Leveraging Transfer Learning for Feature Engineering for Classification Task

Tony Ni, Kyle Truong
Department Mathematics and Statistics
Binghamton University
tni2, ktruong2@binghamton.edu

Abstract

This paper explores efficient image classification using a dataset of 2,000 cat and dog images. Various classical and modern feature extraction techniques, including resizing, PCA, edge detection, logistic regression, SVM, a self-built CNN, and ResNet-50, are employed and compared. ResNet-50 demonstrates superior accuracy at the expense of longer training times. To address this, a hybrid model combining ResNet-50 for feature extraction and SVM for classification achieves a balanced performance. Future work involves exploring different pre-trained models and evaluating ResNet-50 feature extraction on edge-detected images for potential enhancements. The project is available on GitHub at https://github.com/ktnys/catdog.

1 Introduction

Image classification is a difficult task because it requires the computer to first detect the target in the input image. For computers, images are represented as matrices where each element corresponds to a pixel's intensity or color value. For grayscale images, a 2D matrix representation is sufficient, but for colored images, we require a 3D tensor (height, width, channel) representation. To the computer, one set of pixels isn't any more important than another set, thus detecting a target in the image is difficult. [1] We would need to implement techniques to train the computer to detect the important features (in this case the pixel edges) of the target in the image.

This project will analyze 2,000 images of colored images of cats and dogs (1000 of each) and we want to classify the images as cats or dogs in the fastest and most accurate time. We try various classical digital image processing techniques for feature extraction followed with modern techniques.

We implement different feature engineering techniques (PCA, Frequency Domain Filtering, Spatial Filtering, Convolution) on images for faster and more accurate classification using both classical (Logistic Regression, SVM) and modern classification algorithms (CNN, ResNet-50). Our goal is to leverage our results to create a robust methodology for accurate image classification using computationally cheaper classification algorithms. Our implementation is available at https://github.com/ktnys/catdog

2 Related work

In 2019, Zhang and Wu proposed a motion classification algorithm using linear decision and support vector machines (SVM), inspiring our consideration of SVM for image classification with appropriate feature extraction. Instead of employing a kernel function in Linear Discriminant Analysis (LDA), as suggested by Zhang and Wu, we opted for ResNet-50 for extracting optimal classification features,

MATH457: Statistical Learning, Fall 2023, Binghamton University, NY. (LateX template borrowed from NIPS 2017.)

given its pretrained capabilities. Furthermore, the utilization of Convolutional Neural Networks (CNNs) for feature extraction, particularly the ResNet architecture introduced by He, Zhang, and Pan, has gained attention for its effectiveness in learning features. Building on the visualization techniques pioneered by Zeiler and Fergus, we explore ResNet-50's intermediate feature layers to deepen our understanding of feature extraction and model representation. [3,5]

3 Dataset and Features

We obtained our dataset from Kaggle's Cat vs Dog competition hosted in 2013. Our entire dataset contained 25,000 images of cats and dogs evenly split. We choose to work with only 2,000 of the 25,000 for computational convenience, but we expect reproducible results in larger samples. All the images were colored and in different resolutions. We then split the 2,000 images 80-20 for training and validation.

3.1 Preprocessing

Resizing: We standardized image sizes for training and validation, employing `OpenCV` to resize all 2000 images to 224x224 resolution—a common practice in deep learning and necessary for ResNet-50. The resized images were then saved to a dedicated

PCA From the resized images, we isolated the RGB channels, performed PCA individually on each channel with a selection of 10 principal components, and subsequently merged the three channels to generate a lower-resolution image, eliminating redundant information. All PCA images are stored in a separate folder.

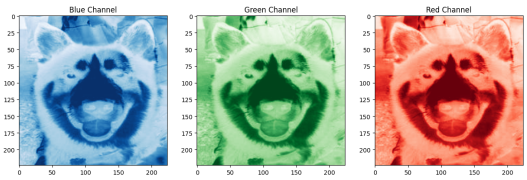

Figure 1: An random image in the dataset split into 3 color channels.

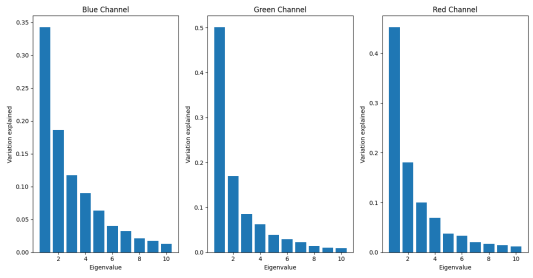

Figure 2: The percentage variation explained by the principal components.

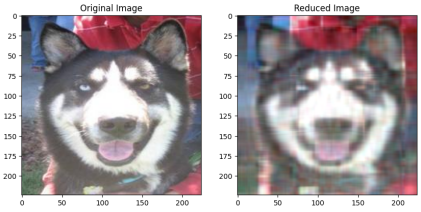

Figure 3: Left: Original Image Right: Merged image after selecting 75 principal component

Edge Detection Using the resized (Non-PCA) images, we applied spatial and frequency domain filtering to engineer new features by transforming the original ones. Specifically, we employed the Weighted Averaged Filter for spatial filtering (smoothing) and the Laplacian Filter for frequency domain filtering (edge detection). The process involved smoothing the image with the Weighted Average Filter followed by sharpening the edges with the Laplacian Filter, and the resulting images were saved to a dedicated folder.

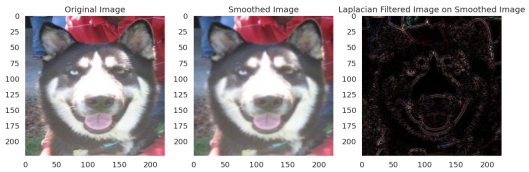

Figure 4: Left: Original Image Right: Merged image after selecting 75 principal component

Edge Detection and PCA Taking the filtered images, we applied the same PCA procedure as above. We split the images into the 3 channels, performed PCA and selected 75 components. We saved all the images to a separate folder.

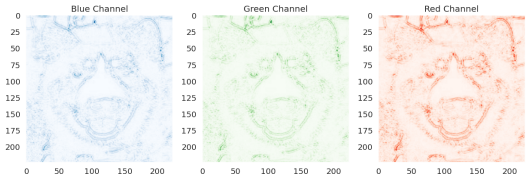

Figure 5: An random image in the dataset split into 3 color channels.

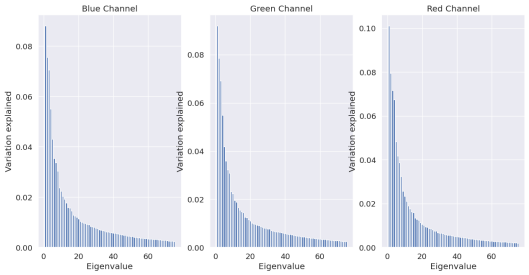

Figure 6: The percentage variation explained by the principal components.

ResNet-50 Feature Extraction Take all the resized images, and input them into the ResNet-50 model. ResNet-50 extracts feature from input images through a series of convolutional layers, activation functions, pooling layers, and residual connections.

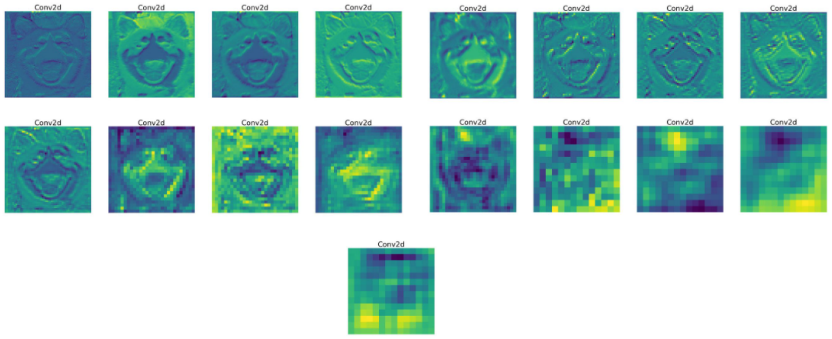

Figure 7: Visualizing intermediate feature layers in ResNet-50

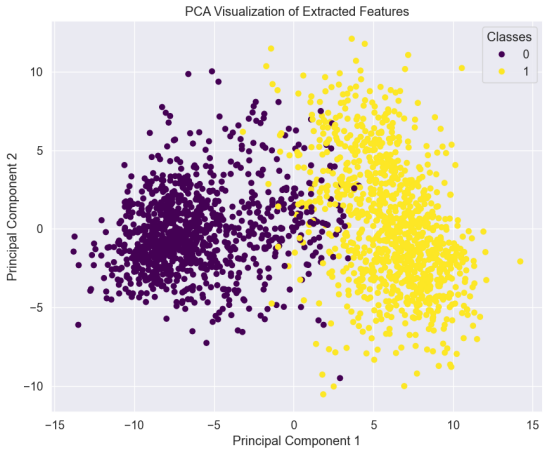

Figure 8: Project of the features extracted by ResNet-50 into two dimensions using PCA

4 Methods

4.1 Principal Component Analysis

PCA, or Principal Component Analysis, serves as a dimensionality reduction technique. In the context of images, PCA transforms the original pixel values into a new set of uncorrelated variables known as principal components. The objective is to capture the most critical information in the data, providing a more concise representation of the image.

4.2 Logistic Regression

We initially employed logistic regression, a widely used method for binary classification tasks. Analyzing image features, it provided a probability output for each image being a cat or dog. The logistic function transformed this output to a range between 0 and 1. The simplicity of logistic regression facilitated interpretability and established a robust baseline for comparative analysis as we progressed to more sophisticated models.

4.3 Support Vector Machine

Support Vector Machines (SVMs) are a machine learning method for classification tasks. SVM aims to identify the hyperplane that effectively separates different classes by utilizing support vectors—points in each class closest to the hyperplane. SVM is versatile, handling non-linear relationships between classes through kernel functions.

4.4 Simple CNN

Our CNN contains three convolutional layers with ReLU activation and max-pooling, transitioning to fully connected layers through flattening. The model uses two dense layers with ReLU activation in the fully connected segment, ending with an output layer adaptable for binary or multi-class classification

5 Results

5.1 Standalone Logistic

The first standalone logistic model had a training time of 1.28 minutes, accuracy of .54, and an AUC score of .54. This model, while quick to train, performed poorly as it predicted almost as many wrongs as it did right.

5.2 Logistic Regression With PCA

The second model, applying PCA, had a training time of 1.16 minutes, accuracy and AUC scores of 0.58 each—showing slight improvement but still performing suboptimally. The simplicity of logistic regression, beneficial for interpretability, may limit its capacity to discern the complexities inherent in distinguishing between cats and dogs based on image features, likely due to non-linear relationships. To address this, we progressed to more advanced image classification methods.

5.3 Self-Built CNN

We initially developed a custom model, which exhibited a training time of 4.609 minutes, accuracy and AUC scores of 0.6675 and 0.67, respectively. Although the accuracy fell short of expectations, it represented an improvement over previous models. However, overfitting became an issue, evident from the graph where validation accuracy remained below 0.70, contrasting with the increasing training accuracy approaching 0.95.

5.4 ResNet-50

Moving on, we chose the pre-built ResNet-50 architecture as our next model. The ResNet architecture has been widely recognized as an immensely powerful deep learning neural network in classification

tasks. The specific variant we utilized, the ResNet-50, consists of 50 layers. Applying it to our data set, it had an evaluation time of 19.454 minutes and an accuracy of .9775. As you can this model has significantly outperformed any of our previous models when it comes to accuracy, but did take much longer.

5.5 SVM with ResNet-50

To tackle the problem of time in the next model, we adopted ResNet-50 for its feature extraction capabilities, and combined it with a SVM model. It took ResNet-50 1.5 minutes for feature extraction, and the SVM model roughly .0342 seconds to train. This model yielded an accuracy of .97. Utilizing both ResNet-50 and SVM allowed us to greatly reduce the training time while maintaining accuracy.

5.6 Result Summary

Model	Training Time (Min)	Accuracy
Logistic Regression	1.279	0.54
Logistic Regression (PCA)	1.162	0.58
SVM	6.981	0.5375
SVM (Edge, PCA)	3.35	0.6025
SVM (ResNet-50)	1.5	0.97
CNN	4.609	0.6675
ResNet-50	19.454	0.9775

Figure 9: Summarizing of all the model results. Here we see that using ResNet-50 to extract features and then SVM to separate the classes produced the highest accuracy and almost the fastest time.

6 Conclusion

Utilizing transfer learning, particularly with models like ResNet-50, proves effective for feature extraction rather than direct detection, as these models are pretrained to recognize image patterns. This approach ensures both high classification accuracy and efficient training. Figure 8 illustrates the clear representation of data classes through feature extraction with ResNet-50. Our recommendation is to leverage pretrained deep learning models for optimal feature extraction, followed by classification using computationally less expensive algorithms. While Figure 9 shows a slightly lower accuracy of SVM with ResNet-50 feature extraction compared to using ResNet-50 alone, the trade-off is a significantly faster training time

In future experiments, exploring various pretrained models and comparing their accuracies based on feature extraction will be insightful. Additionally, our current feature extraction involved plain resized images; considering ResNet-50 feature extraction on edge-detected images might yield more linearly separable features, as it would focus specifically on the edges of the target. This approach could potentially enhance classification performance

References

[1] Gonzalez, R. C., & Woods, R. E. (2017). *Digital Image Processing*. Pearson.

[2] Marr, D.& Hildreth, E. (1980). Theory of edge detection. *Royal Society of London*. Series B. Biological Sciences, 207(1167), 187–217.

[3] Zhang, F. & Wu, TY. & Pan, JS. et al. Human motion recognition based on SVM in VR art media interaction environment. *Hum. Cent. Comput. Inf. Sci.*, **9**, 0 (2019). https://doi.org/10.1186/s13673-019-0203-8

[4] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *In Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

[5] Zeiler, M.D., & Fergus, R. (2014). *Visualizing and Understanding Convolutional Networks.* In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds) Computer Vision ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8689. Springer, Cham. https://doi.org/10.1007/978-3-319-10590-1_53