Deep Convolutional Neural Networks for Image Classification

Nishant Gupta
Technische Universitt Mnchen
nishant.gupta@tum.de (Mtk. Nr.:03646526)
https://github.com/nishantgupta91/MLFCV/tree/master/Exercise2

***Abstract*—the central idea of this exercise is to use Deep Convolutional Neural Network for the classification of images. Basics of Neural Networks, Deep Learning and Backpropagation algorithm used in Neural Networks are also studied.**

I. Introduction

Deep Learning is a new field which aims to improve current Machine Learning technologies for achieving better results in Artificial Intelligence. Deep Learning is a class of machine learning techniques that exploit many layers of non-linear information processing for supervised or unsupervised feature extraction and transformation, and for pattern analysis and classification [1]. In this exercise, deep convolutional neural network [6] is used to classify images using a powerful framework called caffe [4]. With an expressive architecture and available Python and Matlab wrappers, it is one of the most frameworks being used today in the field of deep learning.

II. Backpropagation Algorithm

Following is the derivation of Backpropgation algorithm for any other function f than the sigmoid function σ.

We can define output error for j^{th} neuron δ_j^L by:

$$\delta_j^L = \frac{\partial C}{\partial z_j^L} \tag{1}$$

This can also be written as:

$$\delta_j^L = \frac{\partial C}{\partial a_j^L}\frac{\partial a_j^L}{\partial z_j^L} \tag{2}$$

Since $a_j^L = f(z_j^L)$, the second term on the right side of equation 2 can be written as $f'(z_j^L)$

Therefore:

$$\delta_j^L = \frac{\partial C}{\partial a_j^L} f'(z_j^L)$$

Output error can be written as:

$$\delta^L = \nabla C \odot f'(z^L) \tag{3}$$

Next step is to backpropagate this error to previous layers. Let l be the previous layer to output layer L. Error at this layer δ^l can be written as

$$\delta^l = \frac{\partial C}{\partial z^l} = \frac{\partial C}{\partial z^l}\frac{\partial a^l}{\partial a^l} = \frac{\partial a^L}{\partial z^l}\nabla C(a^L)$$

$$= \frac{\partial z^L}{\partial z^l}(f'(z^L) \odot \nabla C(a^L)) = \frac{\partial z^L}{\partial z^l}\delta^L$$

$$= f'(z^l) \odot ((w^L)^T \delta^L)$$

Similarly for the previous layers this expression would hold true. Generalizing it to any intermediate layer l, we can write the expression for error δ^l as:

$$\delta^l = f'(z^l) \odot ((w^{l+1})^T \delta^{l+1})$$

III. Dataset Description

The aim of the exercise was to detect the contents of a test set of photographs using a large hand labeled ImageNet dataset used in ILSVRC12 challenge [2] as training dataset.

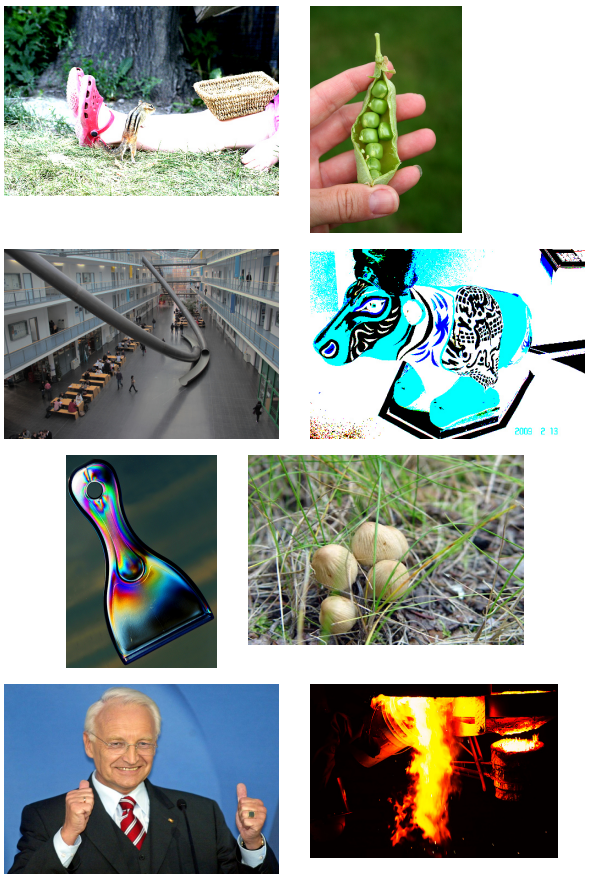

TABLE I: Input Images

Training data contains 1000 categories and 1.2 million images. Test dataset contains random pictures (clicked / downloaded) which are not contained in the ImageNet training data (see Table 1 with 8 photographs)

IV. Data Preprocessing

This approach follows a reference model BVLC Reference CaffeNet [3]. Test set consists of variable-resolution images, while the reference model requires a constant input dimensionality of 256x256. So, all the images in test set are resized to a fixed resolution of 256x256 similar to those in training model. The image was first rescaled such that shorter side was of length 256, and then central 256x256 patch was cropped from the resulting image.

V. Classification Results

My source code [5] for this exercise preprocesses the input images and classifies the content of these images into one of the 1000 classes. Classifier predictions with probability and entropy for the given original input images is shown in the following table.

Image	Class	Contents	Probability	Entropy
IMG_1	8	Hen	0.15	4.50
IMG_2	315	Mantis, mantid	0.23	3.29
IMG_3	421	Bannister, handrail	0.10	4.37
IMG_4	639	Maillot, tank suit	0.05	4.92
IMG_5	673	Mouse, Computer Mouse	0.26	3.07
IMG_6	947	Mushroom	0.38	1.64
IMG_7	834	Suit, suit of clothes	0.64	1.11
IMG_8	980	Volcano	0.17	3.90

As can be seen in the table, most of the images were not classified properly. By looking at the values of probability and entropy, only IMG_7 seems to have a decent score. That is why this is the only image whose contents were identified accurately as suit of clothes. For rest of the images, since the value of probability is low and entropy is high, this means the images have been classified wrong.
Now following table gives results of classification using the same classifier, but for the preprocessed input images.

Image	Class	Contents	Probability	Entropy
IMG_1	292	Tiger, Panthera Tigris	0.11	4.69
IMG_2	945	Bell pepper	0.35	3.17
IMG_3	657	Missile	0.21	2.73
IMG_4	620	Laptop, laptop computer	0.11	4.77
IMG_5	723	Pinwheel	0.13	3.58
IMG_6	947	Mushroom	0.34	1.68
IMG_7	906	Windsor Tie	0.71	1.35
IMG_8	552	Feather boa, boa	0.34	3.90

With some preprocessing steps followed as described in the previous section, overall improvement in terms of getting higher probability and lower entropy was observed for most of the images. IMG_7 is detected to contain a Windsor Tie which is acceptable considering the fact that preprocessing resulted into cropping of central patch of dimensions 256x256 pixels. IMG_2, IMG_3 and IMG_8 showed significant improvement in terms of probability and entropy score compared to the previous case with no preprocessing done.

References

[1] L. Deng and D. Yu (2014) "Deep Learning: Methods and Applications". http://research.microsoft.com/pubs/209355/DeepLearning-NowPublishing-Vol7-SIG-039.pdf

[2] Large Scale Visual Recognition Challenge 2012 (ILSVRC2012). http://image-net.org/challenges/LSVRC/2012/

[3] Krizhevsky, Sutskever, and Hinto (2012) ImageNet Classification with Deep Convolutional Neural Networks.

[4] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama and T. Darrell, "Caffe: Convolutional Architecture for Fast Feature Embedding." arXiv preprint arXiv:1408.5093, 2014.

[5] Source Code: https://github.com/nishantgupta91/MLFCV/tree/master/Exercise2

[6] Deep Learning tutorial deeplearning.net/tutorial/lenet.html