Birds in a Forest

Sneha Mehta SUDO777@VT.EDU
Aditya Pratapa ADYPRAT@VT.EDU
Phillip Summers PTS964@VT.EDU

Abstract

The goal of this project is to use machine learning and computer vision techniques to identify an object in an image. More specifically, given an image of a bird, we would like to identify what species of bird it is. To do this, we have implemented a two step process; first identify the characteristics of the bird in the image, then use those characteristics to predict what species of bird it is. We have obtained approximately 83% for our own implementation of Random Forest using the features predicted from raw images using KNNs to identify various features.

1. Introduction

Bird species identification is a challenging problem as it involves many classification steps. It has many applications such as endangered animal rescue and predatory bird identification(1; 2). This mutli-level classification problem has been extensively studied in various applications for object identification such as flower identification, plant identification etc (3). Many of the previous studies for bird species identification are based on Caltech-UCSD Bird databases (4; 5; 6).

Some of the previous studies for handling this problem involve deep convolutional neural nets (3) or based on color features extracted from data (2). In this study we use a mutli-level classification approach to solve the Bird species identification, where we first identify a label for 25 bird features that can be extracted from an image, as mentioned in a previous study (4). Using the labels obtained for various features, we further give that as an input for another classifier that identifies the bird species based on the labels. For example, a bird with black primary and back color, white throat color and white belly color is more likely to be a Black Footed Albatross. In our approach we first identify the labels for various features such as primary color, back color, belly color, throat color and from an image input. Using these text *feature-labels* obtained from image input, we build another classifier to identify the bird species. In this work we have evaluated various classifier for feature-label identification and then used the input from the feature-labels to build random forest classifier to identify the bird species. The next few sections in this project report detail several aspects involved in data pre-processing, building and evaluating performance of various classifiers.

Virginia Tech CSx824/ECEx424 final project report.

2. Methods

2.1. Extracting bird features from raw images

2.1.1. BIRD FEATURE DATABASE

In order to learn the different features of a bird that can help us distinguish between different bird species, we have used the same 25 features as mentioned in the Caltech-UCSD Birds 200 database (4). Table 1 summarizes the 25 different features and the number of labels each feature can have. For example, the bird's primary color feature can have 15 different labels such as red, blue, black, blue and so on.

As mentioned earlier, there are 6033 bird images in the Caltech-UCSD Birds 200 database. A label for each feature in Table 1 was identified by workers at Amazon MechnicalTurk. There are 288 *feature-labels* in total (i.e., sum of all values in the second column in Table 1 is 288). The input data for each image therefore is a 0,1 vector of size 288:

- 0 if a label for feature is absent
- 1 if a label for a feature is present

For example, if a Turk worker decides that the 'back

color' in an image is blue, then the corresponding value in 1x288 vector for that image is set to 1. As many such workers identify labels for each feature, there may be conflicts for labels. Therefore, the database also includes the certainty with which each of the worker has labeled it:

- 0 if the worker finds that label for a feature is *probably* true,
- 1 if the worker is *sure beyond doubt* and,
- 2 if he is guessing that the label for the feature is present.

We have converted this certainty score to a confidence score [0,1] averaging over all the worker's inputs. In order to obtain this, we have multiplied and averaged responses of each Turk worker as follows:

- 0.5 to the worker's guess, if the certainty value is 0,
- 1 to the worker's guess, if the certainty value is 1, and
- 0.1 to the worker's guess, if the certainty value is 2

While this may not affect features that are absent, we have a value $\in (0, 1]$ for features that may be present in an image. The confidence score $\in [0, 1]$ was converted to a binary {0,1} data using a cut-off value of 0.5 (i.e., everything below 0.5 was set to 0 and 1 otherwise). We have used these values for training the classifiers to automate the process of identifying values for 288 *feature-labels*, in order to eventually use it for bird species identification. If the same feature has multiple entries for confidence score beyond 0.5, we have chosen the label with highest confidence score. For example, if workers feel that the primary color of a bird is green with average confidence score 0.6 and blue with an average confidence score of 0.7 and 0 for all other labels of primary color; we choose blue as the label for the primary color feature for that image. Using this approach we ensured that each image has at most 1 label for each of the 25 features.

2.1.2. BIRD FEATURE EXTRACTION

From the previous step (Section 2.1.1), we have *feature-labels* for training and testing data for 25 classifiers to identify *feature-labels* for each of the 25 features mentioned in Table 1.

The input training and testing data in this step is the vectorized raw RGB image, scaled to a pixel size of 100x100, after applying the segmentation and bounding boxes as obtained from the database. Therefore, the data is of size 6033x30000 (6033 images, and vectorized values for scaled RGB image of size 100x100, i.e., image matrix of size 100x100x3).

It is important to note that not all images may have *feature-labels* for each feature, i.e., some features may be absent in an image, such images from input training data are ignored for training the classifier. Therefore, not all features have the same length of training-testing data vectors. In order to evaluate performance of models for predicting feature-labels for each of 25 features, we have used the following classifiers:

1. SVM with Linear Kernel
2. SVM with Polynomial Kernel of order 3
3. SVM with MLP Kernel (tanh)
4. K-nearest neighbors (KNN) classifier

For all the implementations we have used their MATLAB implementations. The SVM in MATLAB is a One-vs.-All classifier, therefore, we used a multiclass-SVM which essentially enumerates whether or not a feature-label is present, therefore a total of 288 SVM classifiers for identifying a feature-label for 25 features for each image. Based on their performances and due to memory constraints as mentioned in Section 3.1.1, we have chosen KNN classifier to generate feature-labels for all test images and used it for bird species classification.

2.2. Predicting pattern features

For predicting color based features like head color or wing color, we make a single vector of pixel intensities from each of the R,G and B channels of the input bird image and give this vector as input to our SVM and KNN classifiers to predict class for the corresponding bird feature. But for pattern based bird features like head pattern and wing pattern it makes little sense to use pixel intensities as inputs. Input should be something that captures the bird pattern feature in question. So we used visual bag of words to summarize an image based on its SIFT (7) features and pass it as input to our classifier. All pattern based features are belly pattern, back pattern, breast pattern, head pattern, tail pattern and wing pattern. Pattern based features have 4 and 11 classes. All except head pattern have 4 classes whereas head pattern has 11 classes. For 4 class features the classes are solid, striped, spotted and multi-colored and for 11 class features the classes

are malar, eyebrow, capped, eye-ring, unique pattern, striped, spotted, crested, masked, plain and eyeline.

2.2.1. PRE-PROCESSING

For each input bird image we are provided a ground truth segmentation for the image by the MTurk workers as the part of the CUB-200 dataset. The segmentation is a binary image with $I(i,j) = 1$ for pixels where bird is present and $I(i,j) = 0$ for background.

Figure 1. Original image

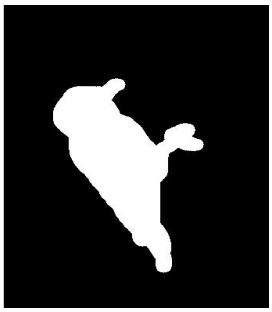

Figure 2. Segmentation

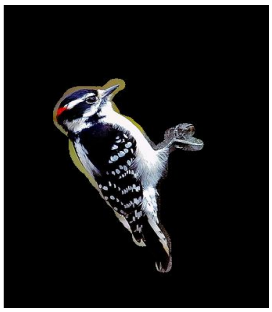

Figure 3. After segmentation

We multiply each channel(R, G, B) of the input by the segmentation to retrieve the segmented image. This retains the pixels corresponding to the bird and sets the background pixels to 0 as shown in Figures 1 to 3. We then crop the segmented image using the ground truth bounding boxes provided in the dataset. Finally we resize the cropped image into a 100x100 image. For all of the above operations MATLAB image processing tool box was used.

2.2.2. EXTRACTING SIFT FEATURES

For each of the images in the database for which the workers have a high confidence (≥ 0.5) we extract SIFT descriptors (7). SIFT features are scale and rotation invariant so two visually similar patches in two or more images will have similar descriptors irrespective of their location or scale. For each of the training image we have nearly 500 128-dimensional SIFT descriptors.

2.2.3. VISUAL BAG-OF-WORDS

We cluster all the SIFT descriptors extracted in the previous stage into $K = 1200$ clusters using kmeans. The value of K is to be determined experimentally. And the value of K can differ for one bird feature to another. For example for head and tail pattern we used K=1150 while for back, breast and belly patterns we used $K = 1200$. For some bird features a value of $K = 1200$ was resulting in empty clusters which is why we brought down the value to 1150. These K SIFT descriptors form our visual vocabulary or the codebook. Each of the descriptors in the code book captures certain visual patches.

2.2.4. IMAGE HISTOGRAM

For each of the nearly 500 SIFT descriptors extracted from an image it is mapped to a visual word from the codebook using euclidean distance as metric. That is, euclidean distance is computed between the given descriptor and each of the words in the codebook and the descriptor is mapped to the word with the least distance from it. For each image a frequency count is calculated for each of the visual words. As a result each image is represented by a K bin histogram. We have effectively reduced the input size from $3 * 100 * 100 = 30000$ to K for an image. And also the input captures information other than simply the pixel intensity values.

2.3. Random Forest Optimization

Intuitively decision trees and random forests fit well with the sort of problem we are solving, given a list of characteristics we want a return of what overarching label those characteristics belong to. In

our case, given its wing size/ torso color/ beak shape/ back pattern etc, we want to know what species of bird has all of those given characteristics. We know that random forests generally perform better than individual decision trees, so wed like to investigate the different aspects of a random forest and try to create an optimal implementation. All models and calculations were executed and evaluated through MATLAB.

We know that tree depth and feature pool size effect a decision trees performance, but in addition to that a random forest has its forest size, percentage of feature pool used, and percentage of samples in each tree used as more parameters. In our bird experiment the feature pool size is not particularly large at 288, so we decided to used the full set as our available feature pool. Percentage of samples available when constructing each tree seems more of a calculation resource saver than providing any increase of accuracy. As a result we are primarily concerned with how maximum depth, forest size, and percentage of samples available per tree affect a forests accuracy. As a rule of thumb, forest size and maximum depth will likely increase the forests accuracy monotonically, so we are also interested in the required calculation time in comparison to accuracy gain.

To start we attempted a grid search using ten different values each for depth, forest size, and percentage. These ranged from 1 to 40, 1 to 1000, 0.1 to 1 respectively. After some time it was apparently that this grid search would take a very long time on my machine, with an eventual estimate of 120 hours. At this point we felt it was prudent to run a smaller scale search on another machine, this time 5x5x5. The parameters ranged from 5 to 30, 10 to 1,000, and .2 to 1 respectively. The results are available in the attached figures. For completeness sake this was run on the 20 newsgroups data set, using the 1,000 highest information gain words across the articles as the feature pool. Plots were created by using MATLABs scatterInterpolant function on a 50 by 50 grid spanning the parameter space as well as the surf function. After successfully parsing our mechanical turk data, our forest evaluation shift to that and away from the 20 newsgroup dataset.

When working on the mechanical turk data, it came to our attention that MATLABs built-in fitctree function was much more efficient timewise than our implementation. Additionally it handled the tree depth parameter for us, and returned higher accuracies than our version. Because it handled the maximum depth parameter for us, our parameter space was reduce from three dimension to two dimension, making grid searches much faster. After finding optimal forest size and percentage of features to use, we compared our new forests accuracy to that of a single decision tree and the famous k-nearest neighbors model.

After we have an acceptable model for converting image features to bird species we plan to use our machine learning/computer vision image features and predict what bird species the image is of. In addition to training the model based on the mechanical turk data and then running the computer vision data through it as the test, we also plan to train the species model on the computer vision data. This second implementation cuts the mechanical turk data completely out of the pipeline from image to species.

3. Results

3.1. Bird feature extraction

3.1.1. Evaluating Classifier Performance

To evaluate the performance of the classifiers, we have used 4 different classifiers namely, SVM classifier with linear, polynomial and MLP kernels and a KNN classifier. As some features may not have any *feature-label* for an image, the input for all 25 feature classifiers is different. It is important to note: we maintained the underlying *feature-label* distribution for each feature and the $90\% - 10\%$ train test split).

As it can be seen from Figure 4, the KNN and SVM with polynomial kernel outperform the rest of the classifiers for most of the 25 features. It is important to note that the different features have different number of feature-labels as mentioned in Table 1. For a 15-class classification for primary color data, we got highest accuracy for KNN classifier with 30% test accuracy. For a feature like eye color, with 11 different labels, we achieved a highest accuracy of 95% for SVM with polynomial kernel of order 3 and KNN test accuracy was close to 91%. However it is important to note that a majority of labels in eye color were black and the classifiers tend to predict black for almost all the images and resulted in a high eye color accuracy.

3.1.2. Feature-label generation

While training SVM requires a one-vs-all classifier, and results in 288 classifiers as opposed to 25 KNN classifiers and resulted in insufficient memory is-

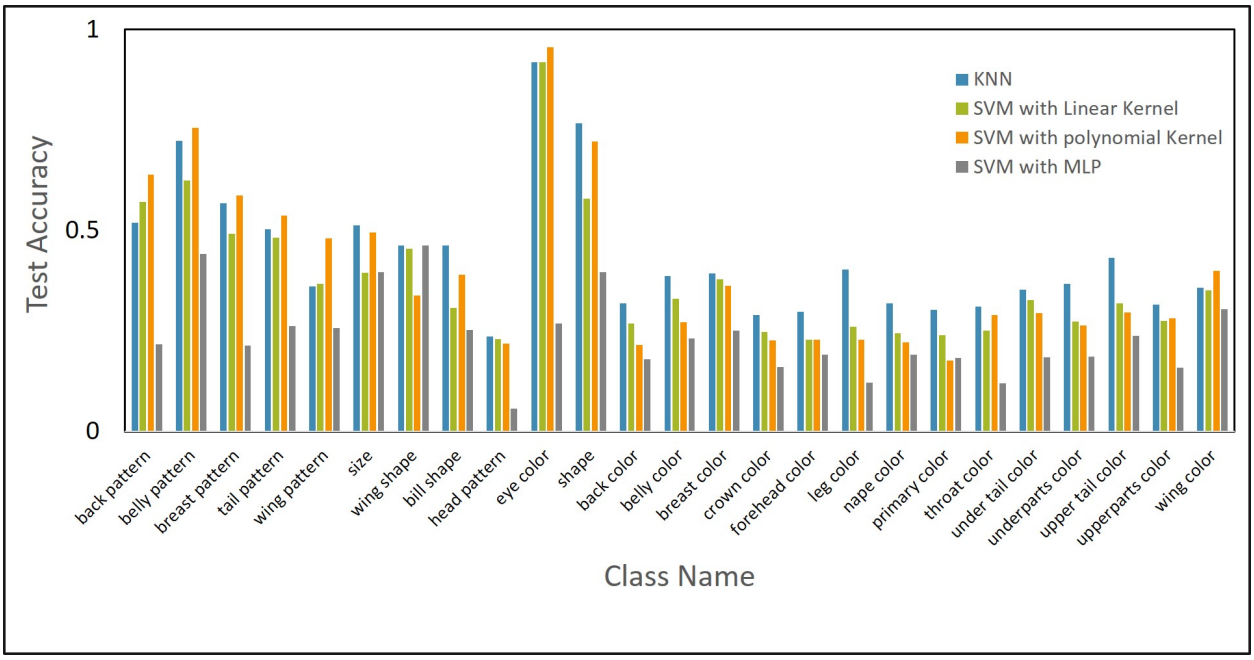

Figure 4. Summary of of all testing accuracies obtained for each classifier for 25 features and $90\% - 10\%$ train-test split for each feature.

sues. We therefore used KNN classifier from the four classifiers under consideration for generating feature-labels for testing the random forest performance to identify bird species.

For predicting feature-labels for bird species classification, we have used 90% of the input data as train data (5430 images) and 10% of the data (603 images) for testing. As training is performed on subset of 5430 images which actually have a Turk worker's determined value, we still predict a label for all 25 features for test data (of 603 images) irrespective of whether or not it is present in that image. Intuitively, any inconsistency in the predicted result for one of the 25 bird features for test images, will be addressed by the Random-Forest implementation depending on how informative that label can be. For example, all bird images may not have a distinguishable eye color, but such feature will have low information gain in the Random-Forest stage, therefore it probably would not affect the final bird species recognition by a lot.

3.2. Bird pattern features

We extracted histograms from the images corresponding to the each pattern feature. This was given as input to SVM and KNN. We trained SVM models with linear, polynomial(order 3) and mlp kernels. Below are the results we got.

Figure 5 compares the SVM and KNN accuracies for all patterns. The best accuracy of 76.6 % was obtained for SVM with polynomial kernel for belly pattern. The head, tail and wing patterns were observed to give low accuracies. This could be because since head, tail and wing parts occupy less number of pixels in the images, it is possible that not many descriptors from these regions have been represented in the codebook.

Next, we compare the performance of patterns giving best accuracy with the performance for those patterns using only image intensity values as input. Figure B.3 gives compares performance for back pattern using old and new methods. Back pattern was observed to give an accuracy 62% for SVM with polynomial kernel. This is slightly lower than old accuracy. For back pattern the accuracy decreased for all SVM classifiers compared to the old method. But accuracy for KNN was observed to have shot up by nearly 10%.

For belly pattern accuracies increased marginally for all methods and a highest overall accuracy of 77% was observed for SVM with polynomial kernel. Results can be observed in Figure B.4

Performance improvement was observed for all methods for breast pattern and a highest accuracy of 70% was observed for SVM with polynomial kernel. Figure B.5 reflects these results.

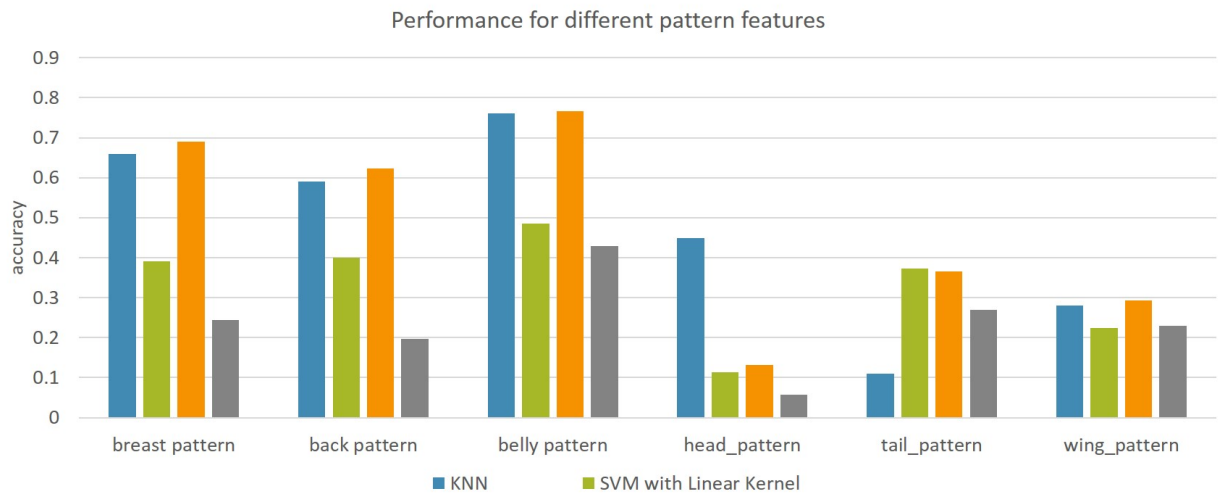

Figure 5. Performance for all pattern based features

Some advantages of this method are that it reduces the data size significantly and hence saves computational resources. Image is represented by a length K vector instead of a length 30000 vector. And it is much simpler to implement and deploy than the methods giving state of the art performances using convolutional neural networks (3). But also features have to be pre-computed and stored beforehand unlike convnets that learn features while training. The K value has to be determined experimentally.

3.3. Random Forest Optimization

As expected increasing the depth of the trees in a random forest, as well as increasing the number of trees in a forest increase their accuracy. There is some randomness in results however, as the success of a forest is in part dependant on the random assignments of features into the trees. For example one forest with 300 trees might have .68 accuracy, and a similar forest with 400 trees might have .67 accuracy. Generally speaking however, accuracy only goes up with deeper trees and wider forests. Figure B.6 Compared to required calculation time however, there is stark diminishing returns for the two parameters. For example in the 20 newsgroup dataset a forest of 200 trees and 20 depth might have .56 accuracy and take 3 minutes to compute, while a forest of 1,000 trees and 30 depth might have a .59 accuracy and take 30 minutes to compute Figure B.7. If either the depth or forest size remains small, calculation time remains small, but if both are large they build off each other and can achieve very long calculation times. It appears that time increases linearly with forest size, and super-linearly with maximum depth.

Feature percentage is a less straightforward parameter to evaluate. If 100% is used then every tree in the forest is identical and has the same result as if the model was a single full tree. From there accuracy is improved by decreasing the percentage used, but the other extreme is that 0% of the features are used in each tree and no information is gained. Generally, the more trees in a forest the lower the optimal percentage of features is. For one data set it may be optimal to use 50% of the features in a forest of five trees Figure B.8, and for the same data set it may also be optimal to use 10% of the features in a forest of 500 trees. Figure B.9 This larger forest with less features has a higher prediction accuracy than the smaller forest if all other parameters are held constant.

After parsing the mechanical turk data and running grid search over it, the optimal parameters seemed to be some percentage of features between .20 and .40, and a forest size of anything over 100. Any parameters within those margins returned a .70 to .72 prediction accuracy ratio, with variance due to the stochastic nature of the forest. Using the fitctree function, forest size and percentage of features affected calculation time very similarly to the forest size and max depth did in the 20 newsgroups data. Figure B.10 Its very likely that the fitctrees maximum depth is dependant on the number of features passed to it, in this case passing it 10% features creates more shallow trees than passing it 90% features. Interestingly though, some of the forests

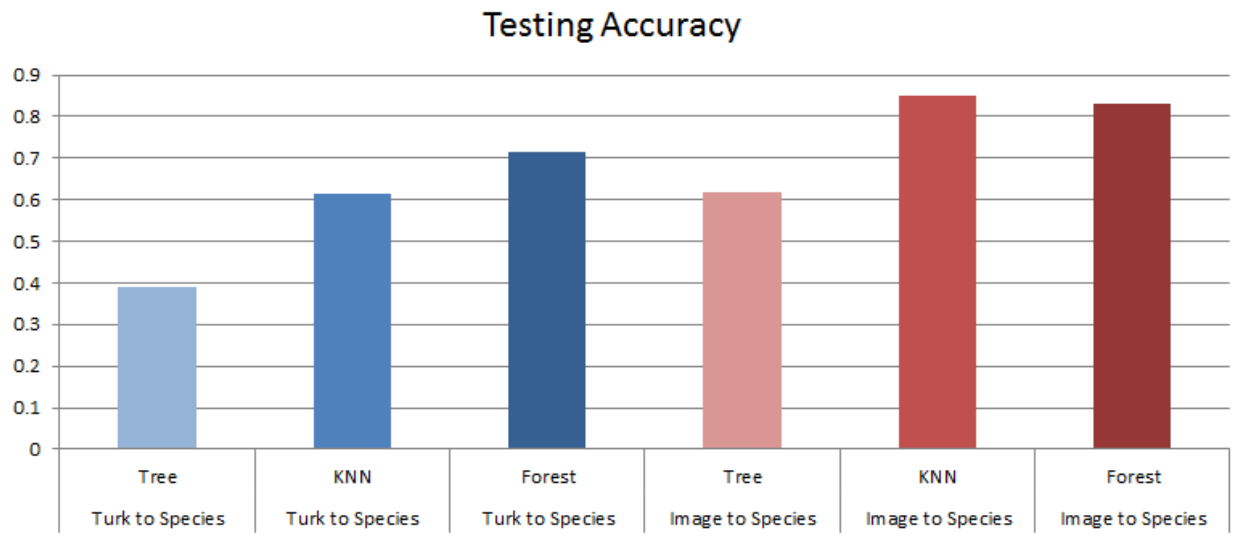

Figure 6. Summary of all testing accuracies for bird species as a whole.

worst performance was in the 100% features used space, which also had the highest required calculation time.

Figure B.11 In this case optimal model accuracy is not dependant on higher and higher calculation requirements. Overall, with our training and testing on mechanical turk data (confidence ratings for each feature rather than binary features) a single decision tree had a .38 testing accuracy, k-nearest neighbors had a .61 testing accuracy, and our random forest had a .72 testing accuracy.

3.4. Putting it all together

After receiving our computer vision predictions for image features for our testing set, we ran that data through our preexisting model which was trained on the mechanical turk data. The result was a .83 testing accuracy, actually higher than when are image features were from the mechanical turk confidence scores. This result is unintuitive and difficult to explain. Perhaps because the mechanical turk data was curated by multiple individuals with an imperfect labeling scheme (i.e., a bird with a white and black torso might be black to one person and white to another), the computer vision might be more consistent in what it classifies a specific species to look like. Using our different models for this computer vision train to computer vision test, a single decision tree had a testing accuracy of .627 and k-nearest neighbors had a testing accuracy of .852. These results corroborate the phenomena of vision training improving our overall species prediction accuracy, with an overall increase in all models after switch from the turk confidence score set (see Figure 6).

4. Conclusions

Overall our plan to take a bird image and predict its species was successful. We classified many different attributes of an image, the vast majority of which had a successful prediction rate of less than 70%, and combined them in such a way as to accurately predict the species with over 80% accuracy. Keeping in mind that each image could have been one of 200 different species, approximately a 0.5% guessing accuracy, this accuracy is a huge improvement. With the help of image segmentation, our random forest model at 83% and KNN at 85%. We however did not make a direct comparison with other methods that tackle a similar bird identification problem(3; 2), as they have their own methods for background removal and feature identification instead of directly using segmentation provided by MechanicalTurk workers.

In the implementation, one versus all kernel SVMs occasionally outperformed KNN in image feature recognition. However the SVM models took up unsustainable amounts of memory, so we opted to use KNN to predict image features. Optimizing a random forest gave us some insights into how all their different parameters interact with each other, and outperformed both single decision trees and KNN when testing on mechanical turk data. How-

ever once the image to species pipeline was constructed, KNN performed on par or better than our random forest. This may be because the mechanical turk data was confidence scores from 0 to 1, where the computer vision prediction was a single binary label for each image feature. We propose that this work can be further extended by using various computer vision techniques for background removal from raw bird images instead of using segmentation by manual effort thereby automating the process of bird species identification.

5. References

References

[1] Nadimpalli U.D., et al., *A comparison of image processing techniques for bird recognition*, Biotechnology Progress, vol. 22, no. 1, pp. 913, 2006.

[2] Marini, et al., *Bird species classification based on color features*", Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on. IEEE, 2013.

[3] Branson S., et al., *Bird species categorization using pose normalized deep convolutional nets.* arXiv preprint arXiv:1406.2952 (2014).

[4] Welinder P., et al., *Caltech-UCSD Birds 200*, California Institute of Technology, CNS-TR-2010-001, 2010.

[5] Wah C., et al., *The Caltech-UCSD Birds-200-2011 Dataset.*, Computation & Neural Systems Technical Report, CNS-TR-2011-001.

[6] Branson S., et al., *Visual Recognition with Humans in the Loop*, European Conference on Computer Vision (ECCV), Heraklion, Sept., 2010.

[7] David G.L., *Distinctive image features from scale-invariant keypoints.* International Journal of Computer Vision 60.2 : 91-110, 2004.

Appendices

A. Table containing information on distribution of various features image data

Feature	Number of labels	Number of images containing each feature
back color	15	3169
back pattern	4	3105
belly color	15	4239
belly pattern	4	4322
bill shape	9	4387
breast color	15	4521
breast pattern	4	4511
crown color	15	4858
eye color	14	5049
forehead color	15	4685
head pattern	11	3341
leg color	15	2811
nape color	15	4024
primary color	15	4721
shape	14	4571
size	5	4161
tail pattern	4	2738
throat color	15	4727
under tail color	15	2721
underparts color	15	4246
upper tail color	15	2636
upperparts color	15	3767
wing color	15	4050
wing pattern	4	3968
wing shape	5	1208

Table 1. Summary of 25 different features extracted from each image and the number labels each of the features can take. Not all 6033 images from the input contain all the features, the third column shows how many images out of 6033 have a label with ≥ 0.5 confidence score for Turk worker's labeling.

B. Accuracy for various SVMs and KNN separated for features with different number of classes

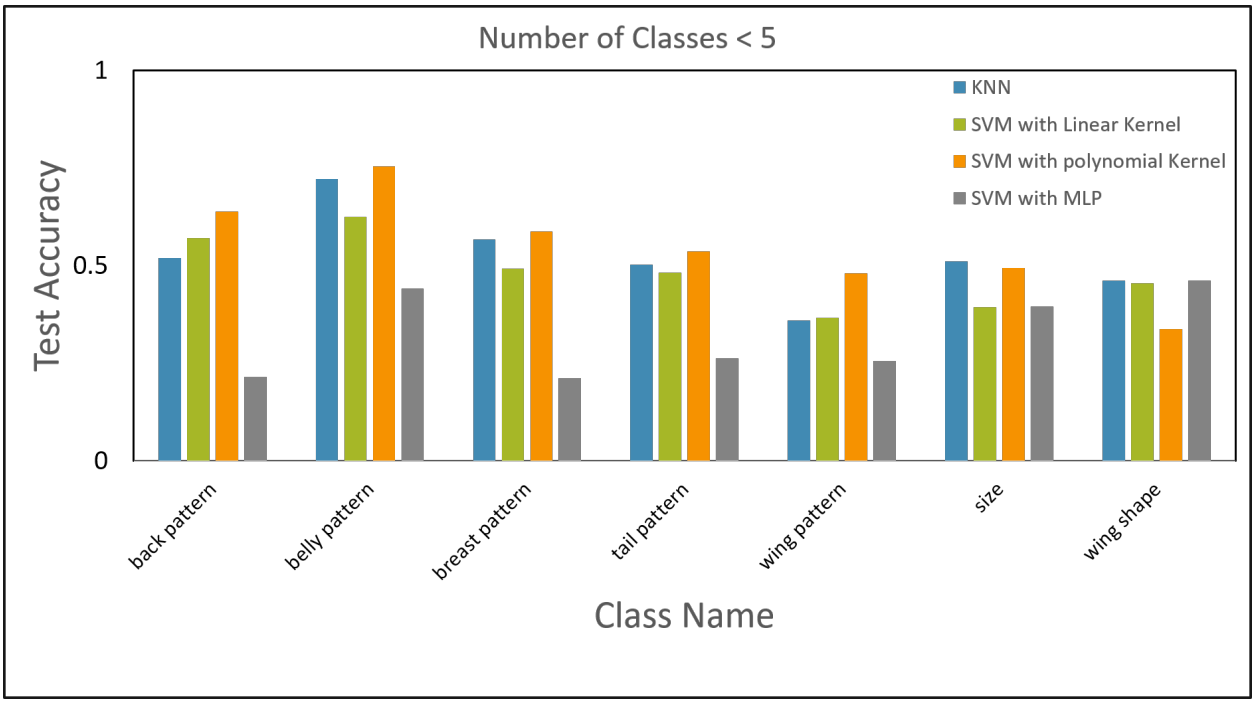

Figure B.1. Summary of of all testing accuracies obtained for each classifier for features which have greater than 14 labels each and $90\% - 10\%$ train-test split for each feature.

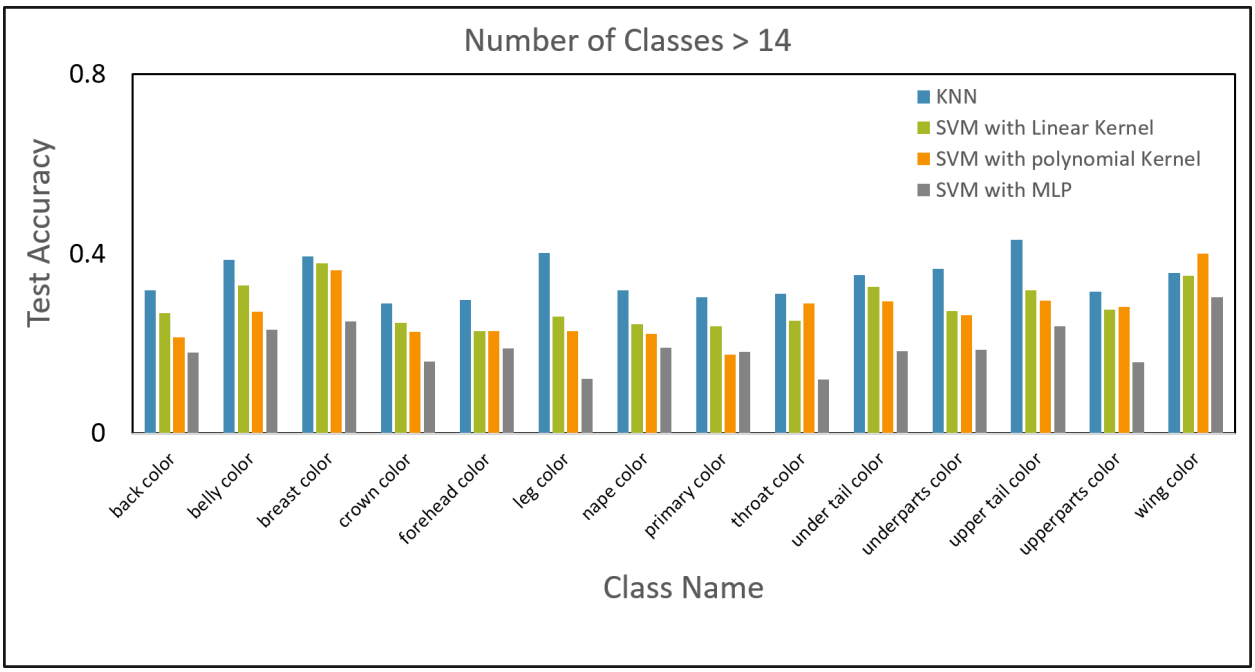

Figure B.2. Summary of of all testing accuracies obtained for each classifier for features which have greater than 14 labels each and $90\% - 10\%$ train-test split for each feature.

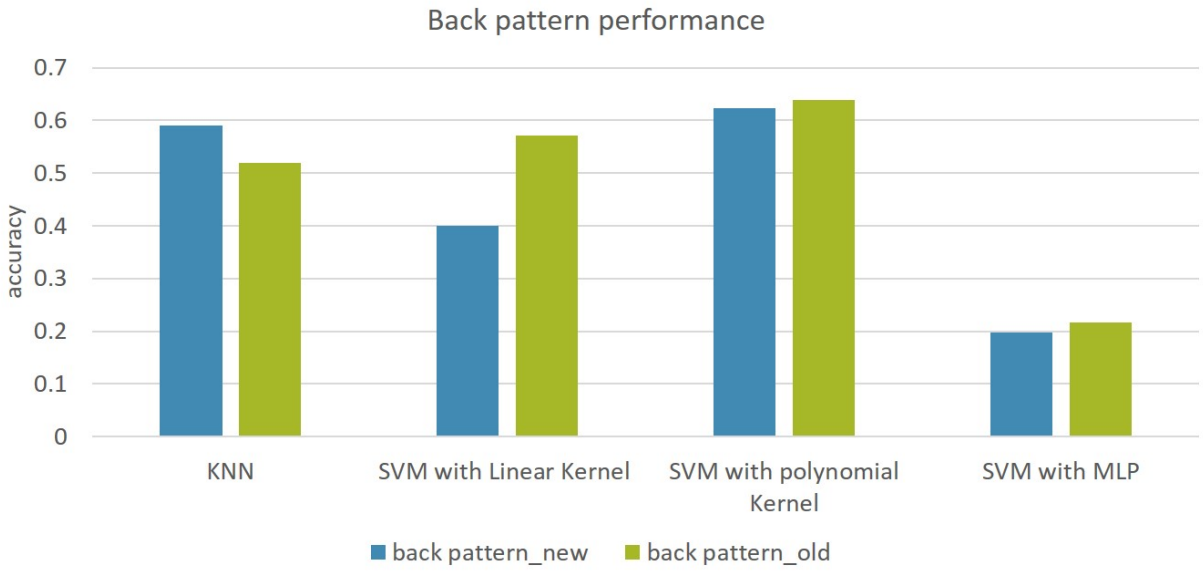

Figure B.3. Comparing original and new back pattern performance

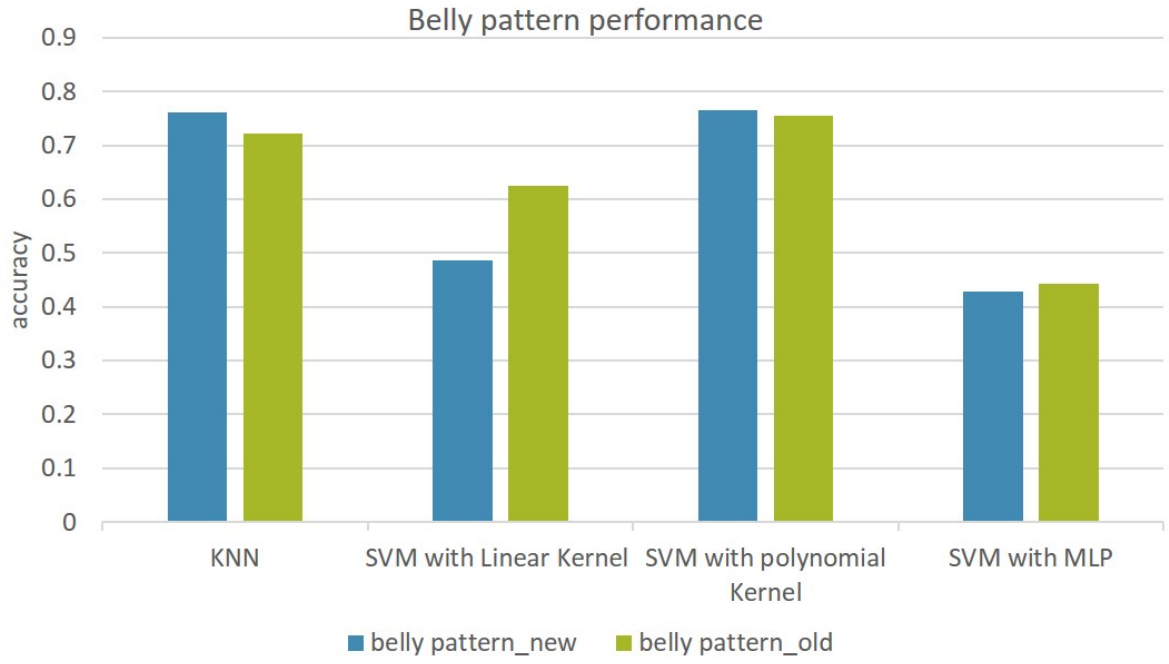

Figure B.4. Comparing original and new belly pattern performance

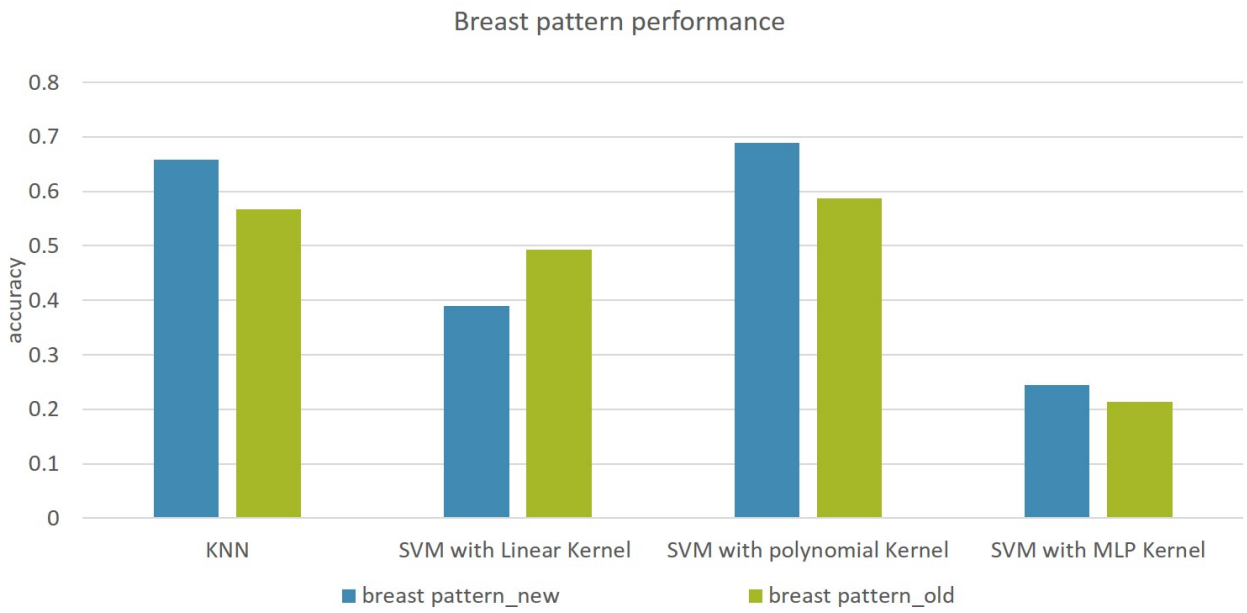

Figure B.5. Comparing original and new breast pattern performance

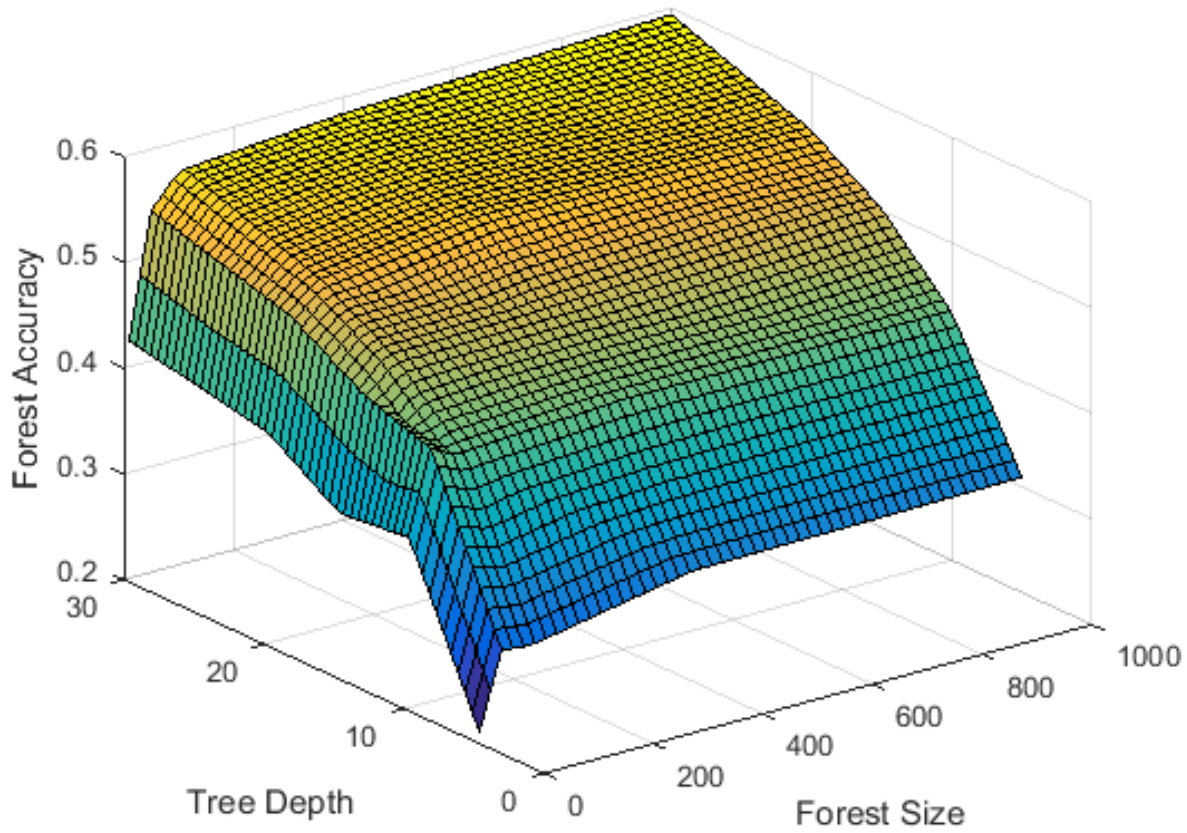

Figure B.6. Surf plot for accuracies of 20newsgroups data using 0.20 feature pool in each tree.

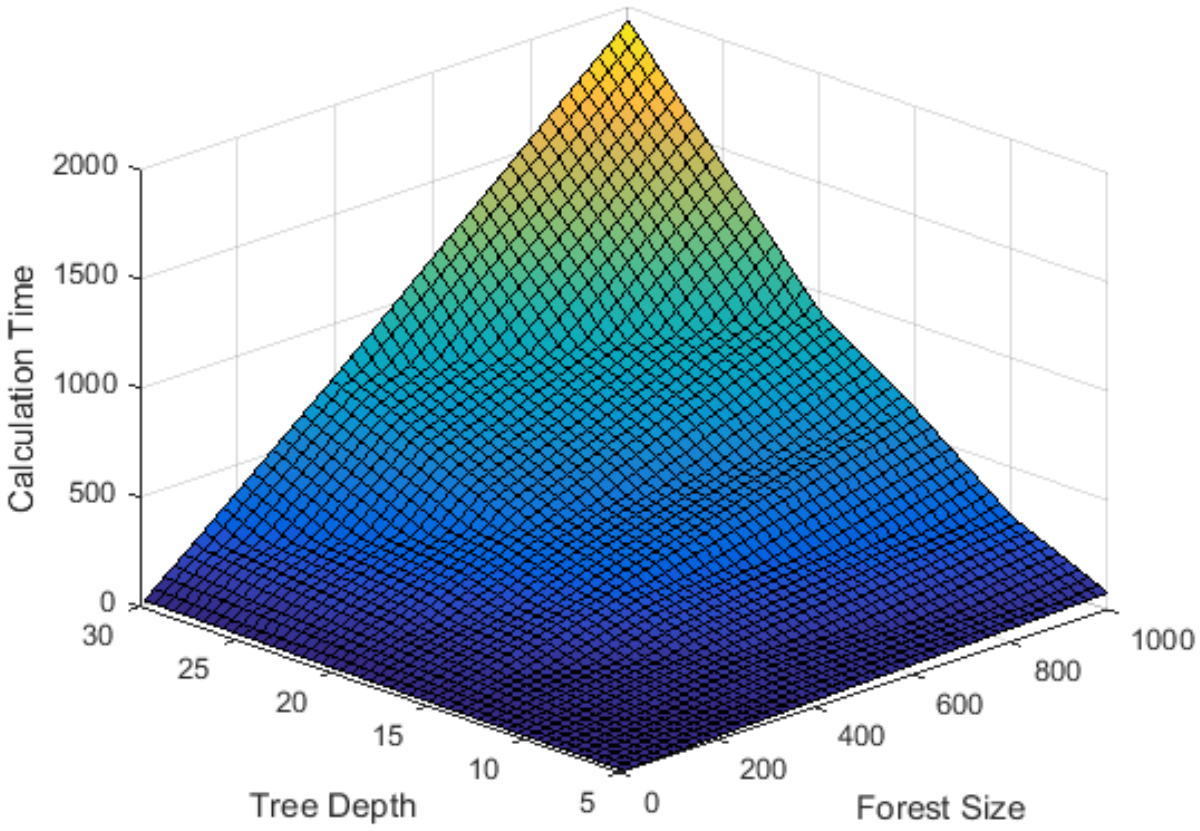

Figure B.7. Surf plot for calculation time of 20newsgroups data using 0.20 feature pool in each tree.

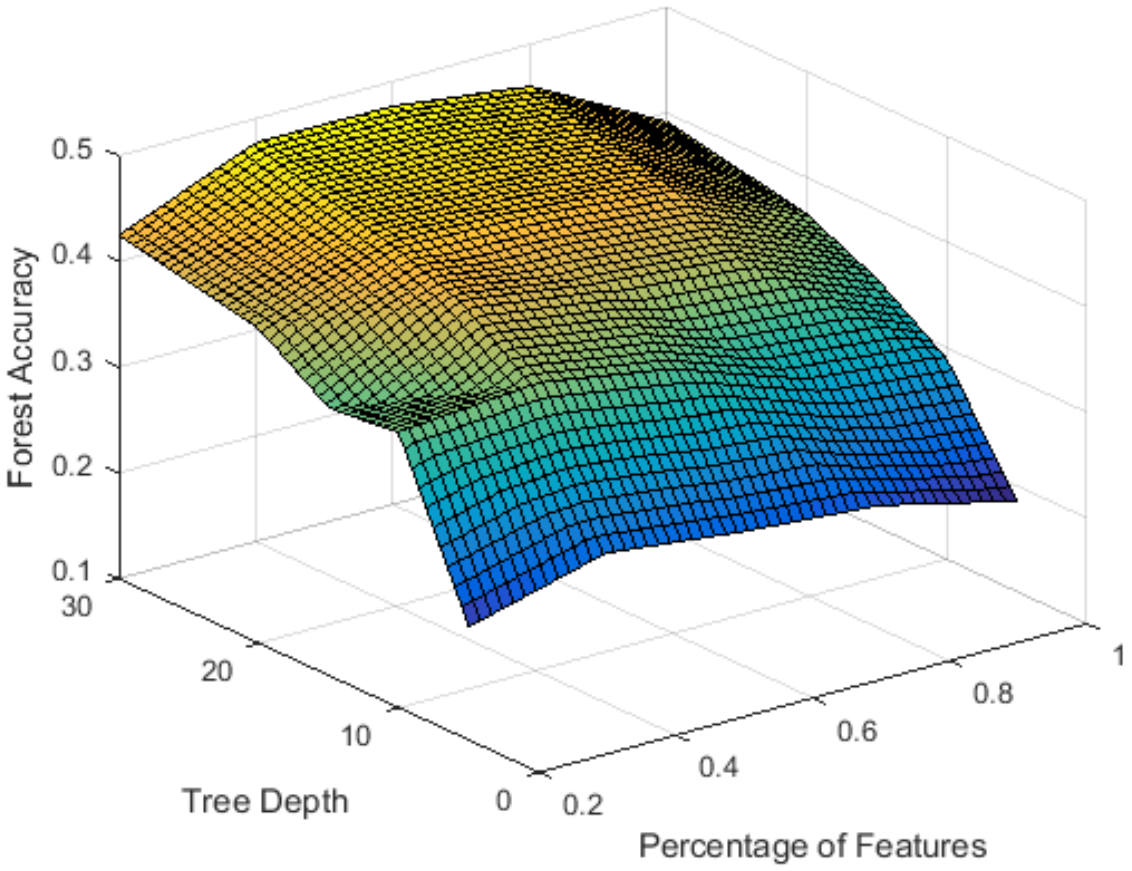

Figure B.8. Surf plot for accuracies of 20newgroups data using 5 trees in each forest.

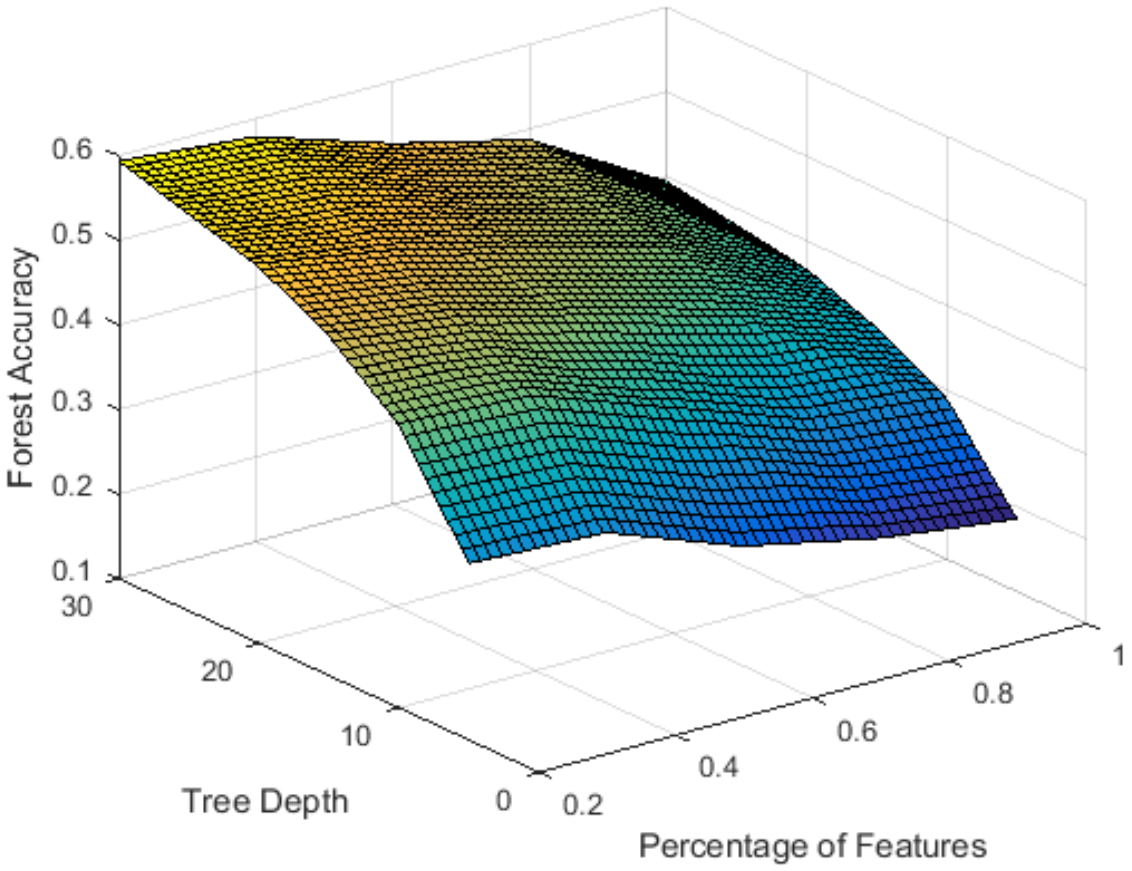

Figure B.9. Surf plot for accuracies of 20newgroups data using 500 trees in each forest.

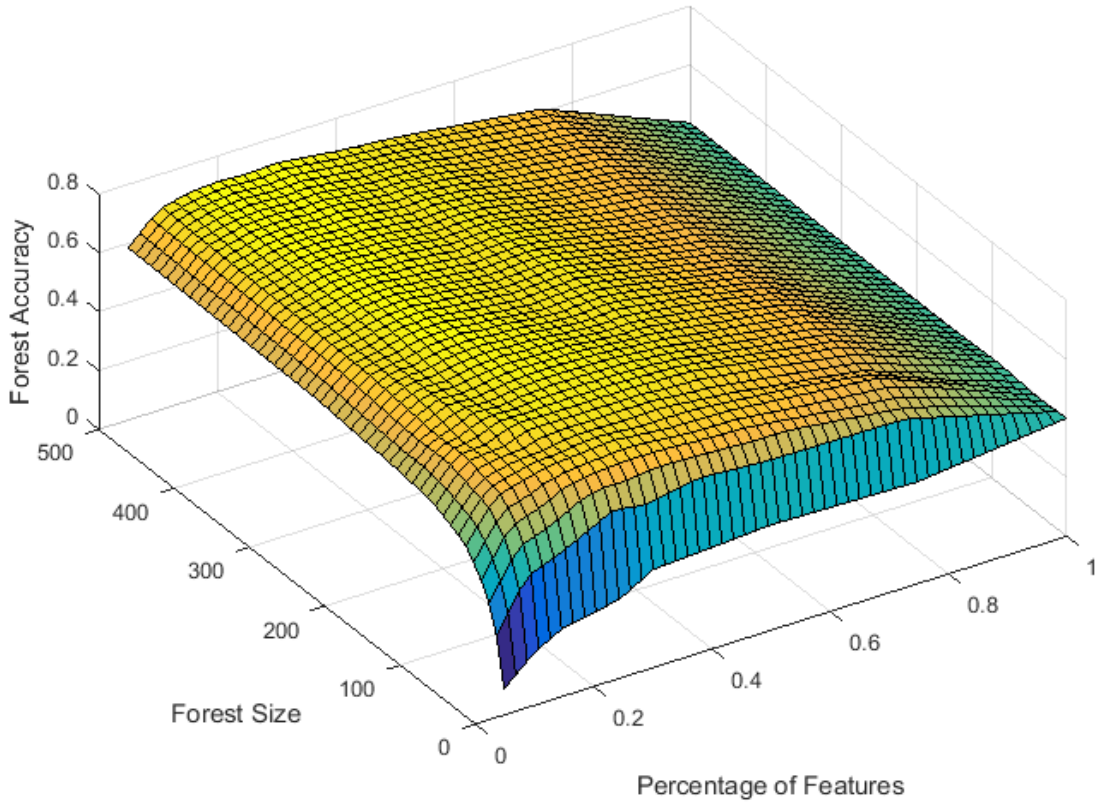

Figure B.10. Surf plot for accuracy of mechanical turk data for bird features.

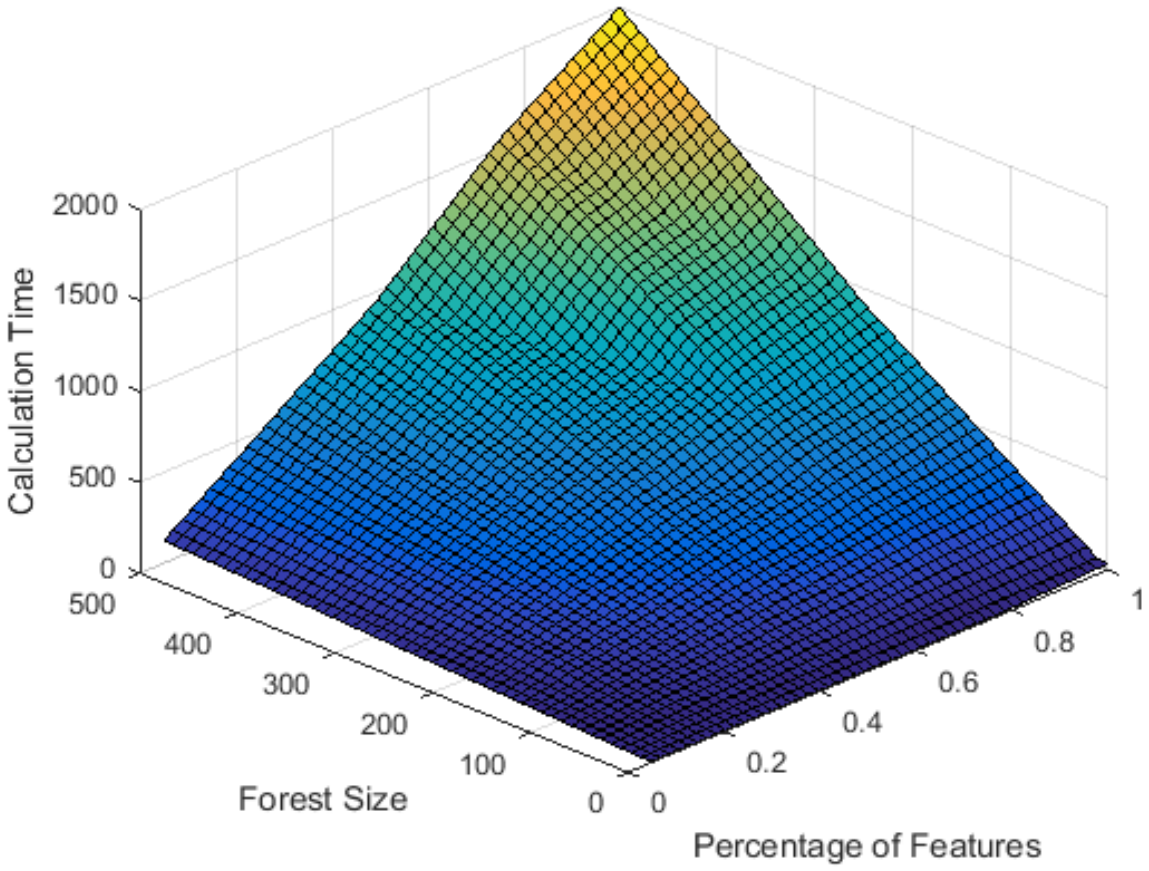

Figure B.11. Surf plot for calculation time of mechanical turk data for bird features.