Image Classification Using Hand-crafted Features

Yifan Zheng

***Abstract*—In this report, we build hand-crafted image features based on color, image shape, texture properties as well as principal components analysis (PCA) for image classification. These features are then used to train some machine learning classification algorithm like logistic regression, support vector machine (SVM), etc. We analyze the dataset and effective of these features through exploratory data analysis (EDA). The performance of each classification model is also reported.**

I. Introduction

With the advent and fast development of deep learning (DL) algorithms in the field of image classification, it seems traditional computer vision (CV) techniques have been outperformed on many CV tasks like object detection and classification [1]. Nevertheless, on small dataset, deep neural network may be an overkill in terms of computing power, efficiency and performance improvement compared to traditional methods. Besides this, DL is an end-to-end training method which usually lacks interpretation of the algorithm while traditional CV algorithms are based on constructing meaningful features from raw image data which can give us some intuition on how an image is actually classified.

In this report, we will stick to the traditional image classification methods. The goal of our work is to train some traditional machine learning models on a small image data set and select out the best model to predict the classes on a test image data set. The report is organized as follows: in Sec. II, we discuss pipeline of the project with an emphasis on feature extraction and selection. We also do some exploratory data analysis (EDA) to get some understanding of the data set itself and the features. Also in Sec. II, we demonstrate the performance of each model we trained and select out the best one to classify the test images. Finally, in the conclusion section (Sec. III), we summarize what we have found and discuss directions to make our classifier better.

II. Procedures of the Project

The pipeline of our image classification project is summarized in Fig. 1. In the following subsections, we are going to explain these steps in more detail.

A. Description of the data set

The learning data set consist of 1501 labeled images in 20 categories. During the exploratory step, we noticed there are 16 grayscale pictures while all others are colored. This matters when we generate some features from these images. In the first step, we read all the learning set images and stored them as high dimensional vectors with corresponding labels in a single data frame. For any practical classification purpose, knowing whether the classes are balanced or not is important. So we plotted the class frequency as shown in Fig. 2. The plot tells us it is a balanced data set and we don't need to worry about class imbalance problems when we train the models. For the test set, it contains 716 unlabeled data within the same 20 categories. Another primitive data we looked at is the image size statistics for the learning data set since we may need to resize the pictures to generate certain features. This information is summarized in Fig. 3. To get some sense of the color variance for each class, we plotted the violin graph of average red intensity of each image for each class in Fig. 4

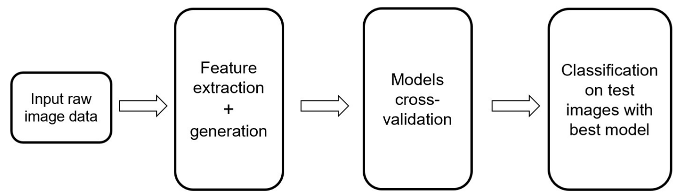

Fig. 1. Pipeline of this image classification project

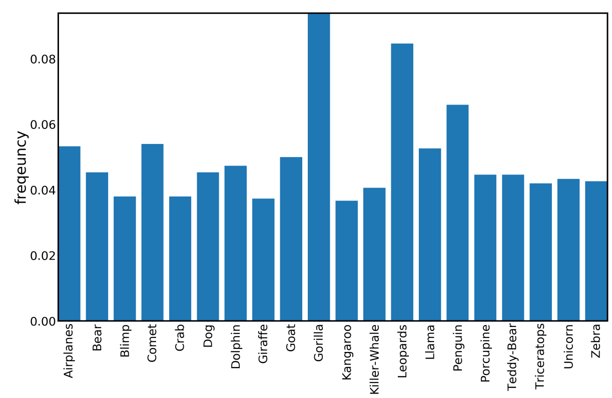

Fig. 2. Class frequency of the learning set. This shows it is a balanced class since highest class number is roughly only as twice as the lowest one.

B. Feature extraction and selection

There is an extensive study in the literature showing how to construct features from image data [2]. For example, Scale-invariant feature transform (SIFT) descriptor is a popular appearance-based feature detection and extraction technique [3]. It has the advantage of generating scale invariant features. Speeded up robust features (SURF) is another appearance-based local feature descriptor, which is faster and considered more robust than SIFT [4]. Given the size of the image, the

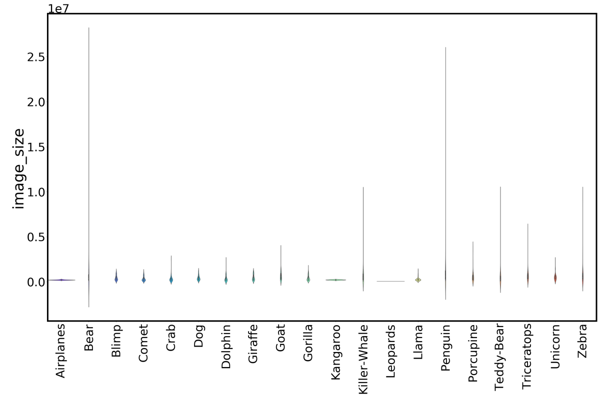

Fig. 3. Violin graph for image size vs. classes. Notice how the distributions vary significantly from class to class. This shows the information loss resulting from resizing a picture is different for different classes.

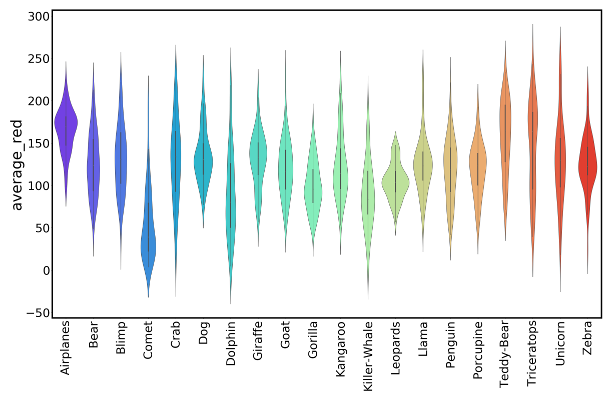

Fig. 4. Violin graph for average red intensity vs. classes. Notice how the median of the comet class is different from other classes. This can be a good indicator to separate comet out.

exact number of features these descriptors generate depend on the patch size we choose for each image. To get a reasonable classification result on our data set, the size of the feature vector they need to generate is likely with the order of magnitude 1000 for each image. This can make the training process computationally expensive. In this project, due to time limit, we used other feature generation methods to generate 25 features and selected 17 from them. These features are summarized in Fig. 5

Let's examine these features more carefully. The first set of feature we extracted are the top components given by principal components analysis (PCA). This is considered as global features since the whole image vector has to be given to compute the covariance matrix. To generate these PCA features, we resized all the image to 340×403 (height and width) and converted the 16 grayscale images to colored one since PCA requires equal vector length for every input image. Only top 10 components are selected since they already provide about 60% variance explanation as shown in Fig. 6 To give a intuitive demonstration of how these PCA features are useful in classifying the data set, a scattering plot of the

Feature categories	**Feature**
Global features	Top 10 PCA components
GLCM features	Contrast Homogeneity ASM Correlation
Color features	Hue Saturation
Shape features	Number of Harris corner
Unused features	Dissimilarity (GLCM) Energy (GLCM) Red Green Blue Value Image size Aspect ratio

Fig. 5. Image feature extracted and selected. Eight features are dropped either because they have a strong correlation with other features or don't characterize the image.

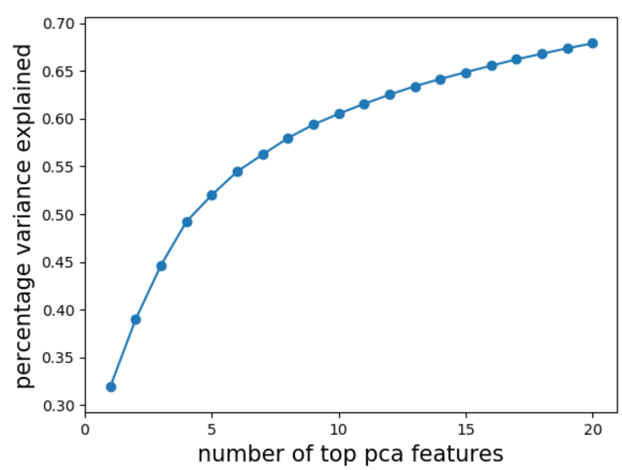

Fig. 6. Principal components variance explanation. Notice how the top 10 components already explain more than 60% of the variance and it grows slower after 10 components.

top two components for three classes is plotted in Fig. 7.

The second set of feature is extracted from the so-called gray-level co-occurrence matrix (GLCM). GLCM is a recurrence matrix characterizing texture of an image by calculating how often pixel pairs with specific values and in a specified spatial relationship occur in an image [5]. The images are resized in the same way as before and then converted to grayscale pictures to compute the GLCM. Six statistical properties can be calculated from this matrix and we choose four of them to be used as features. To show how these texture features can be useful in separating different classes, we plotted a violin graph for the correlation feature as in Fig. 8

The third set is color based features. They are computed as the average intensity of each color channel for every image.

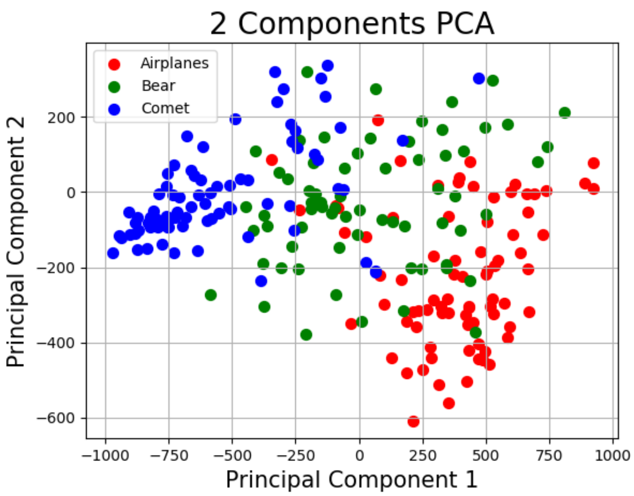

Fig. 7. Top two components scattering plot for three classes. This plot shows a good separation of these three classes.

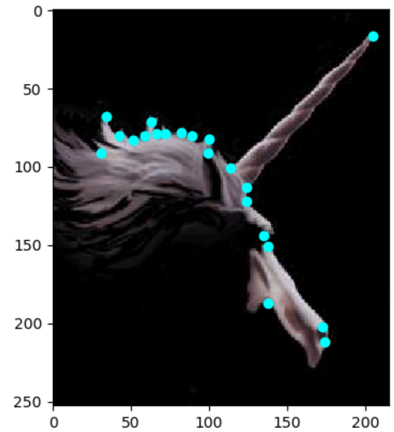

Fig. 9. Harris corner detection for a unicorn picture. Notice how it detected the horn of the unicorn.

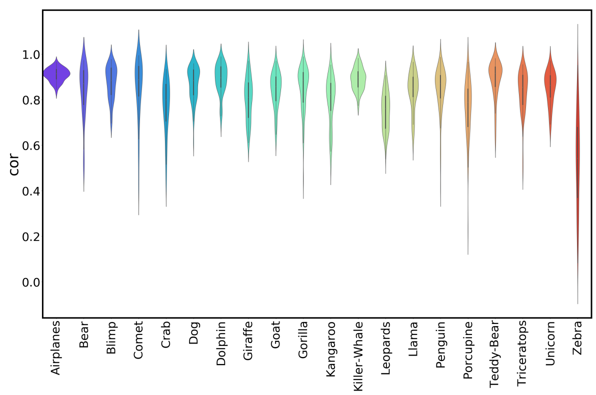

Fig. 8. Violin graph for the correlation feature based on GLCM. Notice how the distribution vary between zebra class and other classes. This may be a good indicator to separate the zebra class from others.

For the grayscale images, we first converted them to color image. How this transformation is done should not impact our classifier a lot since there are only 16 of them. The fourth set is shape based feature. We counted the corner number of each image using Haris method. This feature may be useful in characterizing classes with sharp corners like the unicorn class or triceratops class. One example showing how the Haris corner detector works is illustrated in Fig. 9

Next we will discuss how we performed feature selection to choose the final 17 features out of 25 features. We do realize there are systematic ways to do this like the wrapper based techniques [6]. However, due to time constrain, we used correlation analysis and some hand-weaving arguments to rule out 8 features. The two features, image size and aspect ratio are dropped because intuitively they don't give any information on the actual content of the pictures. Unless the test set pictures are obtained in a similar fashion as the learning set pictures, it is unreasonable to believe they would have a similar statistics for the two features on different classes. The other six features are dropped because they have a strong correlation (above 0.8) with other features as shown in the correlation heatmap (See Fig. 10).

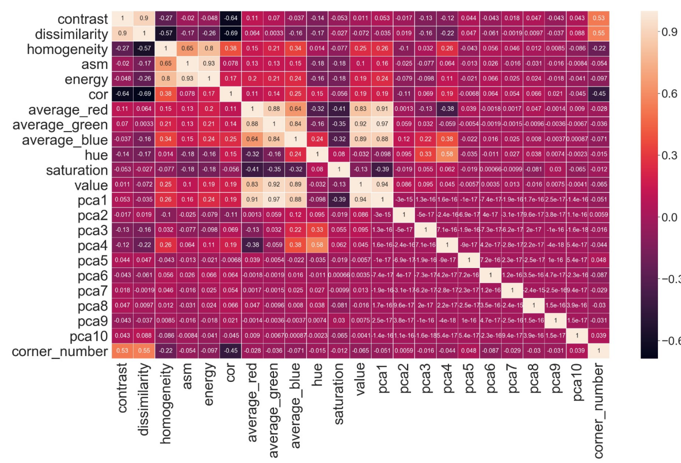

Fig. 10. Correlation heatmap of features. Some features have a high correlation close to 1

It is worth pointing out how we initially thought the red intensity can be useful in classifying comet pictures but we have to drop it due to its high correlation with first principal component.

C. Feature generation summarize

Of all the features we generating, the first two component of PCA can be the most efficient in classification, which can be directly seen from Fig. 7. And the features generated based on image size can be of least distribution since the size of image seems to be random and have no distinct pattern among classes. What's more, we added an interesting feature harris corner to detect a unicorn picture because of the distinct horn of unicorns. At the beginning, the feature red intensity seemed to be an efficient feature for classification, since this feature shows obvious difference among classes. However, this feature had high correlation with the PC1, and then we eventually dropped it. After all the work, we eventually keep 17 features. We didn't add more features in case of overfitting.

D. Classifier training and hyperparameter tuning

Now we have extracted and selected 17 features. Before we train our models on the learning set, we also did a standard scaling on these features. Then We trained five models in total including support vector machine (SVM), logistic regression, k-nearest neighbours (KNN), decision tree, and random forest. To tune their hyperparameters, we used the five-fold cross validation with classification accuracy score as the metric. A grid search for the optimal hyperparameters is performed. The detailed performance of each model is summarized in Fig. 11 It is worth noticing that the decision tree model has the worst performance while all other models are comparable. This is probably because the decision boundary among the classes are not parallel to feature directions as shown in Fig. 7. Among all the five models, the SVM model has the best performance. We trained SVM on the entire learning set with the optimal hyperparameters. A classification on the test set is then generated with the trained model.

Model	Grid search range	Optimal hyperparameters	Best score
SVM	Linear kernel, $C = 0.1, 1, 10, 50$ rbf kernel, $C = 0.1,, 1, 10, 50$, $\gamma = 0.005$ to 10 Poly kernel, $C = 0.1,, 1, 10, 50$, $\gamma = 0.005$ to 10	rbf kernel $C = 50$, $\gamma = 0.008$	0.392
Logistic regression	L2 penalty, multinomial, $C = 1, 10, 50$ L1 penalty, multinomial, $C = 1, 10, 50$	L2 penalty, $C = 10$	0.372
kNN	$k = 1$ to 50	$k = 17$	0.347
Decision tree	Loss function: Gini impurity, cross entropy Max depth = 1 to 30	Gini impurity Max depth = 9	0.278498
Random forest	Loss function: Gini impurity Number of tree = 100, 200 Max depth = 7, 50, 100 Max samples ratio = 0.1 to 0.9 Max features ratio = 0.1 to 0.9	Gini impurity Number of trees = 200 Max depth = 50 Max sample = 0.7 Max feature = 0.9	0.388

Fig. 11. Model performance. The score refers to validation set only.

III. CONCLUSION

In this report, we discussed how features are built and selected and then compared the performance of five machine learning models. There are surely a lot of space to improve the accuracy of our prediction as the validation score is still very low. For traditional CV methods, a good classifier cannot be built without carefully chosen the features and often it requires a domain knowledge on the specific data set. The current dimension of feature space is very limited (17 only). One thing we can do is to use the more complicated SIFT or SURF descriptor to increase the dimension of our feature space and it should capture more information in the data set. To overcome the lack of domain knowledge, we may use some automated feature mining techniques as well [7].

REFERENCES

[1] N. O'Mahony et.al., "Deep learning vs. traditional computer vision" in Comp. Vision Conf.(CVC), Las Vegas, Nevada, United States, April, Dec. 2019

[2] D. Masri, Z. Aung, and W. L. Woon "Image classification using appearance based features" in 11th intern. conf. on innov. in infor. tech. (CIIT), 2015

[3] D. Lowe "Distinctive image features from scale-invariant keypoints". International journal of computer vision, 2004

[4] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeeded-Up Robust Featurs", Preprint on Elsevier, 2006

[5] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textual features for image classification", IEEE transactions on systems, man and cybernetics, Vol. SMC-3, No. 6, pp. 610-621, Nov. 1973

[6] R. Lohavi and G. John, 'Wrappers for feature subset selection'. Artificial intelligence, 97(1):273-324, 1997

[7] P. Dollar, Z. Tu, H. Tao and S. Belongie, "Feature mining for image classification" in conference on computer vision and pattern recognition (CVPR), 2007