# Tackling Black Box Learning using Neural Networks

Titas Nandi

Supervisor: Dr. Arijit Mondal

IIT Patna

April 19, 2017

## ICML Black Box Challenge

- Train a classifier on a dataset that is not human readable
    - Without the knowledge of what the data consists of
- Designed to reduce the usefulness of having a human researcher working in loop with the training algorithm
- Organized by Yoshua Bengio, Ian Goodfellow and Dumitru Erhan as part of **ICML 2013 - Challenges in Representation Learning** [1]

# Dataset

- Problem of Semi-supervised Deep Learning

### Dataset is divided as

- *Supervised data* - 1000 labeled examples with 1875 features classified into 9 classes
- *Unsupervised data* - 135,735 unlabeled examples again with 1875 features
- *Test data* - 10,000 examples split into
  - 5000 public set examples
  - 5000 private set examples

## Baselines

- Random Baseline - **11.1 %**
- Logistic Regression - **21.1 %**
- ZCA + 1 layer net - **41 %**
- ZCA + 3 layer net - **51.5 %**

# Benchmark Results

### First Position

Sparse Filtering + Feature Selection + SVM with linear kernel - **70.22 %**

### Second Position

Pseudo Labels + Denoising Autoencoder + Dropout - **69.58 %** [2]

### Third Position

Horizontal and Vertical Ensemble for Classification - **69.14 %**

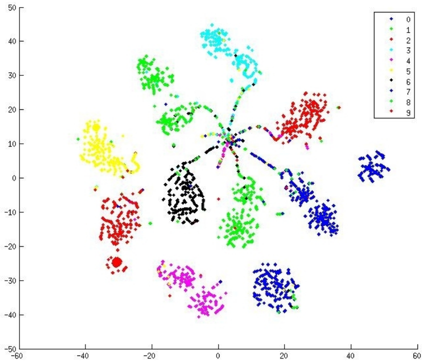
# Pseudo Labels

- Generate pseudo labels for unlabeled data

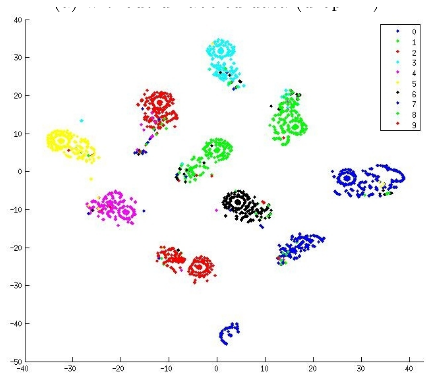
### Method

- run a classifier on labeled examples
- determine probable labels for the unlabeled data
- use both sets of data together for training
- recalculate pseudo labels every weight update

- minimizes conditional entropy of class labels for unlabeled data [3]
- prefers low density separation between classes




(a) without unlabeled data (dropNN)




(b) with unlabeled data and Pseudo-Label (+PL)

Figure: t-SNE 2-D embedding of the network output of MNIST test data

## Sparse Filtering Approach

- **Unsupervised feature learning**
- A major performance constraint of sparse RBMs or autoencoders is hyperparameter tuning
- Optimizes a simple cost function - **sparsity of L2-normalized features** [4]
- Learn sparsely activated features by
  - **Population Sparsity**
  - **Lifetime Sparsity**
  - **High dispersal**

# Sparse Filtering + Supervised Training

- Break the large unsupervised data into **5000** example chunks
- Train a **feedforward Sparse Filter** on these chunks
  - each chunk will be pulled in for training in **data batches** of given count
  - produce 10 feature sets having revised weights
- Picked out the top performing **120 features** out of 1875 initially

# Implementation

- Find the **revised representation** for the training and test data
- Train a **feedforward** Neural Network on the supervised data using these revised weights
- Experiments with neural net architecture

# Architectural experiments

| Num | N | L | Act | D | Opt | Epoch | Batch Size | Acc |
|---|---|---|---|---|---|---|---|---|
| ***Best*** | **1500** | **2** | **sigmoid** | **0.4** | **adam** | **200** | **128** | **64.74** |
| ***1*** | 1000 | 1 | relu | 0.4 | adam | 20 | 128 | 60.12 |
| ***2*** | 200 | 2 | sigmoid | 0.4 | adam | 20 | 128 | 51.22 |
| ***3*** | 1000 | 2 | sigmoid | 0.4 | adam | 100 | 128 | 64.02 |
| ***4*** | 1000 | 3 | sigmoid | 0.4 | adam | 100 | 128 | 63.86 |
| ***5*** | 1000 | 2 | sigmoid | 0.4 | adam | 1000 | 128 | 63.80 |
| ***6*** | 1500 | 2 | sigmoid | 0.5 | adam | 200 | 128 | 64.50 |
| ***7*** | 2000 | 2 | sigmoid | 0.4 | adam | 200 | 128 | 64.66 |
| ***8*** | 1500 | 2 | sigmoid | 0.3 | adam | 200 | 128 | 64.66 |
| ***9*** | 1500 | 2 | sigmoid | 0.4 | adam | 200 | 256 | 64.42 |
| ***10*** | 1500 | 2 | sigmoid | 0.4 | sgd | 200 | 128 | 39.50 |
| ***11*** | 1500 | 2 | relu | 0.4 | adam | 200 | 128 | 61.72 |

Table: Neural Network Experiments on sparsed features
(N = neurons, L = layers, Act = activation, D = dropout, Opt = Optimizer)

# Computation of Pseudo Labels

- Train a feedforward neural net on the supervised examples
- Find probable labels of the unsupervised data
- Retrain the neural network with the **combined** data
- At this point, the network might not have learnt the pseudo labels properly or might be **overfitted**
- Retrain the network until **convergence** (till there are no significant changes in predicted labels)

# Pseudo Labels Method: Results

| Iterations | 1 hidden + 1000 neurons | 2 hidden + 1500 neurons each |
|---|---|---|
| *1* | 56.04 | 47.86 |
| *3* | 55.48 | 47.98 |
| *6* | 55.26 | 48.16 |
| *10* | 55.00 | 48.10 |

Table: Pseudo Labels training after specific iterations of the algorithm

# Irregularities

- Giving same weights to both supervised and unsupervised data
- Need to change weight coefficients of unsupervised data in a time dependent manner
- In some cases, maybe the system is actually moving away from true labels
- The code for both the implementations is available on **https://github.com/TitasNandi/ICML-BlackBox-Challenge**

# Future Work

## Future Work

- Address irregularities in Pseudo Label training
- The success of these methods is **powerful**
  - Reduces annotation overload massively
  - **Black Box** Learning in true sense
- Extend it to data from cQA sites

# References


- I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, *et al.*, "Challenges in representation learning: A report on three machine learning contests," in *International Conference on Neural Information Processing*, pp. 117–124, Springer, 2013.
- D.-H. Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on Challenges in Representation Learning, ICML*, vol. 3, p. 2, 2013.
- Y. Grandvalet, Y. Bengio, *et al.*, "Semi-supervised learning by entropy minimization.," in *NIPS*, vol. 17, pp. 529–536, 2004.
- J. Ngiam, Z. Chen, S. A. Bhaskar, P. W. Koh, and A. Y. Ng, "Sparse filtering," in *Advances in neural information processing systems*, pp. 1125–1133, 2011.