FPGA Board State Detection: SVM versus CNN

Medina Baitemirova
Stanford University
medinab@stanford.edu

1 Introduction

With the current pandemic and changing trends in education, there has been an increased demand for remote education solutions, especially in STEM fields. LabsLand is an education technology startup that provides remote access to STEM laboratories located in various parts of the world. In May 2021, there were 50,000 laboratory session provided by LabsLand and the numbers are expected to grow.

One LabsLand laboratory offers access to Intel field-programmable gate array (FPGA) boards. Students can develop code and test it using those boards. Anytime a code is pushed to the board, one or more light indicators on the board are turned on. With the growing number of users, a problem has been identifying when light indicators are on or off due to varying light conditions and due to different positions of those boards.

Computer vision is the field of computer science that enables obtaining meaningful information from a variety of visual inputs, including images and videos. It has seen tremendous growth in recent years, with applications ranging from facial recognition to automated content moderation.[1] Since computer vision aims to automate tasks that human vision can perform, this project aims to employ computer vision techniques and automate FPGA board light status detection.

The input to our algorithm are images of FPGA boards with each of the 10 light indicators turned on or off. We then use support vector machines (SVM) and a convolutional neural net (CNN) to output a label indicating whether the light is on or off.

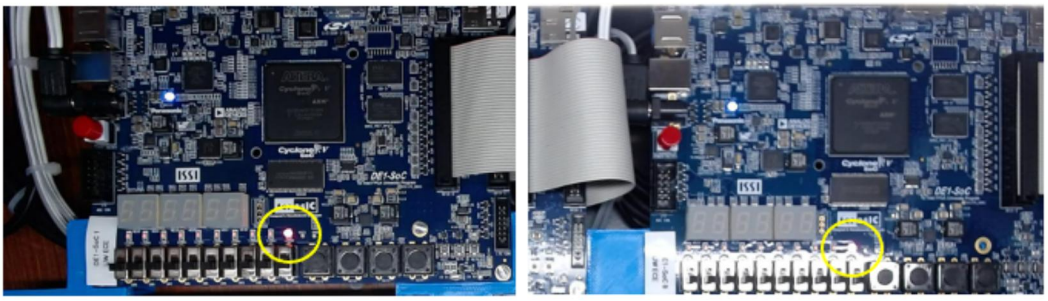

Figure 1: Examples of light indicators that are on

2 Related Work

Classical Data Augmentation: One of the mostly employed data augmentation methods is to perform affine image transformations, such as rotation, refection, scaling, shearing and color modifications. According to one paper, classical data augmentation techniques was one of the most successful strategies when classifying an ImageNet subset. Authors evaluate data augmentation techniques by restricting the dataset to only 2 classes.[2] While classifying dogs versus cats is a significantly different task than detecting the state of FPGA boards via light detectors, data augmentation is a reasonable strategy to employ. However, not all data augmentation techniques can

be generalized. For example, horizontal flipping, in some medical images and in our case, can lead to a change of the label.[3] Another paper analyzed the effect of classical data augmentation and found that it significantly improved model's performance.[4] While authors did a good job evaluating different models, one of the limitations of the paper is that they didn't evaluate the impact of each data augmentation method individually, which could be useful in our case.

Deep Transfer Learning (DTL): One paper evaluated common state-of-the-art deep learning models for detecting COVID-19 using chest CT scans. One of the models they evaluated was VGG-16 that was pretrained on ImageNet. They found that using pretrained weights resulted in good model performance, even if the CT scans were not part of the original training distribution.[4] The paper shows a fast and practical method using pretrained models. However, a limitation of this approach is that the sensitivity of the best performing model was 77.66%, which is low for healthcare. And while they addressed the class imbalance in their evaluation metrics, they didn't perform data augmentation to address class imbalance before the training.

Addressing Variations in the Lighting: Another group of researchers evaluated a CNN in the context of detecting road boundary lanes. Their dataset included photos taken during day time, night time, and with low light, which is very similar to our dataset. However, their approach combined CNN with line detection, which is useful in case of road lanes. They didn't report the performance of the CNN without the line detection, which could be useful in our case.[5]

3 Dataset and Features

Data Generation: Data was generated by capturing photos from a video stream at three different locations at different times of the day to capture varied light conditions. The purpose was to generate images with each of the 10 LED lights on, and images with all of the LED lights off, resulting in 11 classes for the classification task. Examples are illustrated in Figure 1. Number of images per location and the number of images used for training/validation/test are summarized in Figure 2.

Data preprocessing: VGG-16 was pretrained on ImageNet and the input sizes were 224x224. Since our images ranged from 540x960 to 720x960 in size, they were resized to match VGG-16's original input size. Moreover, VGG-16 expected a normalized input, therefore, we applied normalization as well. We used the same mean and standard deviation used for ImageNet.

Location	Number of boards	Total number of images	Training	Validation	Test
Pamplona, Spain	9	3960	2376	792	792
Pamplona, Spain	6	2640	1584	528	528
Seattle, USA	7	3080	1848	616	616

Figure 2: Summary of data statistics per location

Histogram of Oriented Gradients (HOG) for SVM: Feature descriptors encode distinctive features of an image in an informative and non-redundant way. One of the popular feature descriptors is the Histograms of Oriented Gradients (HOG) that uses a gradient-based representation. An example of an FPGA board image and a corresponding HOG of a fixed 128x64 size is in Figure 3.

Data Augmentation for VGG-16: We have performed data augmentation on input images to increase the training set size and regularize the network to avoid overfitting. More specifically, we employed affine data augmentation techniques to proxy varying positions of boards and varying camera angles. Moreover, to adjust for varying light conditions and differences due to cameras we used varying brightness, contrast, saturation, and hue.

4 Methods

For the purpose of this project, we evaluated SVM and VGG-16, a popular CNN.

Support Vector Machines (SVM): We selected SVM as the baseline model for our project because it is viewed as one of the benchmarks in machine and statistical learning. It is originally a binary

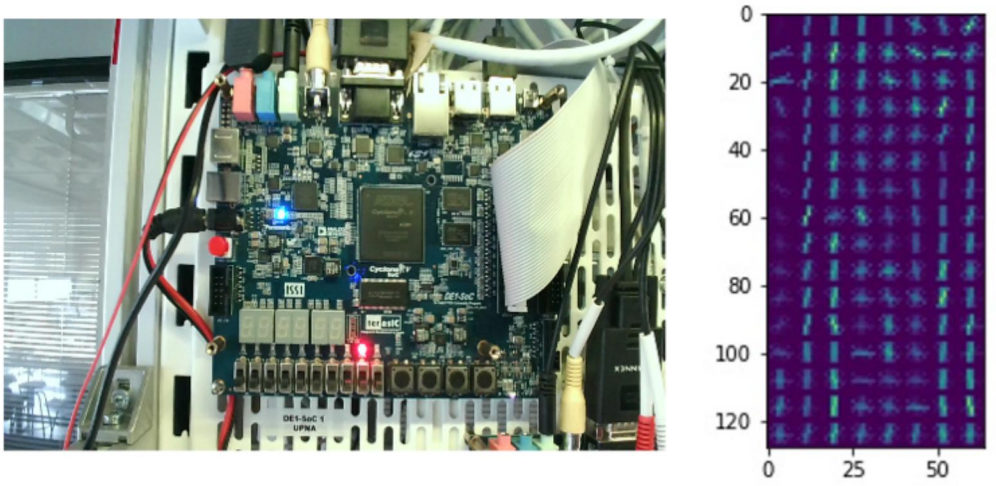

Figure 3: HOG example

classification model, but has been adapted for multi-class classification tasks. SVM works by creating a hyperplane that separates the data into different classes. The optimal hyperlane is the one that has the largest margin, which is a distance between the hyperlane and the closer data points, also known as support vectors. Support vectors influence the position of the hyperplane and make SVM less sensitive to outliers. When linear separation is not possible, data can be mapped into a higher dimensional feature space that allows linear separation. To avoid associated high computational costs a kernel trick is employed that allows performing the computation in the lower dimensions using a dot product. The SVM used in our project uses a one-versus-one strategy that fits one classifier per each pair of classes and selects the class with the most number of votes when classifying. We employed a soft margin with the following loss function:

$$min_{w,b}||w||_2^2 + C\sum_j \ell_{hinge}(y_j, w \cdot x_j + b)$$

Where the first term is the regularization term and the second term is the empirical risk minimization using hinge loss.[6] Among limitations of SVM is the fact that it expects a numeric input, therefore, it is necessary to perform feature extraction from the images.

Convolutional Neural Network: Due to the image complexity, however, we expected that deep learning methods will perform better and have evaluated a deep CNN, VGG-16, that has 134,305,611 parameters, 16 layers (13 convolutional, 3 fully connected). It was pretrained on ImageNet with an input size of 224x224x3 but it can also work with 256x256x3 images. Convolutional layers are followed by a rectified linear activation unit (ReLU), which is an activation function that outputs the input value directly when it is more than zero, and returns zero otherwise. There are 5 max pooling layers that down-sample the input.[7] The output node uses softmax function that converts the output of the final layer into a vector of probabilities that all sum to 1.[8] The softmax formula is as follows:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Visualization using Grad-CAM: We produced a coarse localization map of the final convolutional layer of VGG16 using Gradient-weighted Class Activation Mapping (Grad-CAM).[9]

5 Experiments, Results, and Discussion

Evaluation metrics: Since our data was balanced, we used accuracy to evaluate models' performance, which measures proximity of predicted labels to ground truth labels. To describe model's

performance, we used a confusion matrix, since it directly illustrates correct and incorrect classifications.

Baseline SVM: We estimated the hyper-parameters of SVM using grid search with 5-fold cross-validation, which resulted in a gamma value of 0.0001 and a C-parameter value of 1. SVM using a linear kernel achieved an accuracy of classification of 82.22%.

VGG-16: We used default VGG-16 settings: a learning rate of 1e-5, cross-entropy loss function, batch size of 40, and an Adam optimizer, which is a replacement optimization algorithm for stochastic gradient descent in deep learning.

Training, validation, and test accuracy values for each type of data augmentation are reported in Figure 4.

Data Augmentation	Training Accuracy	Validation Accuracy	Test Accuracy
None	93.43%	92.68%	**94.59%**
Center crop	85.79%	85.03%	84.70%
Random resize	89.75%	91.87%	93.27%
Random rotation	92.49%	92.68%	**94.50%**
Random color change	93.52%	92.97%	**94.65%**
All	88.81%	91.06%	92.46%

Figure 4: Training, validation, and test accuracy for various data augmentation methods.

The confusion matrix is illustrated in Figure 5. The salience mapping using Grad-CAM is illustrated in Figure 6.

174	0	8	0	0	0	0	0	0	7	3
2	183	5	0	0	0	0	0	0	0	0
4	0	188	0	0	0	0	0	0	0	0
3	0	6	182	3	0	0	0	1	0	0
0	0	8	0	184	4	0	0	0	0	0
1	0	6	0	0	177	0	0	0	0	0
1	0	2	0	0	0	189	0	0	0	0
7	0	7	0	0	0	0	181	0	1	2
4	0	4	0	0	0	0	0	179	5	0
0	0	4	0	0	0	0	0	1	187	0
2	0	9	0	0	0	0	0	0	8	173

Figure 5: Confusion matrix for the VGG-16 model with random brightness, contrast, and saturation adjustments

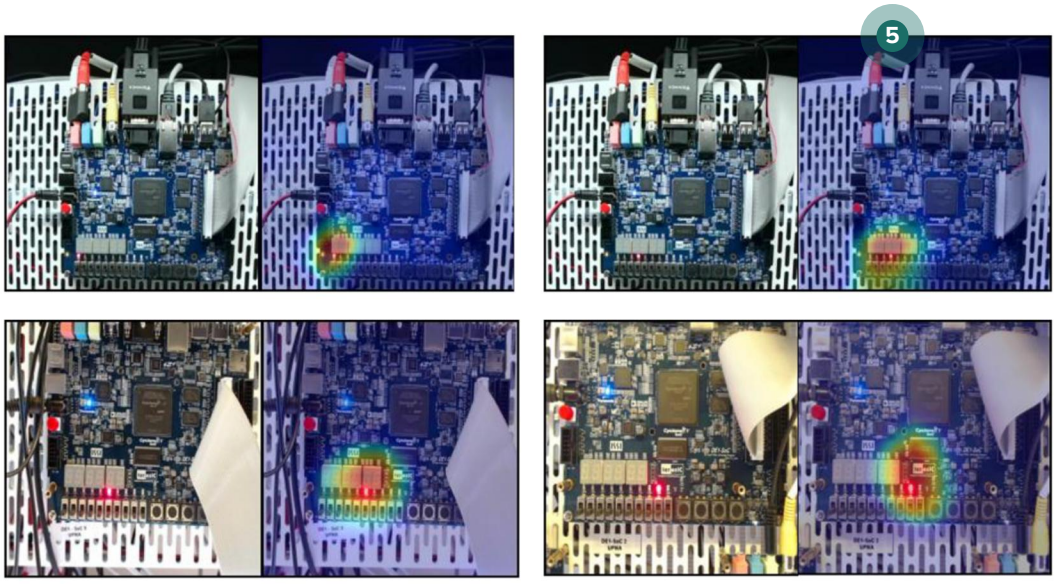

Figure 6: Examples of Grad-CAM generated maps

Overall, VGG-16 performed better than the baseline model. Surprisingly, the performance of VGG-16 without any data augmentation was comparable to the best performing model using data augmentation. One reason could be that our initial dataset taken at different times of the day and different locations already incorporated rotation and color invariance and had an adequate number of training examples. The best performing data augmentation techniques were Random Rotation with an angle of 15° and Random Color Change. Randomly rotating each image mimics the camera angles and positions, therefore, the model generalizes well to a variety of angles. In real-life, camera angles don't have a lot of variation, therefore, the angle of rotation we chose was reasonable. Randomly adjusting the brightness, contrast, and saturation mimics differences in images due to different cameras and different conditions, such as day versus night. Therefore, in addition to increasing the training dataset size, these data augmentation techniques also incorporated rotation and color invariance into the model. One data augmentation technique that performed worse than others was Center Crop. Even though we resized each image to 256x256 before cropping to 224x224, when analyzing FPGA boards we found that some images had the light indicators at the edge of the image. That means that Center Crop, in some cases, was cropping out the light indicators. According to Figure 4, there wasn't a significant variation between training, validation, and test accuracy values for each model. Therefore, the model didn't overfit to the training dataset but further evaluation is needed. Figure 6 shows maps generated by Grad-CAM and we can see that the model was not looking at spurious features but at light indicators that were on when classifying.

6 Conclusion and Future Work

For this project, our goal was to automate FPGA board light indicator status detection and evaluate two methods: SVM and a CNN. The task was classifying 11 classes: one class per each of the 10 light indicators that is on and one class with all of the light indicators turned off. We found that VGG-16 has a high accuracy without any data augmentation techniques and has the best performance with random brightness, contrast, and saturation adjustment. Future work includes fine-tuning parameters for VGG-16 and computing mean and standard deviation statistics for normalization. Since our image domain wasn't similar to any of the ImageNet domains, the fact that we used ImageNet mean and standard deviation statistics could have negatively impacted the model's performance. Another next step could be using deep learning methods for data augmentation, given that most of the classical data augmentation did not have a significant impact, and one of them even hurt the model's performance. Lastly, future work could include trying different models. Even though VGG-16 has made significant contributions to the field of image processing and is still considered as one of the popular models, it is becoming outdated and newer, smaller models could be evaluated.

References

[1] Tarleton Gillespie. Content moderation, ai, and the question of scale. *Big Data amp; Society*, 7(2):205395172094323, 2020.

[2] Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning, 2017.

[3] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1), 2019.

[4] Mohamed Loey, Florentin Smarandache, and Nour Eldeen M. Khalifa. A deep transfer learning model with classical data augmentation and cgan to detect covid-19 from chest ct radiography digital images. 2020.

[5] Satish Kumar Satti, K. Suganya Devi, Prasenjit Dhar, and P. Srinivasan. A machine learning approach for detecting and tracking road boundary lanes. *ICT Express*, 7(1):99–103, 2021.

[6] David Sontag. Support vector machines (svms) lecture 2.

[7] Xiangyu Zhang, Jianhua Zou, Kaiming He, and Jian Sun. Accelerating very deep convolutional networks for classification and detection, 2015.

[8] Softmax function, May 2019.

[9] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, Oct 2019.