Udacity MLND Capstone Proposal
On Classification of the STL-10 dataset using Deep Convolutional Networks and Transfer Learning

Tanbir Ahmed

1 Domain background

Computer vision is a vast and inter-disciplinary field that seeks to automate tasks performed by human vision-system from object detection to driving a vehicle or flying an aircraft. Many different areas and sub-areas have emerged over the years to cover various aspects of computer vision. Object-detection on images still remains as one of the toughest among the fundamental areas. Datasets continue to pose challenges as researchers continue their effort to overcome the learning barriers. The motivations behind investigating the STL-10 dataset (described in the dataset section) in this project are as follows:

- *Learning state-of-the-art models and algorithms used for classification*,
- *Investigating networks from scratch and various pretrained-models for transfer learning*,
- *And most importantly, working on a challenging benchmark dataset that presents plenty of room for improvement in performace and developing scalable learning models*.

For historical purpose, the interested readers may consult [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17].

2 Problem statement

The main objective of this project is to apply deep-learning techniques to investigate the object-detection on images (image-classification) problem conducting expriments on the STL-10 dataset. In this dataset, the training subset is smaller than the test subset and the state-of-the-art has apparently significant scope for improvement.

Though, getting a state-of-the-art result is not within the scope of this project, obtaining a competitive baseline result using a basic CNN (Convolutional Neural Network) is well within its purpose. The availability of the unlabeled dataset and several pre-trained imagenet models open up possibilities for further improvement using the baseline model on an augmented dataset.

It is highly tempting to try as many different known techniques as possible on the above problem, yet time and computing resources may constrain the options. The Benchmarks section describes more about the possibilities.

3 Data sets and inputs

As presented at Adam Coates' STL-10 page, the STL-10 dataset [4] is an image recognition dataset for developing unsupervised feature learning, deep learning, and self-taught learning algorithms. The dataset has the following properties:

- *There are 10 classes: airplane, bird, car, rat, deer, dog, horse, monkey, ship, truck*
- *Total* 5000 *training images and* 8000 *test images each with shape* $96 \times 96 \times 3$ *for supervised learning. There are* 500 *training images (*10 *pre-defined folds) and* 800 *test images per class.*
- *100000 unlabeled images for unsupervised learning with images from a similar but a broader distribution of images.*

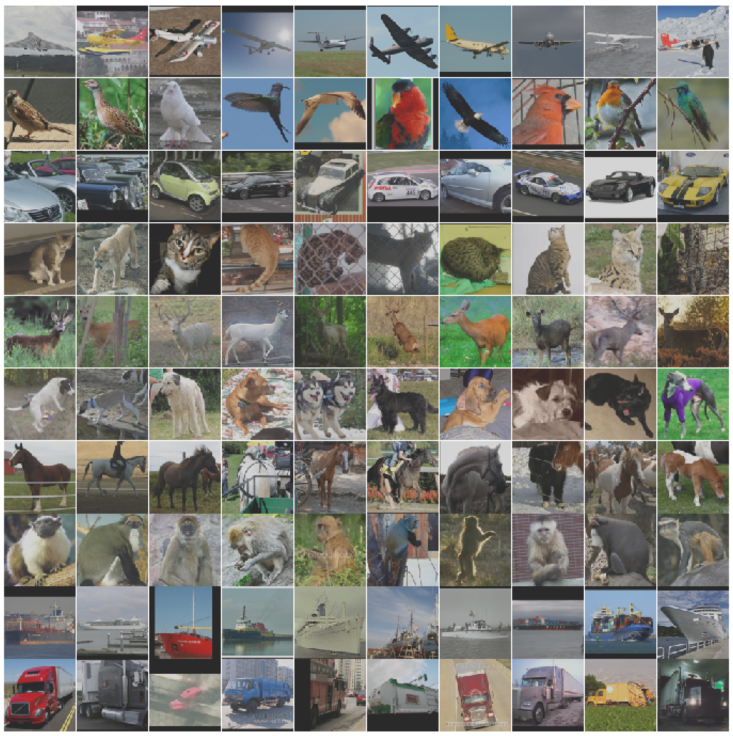
Figure 1: STL-10 dataset example

These images were acquired from labeled examples on Imagenet. The dataset is inspired by the CIFAR-10 dataset ($60000 \times 32 \times 32 \times 3$ with 50000 training images and 10000 test images). But higher resolution makes the STL-10 dataset a challenging benchmark for developing scalable learning models.

4 Solution statement

Deep learning techniques have been extremely successful in classification of large image datasets. One of the fundamental challenges is scalability of the designed models. The STL-10 dataset is a perfect example where obtaining model scalability is a major challenge given the training subset is smaller compared to the test subset. The availability of a large unlabeled subset and several pre-trained imagenet models would allow transfer learning and improvement of performace towards obtaining scalability. In this project, I attempt to explore such a possibility.

5 Benchmark

Classification results including the state-of-the art for the STL-10 dataset are as follows (taken from Rodrigo benenson's web-site at [1]):

Result	Method / Publication title	Reference
74.33%	Stacked What-Where Auto-encoders	Graham [7]
74.10%	Convolutional Clustering for Unsupervised Learning	Springenberg et al. [13]
73.15%	Deep Representation Learning with Target Coding	Yang et al. [16]
72.80%	Discriminative Unsupervised Feature Learning with Convolutional Neural Networks	Dosovitskiy et al. [5]
70.20%	An Analysis of Unsupervised Pre-training in Light of Recent Advances	Le Paine et al. [8]
70.10%	Multi-Task Bayesian Optimization	Swersky et al. [14]
68.23%	C-SVDDNet: An Effective Single-Layer Network for Unsupervised Feature Learning	Wang and Tan [15]
68.00%	Committees of deep feedforward networks trained with few data	Miclut et al. [11]
67.90%	Stable and Efficient Representation Learning with Nonnegativity Constraints	Lin and Kung [9]
64.50%	Unsupervised Feature Learning for RGB-D Based Object Recognition	Bo et al. [2]
62.32%	Convolutional Kernel Networks	Mairal et al. [10]
62.30(±1)%	Discriminative Learning of Sum-Product Networks	R. Gens et al. [6]
61(±0.58)%	No more meta-parameter tuning in unsupervised sparse feature learning	Romero et al. [12]
61.00%	Deep Learning of Invariant Features via Simulated Fixations in Video	Zou et al. [17]
60.10%	Selecting Receptive Fields in Deep Networks	Coates and Ng. [3]

Figure 2: Some best known results on the STL-10 dataset

I would like to obtain a performance that falls within the range in the above list (achieve around 60% accuracy on the test dataset) with a convolutional neural network built from the scratch with tuned hyperparameters but without preprocessing or augmentation of the training dataset. Then I would like to improve it as far as possible using pre-processing and augmentation with help from pre-trained imagenet models to label the unlabeled dataset. I may possibly also apply some techniques and ideas as in the references if the time and scope allows.

6 Evaluation metrics

The evaluation metric used is the *multi-class logarithmic loss* (also known as *categorical cross entropy* in machine learning context). It measures the performance of a classification model whose output is a probability (output of *softmax* activation) between 0 and 1. Let N be the number of images and M be the number of class labels. For $1 \leqslant i \leqslant N$ and $1 \leqslant j \leqslant M$, let $y_{i,j}$ denote the indicator variable which equals to 1 if and only if observation i belongs to class j. Let

$p_{i,j}$ denote the predicted probability that observation i belongs to class j. Then the multi-class logarithmic loss is

$$-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{i,j}\ln\left(p_{i,j}\right).$$

7 Project design

Language: Python 2.7+
Libraries to be used: Keras, Tensorflow, Scikit-learn, Tensorflow-gpu
Outline:

- Designing a CNN (or preferably a class of CNNs) from scratch to train the labeled dataset to produce a performance in the range of 60% in test accuracy.
    - ◇ Discovering general pattern in architechture and trends in the hyperparameters towards improved performances.
    - ◇ Finding restarts and finetuning strategies towards improved performances.
- Extracting features form pre-trained imagenet models such as ResNet/InceptionV3 on the labeled data and saving the bottleneck features
- Finetuning the imagenet model using bottleneck features as input to obtain a performance in the range of 90% in test accuracy.
    - ◇ This will be used as a reliable classifier model
    - ◇ Classifying the unlabeled subset using threshold on the softmax probabilities using this classifier model
- Augmenting the original labeled training dataset using the newly labeled unlabeld data
- Applying the model from scratch on the augmented training subset to obtain improved test accuracy

References

[1] R. Benenson, Discover the current state of the art in objects classification. (Retrieved: 2018-03-27) *http://rodrigob.github.io/are_we_there_yet/build/classification_datasets_results.html.*

[2] L. Bo, X. Ren, D. Fox, Unsupervised Feature Learning for RGB-D Based Object Recognition, *ISER*, (2012).

[3] A. Coates, A. Y. Ng, Selecting Receptive Fields in Deep Networks, *NIPS*, (2011).

[4] A. Coates, H. Lee, A. Y. Ng, An Analysis of Single Layer Networks in Unsupervised Feature Learning, *AISTATS*, (2011).

[5] A. Dosovitskiy, J. T. Springenberg, M.n Riedmiller and T. Brox, Discriminative Unsupervised Feature Learning with Convolutional Neural Networks, *NIPS*, (2014).

[6] R. Gens and P. Domingos, Discriminative Learning of Sum-Product Networks, *NIPS*, (2012).

[7] B. Graham, Fraxtional Max-Pooling, *http://arxiv.org/abs/1412.6071*, (2015).

[8] T. Le Paine, P. Khorrami, W. Han, T. S. Huang, An Analysis of Unsupervised Pre-training in Light of Recent Advances, *ICLR*, (2015).

[9] T. H. Lin, H. T. Kung, Stable and Efficient Representation Learning with Nonnegativity Constraints, *ICML*, (2014).

[10] J. Mairal, P. Koniusz, Z. Harchaoui, C. Schmid, Convolutional Kernel Networks, *https://arxiv.org/abs/1406.3332*, (2014).

[11] B. Miclut, T. Kaester, T. Martinetz, E. Barth, Committees of deep feedforward networks trained with few data, *https://arxiv.org/abs/1406.5947*, (2014).

[12] A. Romero, P. Raveda, C. Gatta, No more meta-parameter tuning in unsupervised sparse feature learning, *https://arxiv.org/abs/1402.5766*, (2014)

[13] J. T. Springenberg, A. Dosovitskiy, T. Brox, M. Riedmiller, Striving for Simplicity: The All Convolutional Net, *https://arxiv.org/pdf/1412.6806.pdf*, (2015).

[14] K. Swersky, J. Snoek, R. P. Adams, Multi-Task Bayesian Optimization, *NIPS*, (2013).

[15] D. Wang, X. Tan, Unsupervised Feature Learning with C-SVDDNet, *https://arxiv.org/abs/1412.7259*, (2014).

[16] S. Yang, P. Luo, C. C. Loy, K. W. Shum, X. Tang, Deep Representation Learning with Target Coding, *AAAI*, (2015).

[17] W. Y. Zou, S. Zhu, A. Y. Ng, K. Yu, Deep Learning of Invariant Features via Simulated Fixations in Video, *NIPS*, (2012).