Northwestern University

Card Classification

EECS 349 Machine Learning

Group 18
Elton Cheng
Yuchen Rao
Weiyuan Deng

Instructor
Douglas Downey

06/03/2017

Objective and Introduction

The goal of our project is to classify cards for a game of BlackJack. Given an image of a card, can the computer correctly classify it? (Ace, 2-10, Jack, Queen, King).

This task explores the idea of object recognition, a tool that is being used more and more in fields, such as self-driving cars and pick/place and sorting robots. Object recognition can provide more information for computers to make decisions on by being able to tell apart a red balloon to a stop sign, or a bottle of gatorade from dish soap.

We used playing cards specifically because for people, it is easy for us to classify each card, as we can tell them apart from the letters, numbers, artwork, etc. of the cards. The challenge for a computer to classify these cards comes in trying to recognize these features of the cards, and being able to interpret these features correctly. Each of the cards are unique enough that it can be a challenge for a computer to try and classify them all.

Data and Approach

The original dataset contained 1,040 images of one deck of 52 cards. Each card has 20 pictures with the same background in different orientation.

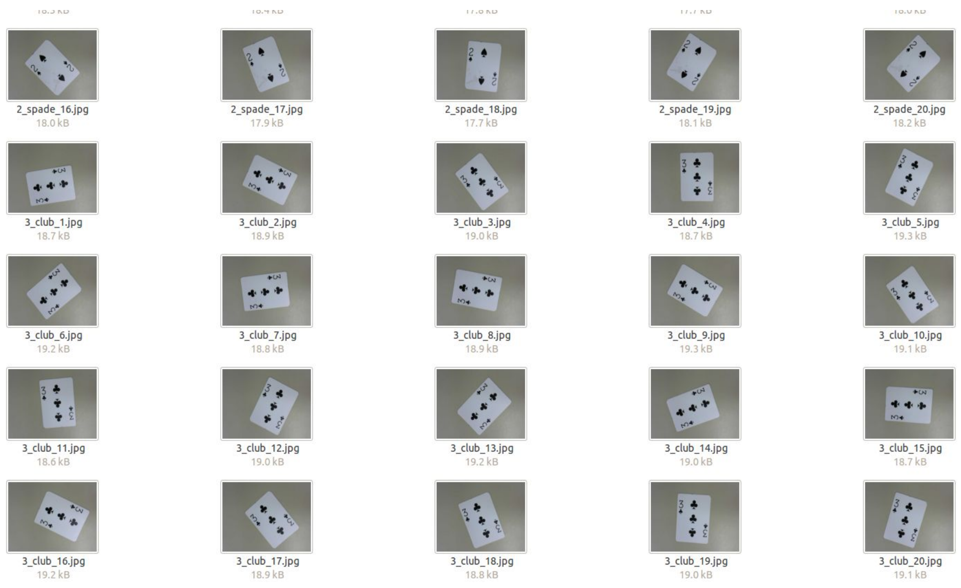

Figure 2. Dataset contains 1040 images of a deck of 52 cards

We obtained the images by taking pictures of each card with an outer webcam fixed at a constant height and labeled them by changing file name.

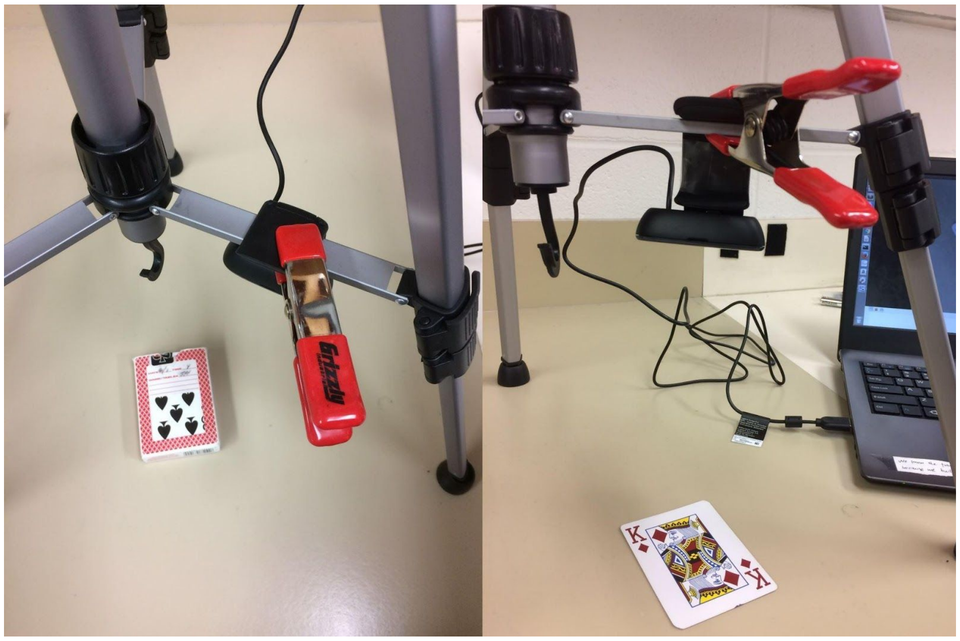

Figure 3. Camera with fixed height is getting picture of cards

We applied 10 data augmentation methods (translations, rotations, scaling, zooming, stretching and compressing) to each image in the original, resulting in a new augmented dataset of 10,400. The reason for doing this step is to explore the feature set the algorithms can explore and increase the accuracy when classifying the cards.

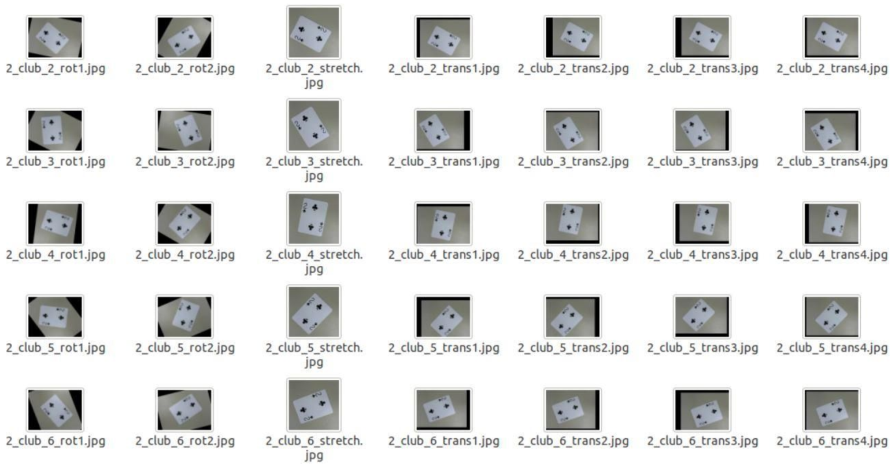

Figure 4. Dataset after data augmentation

Algorithms we initially looked at were K-Nearest Neighbor (KNN, k = 5), Support Vector Machine (SVC, linear kernel), and Neural Net (NN, 15 Layers) algorithms from scikit-learn and

Convolutional Neural Net (CNN) algorithms from the tflearn package. 10-fold cross validation was used to verify the accuracy of the models we generated.

Dense SIFT (DSIFT) was used to extract the features that the model will train on for sckit-learn algorithms. The feature we used for CNN is convolutional features, created by the convolution filter of the algorithm. When using the tflearn package, the image inputs were scaled down from 800x600 to 50x50 in order to reduce training time.

We tried preprocessing the dataset by getting edge information of the cards as features; however, this pre-processing method did not improve performance and we did not continue using it.

Results

The training data set is split up into 70% used for training and validation, and 30% used for testing. The ZeroR for this data set is expected to be 7.69%, since we are currently only looking at recognizing the value of the cards, not the suits. Our initial results with the algorithms found in scikit-learn are:

Algorithm	KNN (%)	SVM (%)	NN (%)	CNN(%)
Accuracy (cross-validation)	25.40	53.00	53.60	60.00
Accuracy (testing)	27.10	50.60	51.50	77.41

Using TensorFlow, the parameters for the CNN algorithm were 4 layers with 128 neurons. As the accuracy for different models we generated is relative low, data augmentation is applied to enlarge the data, and to expand the feature set the algorithms can explore. We decided not to continue looking at K-NN due to its low initial accuracy. Also, because CNN is a better neural net method for images than NN, we decided to focus on CNN and SVM.

The accuracy of SVM and CNN with data augmentation are shown below. The parameters for this model of CNN was 2 layers with 250 neurons. It was found that more layers did not necessarily increase accuracy, while more neurons improved accuracy.

Algorithm	SVM with Data Aug (%)	CNN with Data Aug(%)
Accuracy (cross-validation)	87.35	97.13
Accuracy (testing)	86.00	98.31

Conclusion and Future Work

Our initial training and testing with the 1040 images dataset has a relative low accuracy (~50%), so we decided to apply data augmentation to our original dataset to expand our feature set. We also used Dense SIFT algorithms, a tool used for object *detection,* to extract SIFT features from each image. The SIFT algorithm available in OpenCV was used to do this.

Using data augmentation, the accuracy of the SVM model has a significant increase of around 35%. That is because after augmenting images, we have more data in different directions and sizes, which increase the size of the dataset and features can be captured by the model.

By analyzing the results of different learners corresponding to their features, our group was able to get a model with the highest accuracy of 97.13%, which was given by CNN with convolutional features.

The reason why CNN outperformed SVM was because the SIFT features used by the SVM algorithm are usually meant for object *identification*, not classification. CNN uses many layers and filters to get convolutional features for each image which can provide features for classification.

Our classifier can only classify cards with the deck of cards it was trained with; it will not work as well with other decks of different design. The camera used for taking pictures was fixed in a constant height, which is not flexible in application. Therefore, we will enlarge our dataset with more cards in different design and take pictures with the camera in changing height or different background and lighting environment. Apart from the accuracy, we can expand our project by also classifying cards with value and suits, as well as implement a basic AI that can play BlackJack based on the cards identified on the table.

Tasks Completed by Group:

Weiyuan Deng: Labeling data, website, final report.

Yuchen Rao: Data capturing, wrote machine learning code, final report.

Elton Cheng: Data capturing, wrote data augmentation code, edited final report.