Investigating the Use of Synthetic Data for Training Machine Learning Models for Classification Tasks on the Rock Paper Scissors Dataset

Sonali Mahendran
Princeton University
sonalim@princeton.edu

Johan Ospina
Princeton University
jospina@princeton.edu

Abstract

Currently, the quality of machine learning vision algorithms is directly tied to the quality and quantity of images collected for training. It can be difficult and inefficient to collect and label these Natural Images (NI). Thus, Computer Generated Imagery (CGI) can pose as a scalable solution to this ubiquitous machine learning need for big data. Recent advances in CGI have allowed the increase in throughput of photorealistic renders. In this project, we investigate whether salient features can be identified in CGI data and used in classification tasks for NI. We focus this investigation on the simple 3-way classification of poses in the classic Rock-Paper-Scissors (RPS) game. This is a simple task for humans, which means there exist some features that uniquely describe each of the poses, regardless of dataset membership. We present an analysis of the impact of three feature extraction methods on Support Vector Machine (SVM) classifier performance for this RPS classification task. The SVM trains on features extracted in three different ways from CGI data, and its performance is evaluated on a test set of NI.

1 Introduction

Artificial intelligence is becoming increasingly dependent on large datasets. Some applications, such as social media, reap the benefits of an increasingly large set of labeled data (tagged images in the example of social media). For this reason, facial recognition is one of the most accurate AI technologies (White et al.). On the other hand, certain applications struggle with acquiring effective data to train their models. Until this challenge is solved, these applications will not achieve the high quality performance required for their downstream tasks. Recent work has investigated the task of training models to differentiate between synthetic and NI (Ng and Chang, 2006). Others have studied the impact of using synthetic data to train deep learning models (Tremblay et al., 2018). In this work, we use a CGI dataset to train a classifier to differentiate among the NI of three hand gestures involved in the game of Rock-Paper-Scissors (RPS). The images from the CGI dataset are different from those of the NI dataset in terms of pose shape, lighting, etc. However, we hypothesize that there exist some salient features that are unique to each class and are shared across the two datasets. To investigate the correctness of this hypothesis, we implement three feature extraction methods: Principal Component Analysis (PCA), Histogram of Oriented Gradients (HOG), and Bag of Visual Words (BOVW). We then compare the classification performance for the PCA-SVM, HOG-SVM, and BOVW-SVM setups.

The rest of this report is organized as follows. Section 2 summarizes related work on use of CGI images for machine learning applications. Section 3 provides an overview of the data. Section 4 discusses the steps involved in data preprocessing. Section 5 details the feature extraction methods we implemented. Section 6 describes the SVM classifier. Section 7 presents the experimental results. Section 8 concludes this report.

2 Related Work

Understanding the potential of using synthetic datasets in machine learning applications is an active area of ongoing research.

d'Acremont et al. address object identification and recognition in the wild for infrared imaging in defense applications. No large-scale groundtruthed dataset is available for training deep learning models for this application. They propose a compact and

fully convolutional CNN architecture with global average pooling and show that this model achieves state-of-the-art object identification performance compared to other CNNs, when trained on realistic simulation datasets. d'Acremont et al. conclude that realistic synthetic images are required during the training phase to achieve good recognition performance. Further, the use of a Global Average Pooling layer was key in achieving the state-of-the-art performance in the identification task (d'Acremont et al., 2019).

Movshovitz et al. focus on semi-automating dataset creation by using synthetic data and use this dataset in the tak of object viewpoint estimation. A large labeled dataset of cars was rendered in viewpoint space using state-of-the-art rendering software. They studied the impact of rendering parameters on estimation performance and concluded that realism is important. Further, it was shown that generalizing from synthetic data was not harder than the domain adaptation required between two natural-image datasets. Moreover, combining synthetic images with a few examples of natural data improves the accuracy of viewpoint estimation (Movshovitz-Attias et al., 2016).

Sun et al. investigate the potential for using virtual data rendered from 3D models in training object detectors. Their experiments show that detectors trained on virtual data and adapted to natural image statistics achieve performance comparable to that of detectors trained on natural image datasets like ImageNet. Further, Sun et al. conclude that non-photorealistic data works as well as a dataset comprised of more realistic renders. They evaluated their detectors on rigid man-made objects (Sun and Saenko, 2014).

Julien de la Bruère-Terreault developed a computer-vision and machine-learning-driven version of the Rock-Paper-Scissors game. The game is between the computer and a human. The game uses a Raspberry Pi computer and Raspberry Pi camera. The camera takes a picture of the the human's gesture while simultaneously playing *rock*, *paper*, or *scissors*. After identifying the human's gesture and applying the rules of the game, the computer maintains the scores for both players.

The classifier has been trained on previously labeled images corresponding to *rock*, *paper*, and *scissors*. The PCA components are extracted from each class of the training data such that at least 80% of the total variance is explained by the principal components. These principal components are the input to an support vector machine (SVM) classifier. Given an input image, the SVM will return a prediction for the class to which the image belongs (de la Bruère-Terreault, 2019). This work has heavily influenced the direction of our project.

3 Description of the Data

Canadian aerospace engineer Julien de la Bruère-Terreault used a Raspberry Pi camera to capture consistent images of people making *rock*, *paper*, and *scissors* gestures. The data contains a total of 2188 RGB images of 300 × 200 pixels in .png format. There are 726 images of *rock*, 710 images of *paper*, and 752 images of *scissors* hand gestures. The dataset contains a combination of right and left-handed poses. All images are taken on a green background with relatively consistent lighting and white balance (de la Bruère-Terreault). Throughout the rest of this report we refer to this dataset as the *Natural Image* (NI) dataset.

The CGI dataset contains 2892 images of diverse (varying in race, age, and gender) hands in *rock*, *paper*, and *scissors* poses. There are 840 images of *rock*, 840 images of *paper*, and 840 images of *scissors* hand gestures. All of the images are of right-handed poses against a white background, and each image is 300 × 300 pixels in 24-bit color (Moroney).

Sample images from both datasets are shown in Figures 1, 2, and 3.

4 Data Preprocessing

The first step of our pipeline is to equalize the image backgrounds and resize the images.

Figure 1: Examples of *rock* images in the natural (left) and CGI (right) datasets

Figure 2: Examples of *paper* images in the natural (left) and CGI (right) datasets

Figure 3: Examples of *scissors* images in the natural (left) and CGI (right) datasets

4.1 Background Equalization and Image Resizing

The NIs are 300×200 pixels; whereas, the *CGI* are 300×300 pixels. To ensure both datasets contain images of all the same size, we added 50 rows of zero-padding above and below the original image. This ensured both datasets contained 300×300 pixel images. Next, we made sure that both datasets have the same color background. The *CGI* dataset is set on a white background, and the NI dataset is set on a green background. We handled the difference in background colors by implementing an equalization step. In this step, the pixels were thresholded on their hue distance from the background hue. The images were then converted to grayscale. The NI dataset consists of images of both right-handed and left-handed poses. The *CGI* dataset, on the other hand, consists of only right-handed poses. For this reason, we created a copy of all the *CGI* images and flipped them over the y-axis. In this way, we ensured that both datasets have right and left-handed RPS poses.

4.2 Histogram Matching

We experimented with Histogram Matching as a means of normalizing the lighting conditions among images of the two data sets. Histogram Matching is a technique used to match the histogram of pixel intensities in one image to the histogram of another image as shown in (Finklestein, 2018). To implement this technique, we created a gradient image, where the image was a smooth gradient from 0-255, as shown in Figure 4. This image was used as the template for matching in the Histogram Matching process. By applying this image as the reference for histogram matching, we attempted to equalize the values of the pixel frequencies between the two data sets. Examples of Histogram-Matched images are shown in Figure 5.

5 Feature Extraction Approaches

We chose to implement three feature extraction techniques: PCA, HOG, and BOVW. As each feature extraction technique revealed interesting aspects about our data, we selected and implemented the next technique accordingly.

5.1 Principal Component Analysis (PCA)

Each image in our dataset contains 90,000 pixels (300×300 pixels). PCA allows us to extract low-dimensional representations of this information.

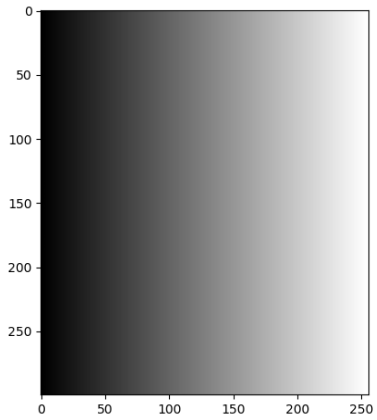

Figure 4: The reference image used as the template for matching

Figure 5: Examples of Histogram-Matched images from the CGI (left) and NI (right) datasets

PCA components are orthogonal vectors in the original feature space that are not correlated. The first component represents the direction of maximum variance in the data, the second component represents the direction of the second highest amount of variance in the data, and so on, in order of decreasing value of explained variance.

To extract principal components, we used the PCA transformer from the Scikit-Learn decomposition module (Pedregosa et al., 2011). We followed de la Bruère-Terreault's Jupyter Notebook to analyze the PCA components of the images in our dataset and determine how many principal components we needed to use to reconstruct images from either dataset (de la Bruère-Terreault, 2019). For the NI dataset, the first 40 principal components explained more than 80% of the total variance. For the CGI dataset, only about 70% of the total variance is explained by the first 40 principal components. Thus, for the CGI dataset, we used 70 principal components, as this explained roughly 80% of the total variance in the CGI dataset.

5.2 HOG

HOG is a feature descriptor used for object detection due to its success in recognizing shape-consistent objects (Russakovsky, 2019). This technique counts the frequencies of gradient orientations in localized regions of an image.

The image is first divided into smaller regions called "cells". For each cell, a histogram of edge orientations is computed for all of the pixels in that cell. The cells are then sorted into angular bins according to the gradient orientation. Adjacent cells are grouped into "blocks". This grouping is necessary for grouping and normalization of histograms. The block histogram refers to the normalized group of histograms, and the set of these block histograms is the HOG descriptor (Intel, 2019).

We used the HOG implementation from the Scikit-Learn feature module (van der Walt et al., 2014). We found that 9 orientations, 30×30 pixels per cell, and 2×2 cells per block yielded best results. The cells per block parameter was set as per (Dalal and Triggs, 2005).

5.3 BOVW

The last dimensionality reduction technique we focused on was Bag of Visual Words. The main benefit of this method is that it quantifies key points in an image so that it can determine what is contained in the image. This method does not account for the object's euclidean configuration. Instead, it only accounts for the features it can detect.

There were no readily available modules to implement BOVW, so we had to develop our own implementation of BOVW. For our implementation, we used SIFT features for each image in our CGI training data set. We chose SIFT features because they are robust to scaling, translation, and intensity (Lowe, 2004). These features were then fed to K-Means Clustering from SciKitLearn (Pedregosa et al., 2011). Once the means were identified, we generated a histogram of the nearest means for each feature in every image in both our

train and test data.

6 Support Vector Machine (SVM) Classification

SVMs have been shown to generalize well on difficult image classification tasks, where the only features are high dimensional histograms (Chapelle et al., 1999). The SVM algorithm uses the “kernel trick" to transform the data, which allows it to handle nonlinear input spaces. The algorithm then fits an optimal boundary hyperplane to separate the different classes in the data by maximizing the margin between the examples and the hyperplane. We use the SVM module from Scikit-Learn to construct our support vector classifier (Pedregosa et al., 2011). As per (de la Bruère-Terreault, 2019), we use an ‘rbf’ kernel. We implement five-fold cross-validated grid-search to optimize the following parameters: number of principal components, the regularization parameter, and the kernel coefficient for ‘rbf’. Provided an input of extracted features, the classifier outputs the class ‘rock’, ‘paper’, or ‘scissors’.

Table 1 shows the experiments we ran and the data we used for the train and test set in each experiment. We used NI/NI splits to ensure that the feature extraction methods we implemented were suitable for our data. To test our hypothesis, for each feature extraction method we implemented, we trained the SVM classifier on a CGI train set and evaluated the classifier on a test set of NI.

Table 1: Experiments

Experiments	Train Set	Test Set	Features
1	NI	NI	PCA
2	CGI	NI	PCA
3	NI	NI	HOG
4	CGI	NI	HOG
5	NI	NI	BOVW
6	CGI	NI	BOVW

7 Results and Discussion

7.1 PCA

7.1.1 PCA Visualization

When analyzing the PCA feature extraction method, we found it helpful to visualize the image reconstructions. Figures 6 and 7 show the images reconstructed from principal components for an NI and a CGI, respectively.

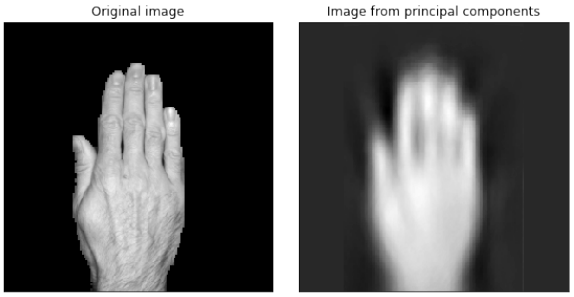

Figure 6: Reconstruction of an NI from its PCA components

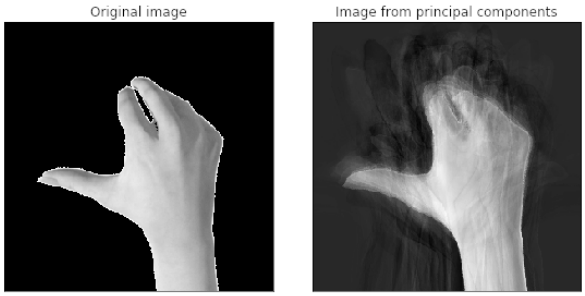

Figure 7: Reconstruction of a CGI image from its PCA components

Next, we used the PCA components extracted from the NI dataset to reconstruct a CGI (Figure 8, Top) and an NI (Figure 8, Bottom). The reconstruction of the CGI is not reflective of the “*scissors*" class; however, the reconstruction of the NI is reflective of the “*scissors*" class.

We then used the PCA components extracted from the CGI dataset to reconstruct an NI (Figure 9, Top) and a CGI (Figure 9, Bottom).

In this case, the reconstruction of the NI was poor, and the reconstruction of the CGI was rather reflective of the “*scissors*" class in the CGI dataset.

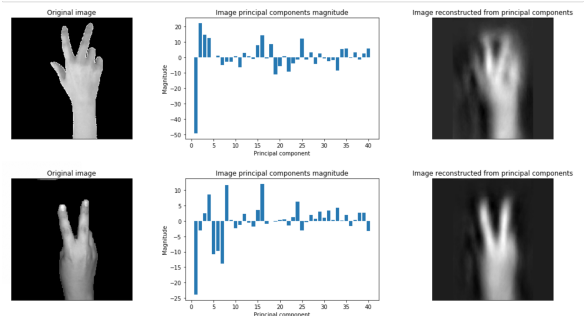

Figure 8: Reconstruction of a CGI and an NI from PCA components of the NI data set

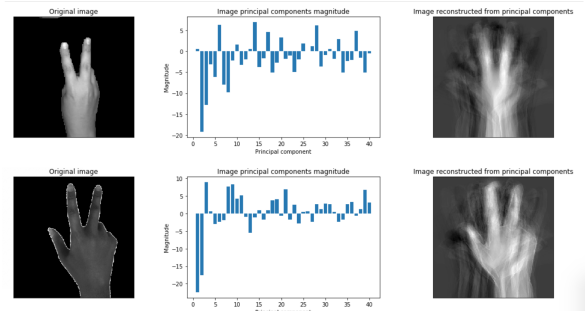

Figure 9: Reconstruction of an NI and a CGI from PCA components of the CGI data set

The average images of each class from both datasets (Figure 10) show that the CGI dataset is not representative of the NI dataset, to which it needs to generalize. There are clear, distinct patterns for each class in the NI average images. However, in the CGI average images, we see some “ghost" images. These “ghost" images represent the intra-class variation within each class. In other words, there are several representations of the same class in the CGI dataset. Some of these representations never show up in the NI dataset. One such example is the representation of “*scissors*". In the CGI dataset, all of the “*scissors*" images have a protruding thumb; however, this feature is never present in the NI dataset.

For all of these reasons, the PCA components extracted from the CGI dataset are not sufficient to effectively describe the images from the NI dataset.

7.1.2 PCA Experimental Results

According to our hypothesis, after training on the PCA features for the CGI train set, we expected the SVM classifier to achieve high performance on the NI test set. However, we found that the model trained on CGI does not generalize well to the NI and that the model trained on NI does not generalize well to the CGI. The F1-score for rows 4 and 8 in Table 2 reflect this issue. Further, the figure below is the confusion matrix that results from training the SVM on PCA features for CGI and evaluating it on NI. The confusion matrix indicates that the classifier is more likely to predict “Rock", regardless of the class.

There are several factors that may contribute to this issue. One factor we tried to address was the difference in lighting condi-

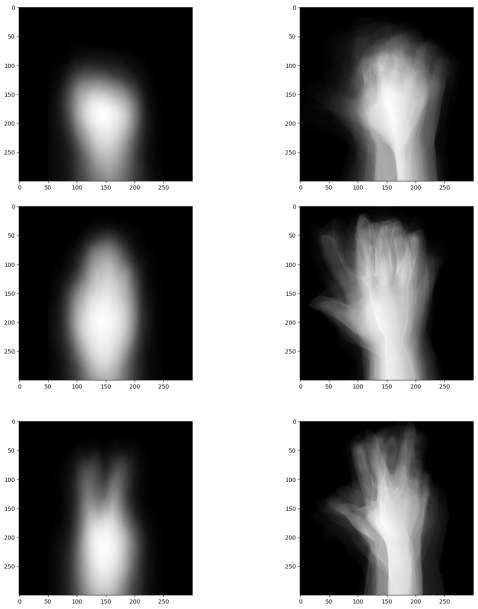

Figure 10: Average images of “*Rock*", “*Paper*", “*Scissors*" for both datasets (NI, left) and (CGI, right)

	R′	**P′**	**S′**
R	77	9	19
P	87	13	5
S	47	14	44

Figure 11: Confusion Matrix on CGI/NI with histogram matching Off

Table 2: PCA-SVM results

Exp	Train Set	Test Set	HM	F1-score
1	Natural	Natural	on	0.978
2	Natural	Natural	off	0.981
3	Natural	CGI	on	0.505
4	Natural	CGI	off	0.425
5	CGI	CGI	on	1.0
6	CGI	CGI	off	1.0
7	CGI	Natural	on	0.330
8	CGI	Natural	off	0.425

tions between the CGI and NI datasets. To do this, we implemented Histogram Matching.

Once we implemented Histogram Matching, we found that the classification performance was lower (Table 2) when a CGI train set and an NI test set were used. Because the focus of this project is to train a classifier on CGI and evaluate it on NI, we decided to stop using histogram matching with the remaining feature extraction setups.

PCA is a very statistical method which looks only at pixel values and finds lower-dimensional components that best describe the data. For this reason, it makes sense for it to perform poorly at describing data from a different distribution, especially when that distribution has different lighting conditions and poses. Thus, we decided that basing our feature extraction on solely pixel intensities is insufficient for this task.

7.2 HOG

Given images from either the CGI or NI dataset, it is easy to determine the class just by visual inspection. This means that there exist some structural features that are shared by images in both datasets and lend themselves to improved classifier performance. For this reason, we chose to implement HOG feature extraction.

The HOG representations of a single image for each class are shown in Figure 12. These figures clearly show the structural information available for each class in each dataset. Figure 13 shows the average HOG representations for "*Rock*", "*Paper*", and "*Scissors*" in both datasets.

The results for the HOG-SVM setup are

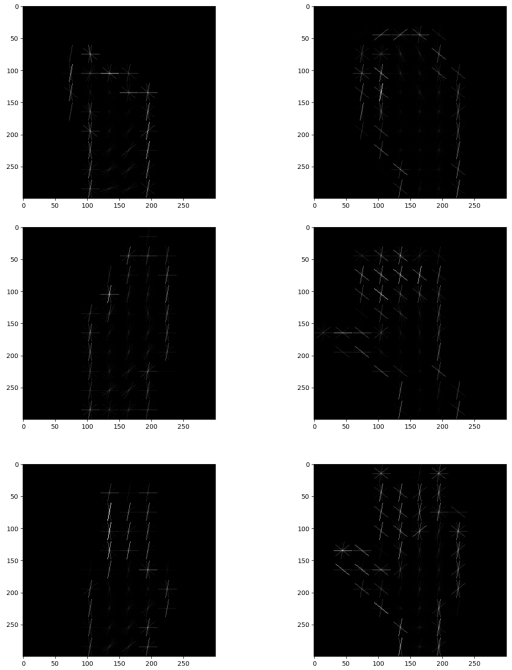

Figure 12: Single HOG representations for "*Rock*", "*Paper*", and "*Scissors*" (NI, Left) (CGI, Right)

shown in Table 3. The F1-score for Experiment 3 was 0.530, which was higher than that of the PCA-SVM setup. We believe that HOG works sightly better than PCA since the features are now capturing structural cues from their respective poses, instead of only pixel value information. This might explain why this classifier is more likely to predict "*Paper*". While the F1-score for Experiment 3 is not as good as that for the NI/NI split, the increase from the F1-score of Experiment 8 in Table 2 does show progress in the right direction.

Table 3: HOG-SVM results

Exp	Train Set	Test Set	F1-score
1	NI	NI	0.984
2	CGI	CGI	0.994
3	CGI	NI	0.530

7.3 BOVW

BOVW is a robust dimensionality reduction technique for classes that have little intra-class variation and, thus, can be described by the features of a visual dictionary cre-

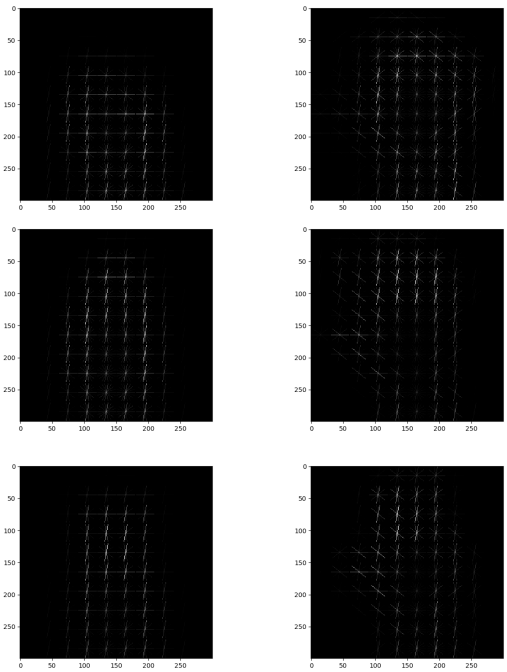

Figure 13: Average HOG representations for "*Rock*", "*Paper*", and "*Scissors*" (NI, Left) (CGI, Right)

ated from training data (Russakovsky, 2019). For our data, this method proved to be the best technique for dimensionality reduction. We believe this is due to the fact that the BOVW model relies on features without accounting for their spatial layout. Of our 3 feature extraction techniques, this proved to be the best with an F1-score of 0.606 (Table 4)and good precision (albeit recall values of 0.3/0.4). Another key result is the classifier's performance on the NI/NI split; it achieved an F1-score of 0.997 with the parameters in Table 4. This was the highest F1-score of all of the feature extraction methods we implemented when the train and test set were both NI.

Table 4: BOVW-SVM results (1000 vocab size)

Train Set	Test Set	Train Set	F1-score
NI	NI	1785	0.997
CGI	NI	1200	0.511
CGI	NI	1500	0.600
CGI	NI	2100	0.606

BOVW's successful performance on this classification task can be attributed to the fact that it is able to find important features between these two data sets that are more robust to the variation between the datasets, unlike PCA and HOG.

The average histograms of BOVW features for each class are shown in Figure 14. Note that all three histograms have a similar shape, with a few discerning peaks that are unique to a particular class. We tried many different vocabulary sizes, and we found that a vocabulary size of 1000 words was the best hyperparameter for our data.

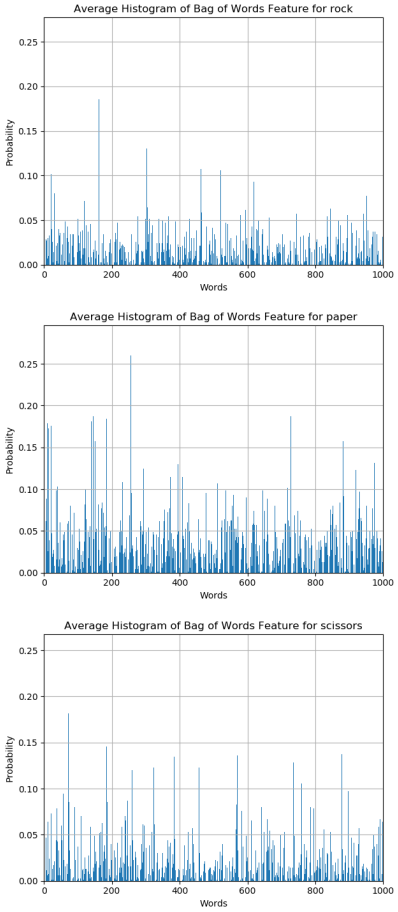

Figure 14: Average histograms of BOVW features for "*Rock*", "*Paper*" and "*Scissors*", respectively

8 Conclusion

By visual inspection, we find that the CGI and NI datasets are inherently different. Differences, such as pose shape and lighting conditions, introduced significant variance between corresponding classes across the two datasets. For example, there is significant

variance among the images within each class in the CGI dataset. Further, the poses for each class of the CGI dataset are different from that of the NI dataset. These conditions led the PCA-SVM setup to achieve low precision and recall. The HOG-SVM setup achieved slightly better performance since HOG features captured the structural composition of the images. The BoVW-SVM setup achieved the highest performance of all three setups. We believe this is due to the fact that BoVW relies on features that are not specifically tied to the spatial orientation of pixels. Finally, these results demonstrate that though these two datasets are different, they share key features that can be used to improve a classifier. As an extension of this work, one could try to design systems that allow him/her to specify the generation of CGI to more closely align with the NI data that will be encountered in downstream tasks. Other extensions of this work could entail implementation of other dimensionality reduction methods or pyramidal bag of words searches.

9 Appendix

	R′	**P′**	**S′**
R	104	1	0
P	3	100	2
S	0	1	104

Figure 15: PCA Confusion Matrix on NI/NI with histogram matching On

	R′	**P′**	**S′**
R	103	0	2
P	2	101	2
S	0	0	105

Figure 16: PCA Confusion Matrix on NI/NI with histogram matching Off

	R′	**P′**	**S′**
R	66	39	0
P	80	23	2
S	46	44	15

Figure 17: PCA Confusion Matrix on CGI/NI with histogram matching On

	R′	P′	S′
R	39	56	10
P	2	95	8
S	13	35	57

Figure 18: BOVW Confusion Matrix on CGI/NI with histogram matching Off for 2100 and 1000 vocabulary size

	R′	P′	S′
R	42	61	2
P	4	85	16
S	8	65	32

Figure 19: PCA Confusion Matrix on NI/CGI with histogram matching On

	R′	P′	S′
R	105	0	0
P	75	29	1
S	57	15	33

Figure 20: HOG Confusion Matrix on CGI/NI with histogram matching Off

References

Julien de la Bruère-Terreault. Rock-Paper-Scissors Images. https://www.kaggle.com/drgfreeman/rockpaperscissors#README_rpc-cv-images.txt. Accessed: 2019-11-05.

Julien de la Bruère-Terreault. 2019. rps-cv. https://github.com/DrGFreeman/rps-cv.

Olivier Chapelle, Patrick Haffner, and Vladimir Vapnik. 1999. Support vector machines for histogram-based image classification. In *IEEE transactions on Neural Networks*, pages 1055–1064.

Antoine d'Acremont, Ronan Fablet, Alexandre Baussard, and Guillaume Quin. 2019. Cnn-based target recognition and identification for infrared imaging in defense systems. In *Sensors (Basel)*, page 2040.

N. Dalal and B. Triggs. 2005. Histograms of oriented gradients for human detection. *CVPR*.

Adam Finklestein. 2018. Lecture 2 image processing. https://www.cs.princeton.edu/courses/archive/spr19/cos426/precepts/02-image_processing.pdf. Accessed: 2020-1-13.

Intel. 2019. Histogram of oriented gradients (hog) descriptor. https://software.intel.com/en-us/ipp-dev-reference-histogram-of-oriented-gradie

D. Lowe. 2004. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110.

Laurence Moroney. Rock Paper Scissors Dataset. http://www.laurencemoroney.com/rock-paper-scissors-dataset/. Accessed: 2019-11-05.

Yair Movshovitz-Attias, Takeo Kanade, and Yaser Sheikh. 2016. How useful is photo-realistic rendering for visual learning? In *European Conference on Computer Vision*, pages 202–217.

T. Ng and S. Chang. 2006. Classifying photographic and photorealistic computer graphic images using natural image statistics. *ADVENT Technical Report*, 220.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Olga Russakovsky. 2019. Lecture 9 object classification. https://www.cs.princeton.edu/courses/archive/fall19/cos429/slides/cos429_fall2019_lecture9_ObjectClassification.pdf. Accessed: 2020-1-12.

Baochen Sun and Kate Saenko. 2014. From virtual to reality: Fast adaptation of virtual object detectors to real domains. In *British Machine Vision Conference*, page 3.

Jonathan Tremblay, Aayush Prakash, David Acuna, Mark Brophy, Varun Jampani, Cem Anil, Thang To, Eric Cameracci, Shaad Boochoon, and Stan Birchfield. 2018. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. 2014. scikit-image: image processing in Python. *PeerJ*, 2:e453.

David White, James D. Dunn, Alexandra C Schmid, and Richard I. Kemp. Error rates in users of automatic face recognition software.