256 kernels of size 5×5×48 to filter. Layers are connected to each other. The third layer is normalized and pooled. Each fully-connected layers contain 4096 neurons.

3. Result

Their results on ILSVRC-2010 are summarized in Figure 4. Their network achieves top-1 and top-5 test set error rates of 37.5% and 17.0%.



Model	Top-1	Top-5
Sparse coding [2]	*47.1%*	*28.2%*
SIFT + FVs [24]	*45.7%*	*25.7%*
CNN	**37.5%**	**17.0%**

Figure 4: Comparison of results on ILSVRC-2010 test set.

4. Weakness in Design

They introduce many new functions, such as non-linear ReLU active function and local response normalization, and give empirical proof of their improvement by these changes. But they only show when they apply one of new features and the subsequent accuracy improvement, rather than combining them to explore a different result. That means they only have one Control group for all experiments to test if the feature is helpful or unhelpful, thus even though their experiment shows each of the new feature separately is helpful, they didn't prove the combination of new features have better performance.

Visualizing and Understanding Convolutional Networks [3]

1. Research Overview

Since CNNs is just like a black box in neutral network design, researchers from NYC introduced an original visualization technique, which gives details into the function of layers, and the classifier's operation. They were able to find model architectures that performs better than Krizhevsky et al. on the ImageNet classification benchmark. In addition, an ablation study was performed to find out the performance contribution in different model layers. Their ImageNet model generalizes well to other datasets too: it convincingly better than the current state-of-the-art results on Caltech-101 and Caltech-256 datasets, when the softmax classifier is retrained.

2. Design

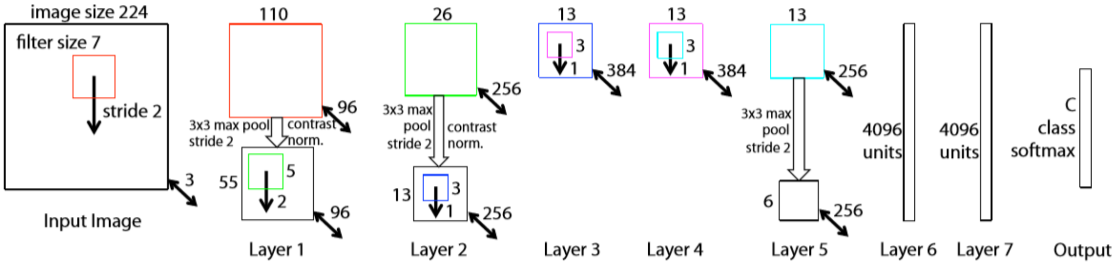
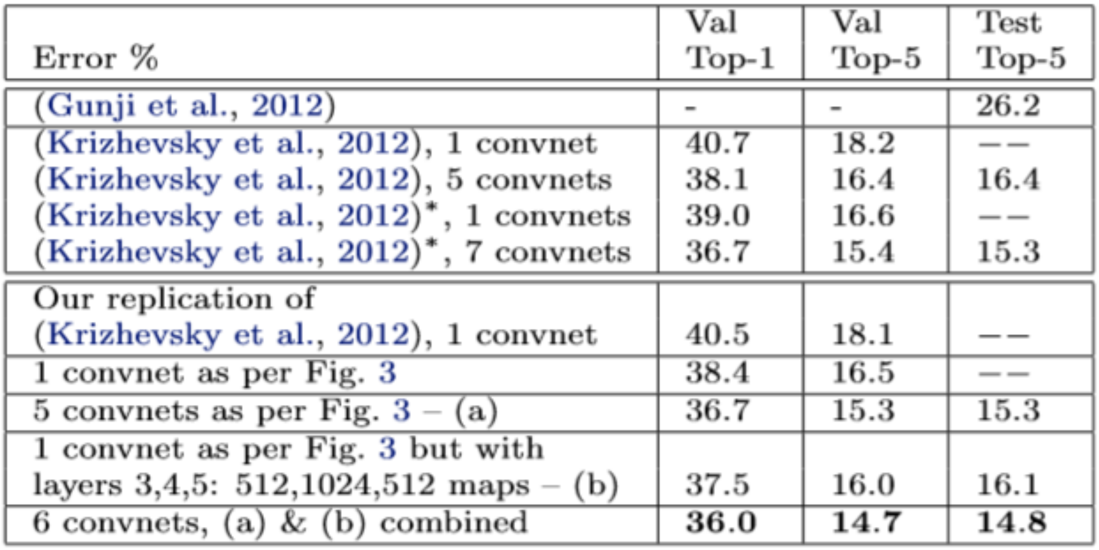


Figure 5: Architecture of the 8 layer convnet model.

Figure 5 shows architecture of their 8-layer convnet model. Input is a crop of 224 x 224 x 3 image. It's convolved with 96 different first red layer filters, the size is 7 x 7 for each of them, and has a stride of 2 in both x and y. In each layer, the kernel passed to rectified linear function, pooled, and contrast normalized feature maps. Only the last two layers are fully connected, and the features are taking from the top convolutional layers as input in vector. The final layer is a C-way softmax function, and C is the number of classes. The shape of all filters and feature maps are square.

The architecture is similar to that used by [1] (Krizhevsky et al., 2012) for ImageNet classification. One difference is that the sparse connections used in Krizhevsky's layers 3,4,5 (because the model being split across 2 GPUs) were replaced with dense connections in NYC's model. Another significant difference is relating to layers 1 and 2 were made to change convolution kernels or filter size and stride.

3. Result



Error %	Val Top-1	Val Top-5	Test Top-5
(Gunji et al., 2012)	-	-	26.2
(Krizhevsky et al., 2012), 1 convnet	40.7	18.2	−−
(Krizhevsky et al., 2012), 5 convnets	38.1	16.4	16.4
(Krizhevsky et al., 2012)*, 1 convnets	39.0	16.6	−−
(Krizhevsky et al., 2012)*, 7 convnets	36.7	15.4	15.3
Our replication of (Krizhevsky et al., 2012), 1 convnet	40.5	18.1	−−
1 convnet as per Fig. 3	38.4	16.5	−−
5 convnets as per Fig. 3 – (a)	36.7	15.3	15.3
1 convnet as per Fig. 3 but with layers 3,4,5: 512,1024,512 maps – (b)	37.5	16.0	16.1
6 convnets, (a) & (b) combined	**36.0**	**14.7**	**14.8**

Figure 6: ImageNet 2012 classification error rates.

The final result indicates they achieve a better Top-1 error rate compare to [1].

4. Weakness in design

They built a similar model as [1], and use a visualization method to change their parameters to see if there was any improvement in the prior model, which is helpful to analyze. But they only compared their new model with the deep convolution model designed by [1], which may lead to some limitation results. For example, they may ignore common deeper neural network shortcomings, like gradient vanishing problem and gradient exploding problem.

Scale-Invariant Convolutional Neural Network [2]

1. Research Overview

In this paper, they propose a scale invariant convolutional neural network (SiCNN), a model designed to incorporate multi-scale feature exaction and classification into the network structure. SiCNN is a multi-colunmn architecture, and each column is subject to a specific scale. Different from previous multi-column strategies, these columns use the same set of filter parameters through a scale transformation with them. This designs can reduce scale variation without changing the model size. Based on their experiments, the results show that SiCNN identifies features at diverse scales, and the classification result shows strong robustness against object scale variations.

2. Design

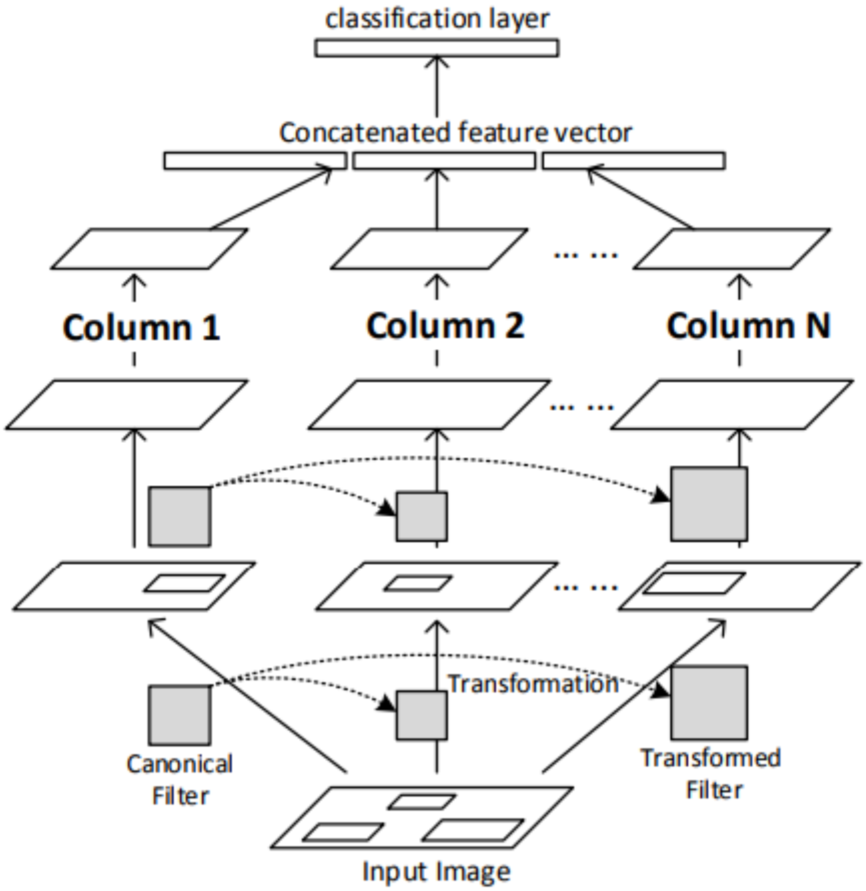
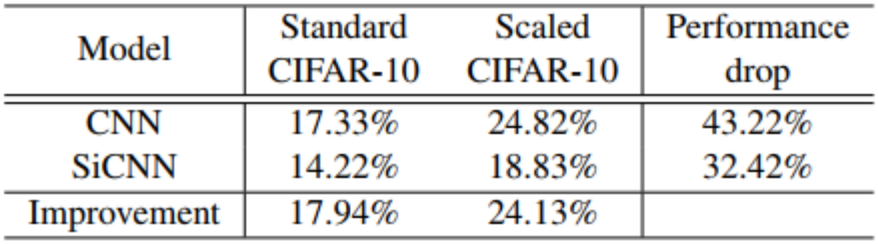


Figure 7: Architecture of SiCNN.

SiCNN applies various columns of convolutional stack with different filter sizes to catch objects that has unknown scales in input images. The input image is fitted into all the columns from bottom to up. Max-pooling is applied to several convolutional layers in each column. Even though these columns use different filter size, they share a set of common parameters with their filters. In each layer, Canonical column (Column 1 in Figure 7) keeps canonical filters. Other columns are called scale columns, covert these canonical filters into their own filter. As a whole, a canonical filter and the transformed filters recognize its pattern in multiple columns at different scales concurrently. Thus, a single pattern with various scales generate one or more columns.

In their architecture, they simply concatenate the top-layer feature maps from all the columns into a feature vector. The final classification layers (a softmax layer in the simplest case) take this feature vector as input.

3. Result



Model	Standard CIFAR-10	Scaled CIFAR-10	Performance drop
CNN	17.33%	24.82%	43.22%
SiCNN	14.22%	18.83%	32.42%
Improvement	17.94%	24.13%	

Figure 8: Classification error rate, tested on standard CIFAR-10 and scaled CIFAR-10.

Figure 8 compares the error rate results of CNN and SiCNN. Both of them are trained on standard CIFAR-10 dataset. Statistically, SiCNN achieves notable gain on standard CIFAR-10. More obvious results in performance drop, CNN has a performance drop of more than 43.22%, and SiCNN drops by 32.42%. Researchers manually test the error cases, and find that the simple central-crop-resize has cut off many important features in the scaled CIFAR-10. As a result, we hypothesize SiCNN will work better on multi-scale datasets with higher quality.

4. Weakness in design

They design a new CNN network called SiCNN to address scale variant problem in image classification. They made experiments with CNN and SiCNN in same dataset and get result the new network has better performance of the standard one. Weakness here is they set same depth in each experiment, thus when changing the depth of neural network, whether performance of SiCNN better than standard CNN is unknown.

Deep Residual Learning for Image Recognition [4]

1. Research Overview

Due to Deeper neural networks are more difficult to train, has gradient vanishing problem and gradient exploding problem. Researcher from Microsoft design a new network architecture - residual learning framework to solve the problems in deep neural network. On the ImageNet dataset they evaluate residual nets with a depth of up to 152 layers, with ensemble these residual nets achieve 3.57% error on the ImageNet test set. They won the 1st place on the ILSVRC 2015 classification task.

2. Design

A degradation problem has been exposed when deeper network start converging. With the network depth increasing, accuracy gets stuck, and then degrades speedily. The degradation is not caused by overfitting, and adding more layers to a suitable deep model builds a higher training error.

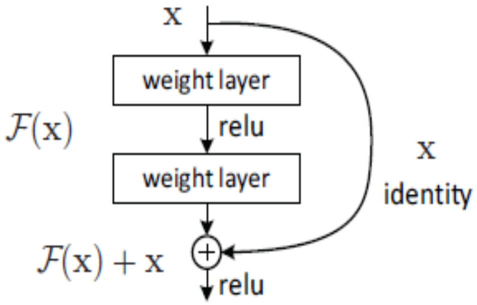


Figure 9: Residual learning: a building block.

Instead of desiring each few stacked layers fit a desired underlying mapping directly, researchers precisely let these layers fit a residual mapping. Then they applied this framework to 34 layers CNN network.

3. Result

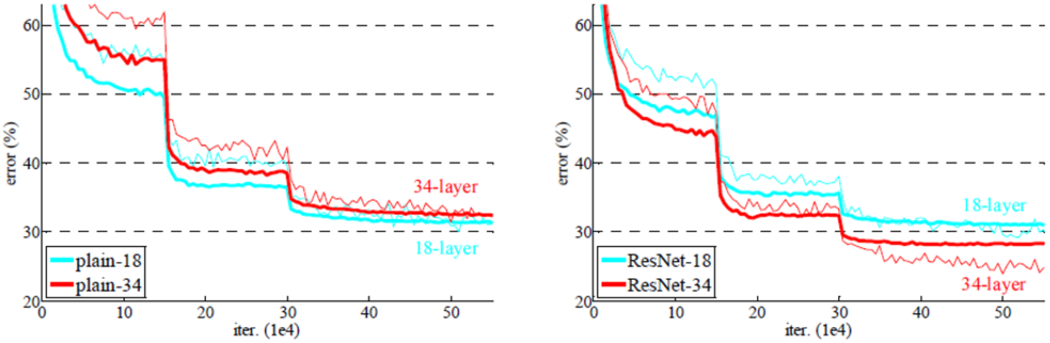


Figure 10: Training on ImageNet.

In figure 10, the thin curves show training error, and bold curves show validation error of the center crops. The left graph is plain networks of 18 and 34 layers. The right graph is ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts. All results show ResNet has improved degradation problems in CNN.

4. Weakness in design

This paper considers any factors may affect final result, and discuss each parameter in very detail, including different dataset, resNet performance in different layers, and how to control resNet parameters. So the weakness in design is minor.

Importance/Benefits of the Study

Image classification is more and more popular today, people produce many pictures and computer can classify them into different categories. This research can help in object recognition in CCTV, human Re-find in previous videos, check suspicious human activity in street videos, and product image classification in shopping website. It can be also applied in robots, since picture in computer view is actually different from human eyes, if we want robots act like human, then robots need human eyes and brains to recognize object, and classify object, they need to know the bonds between picture itself with its real meaning in human world.

Actually, some machine learning algorithm has already achieved human recognition level, but for some confused pictures with problems in prior section mentioned, there should be far way to go for image classification.

Research Design

Research Overview

Our research is based on a Kaggle image classification competition. The goal of this competition is to build a model that automatically classifies the products based on their images. Note that a product can have one or several images associated with it, hence there are different challenges:

- Massive images cause interference among images, therefore lowering the accuracy.
- The difference between each label is limited, so the prediction could be ambiguous.
- As for the contest, one problem is under one product, but could have multiple associated images. The associated images might lead the prediction into a wrong category.

Data Collection [8]

Ours dataset, different from the one above, is provided by a shopping website. The dataset consists of 4 files: train.bson, train_example.bson, test.bson, category_names.csv.

These file descriptions are copied from Kaggle.com

- " train.bson - (Size: 58.2 GB) Contains a list of 7,069,896 dictionaries, one per product. Each dictionary contains a product id (key: _id), the category id of the product (key: category_id), and between 1-4 images, stored in a list (key: imgs). Each image list contains a single dictionary per image, which uses the format: {'picture': b'...binary string...'}. The binary string corresponds to a binary representation of the image in JPEG format. This kernel provides an example of how to process the data.
- train_example.bson - Contains the first 100 records of train.bson so you can start exploring the data before downloading the entire set.
- test.bson - (Size: 14.5 GB) Contains a list of 1,768,182 products in the same format as train.bson, except there is no category_id included. The objective of the competition is to predict the correct category_id from the picture(s) of each product id (_id). The category_ids that are present in Private Test split are also all present in the Public Test split.

- category_names.csv - Shows the hierarchy of product classification. Each category_id has a corresponding level1, level2, and level3 name, in French. The category_id corresponds to the category tree down to its lowest level. This hierarchical data may be useful, but it is not necessary for building models and making predictions. All the absolutely necessary information is found in train.bson. "

Data Preprocessing

Preprocessing combines all images under one product to one image as shown below. We read the bson file to MongoDb, then read the binary image and combine them to one image.

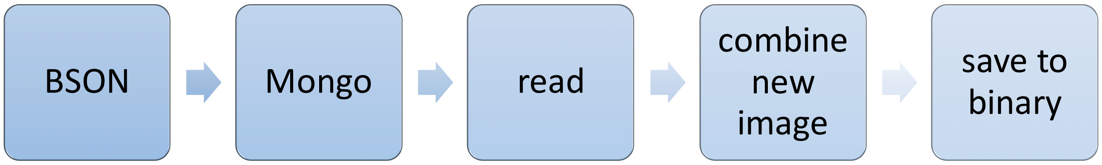

BSON → Mongo → read → combine new image → save to binary

Figure 11: Preprocessing progress

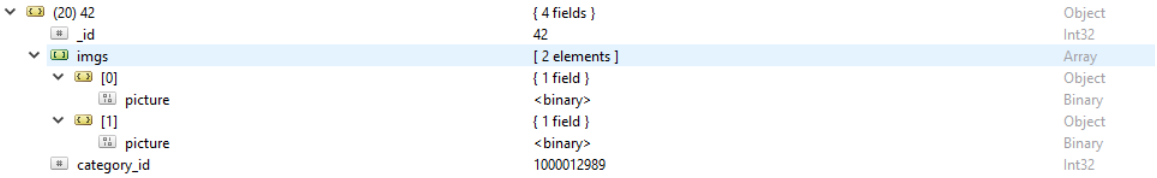
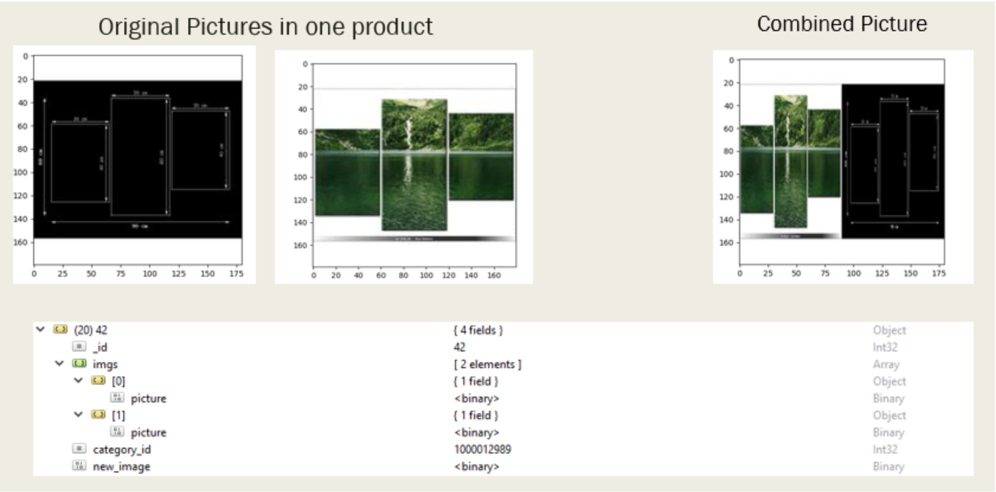

```
(20) 42                  { 4 fields }     Object
  _id                    42               Int32
  imgs                   [ 2 elements ]   Array
    [0]                  { 1 field }      Object
      picture            <binary>         Binary
    [1]                  { 1 field }      Object
      picture            <binary>         Binary
  category_id            1000012989       Int32
```

Figure 12: Example of data in database.

This is a sample saved in MongoDb. As can be seen from this picture, it contains category_id, so this is a train data.



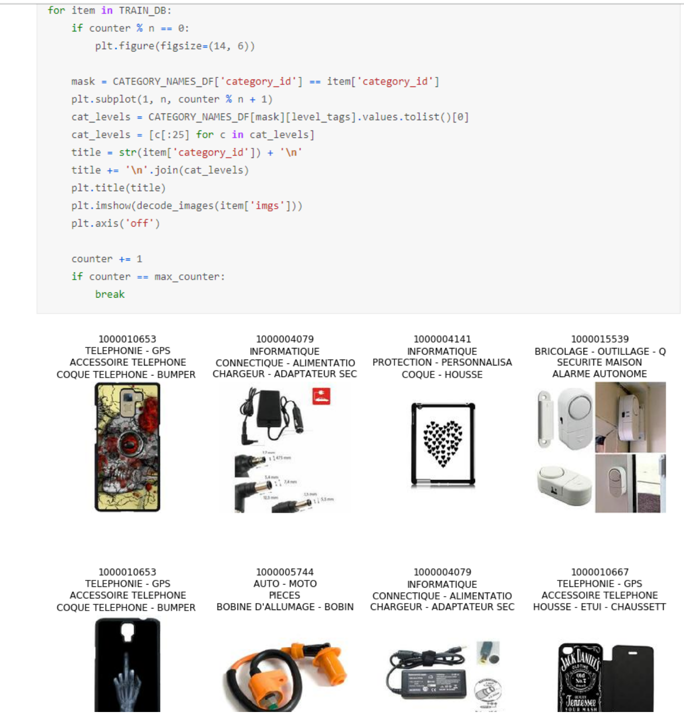
```
(20) 42                  { 4 fields }     Object
  _id                    42               Int32
  imgs                   [ 2 elements ]   Array
    [0]                  { 1 field }      Object
      picture            <binary>         Binary
    [1]                  { 1 field }      Object
      picture            <binary>         Binary
  category_id            1000012989       Int32
  new_image              <binary>         Binary
```

Figure 13: Example of new combined image.

Then we combine two pictures under this product to one new_image, thus we have one input for neural network, and one prediction vector output from the network.

Reward Policy

To add a reward policy that we may use in future, we do a second preprocessing on each product. We add a 3 level category name to each product, so that if our model predicts a wrong but similar result, we can give this model a reward, to show it the result is very close. Figure 14 shows each product was integrated to one picture, each of them with 3 level category name in the header and specific category id.



```
for item in TRAIN_DB:
    if counter % n == 0:
        plt.figure(figsize=(14, 6))

    mask = CATEGORY_NAMES_DF['category_id'] == item['category_id']
    plt.subplot(1, n, counter % n + 1)
    cat_levels = CATEGORY_NAMES_DF[mask][level_tags].values.tolist()[0]
    cat_levels = [c[:25] for c in cat_levels]
    title = str(item['category_id']) + '\n'
    title += '\n'.join(cat_levels)
    plt.title(title)
    plt.imshow(decode_images(item['imgs']))
    plt.axis('off')

    counter += 1
    if counter == max_counter:
        break
```

Figure 14: Example of three level labels.

Model Design

At first, we use resNet-50, which is a 50 layers residual network, and give every image a resolution of 180*180. We use all parameters as [4] mentioned.

Then we change the image resolution, drop some useless pictures in one product, and through a random crop and scale method, we get 224*224 resolution, which better fits our trained model. Then we change the learning rate and use deeper neural network-152 resNet. This gave us a better accuracy.

Data Analysis

In the prior design, we got top-1 accuracy about 53%.

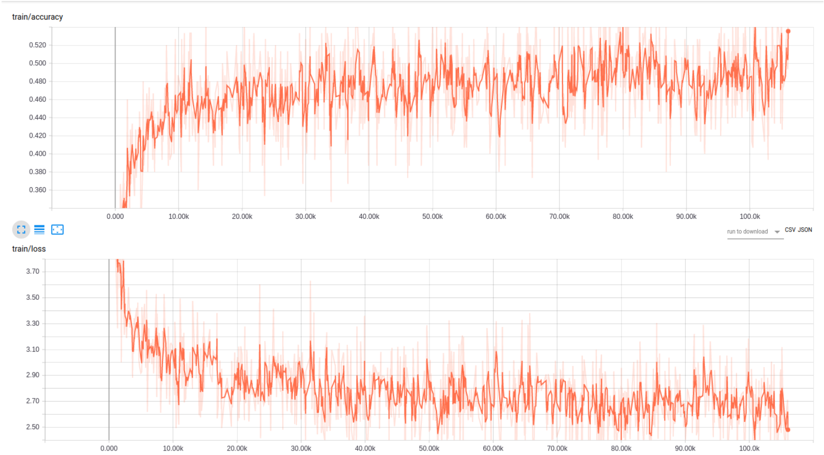


Figure 15: Top-1 accuracy in iteration 1.

In the improved design, the Top-1 accuracy raised to 70.4%.

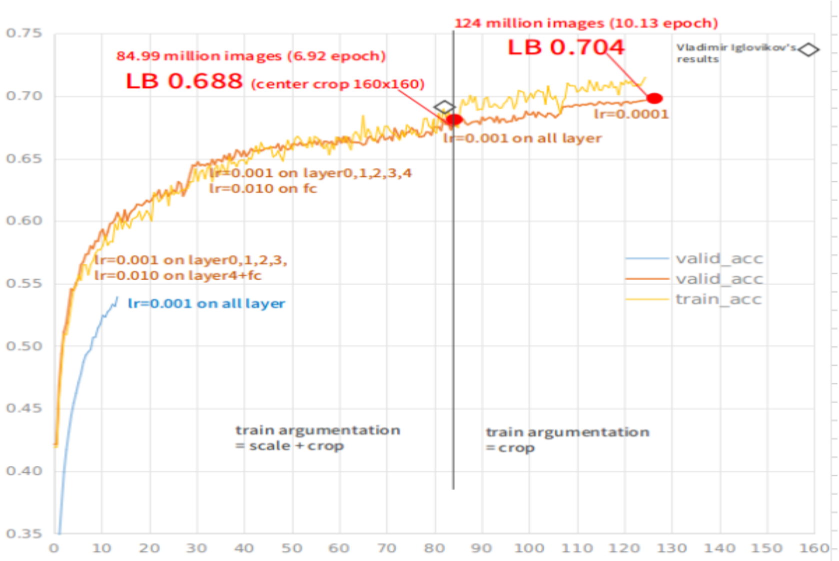


Figure 16: Top-1 accuracy in iteration 2.

The result shows the deeper resNet gets better performance. And setting a higher learning rate in the full connect layer will help it learn more abstract features. In addition, random crop and scale can also achieve Data Augmentation.

Schedule

Time	To Do/ Done
Sep 2017 - - Oct 2017	Data Preprocessing
Oct 2017 - - Dec 2017	Training and making changes
Dec 2017 - - Feb 2018	Adding Reward Learning policy
Feb 2018 - - April 2018	Change output policy Thanks to current Top-1 contributor: https://www.kaggle.com/c/cdiscount-image-classification-challenge/discussion/44581
May 2018	Apply all new features to new model

Facilities and Special Resource

At least one 1080Ti video card or more.

32GB RAM or more.

Python2.7 or 3.X environment.

Tensorflow or Pytorch installed.

Cuda installed.

OpenCV installed.

Deliverables

- ✓ Transforming the res-net from ImageNet into the kaggle contest data set, includes:

i. inherit the trained model parameters,

ii. change learning rate,

iii. output has possibility of 5720 classes instead of 1000.

iv. Change dataset and add new combined images in each product.

- ✓ Add more three level class info into one product info
- ✓ Change network layers to 50 layers, add some more max pooling.
- ✓ Change network layers to 152 layers, change learning rate in full connected layers
- ✓ Rescale and crop figure to fit flexible layer change.
- ✓ Achieving top-1 accuracy of 70.4%.

Reference

[1] Krizhevsky, A. (n.d.). *ImageNet Classification with Deep Convolutional Neural ...* Retrieved 125, 2017, from http://www.image-net.org/challenges/LSVRC/2012/supervision.pdf

[2] Xu, Y., Xiao, T., Zhang, J., Yang, K., & Zhang, Z. (2014). Scale-Invariant Convolutional Neural Networks. *arXiv: Computer Vision and Pattern Recognition*. Retrieved 12 5, 2017, from https://arxiv.org/abs/1411.6369

[3] Zeiler, M. D., & Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. *arXiv: Computer Vision and Pattern Recognition*, 818-833. Retrieved 12 5, 2017, from https://arxiv.org/abs/1311.2901

[4] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[5] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun. Object detection networks on convolutional feature maps. arXiv:1504.06066, 2015.

[6] P.Y. Simard, D. Steinkraus, and J.C. Platt. Best practices for convolutional neural networks applied to visualdocumentanalysis. InProceedingsoftheSeventhInternationalConferenceonDocumentAnalysis and Recognition, volume 2, pages 958–962, 2003.

[7] Nath S S, Mishra G, Kar J, et al. A survey of image classification methods and techniques[C]//Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2014 International Conference on. IEEE, 2014: 554-557.

[8] https://www.kaggle.com/c/cdiscount-image-classification-challenge

[9] http://cs231n.github.io/classification/