IE 506: COURSE PROJECT

Learning to Learn: Model Regression Networks for Easy Small Sample Learning

Team AI-nthusiasts

Om Prabhu (19D170018)
Naman Gupta (190010048)

OUTLINE

- PROBLEM STATEMENT:
  - Overview
  - Technical Details
  - Previous Works
- PROBLEM DETAILS:
  - Model Regression Networks
  - Generation of Model Pairs
  - Proposed Method
- PAPER IMPLEMENTATION:
  - Data Processing
  - Technical Details
- STATUS OF WORK:
  - Result Replication Status
  - Modifications Proposed

PROBLEM STATEMENT : Overview

- **Objective:** Using experience from already learned example classes to facilitate learning of novel classes.
- **Motivation:** The authors hypothesize the existence of a generic, category agnostic transformation from small-sample models to underlying large-sample models, which is empirically validated in the paper. A major motivation is the transferability of feature extraction in deep CNNs trained on large object categories to novel small categories.
- **Background:** Over the past decade, ML techniques to deal with big data have emerged - however, data acquisition is either expensive or very tough due to its limited nature in most practical applications. The most common applications with limited training data involve visual phenomena such as object recognition & error handling by robots in natural environments (e.g., emergency response systems in self-driving cars).

PROBLEM STATEMENT : Technical Details

- The paper makes a three-fold contribution:
    - Construction of a training “model set” - this set consists of model pairs learned from small and large sample sets on various categories
    - Introduction of a model regression network to learn and identify generic transformations between the individual models
    - Implementation of the regression network to facilitate recognition of novel categories from few samples

PREVIOUS WORKS - 1
Tabula Rasa: Model Transfer for Object Category Detection (Aytar and Zisserman, 2011)

1. <u>Concepts:</u> Using a well trained detector for a source category, learn an object detector for a visually similar target category (e.g., bicycle classifier learned using a motorbike classifier as the source
2. <u>Model:</u> Stochastic gradient descent to train SVMs for learning small-sample models
   - Training set contains annotated images with tight bounding boxes
   - Switch between optimizing latent variables and SVM objectives
3. <u>Observations & Results:</u>
   A. Transfer learning offers considerable benefits including higher start, higher slope and a higher asymptote for performance levels
   B. Performance levels for transfer methods are consistently higher than baseline SVM models

PREVIOUS WORKS - 2
Uncovering shared structures in multiclass classification (Weller and Seppi, 2019)

OUTLINE

- PROBLEM STATEMENT:
  - Overview
  - Technical Details
  - Previous Works
- PROBLEM DETAILS:
  - Model Regression Networks
  - Generation of Model Pairs
  - Proposed Method
- PAPER IMPLEMENTATION:
  - Data Processing
  - Technical Details
- STATUS OF WORK:
  - Result Replication Status
  - Modifications Proposed

PROBLEM DETAILS

- Model Regression Networks: Supervised classifier that discriminates between positive and negative instances of a feature space
  - w^0 - classifier learned from few annotated samples
  - w^* - classifier learned from large set of samples of the same category
  - The goal is to generate a classifier w as close to w^* as possible
- Assumption: There exists a generic non-linear transformation T such that for any w^0 and w^*, we have $w^* \approx T(w^0)$

Performance sanity check of regressed models against large and small-sample models

PROBLEM DETAILS

- Generation of Model Pairs: Original training set contains large amounts of labeled data from various categories $\{(x_i, y_i)\}$ where $y_i \in \{1, ..., C\}$, where C = number of categories
  - Produce a collection of model pairs (w^*_j, w^0_j) using original training set for $j = 1, ..., C$
  - Each model is a binary classifier based on separating one category from the rest
- For each category c, we first learn $w^{c,*}$ from a large sample set - ground truth model
- Then, randomly sample $N \ll L_c$ (total number of positive samples for category c) points and train a binary SVM classifier $w^{c,0}$
  - Repeat this process by sub-sampling to obtain a series of model pairs for each category

PROPOSED METHOD

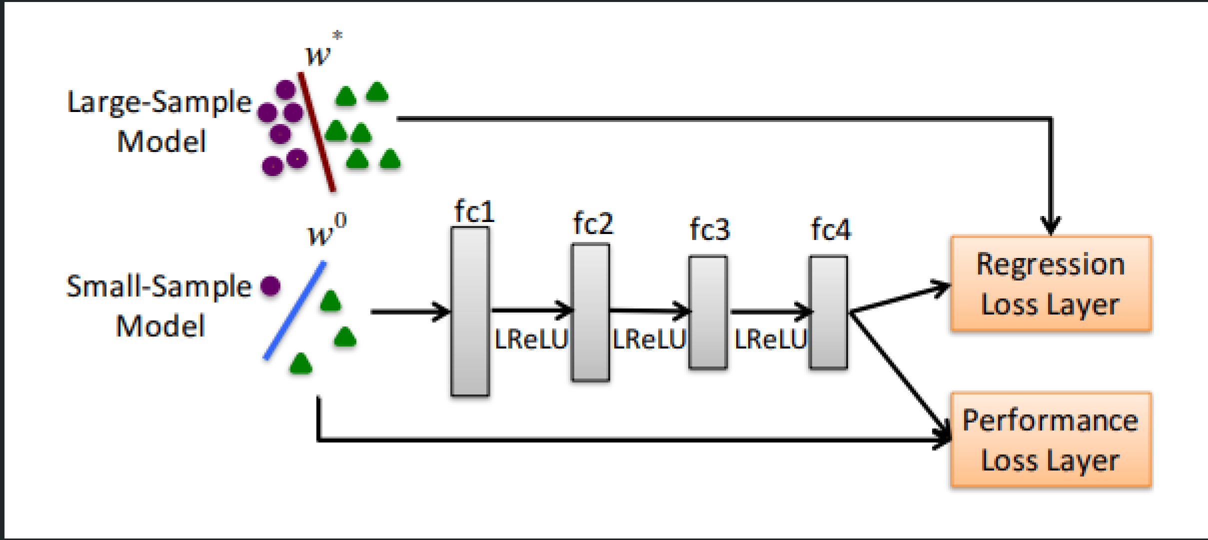
1. Training model set contains one-to-one model correspondence - we aim to learn a mapping *T*: $\mathbf{w}^0 \rightarrow \mathbf{w}^*$
2. Parametrize the transformation as a regression function $T(\mathbf{w}^0, \Theta)$ such that $\mathbf{w}^* \approx T(\mathbf{w}^0, \Theta)$
3. Quality of the approximation is quantified by square of the Euclidean distance

(

$$L(\Theta) = \sum_{j=1}^{J} \left\{ \frac{1}{2} \left\| \mathbf{w}_j^* - T\left(\mathbf{w}_j^0, \Theta\right) \right\|_2^2 + \lambda \sum_{i=1}^{M+N} \left[1 - y_i^j \left(T\left(\mathbf{w}_j^0, \Theta\right)^T \mathbf{x}_i^j \right) \right]_+ \right\}$$

1. Regression function is a multi-layer feed-forward neural network

PROPOSED METHOD - ILLUSTRATION



As observed, regression network consists of 4 fully-connected layers with leaky ReLUs - each layer applies a non-linear transformation *G*. The desired transformation *T* is obtained as a series of layer-by-layer transformations *G*.

OUTLINE

- PROBLEM STATEMENT:
  - Overview
  - Technical Details
  - Previous Works
- PROBLEM DETAILS:
  - Model Regression Networks
  - Construction of Model Pairs
  - Proposed Method
- PAPER IMPLEMENTATION:
  - Data Processing
  - Technical Details
- STATUS OF WORK:
  - Result Replication Status
  - Modifications Proposed

PAPER IMPLEMENTATION - Data Processing

- Training Model Set:
  - Generated using ILSVRC (ImageNet large scale visual recognition challenge) 2012 training data
  - 1000 object categories with 600-1300 images per category (≈ 1.2 million images)
- Pre-Processing Steps:
  - For generation of w^* for each category, use all positive images and nearly equal number of negative images randomly sampled from all other categories
  - w^* is trained with the optimal SVM regularization parameter using 10-fold cross-validation
  - For generation of w^0, we use N = 1, 2, ..., 9, 10, 15, 20, ..., 100 random samples (sub-sampling is repeated 5 times for each value of N)
  - w^0 is trained using different SVM regularization parameters from $10^{\{-2, -1, 0, 1, 2\}}$
  - 700 generated model pairs for each category (hence, size of training model set is 700000)

PAPER IMPLEMENTATION - Data Processing

- A few statistics about the ILSVRC dataset: much larger than other image classification datasets; contains more fine-grained classes compared to PASCAL VOC dataset

Image classification annotations (1000 object classes)

Year	Train images (per class)	Val images (per class)	Test images (per class)
ILSVRC2010	1,261,406 (668-3047)	50,000 (50)	150,000 (150)
ILSVRC2011	1,229,413 (384-1300)	50,000 (50)	100,000 (100)
ILSVRC2012-14	1,281,167 (732-1300)	50,000 (50)	100,000 (100)

Additional annotations for single-object localization (1000 object classes)

Year	Train images with bbox annotations (per class)	Train bboxes annotated (per class)	Val images with bbox annotations (per class)	Val bboxes annotated (per class)	Test images with bbox annotations
ILSVRC2011	315,525 (104-1256)	344,233 (114-1502)	50,000 (50)	55,388 (50-118)	100,000
ILSVRC2012-14	523,966 (91-1268)	593,173 (92-1418)	50,000 (50)	64,058 (50-189)	100,000

- There is a large diversity in the ILSVRC 2012 data based on multiple parameters like object scale, number of instances, image clutter, real world size, etc
- More details in the ILSVRC whitepaper

PAPER IMPLEMENTATION - Technical Details

- Feature Space: Caffe AlexNet CNN pre-trained on ILSVRC 201; all weights are frozen from this data without fine-tuning on any other datasets
- Feature Extraction: For each 256 x 256 image, the 224 x 224 center crop is used to extract features
- Solver: Liblinear is used to generate linear SVM classification models w^0 and w^*
- Training: 700 model pairs generated per category - 685 pairs are used for training and 15 are used for validation per category
- Model: Regression network trained using Caffe; number of units per dense layer is 6144, 5120, 4097, 4097 respectively; negative slope for leaky ReLU is 0.1
  - Loss function is implemented as two layers in Caffe - one focuses on regression accuracy (based on squared Euclidean distance), other on performance loss on training data

PAPER IMPLEMENTATION - Results

- Model evaluated on ILSVRC validation dataset containing 50 images per category - no overlap with training dataset
- Comparison against baselines: All approaches use prior knowledge from the ILSVRC source domain for features, model parameters and fine-tuning for few-shot learning

Source Prior Knowledge Type	Method	Acc (%)
NA	SVM (target only) [43]	62.28
Data	SVM (source only) [43]	53.51
	SVM (source and target) [43]	56.68
Feature	GFK [34]	65.16
	SA [28]	59.30
	Daumé III [17]	59.21
	MMDT [42]	59.21
Model Parameter	PMT [2]	66.30
	Late Fusion (Max) [43]	59.59
	Late Fusion (Lin. Int. Avg) [43]	60.64
Joint	Fine-tuning [43]	61.13
Model Transformation	Model Regression Network (Ours)	**68.47**

OUTLINE

- PROBLEM STATEMENT:
  - Overview
  - Technical Details
  - Previous Works
- PROBLEM DETAILS:
  - Model Regression Networks
  - Construction of Model Pairs
  - Proposed Method
- PAPER IMPLEMENTATION:
  - Data Processing
  - Technical Details
- STATUS OF WORK:
  - Result Replication Status
  - Modifications Proposed

STATUS OF WORK: Result Replication Status

Implemented code from the following GitHub repository to gain a better understanding of the classification models on the ILSVRC dataset:

- longrootchen/ILSVRC-2012-classification-pytorch: ILSVRC 2012 classification playground (github.com)
  - Re-implementation of CNNs on the ILSVRC 2012 training dataset using PyTorch (evaluation metrics are top-1 and top-5 error rates)

```
Top-1 error rate (original): 0.407
Top-1 error rate (re-implemented): 0.412
Top-5 error rate (original): 0.182
Top-5 error rate (re-implemented): 0.182
```

  - Work contribution for basic implementation of classification code: Om Prabhu

STATUS OF WORK: Result Replication Status

Implemented basic classification code in TensorFlow using documentation from the following GitHub repository:

- models/README.md at master · tensorflow/models (github.com)
  - Involved additional steps to convert dataset to a TFRecord format for processing in TensorFlow
  - Used a TF dataset descriptor to store pointers to several pieces of metadata (class labels, train/test split, etc)
  - Automated script for processing ImageNet data
  - Implemented the Inception-ResNet-v2 model code which achieved a 80.4% top-1 accuracy and 95.3% top-5 accuracy
  - Work contribution for alternative implementation of classification model in TF: Om Prabhu

For future implementations, we will try to build a model from scratch instead of using automated scripts for ImageNet data processing

STATUS OF WORK: Modifications Proposed

A few modifications proposed to the model described in this presentation are:

- Instead of using fully connected/dense layers in the regression network, we can try implementing CNNs. This could probably improve the quality of feature extraction from the images and lead to better model performance
- Instead of a SVM classifier, we could explore other types of classifiers such as those that use logistic regression - this would also require converting the hinge loss for the SVM classifier to a logistic loss

We are yet to decide which of these modifications to implement - we would be trying both of them and implementing whichever is easier/shows more pronounced results

REFERENCES

- Research Papers
  - Learning to Learn: Model Regression Networks for Easy Small Sample Learning (Wang, Habert - 2016)
  - Tabula Rasa: Model Transfer for Object Category Detection (Aytar, Zisserman - 2011)
  - Uncovering Shared Structures in Multiclass Classification (Amit, Fink, Srebro, Ullman - 2007)
- GitHub Repositories
  - https://github.com/tensorflow/models/blob/master/research/slim/README.md
  - https://github.com/longrootchen/ILSVRC-2012-classification-pytorch

Disclaimer: The work presented in this document are in no way the work of Om Prabhu and/or Naman Gupta, unless explicitly specified (Work Status Section).