DOG BREED CLASSIFICATION

GROUP –2

SRUJAN SWAROOP - IMT2016033

SIDDARTH REDDY - IMT2016037

DURGA YASASVI - IMT2016060

ROHITH YOGI - IMT2016072

OBJECTIVE

The aim of this project is to detect the dog by localized bounding boxes and categorize them into their respective breed.

DATASET

The Stanford Dogs dataset contains images of 120 breeds of dogs from around the world. This dataset has been built using images and annotation from ImageNet for the task of fine-grained image categorization :-

- Number of categories: 120
- Number of images: 20,580
- Annotations: Class labels, Bounding boxes
- Number of train Images: 12000
- Number of test images: 8580

OVERVIEW

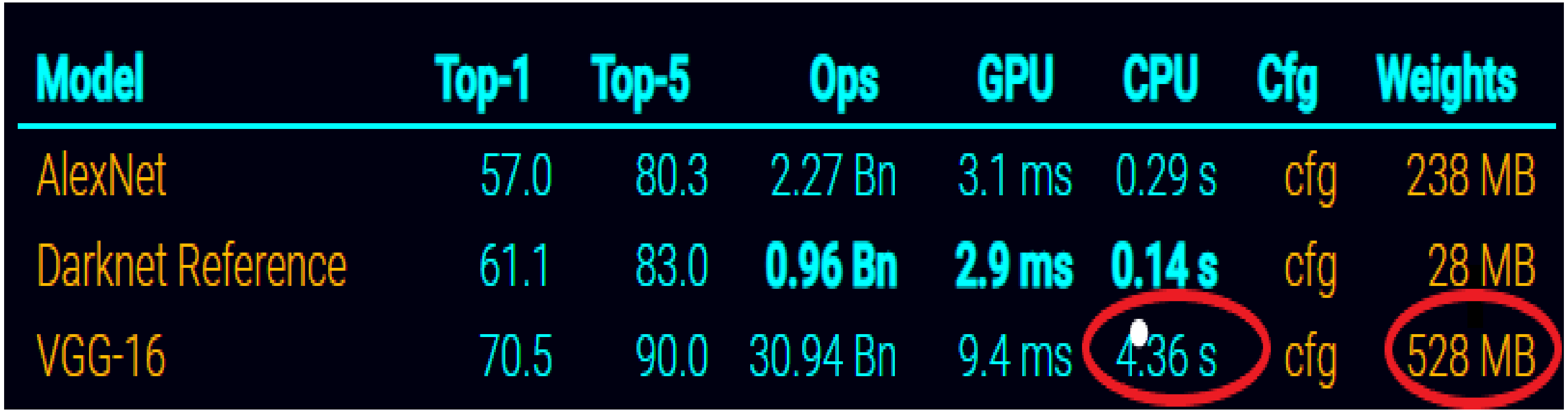
- This is a fine grained detection and classification problem.
- Therefore uses YOLO like model to localize bounding boxes and classify the detected dog. We use YOLO-v1 for object detection and classification.
- YOLO has set of pretrained convolutional layers followed by randomly initialized convolutional layers and fully connected layers.
- Since images of our dataset are directly annotated from ImageNet dataset hence the first few convolutional layers are pre trained on ImageNet dataset.

APPROACH

- As you might have noticed, having only ~20K images of 120 different breeds (~200 images per breed) is not enough to train a deep neural network. Convolutional Neural Network (CNN) here is best option for classification but there are not enough training examples to train it.
- Hence the solution is to use pre-trained network.
- The pre-trained network used is **VGG-16** network instead of conventional Darknet trained on ImageNet Data Set.
- Since there is **high intra breed variation** the convolutional features should be precise enough to find the distinction between different breeds.
- Therefore pretrained model which has higher accuracy on ImageNet data set is preferred.

NEED FOR TRADE-OFF

- **Accuracy** over **Speed**
- VGG-16 is more accurate than Darknet but significantly slower than Darknet, hence cannot perform real time detection and classification.



Model	Top-1	Top-5	Ops	GPU	CPU	Cfg	Weights
AlexNet	57.0	80.3	2.27 Bn	3.1 ms	0.29 s	cfg	238 MB
Darknet Reference	61.1	83.0	**0.96 Bn**	**2.9 ms**	**0.14 s**	cfg	28 MB
VGG-16	70.5	90.0	30.94 Bn	9.4 ms	4.36 s	cfg	528 MB

NETWORK ARCHITECTURE

- Our network was inspired from YOLO v1 architecture.
- In our architecture, we replaced the Darknet framework with **VGG-16** pretrained on ImageNet dataset.
- Followed by the VGG-16 network we added:-
- 1 convolutional layer &
- 3 fully connected layers.

NETWORK ARCHITECTURE

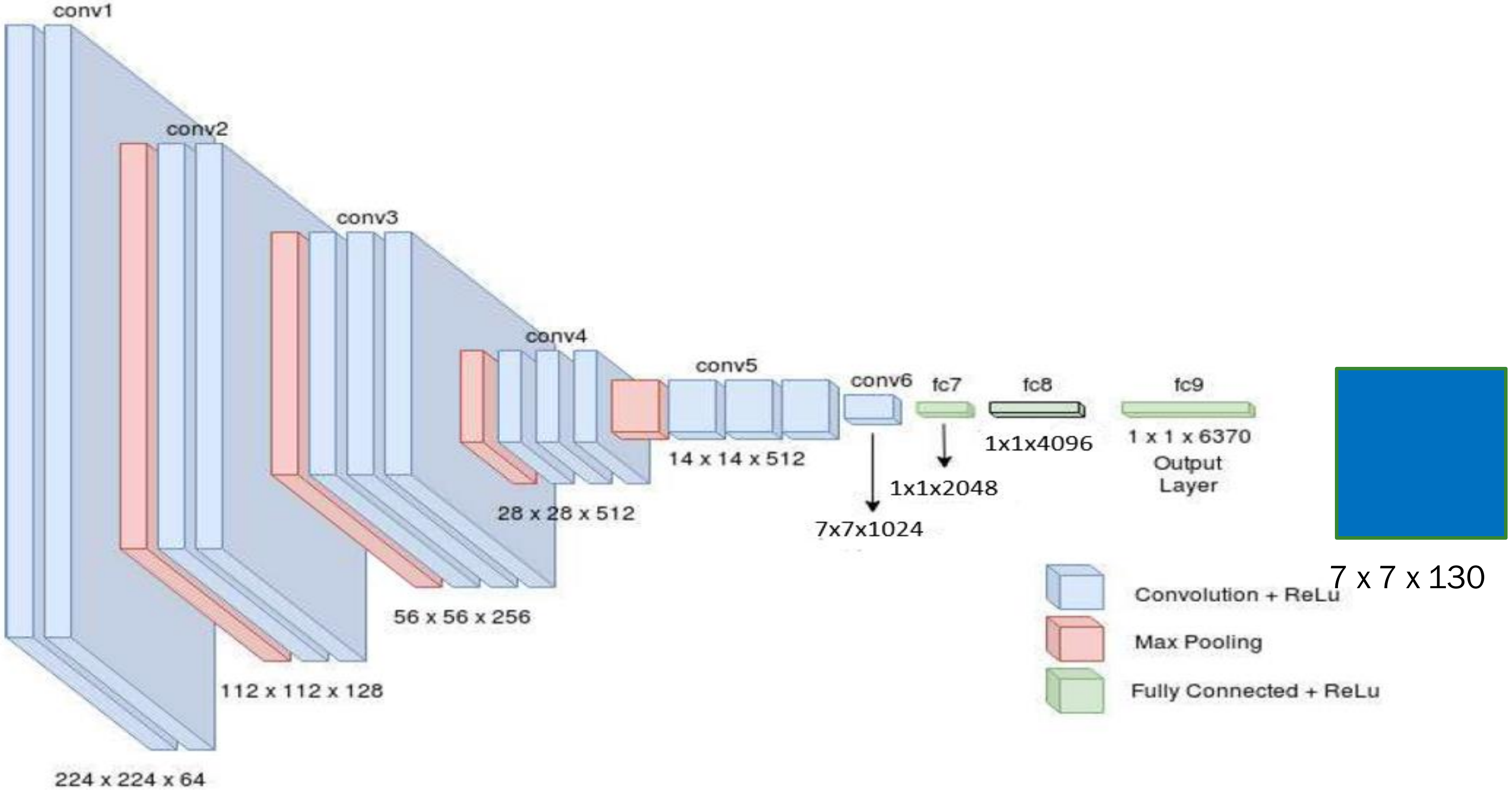


Fig. 2: Our Network Architecture

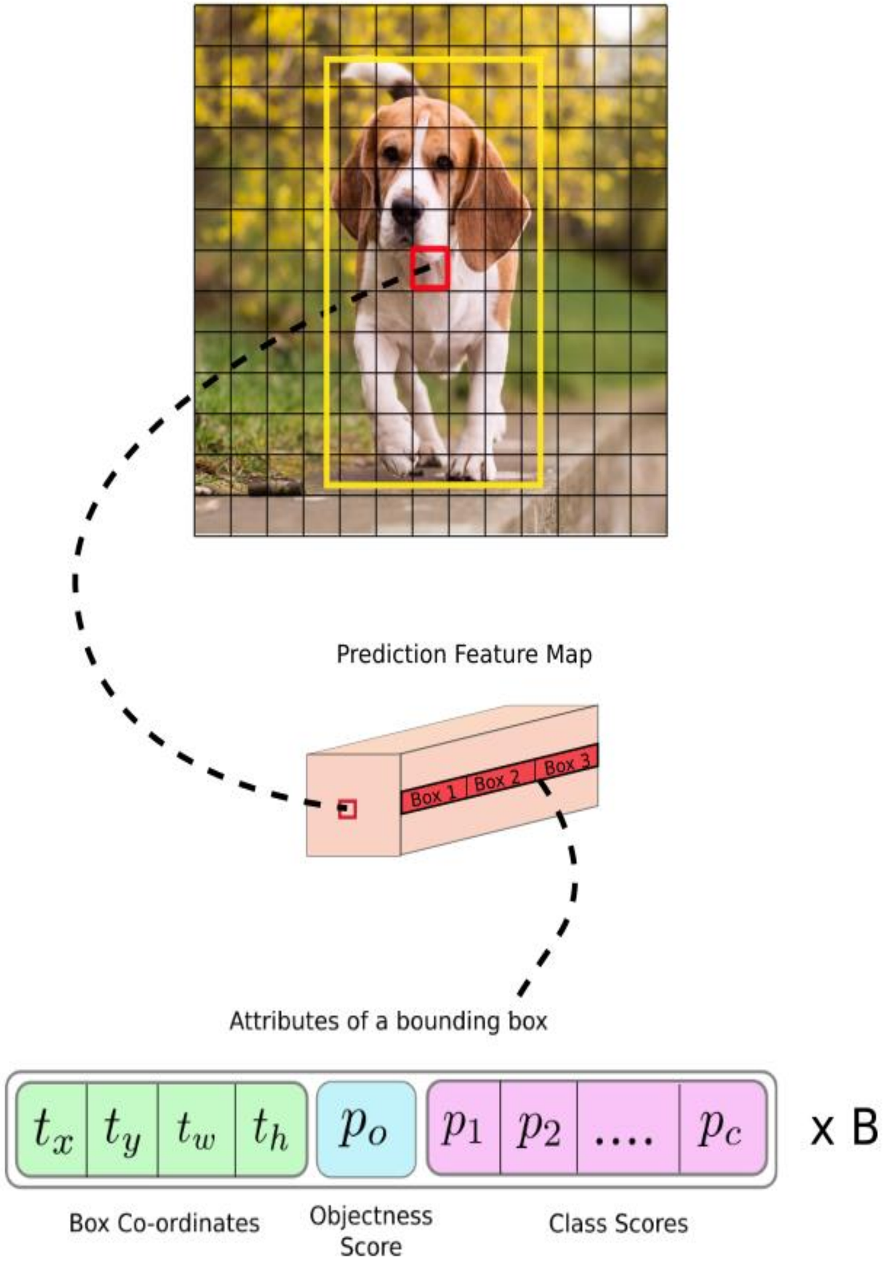
ARCHITECTURAL REASONS

The requirement of number of convolutional layers is less than fully connected layers because :-

- Since the data set is directly annotated from ImageNet dataset and VGG-16 is already pre-trained on ImageNet for classification task.
- Hence the convolutional features required for classification is already learnt ,therefore additional fully connected layers are added to localize the dog by regressing the bounding box co-ordinates for each grid present in the image.
- Since pretrained network is used the task now orients more towards detection rather than classification.

NETWORK ARCHITECTURE

- As it is dog breed specific classification task, the variability between classes is very low hence the requirement of number of convolutional layers after pretrained network is not abundant as features of the dogs are assumed to be learnt during pretraining of VGG-16 on ImageNet dataset.
- Therefore the intention of adding layers turned more towards object detection rather than pure classification. Hence number of convolutional layers added are less as compared to fully connected layers as bounding coordinates needed to be regressed
- The output is reconstructed into *7 x 7 x 130* tensor which encodes all 5 parameters information for all the bounding boxes.


Prediction Feature Map
Box 1
Box 2
Box 3
Attributes of a bounding box
t_x t_y t_w t_h
p_o
p_1 p_2 .... p_c
x B
Box Co-ordinates
Objectness Score
Class Scores

OUTPUT-TENSOR

Final output layer outputs a tensor which has following information.

- Predictions for each grid cell for B bounding boxes
- For each bounding box {x, y, w, h} is predicted along with the corresponding confidence score which tells about how confidence about how accurate the predicted bounding box is.
- Along with above predictions each grid cell also predicts C class conditional probabilities.
- This information is encoded as S x S x(B x 5 + C) tensor. As S is size of each grid, B is number of bounding boxes predicted for each grid cell ,C is the number of classes.

DIMENSIONS OF OUTPUT TENSOR

- grid size(S)=7.
- number of bounding boxes predicted for each grid(B)=2
- number of dog breed classes(C)=120
- Therefore our output tensor is in the form 7 x 7 x (2 x 5 + 120)

  =49 x (10 + 120)

  = 49x130

  =(6370) .
- Dimension of output tensor: 1x 1 x 6370

TRAINING

Transfer learning:

We chose VGG-16 network trained on ImageNet as the dog breed dataset is quite similar to the ImageNet dataset. Also as our dataset is quite heavy as well, we used transfer learning on these pretrained networks for training. So we remove the 3 layers and treat the rest of the network as a fixed feature extractor for our problem specific dataset.

FINE-TUNING

- As our dataset is large enough and has very similar data to ImageNet, we finetune a few layers and with a very few learning rate we train the remaining layers.So,in our model the lower layers contain may be features of dog and higher layers contains breed specific information.
- So to prevent over-fitting and preserve generalization we freeze the lower layers of pre-trained network. To implement this property we enforce a very low learning rate such that loss gradient is not propagated to lower layers of the network.
- Hence our learning rate is :-

learning – rate= 0.0001

- Loss Function: Same loss function was used as given in YOLO v1.

TRAINING

The training is carried through the following mentioned configuration:

- Model trained for 110 epochs.
- Each epoch consisting of 300 iterations.
- Total number of iterations for which the model trained is 33,350 iterations descent is carried through mini batches of size-32(Mini batch gradient descent).
- Each image sample is used in training in normalized form.
- Batch normalization is applied for all the batches and batches of image samples are selected in random fashion.
- Added a dropout layer with dropout probability 0.5 just before final output layer which helps as a regularizer.

TESTING

Inference: We pass the test image through the network which outputs the *7 x 7 x 130.* Now from this tensor we need to perform some post processing to get the final output. We perform confidence thresholding and Non maximum suppression.

Confidence Thresholding: We consider only the predictions above a certain threshold (around 0.2). All the bounding boxes with confidence scores less 20% are not considered further.

Non Maximum Suppression: The main purpose of non-maximum suppression is to pick one of the multiple candidates for object. So it eliminates some candidates that are different detections of the same object. It makes sure that the object is identified only once.

After Non maximum suppression, we output the optimal bounding box fit for the dog present in the image and the class corresponding to the highest class conditional probability.

NON MAXIMUM SUPPRESSION

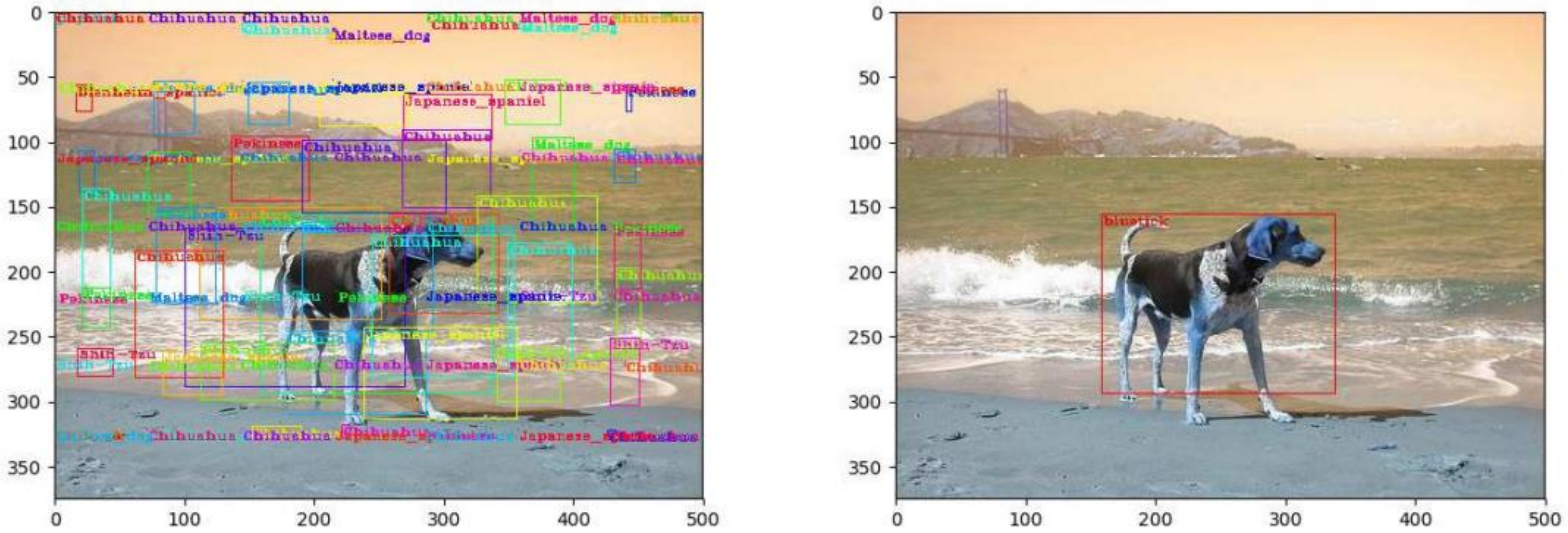


Fig. 6: Image before and after the Non Maximum Suppression

COMPARISONS

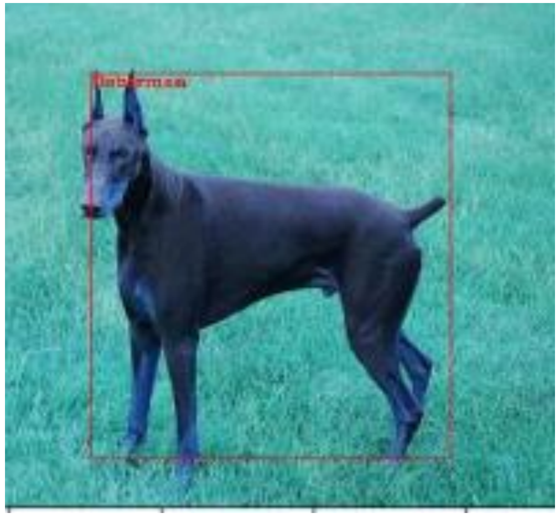
We trained on different pretrained networks like VGG-16, AlexNet and YOLO-tiny:

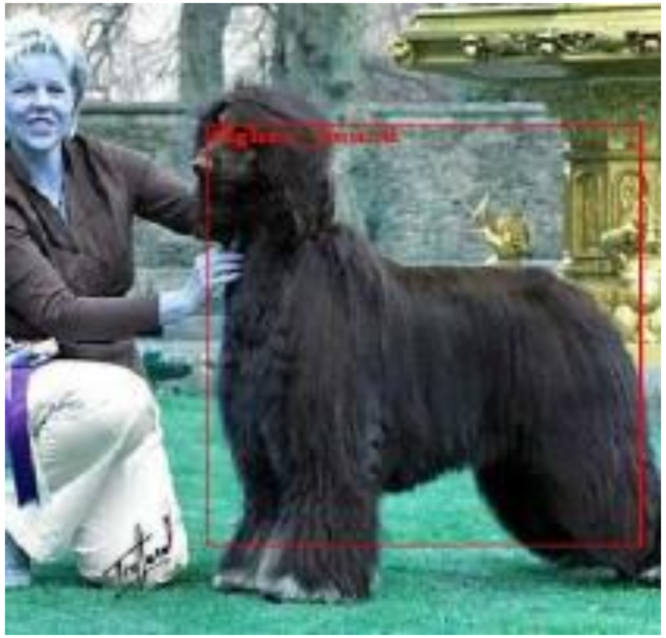
Following are the mAP accuracies obtained for different networks:

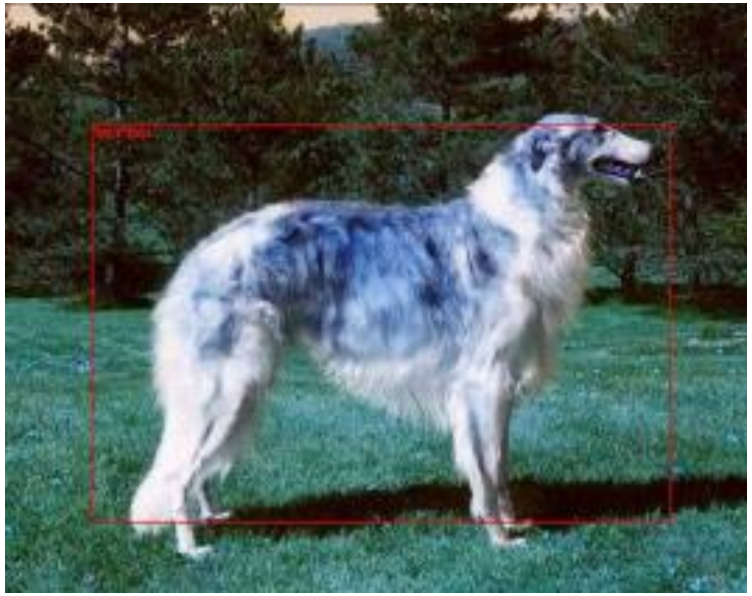
- VGG-16: 72.12%
- YOLO-tiny: 65.89%
- AlexNet: 58.23%

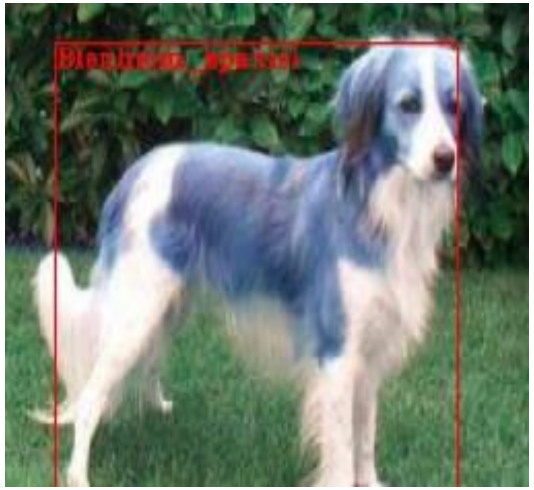
VGG-16 was significantly more accurate than remaining pretrained networks. But in terms of computation darknet was effectively faster than VGG-16. Compared to the rest Alexnet exhibited poor performance.

FEW PREDICTED OUTPUTS









FEW PREDICTED OUTP UTS

Here are the results where there are people present, posture, occlusion, different poses, etc...

But it failed for the case where there are two dogs present together. The model detected it as a single dog.