Object detection using Semi-Supervised Classification

Shubham Kumar Agrawal [1] Pavan Tummala [1] Usama Ashfaq [1] Ranjiraj Rajendran Nair [2] Syed Muhammad Laique Abbas [2]

[1]Faculty of Computer Science: Digital Engineering [2]Faculty of Computer Science: Data and Knowledge Engineering

Motivation and Problem Statement

Motivation: The Proceedings of the European Conference on Computer Vision has conducted a benchmark PASCAL Visual Object Challenge (VOC) evaluating performance on object class recognition (from 2005-2012, now finished). For our task, we examine the *VOC07* dataset which consists of several types of random images collected in January 2007 from *Flickr*. There are five challenges: classification, detection, segmentation, action classification, and person layout.

Problem Statement: Our goal from this challenge is to perform **image classification** from several visual object classes in realistic scenes (i.e. not pre-segmented objects). And so, we will be using certain Semi-Supervised Learning approaches where we have both labeled and unlabeled sets to check if we get superior results as opposed to supervised techniques.

Data set

The PASCAL Visual Object Classes Challenge (VOC 2007) data set consists of about 9,000 256x256 RGB images of 20 classes. The twenty object classes that have been selected are:

- **Person**: person
- **Animal**: bird, cat, cow, dog, horse, sheep
- **Vehicle**: aeroplane, bicycle, boat, bus, car, motorbike, train
- **Indoor**: bottle, chair, dining table, potted plant, sofa, tv/monitor

We represent the images in an array with their labels. The labeled set is partitioned into various split points ranging from [0.1,0.9] for our SSL strategy.

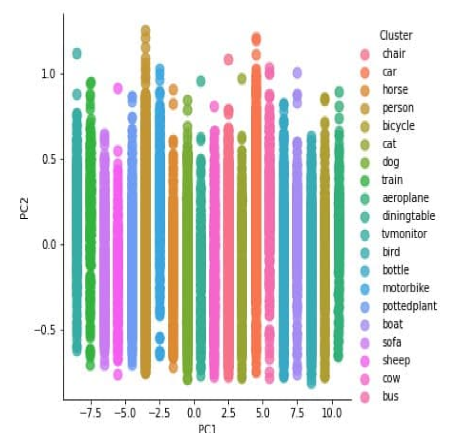

Figure 1. Feature space by PCA

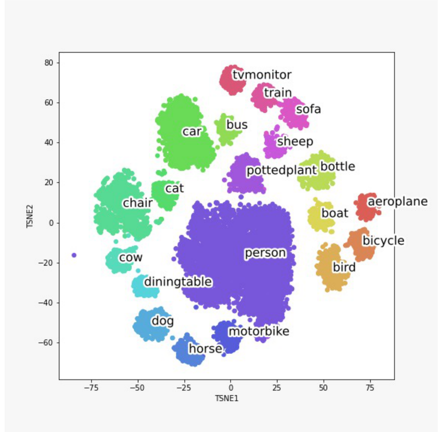

Figure 2. Feature space by t-SNE

SSL Concept

The SSL methods we proposed for solving our task as follows:

- (*Graph-based*) **Label Propagation Algorithm (LPA)**: *Assumption* - Similar images would have similar feature descriptors and so they would be mapped closely in the graph with high weights to the edges connecting to them.
- (*Graph-based*) **Label Spreading Algorithm**: *Manifold Assumption* - the graphs, constructed based on the local similarity between features, provide a lower-dimensional representation of the high-dimensional input images (images on the same low-dimensional manifold should have the same label).
- (*Inductive*) **Semi-Supervised Gaussian Mixture Model (SSGMM)**: *Assumption* - The images come from the mixture model, where the number of features, prior $p(y)$, and conditional $p(x|y)$ are all correct.

Advantages: It highly reduces the amount of annotated data used for our task.
Disadvantages: Since all stated models of ours use an iterative approach for updating it can converge to local minima leading to less stable results.

Implementation

For our task we used **Python 3+** for implementing our concepts. The open tool-kits that we used for our development purposes are primarily ***Jupyter Notebook*** and ***Spyder IDE*** from the ***Anaconda*** distribution software.

We implemented a ***Data Loader*** pipeline where we feed 15,500 extracted images in an array along with their labels which goes for train-test-validate partition. We extracted the 3 mandatory features: *MPEG-7 Color Layout Descriptor, Visual Bag-of-Words (BOV), **S**peeded **U**p **R**obust **F**eatures*. For the first feature, the process comprised of 4 phases namely: *Image partitioning, Representative color selection, DCT transformation, Zigzag scanning*. For the second and third features combined, we extracted the features for each of the images using the functionalities provided by python *OpenCV* library. Furthermore, we constructed a codebook vector using the *k*-means clustering algorithm of a certain vocabulary size using the extracted features. For each feature, we assign a code from the codebook, and then produce a histogram for it, which we used as a feature to the model. For the extra feature, we have the *Local Binary Patterns* and *Color Histogram*.

Figure 3. SURF key point visualization for images

Evaluation

Model selection

On the features we perform class balancing (*undersampling*), feature selection (*ANOVA*), feature reduction (*PCA*) with an enumeration of 2^c (where c is the total extracted features). Based on this, we analyze from a line plot for all the split points ranging from [0.1,0.9] which combination is better to be used for model building. Followed by it we effectively determine from a box plot to pick the best model.

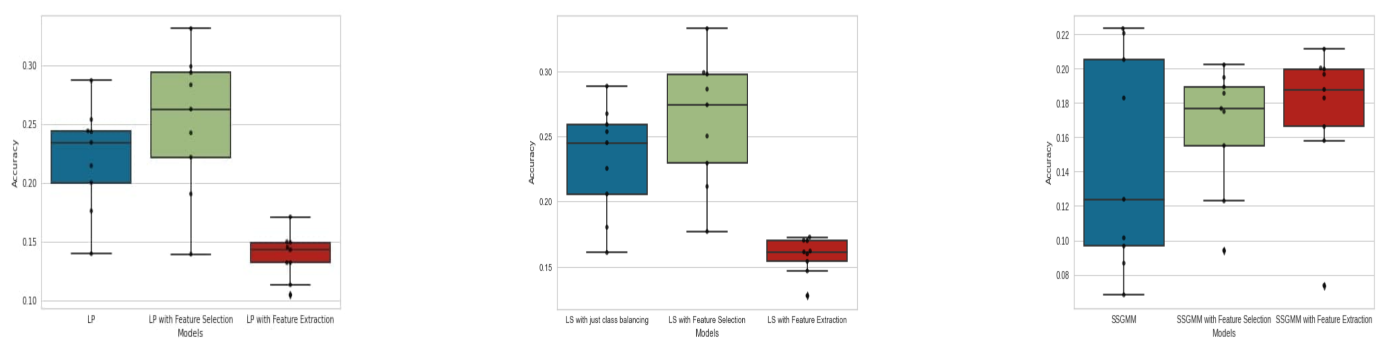

Figure 4. Box and Whisker plots for *Label propagation, Label spreading, SSGMM*

Model evaluation

We are showing a comparative study of *Label Propagation, Label Spreading, SSGMM* on different splits of the data from [0.1,0.9] on the features that we implemented.

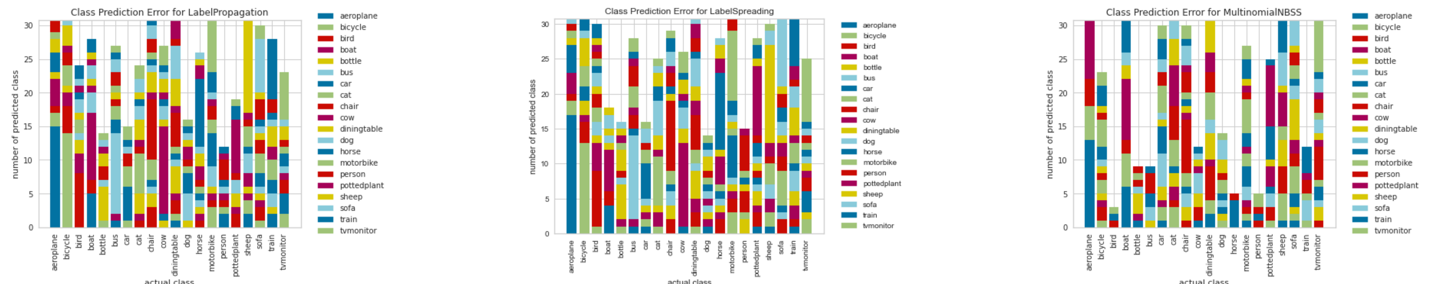

Figure 5. Class prediction error for *Label propagation, Label spreading, SSGMM*

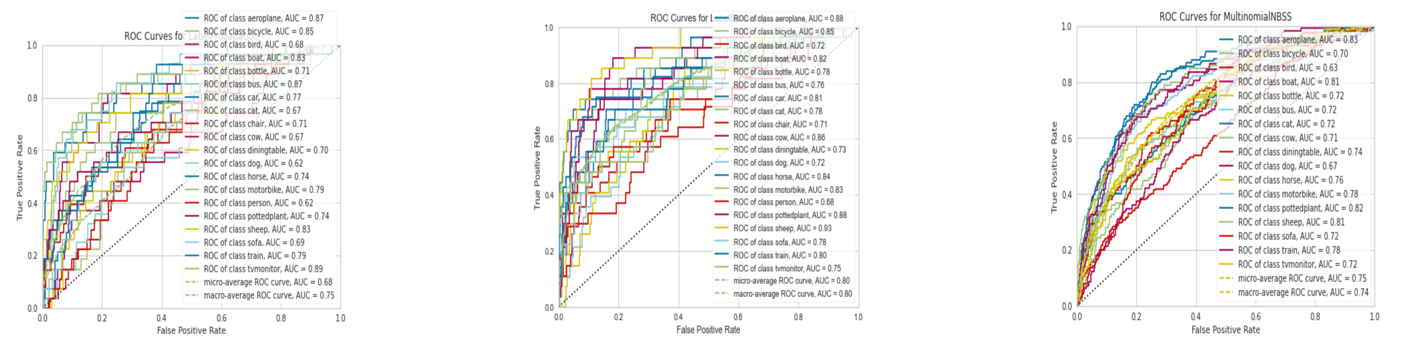

Figure 6. ROC curves for *Label propagation, Label spreading, SSGMM*

We check for 2-feature and 3-feature combinations for all the models to determine the best model with the lowest complexity. We test for 1,500 iterations by tuning parameters α,β,γ and *choice of kernel*. We use **ROC/AUC curves** for every possible cut-off for combination of tests with different feature combinations. Furthermore, we use **class prediction error plot** which shows the actual targets from the dataset against the predicted values generated by our model.

Conclusion

For our task the best SSL model generalization performance is achieved for *Label spreading* with an accuracy of nearly 31%.

Algorithms	Class balancing	Class balancing+ feature selection	Class balancing+ feature reduction
Label Propagation	26.5±2.6%	29.4±2.8%	16.3±2.7%
Label Spreading	27.4±3.1%	31.2±3.4%	17.1±1.2%
SSGMM	22.8±2.8%	20.4±2.2%	20.4±1.7%

Table 1. An overview of the results of our SSL techniques with 95% C.I.

Safe SSL

We trained both on the semi-supervised and supervised models by splitting the data with varying labeled and unlabelled test sizes and compared it with the fixed test set. We then compared the efficiency of SSL to the supervised approach and concluded that the "Safe SSL" assumption **does not** hold as it performed better for the semi-supervised model at the split test data.

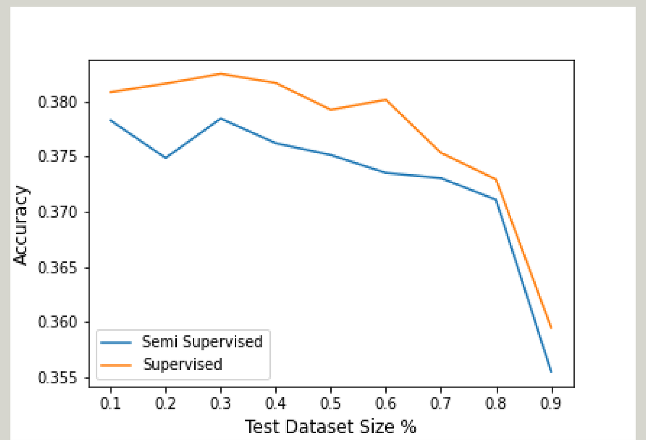

Figure 7. Safe SSL check

Baseline comparison

We compare our work with a similar dataset on *BOV* features which was implemented using additive and exponential kernel-based supervised SVM classifiers. Their performances reported were in the range of 48.9%-52%. In comparison, our SSL techniques attain nearly 31% which are not superior to supervised counterparts.

Future work

- **Feature refining**: Extract relevant features by using Deep Learning-based techniques like Convolutional Neural Networks.
- **Model ensemble**: Combining classifiers by voting or averaging to improve performance.