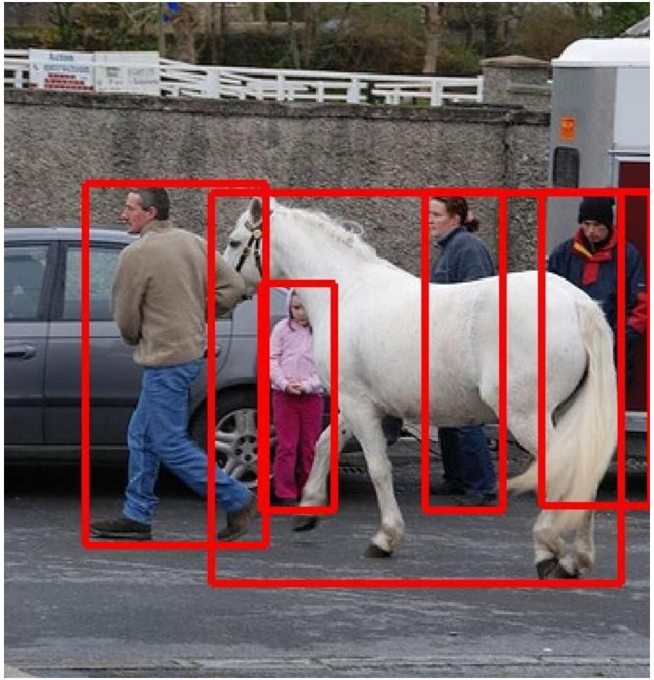
A Novel Representation for Object Detection

Vishakh Hegde and Manik Dhar

Motivation - high level



Objective: Given an image, get a tight bounding box for all the objects in the box

Use cases:

- Self driving cars (detect people)
- Medical Imaging (detect tumor)
- Satellite Imaging (regions of poverty)

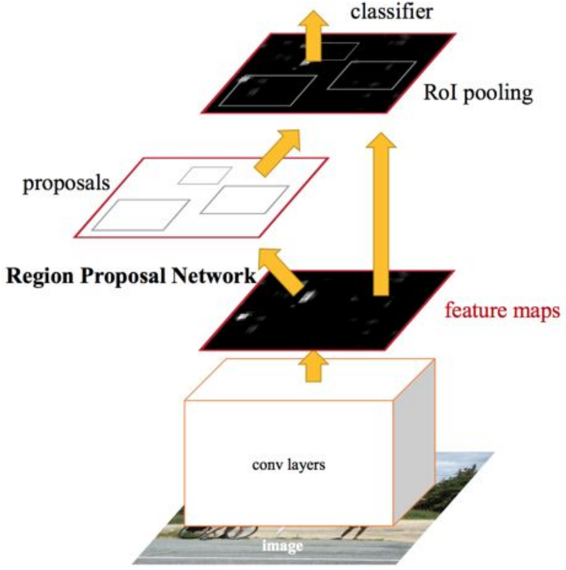
Hence, important to get good representations to detect objects

Related Work and State of the Art

Faster RCNN: Built on top of a standard neural network for classification

Additional layers: RoI pooling, Region Proposal Network

Note: They treat background as just another class



LSDA: Large Scale Detection Through Adaptation

Adapt a classification network for detection

Motivation - Specifics

Observation:

Background is special and is present in all images

Treating Background as a category is not elegant

Question:

Better representation that inherently encodes information about background?

Object

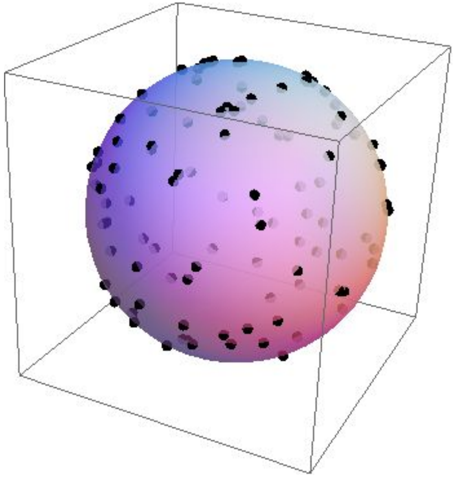
Background

Proposed Solution

Encode background information in final features layer:

Neurons activate for objects

Do not activate for background



Another picture:

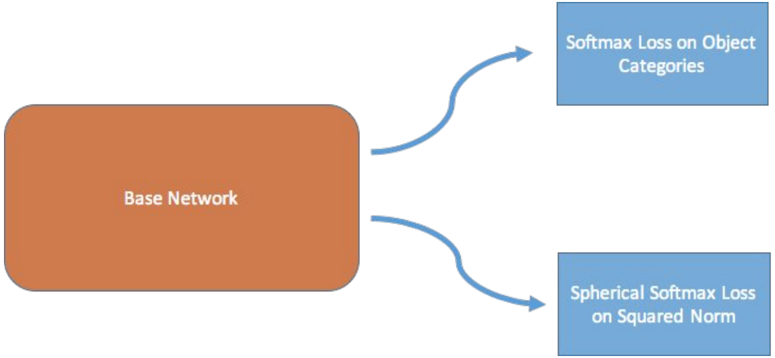
Push background to the origin in a high dimensional feature space

Pull objects to the surface of a unit hyper-sphere

Intuition:

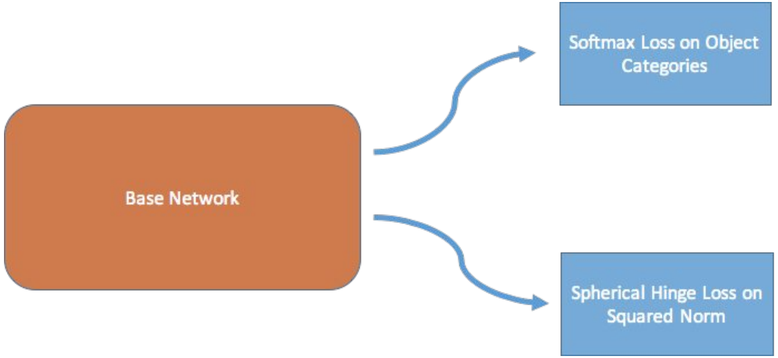
Distribute background information in the network

Our Loss Functions



$$\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m} 1\{y_j^{(i)}=1\}\log(\text{softmax}_\theta(\phi(x^{(i)}),j))$$

$$\text{softmax}_\theta(x,j)=\frac{e^{\theta_j^T\phi(x)}}{\sum_{k=1}^{m} e^{\theta_k^T\phi(x)}}$$



$$\frac{1}{n}\sum_{i=1}^{n}\{(\|\phi(x^{(i)})\|_2^2-1)(-1)^{1\{\|y^{(i)}\|_1=1\}}\}_+$$

where

$$\{x\}_+ = x \text{ if } x>0, \text{ else } 0$$

Introduction

Dataset used: PASCAL VOC 2012 train and val

Engineering Challenges:

Limited compute power and memory

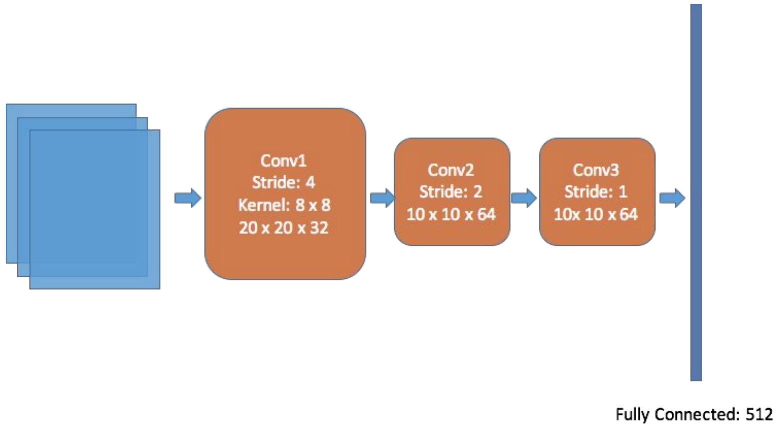
We use:

3 Layer Convolutional Neural Network

A subset of PASCAL VOC for train and val

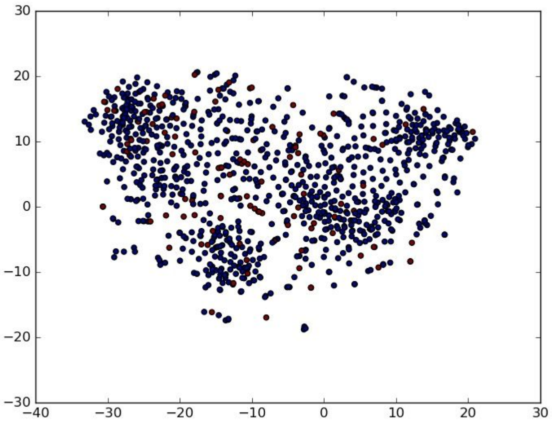
Selective Search boxes from RCNN

90% background, 10% objects: realistic setting

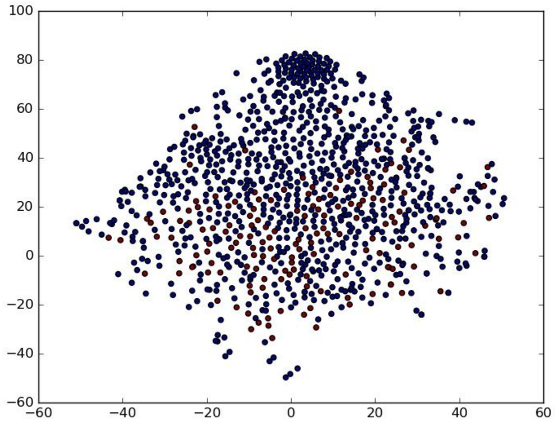


Base network used

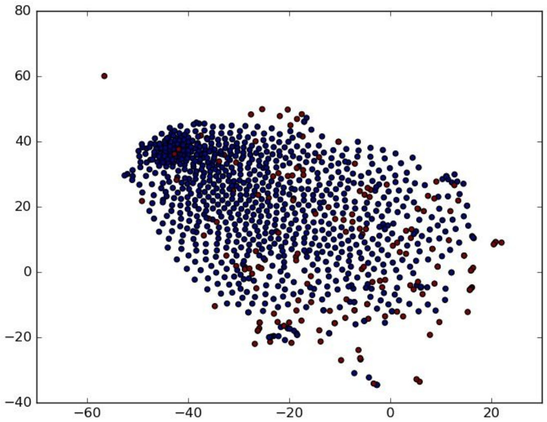
Experiments - tSNE Visualization



Loss used: Softmax
(assuming background to be
another class)

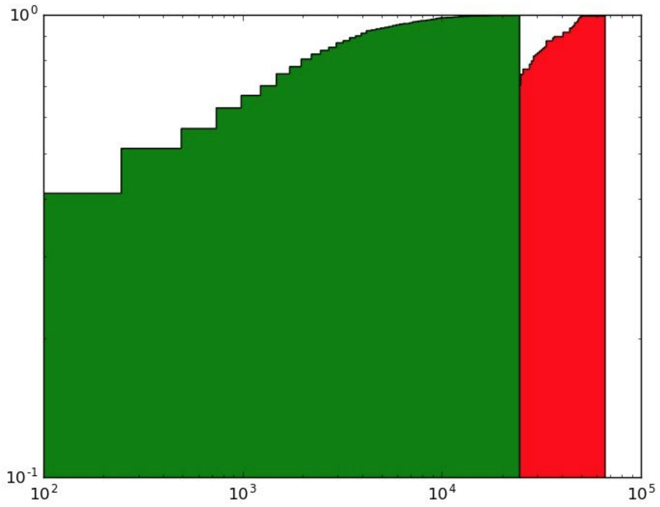


Loss used: Spherical Softmax

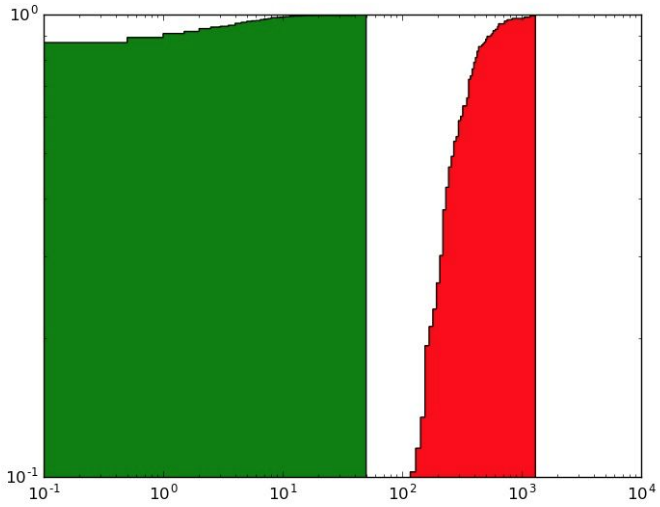


Loss used: Spherical Hinge

Experiments - Visualization



X axis: Squared norm of features
Y axis: Normed Cumulative Frequency
Loss used: Spherical Softmax



X axis: Squared norm of features
Y axis: Normed Cumulative Frequency
Loss used: Spherical Hinge

Experiments - Observations

Loss Type	Classification accuracy (given object)	Classification accuracy	Object or not classification acc
Regular Softmax Loss 2	0.1339	0.764	0.793
Spherical Hinge Loss 4	**0.348**	**0.755**	**0.801**
Spherical Hinge Loss 5	0.241	0.646	0.706
Spherical Hinge Loss 6	0.241	0.717	0.764
Spherical Softmax Loss 1	0.241	0.653	0.716
Spherical Softmax Loss 2	0.259	0.346	0.423
Spherical Softmax Loss 3	0.205	0.722	0.772

We report the ‘best’ validation scores for each model, over 100 epochs

Using spherical hinge loss seems to perform better than treating background as another class

Next Steps - short term

Use a bigger base network and train on full PASCAL VOC dataset

Pretrain base network on ImageNet

Use LwF loss function to transfer knowledge from classification to detection

Next Steps - long term

Make it faster - use RoI and RPN layers

Make it open source

Thank you!