Species Classfication for Birds using Labelled Image Corpus

Kaveri Gupta(u1077655)
Sagar Chaturvedi(u1068847)

1: Introduction

The problem at hand is identification of bird species given an image. This is an image classification problem motivated by the following facts:

1. Although bird species have mostly similar set of body parts but they vary significantly in shape and appearance and are visually indistinguishable, even to expert bird watchers.
2. Depending on the lighting, background and pose, it's difficult to identify birds belonging to the same class..

The dataset that we used is CalTech dataset, CalTech-UCSD Birds-200-2011 (http://www.vision.caltech.edu/visipedia/CUB-200-2011.html), by Wah, C. and Branson, S. and Welinder, P. and Perona, P. and Belongie, S. from 2011. It contains 11,788 images of 200 bird species.

2: Features

We conducted our experiments on 3 types of features as given below:

1. *Hand-crafted features (included in the dataset)*
   Each image in the dataset has 28 attributes, based on 15 body parts of the bird, that have been converted to 312 binary features. All attributes are visual in nature, related to color, shape or pattern of the body part. Each image is also annotated with details of bounding boxes, part locations and attribute labels.

Body Part	Attribute	Binary Features
Beak	Has_Bill_Shape	has_bill_shape::curved_(up_or_down)
		has_bill_shape::dagger
		has_bill_shape::hooked
		has_bill_shape::needle
		has_bill_shape::hooked_seabird
		has_bill_shape::spatulate
		has_bill_shape::all-purpose
		has_bill_shape::cone
		has_bill_shape::specialized

2. *SIFT features*
   We also extracted features from images using SIFT technique implementation of OpenCV

and used keypoint averaging and key-point clustering to get two different sets of features for each image. We did our baseline experiments with the key-point averaged SIFT features.

3. *Tensorflow Features*
   Since SIFT features did not give us very promising results, we used tensorflow InceptionV3 to extract features. It returned 1009 features per image. We ran similar experiments on these features too.

3: Experiments and Results

Since this is a multi-class dataset, we modified the SVM and logistic regression developed as a part of course to accomodate multiple classes. We developed one-vs-rest classifier and pair-wise classifiers for the same. We executed these codes (and a few more from scikit-learn) on aforementioned featuresets and got the results as follows:

1. *Hand-crafted features (included in the dataset)*
   We ran the following experiements on the features that came with the dataset:

   (a) ***Experiment 1:*** *Decision Tree (Developed by us) - ID3*
   We used only 50 categories to test this since it was taking a lot of time for 200 categories. We used **entropy** to calculate information gain. We got good results with maximum accuracy being **56.311%** at a depth of 17. Following are the results with various max-depths of the tree:

Depth	Accuracy
2	15.922%
5	42.718%
8	54.369%

   We observed that the accuracy dropped once we limited the depth of the tree.

   (b) ***Experiment 2:*** *Decision Tree (Python scikit-learn) - ID3*
   Here we used **GINI** to calculate information gain and achieved an accuracy of **40.611%** for **50 labels** with unbounded depth as 42. Following are the results with various max-depths of the tree:

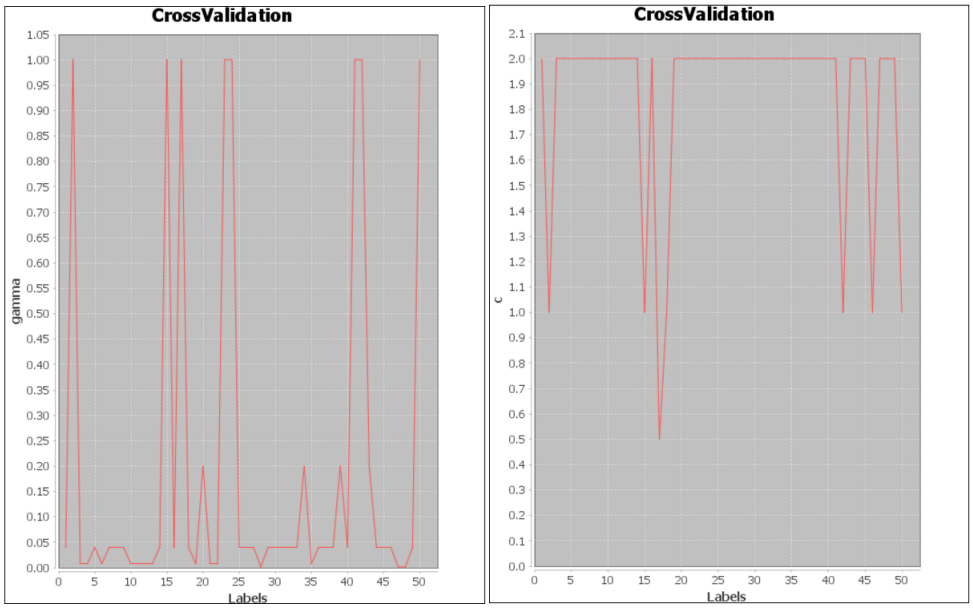
Depth	Accuracy
2	2.076%
5	11.419%
8	18.512%
20	38.754%
42	40.611%

   We observed that the accuracy dropped once we limited the depth of the tree.

   (c) ***Experiment 3:*** *One-vs-Rest using SVM (Developed by us)*
   We developed our own one-vs-rest classifier and achieved accuracy of **25%** without cross-validation. By using a 5-fold cross validation we were able to achieve an accuracy of **31.677%**. We tested with 5 values of both γ_0 and c.

   In One-vs-rest, we created a classifier for each label. Following are the plots of best

values (observed using 5-fold cross validation) of learning rate (γ) and hyperparameter (c) for the classifier of each label:



(d) ***Experiment 4:*** *Pair-wise classification using SVM (Developed by us)*
We randomly picked out two labels and trained our SVM (using cross validation) on them. This was done for 10 such pairs. The model achieved a maximum cross-validation accuracy of 100% for some label pairs. We did this experiment with 50 categories. Since the one-vs-one classification would create 1225 classifiers for 50 labels, we did not continue with it as it was computationally very expensive and time consuming. The results of pair wise are as follows:

Label 1	label 2	Learning Rate - γ	Trade-off - c	CV Accuracy	Test Accuracy
48	35	0.04	4.0	95.0	96.677%
21	14	0.2	4.0	98.333	91.667%
16	12	1.0	2.0	100.0	88.889%
29	24	0.2	2.0	91.667	88.889%
27	41	0.04	4.0	98.333	86.667%
27	2	0.2	4.0	98.333	98.333%
48	13	0.2	4.0	100.0	95.0%
3	41	0.2	4.0	100.0	91.379%
16	12	0.2	4.0	98.333	91.071%
22	4	0.0016	4.0	100	48.077%

(e) ***Experiment 5:*** *One-vs-Rest using Logistic Regression (Developed by us)*
By using a 5-fold cross validation we were able to achieve an accuracy of **32.109%**. We tested with 6 values of σ ($\sigma = 1, 2, 4, 7, 11, 16$). All the 50 classifiers selected best value of σ as 1 after cross validation.

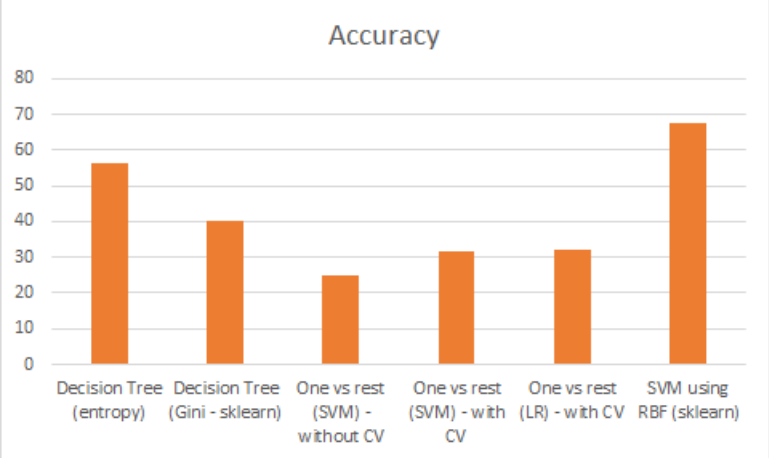
(f) ***Experiment 6:*** *Pair-wise classification using Logistic regression (Developed by us)*
Just like for SVM, here too we randomly picked out two labels and trained our Logistic Regressor (using cross validation) on it and tested the data. This was done for 10 such pairs and a maximum cross-validation accuracy of 98.214% was achieved for some label

pairs. This experiment was done with 50 categories. We did not do one-vs-one since it was taking a lot of time and computation to execute. The results are as follows:

Label 1	label 2	Learning Rate - γ	σ	CV Accuracy	Test Accuracy
28	1	0.001	2.0	86.667	93.220%
36	9	0.001	2.0	93.333	94.915%
24	18	0.001	1.0	93.333	100.0%
19	17	0.001	1.0	96.667	100.0%
36	25	0.001	1.0	93.333	96.667%
20	47	0.001	2.0	90.0	96.610%
25	14	0.001	4.0	95.0	93.333%
1	12	0.001	2.0	90.0	98.214%
18	40	0.001	7.0	95.0	62.222%
39	49	0.001	4.0	95.0	79.661%

(g) ***Experiment 7:*** *SVM with non-linear RBF kernel (scikit-learn)*
We used multi-class classifier provided in scikit-learn library of python to train our model for **50 categories** using 5-fold cross validation for 6 different values of learning rate (γ) and c and achieved an accuracy of **67.462** for $\gamma = 0.01$ and $c = 10$.

The performance comparision of various classifiers on hand-crafted features is as follows:



As it can be seen from the above bar graph, **multiclass RBF performed the best** on this feature set giving us an accuracy of **67.462**

2. *SIFT features*

(a) ***Experiment 8:*** *Decision Tree on key-point averaged features (Developed by us) - ID3*
We used **entropy** to calculate information gain for **200 categories**. The performance of the model was poor and resulted in an accuracy of **0.72%** with max tree depth as 11. Almost no change was observed after limiting the depth of the tree. Following are the results with various max-depths of the tree:

Depth	Accuracy
2	0.80%
5	0.72%
8	0.72%

Since the accuracies were so low, this experiment defined the **baseline** of our project.

(b) ***Experiment 9:*** *Decision Tree on averaged key-point features (scikit-learn) - ID3*
Here we used **GINI** to calculate information gain and achieved an accuracy of **1.44%** which was better than what we achieved from our model but was still not good enough.

(c) ***Experiment 10:*** *One-vs-Rest (SVM) on clustered key-point features (Developed by us)*
We performed key-point clustering using K-means on the features extracted using SIFT. We used different values of k (like 20, 100, 1500) to cluster the keypoints and then used one-vs-rest using SVM to train the model. We used 6 values of γ and c to do 5-fold cross validation for the model. The best accuracy we achieved was **12.278%** with 1500 clusters. Following are the results achieved:

Number of cluster, k	Learning Rate, γ	Trade-off, c	Accuracy
20	0.01	1	5.260%
100	0.01	1	7.813%
1500	0.01	1	12.278%

The performance comparision of various classifiers on SIFT features is as follows:

As it can be seen from the above experiments, the **SIFT features did not perform as well** compared to the hand-crafted features.

3. *Tensorflow Features*

(a) ***Experiment 11:*** *SVM with non-linear RBF kernel*
We used RBF kernel to train a multiclass model using cross validation for **200 categories**. Highly varied results were achieved ranging from accuracies as low as 45.278% to as high as 87.87% for some values of γ and c. Following are some of the cross validation results that were observed:

Learning Rate - γ	c	CV Accuracy	Test Accuracy
0.01	1.0	55.165	87.574%
0.001	10.0	56.077	89.864%
0.001	5.0	35.072	45.278%
0.001	7.0	95.0	62.222%
0.001	4.0	95.0	79.661%

(b) ***Experiment 12:*** *One-vs-rest using SVM (Developed by us)*
Even though we achieved high cross-validation accuracies using one-vs-rest on **50 categories**, the overall accuracy on the test set was just **2.159%**.

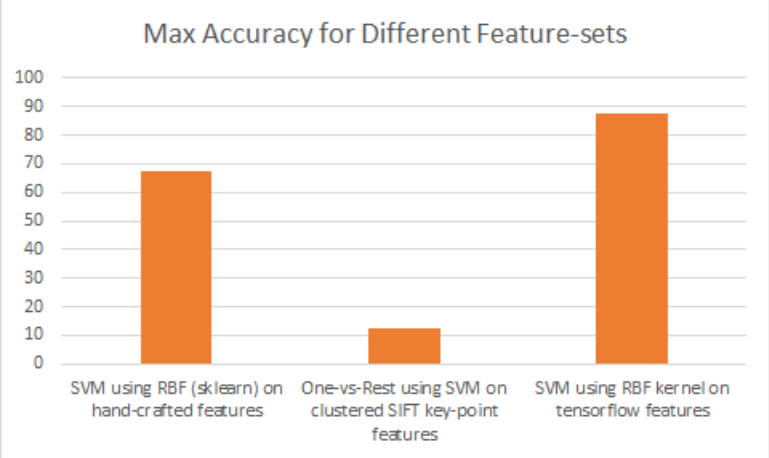
(c) ***Experiment 13:*** *Pair-wise classification using SVM (Developed by us)*
We trained 10 pair-wise classifiers by selecting two random labels at a time and achieved accuracies ranging from **49.153% to 100%**.

As observed from the above experiments, **SVM using RBF kernel produced the best results**, but they were highly unstable.

4: Learnings

Our learnings from the project can be summarized as follows:

1. Following bar graph shows the maximum accuracy achieved on each feature set.



2. SVM non linear RBF kernel performs better than all other classifiers for all the feature sets.
3. Even though the most promising results were achieved using the features that came with the data, since these features are hand-crafted, they will not be available if a real world application was developed. Hence, it would be better too use other feature sets like tensorflow features.
4. As the number of categories increase, the performance of classifiers like one-vs-rest, decision trees, one-vs-one decreases significantly and the computational cost to train them becomes very high.
5. As seen from the above experiments, SIFT features do not appear to be a good representation of this dataset.

5: Future Scope

If time were not a constraint, we would have done the following experiment:

1. More experiments using feature scaling, feature standardization, dimensionality reduction (PCA) with tensorflow features.
2. Analysing why tensorflow features produced unstable results for different SVM parameters.
3. One-vs-One classification on all the feature-sets.