Kaggle Report: Solar PV in Aerial Imagery

Team H - Han Yan, Kate Coulter,
Xuan Yu, Avani Gupta, Daniel Witt

March 6, 2019

Abstract

Image classification by machine learning techniques is an important, growing field with numerous applications. For our in-class Kaggle competition, we developed an algorithm to detect and make a binary decision on whether a solar panel exists in aerial imagery data or not. By researching previous machine learning applications to imagery data of various disciplines, as well as by studying the specific methods and parameters we chose for our model, we offer a model that leverages transfer learning on pre-trained VGG-16 specifications using augmented data, achieving AUC values of 0.998 on the private leaderboard and 0.984 on the public leaderboard, earning fourth place in the competition.

1 Introduction

Image analysis by machine learning techniques is a growing field with applications in numerous disciplines, thus offering an opportunity for interdisciplinary exploration and insight. Algorithms that are able to accurately detect certain features in images create a way for new types of data to become available at a lower cost than it would have been previously. In addition to offering cost relief to manual evaluation, these algorithms could perform more efficiently than humans could analyze the same amount of data.

For this project, we were challenged to create algorithms using machine learning techniques that are able to make a binary decision on whether a solar panel exists or not in each aerial imagery data. As our demands on energy continue to change based on human activity, the ability to quickly analyze where solar panels are currently paired with demographic data that matches the geographic location of these images can be used by policy makers and elected officials for predicting where upgrades to the electric grid infrastructure should be made.

In the following sections we discuss our process for creating our final code to meet this goal. Briefly, we will mention previous studies that contributed to our decisions that form our code as well as explain the motivation behind this project, the data we had available to use, our methodology, results of different algorithms we considered using, and our conclusions on our own work as well as potential future research that could begin using our report

as a platform for new discovery. As a team project, we also describe the different roles team members played for this project.

2 Background

Image classification and recognition holds a very important place in applying machine learning techniques. Although our particular problem involved identifying solar aerial imagery, we explored other applications of machine learning techniques to images to understand best practices and the difference between models and techniques. In this section, we will summarize some of our key findings from these works, noting in places how these studies contributed to choices we made in our own methodology.

2.1 Review of Previous Applications

In "Combining Satellite Imagery and Machine Learning to Predict Poverty", Neal Jean et al. estimated consumption expenditure and asset wealth from combined nighttime maps with high-resolution daytime satellite images. Using a convolutional neural network (further refered to as CNN) trained with this data, they were able to identify image features that successfully explained 75% of the variation in economic experiences at the local level. Their method not only performed well in measuring consumption and wealth remotely, but demonstrated some of the advantages in CNN specific machine learning techniques, such as its performance despite having limited training data. [6]

Andre Esteva et al. demonstrate the successful application of CNN in "Dermatologist-Level Classification of Skin Cancer with Deep Neural Networks". In this paper, the scholars detail how CNN classified skin lesions using only pixels and disease labels as input data. To measure performance, the model was tested against 21 board-certified dermatologists on biopsy-proven clinical images where the doctors and the algorithm performed two binary classifications for each image - keratinocyte carninomas versus benign seborrheic keratoses and malignant melanomas versus benign nevi (essentially, each image was classified as containing a cancerous skin lesion or, rather, containing a benign skin lesion). The CNN performed at the same rate as all tested experts for both tasks, thus again suggesting the potential in CNN across various applications. [5]

These and other image classification techiques are described in George Magoulas and Andriana Prentza's "Machine Learning in Medical Applications". This paper also details the importance of these methods as they can greatly decrease cost for patients who would otherwise need expensive and often invasive procedures to determine diagnoses. [7] Finally, in "Deep Learning for Medical Image Processing: Overview, Challenges, and Future," Muhammad Razzak et al. describe various machine learning image processing techniques as well as their advantages and possible future applications. CNN is described as the technique with the largest potential given its accuracy and efficiency in learning. [12]

2.2 Solar Imagery Literature

There is a rich recent history of applications of CNN and other machine learning techniques in detecting solar arrays. Recently, Duke University released a new set of open-source data of more than 19,000 solar panels. [2] This dataset is particularly useful because it could immediately be used by utility officials to predict where to install or upgrade these systems as specific demands change, but more importantly it is a ground-truth dataset that allows researchers to train algorithms to automate the identification of solar panels, much like we do in this report. [10] One such technique is described in "Automatic detection of solar photovoltaic arrays in high resolution aerial imagery" by Jordan Malof et al. [8] Stanford University's DeepSolar project offers another successful usage of a deep learning model in detecting solar panel on satellite imagery. [17] Additionally, DeepSolar builds on the dataset offered by Duke University's Energy Initiative by building a nearly complete solar installation database for the contiguous United States.

2.3 Specific Considerations for Competition

To further our understanding of CNN techniques, we specifically researched best practices and risks associated with using this method. Our primary concern, as with most machine learning problems, was choosing a model that was just complex enough without overfitting. In the book *Challenges in Machine Generation of Analytic Products from Multi-Source Data*, we focused on understanding the notes from a chapter on this specific issue, "Session 2: Machine Learning from Image, Video, and Map Data." [9] Joseph Mundy describes in this chapter the three phases of satellite imagery analysis, which matched the processes used by the researches previously mentioned. Later in this chapter, different types of training data are explained as a way to choose what type of method to use, further justifying our choice of using CNN. "Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis" serves as a great source for any application of CNN techniques and argues that by far the most important step in using this technique is the way the training data is treated and analyzed, which influenced our choice in parameters and pre-processing steps. [14]

3 Data

Our data consists of a training image set of 1,500 images and a test image set of 588 images, all images in .tif format. We also were provided a sample submission file in the correct format to use as a template for submitting our own class confidence scores based on our training data that we submitted as our file for evaluation. Our training labels document contains two fields - id and label - which correspond to the id of our image file as well as the class of that image, 0 = "no solar array" and 1 = "solar array(s) present". All images in both data sets are tagged image files with three color channels of 101px-by-101px size.

Below are 10 examples of each class taken from the correctly labeled training data set. Figure 1 contains image files 85, 398, 426, 431, 568, 879, 981, 1003, 1241, and 1488. Each of these figures is of class 0 and thus does not contain solar arrays. Figure 2 contains image files 339, 437, 483, 578, 978, 1004, 1096, 1121, 1304, and 1333. Each of these figures is of

class 1 and thus does contain solar array(s).

Figure 1: Class 0 examples

Figure 2: Class 1 examples

By looking at examples from each class we are provided an opportunity to think through what special cases exist for each class so that we can fine tune our code used to predict the class of each image. First, as each image is 101px-by-101px we know that the quality of each image is not necessarily great. Thus, as we resize the image to get a better view of the figure and its features, we risk the image becoming blurry and the clarity diminishes. Also, as you can see in Fig. 1's final example very easily, the building structure that would have the solar array(s) built on them are not necessarily centered, but rather appear anywhere in the image, as this figure has the building in the top right corner of the image and, furthermore, the building takes up a small portion of the figure, with most of the image space showing other features. Furthermore, the fifth example of this class shows two building structures, both of which must be checked for solar array(s). This implies that the subsection based on color contrast cannot be used to view the building as a complete, one time occurring object in each image, but rather as something that can occur more than once in each case and also may not be completely captured as the bottom building in this image appears to be cut off part way through its actual structure.

For cases that do contain solar array(s), examples from Figure 2 show that solar array(s) are of varying size and shape (square or rectangular), do not all necessarily border each other (example 9, for instance has numerous solar array(s) some of which touch other solar array(s) and some of which do not. Finally, example 3 of Figure 2 shows that the color

contrast between solar array(s) and the surface its on is of varying degrees, as this image has a building and solar arrays that are quite similar in RGB depiction.

Note: The data used in this project were all provided by Kyle Bradbury through the Kaggle Competition webpage. [3]

4 Methods

4.1 Overall Workflow

For our base model, we trained KNN and random forest classifiers using scikit-learn and obtained an AUC of 0.6 using cross validation. [1] [11] Afterwards, we focused on using CNN in keras for image recognition, which gave us a significant boost. [15] [4]

4.2 Pre-Processing

For KNN and Random Forest, we read in the data as 3-dimensional array of 101X101 pixel images, averaged the R, G, B layers to obtain grey scale images, and extracted the mean and standard deviation as the two features.

For CNN, we generated batches of tensor images with real-time data augmentation using batch size of 32, the purpose of which is to artificially increase the variation of training set by flipping, rotating, shearing and varying the brightness of the images. Considering the 1500 training data are further split into training and validation, augmenting the data size can help reduce overfitting.

The augmentation techniques we adopted are rotating the images by up to 30 degrees, slightly shifting the image width of height by an fraction of 0.1, as well as horizontally flipping the images, under the assumption that horizontal flipping of images containing houses and solar panels preserve the integrity of the images while vertical flipping could create different contours.

4.3 CNN Model Architecture

We used the VGG-16 model for keras as our base for our model and added a fully connected dense layer to customize the classifier to the solar panel images. We finalized our model by using a sigmoid activation method for binary classification.

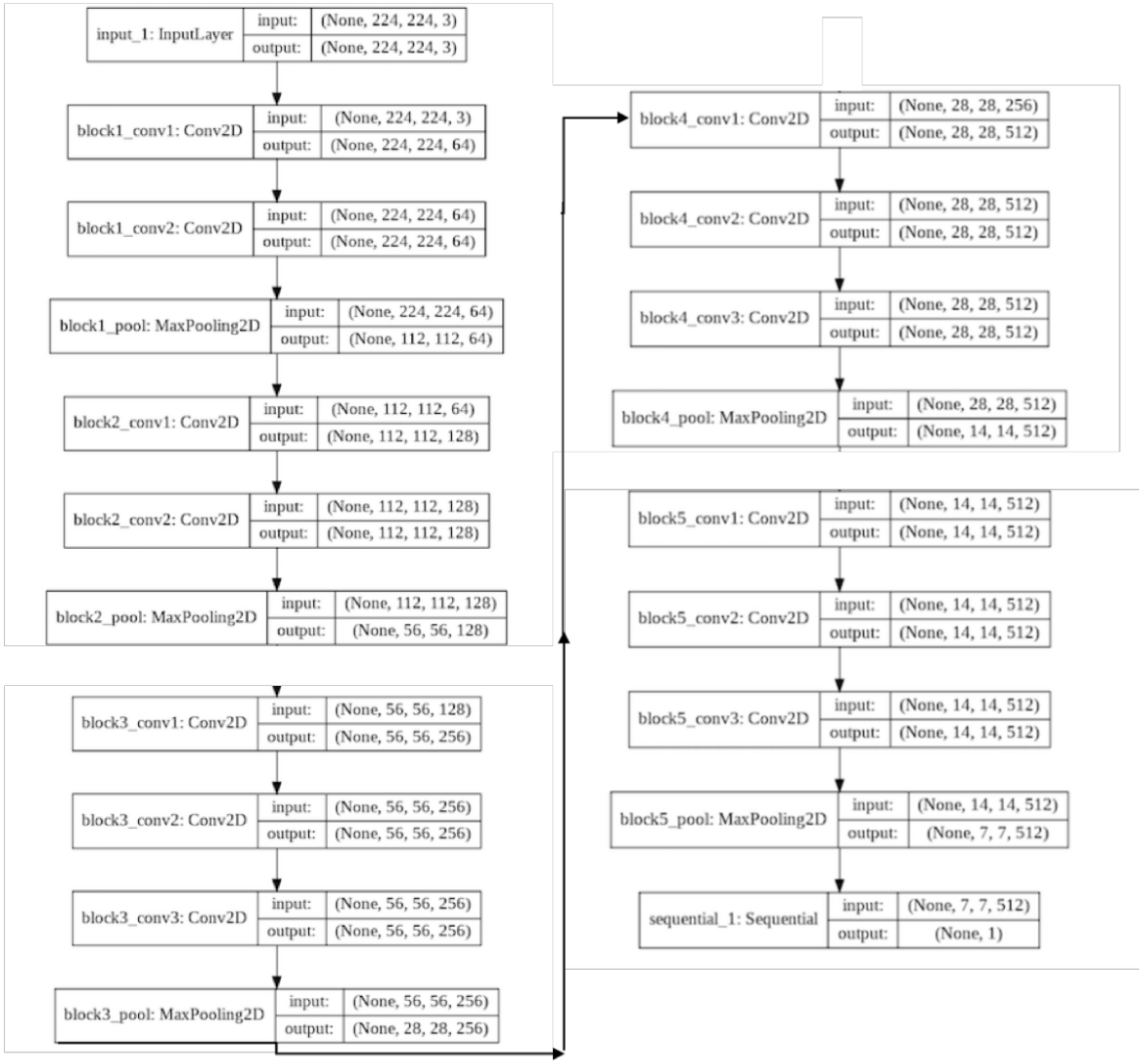

Figure 3: Model Architecture

4.4 Model Tuning

We ran the first CNN models with 40 epochs (each epoch takes around 30 minutes) on CPU and subsequently 100/200/300 epochs on GPU (each epoch takes around 30 seconds). The change in running our model on GPU instead of CPU is detailed by Microsoft Azure. [16] Epoch describes how many times we pass the entire data forward and backward through the neural network. It is divided into smaller batches to feed into the model.

In terms of model parameters, we used SGD (stochastic gradient descent) with momentum as optimizer on a learning rate of 1e-4. [13] This method utilizes exponentially weighted moving average of gradient vectors to accelerate gradients in the right direction, and leads to faster convergence than classical SGD. It is a widely used optimization algorithm.

The model with 200 epochs gave the best AUC of 0.998 between our models on our private testing board and resulted in a value of 0.984 on the public board.

4.5 Performance Measurement

After splitting the data into 80% training and 20% validation, we used binary cross entropy as loss to evaluate how good the predicted probabilities are, and used accuracy on the validation set to evaluate how well it predicts the binary outcome.

5 Results

5.1 Base KNN Model

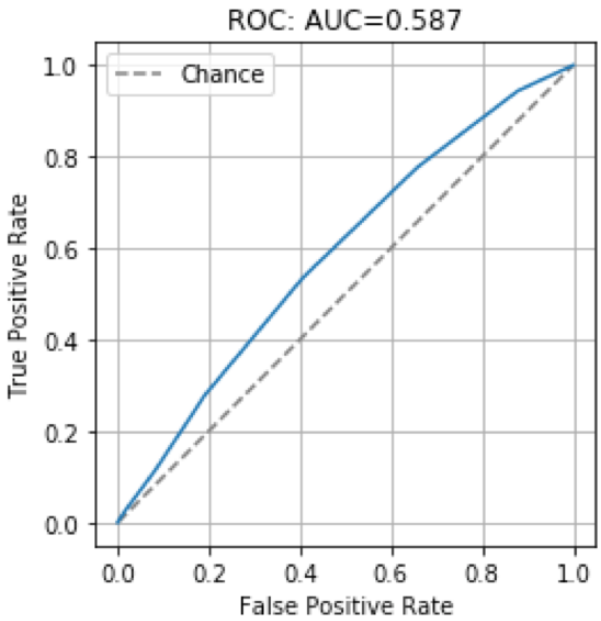

Figure 4: ROC Curve for Base KNN Model

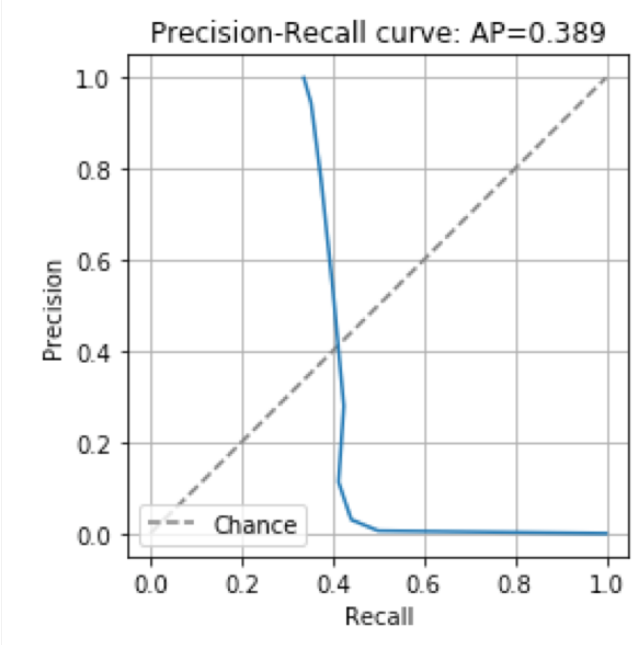

Figure 5: Precision Recall for Base KNN Model

5.2 Base Random Forest Model

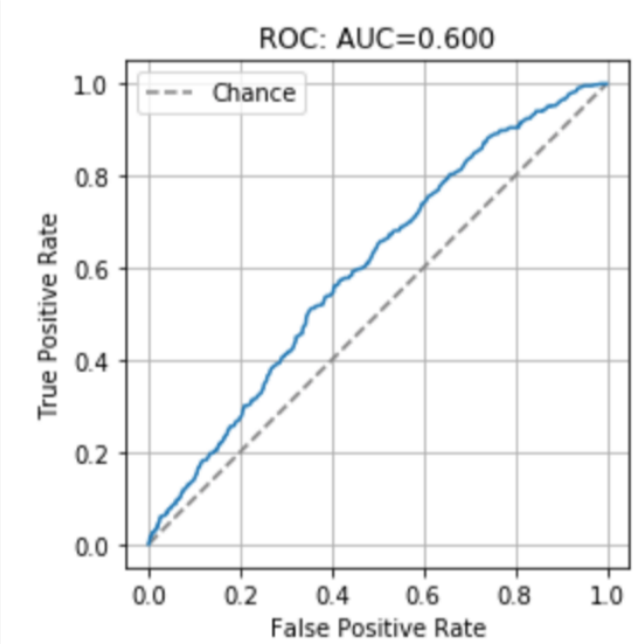

Figure 6: ROC Curve for Base Random Forest Model

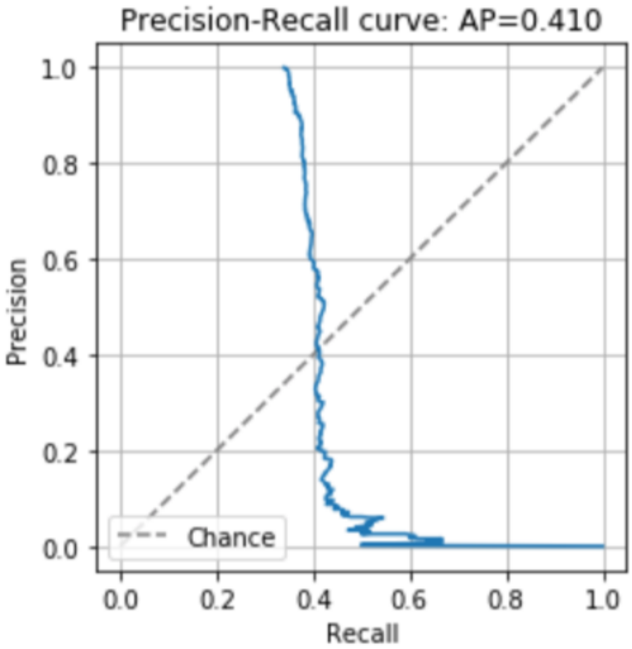

Figure 7: Precision Recall for Base Random Forest Model

5.3 CNN with Epoch 100

Performance of CNN model with epoch = 100.

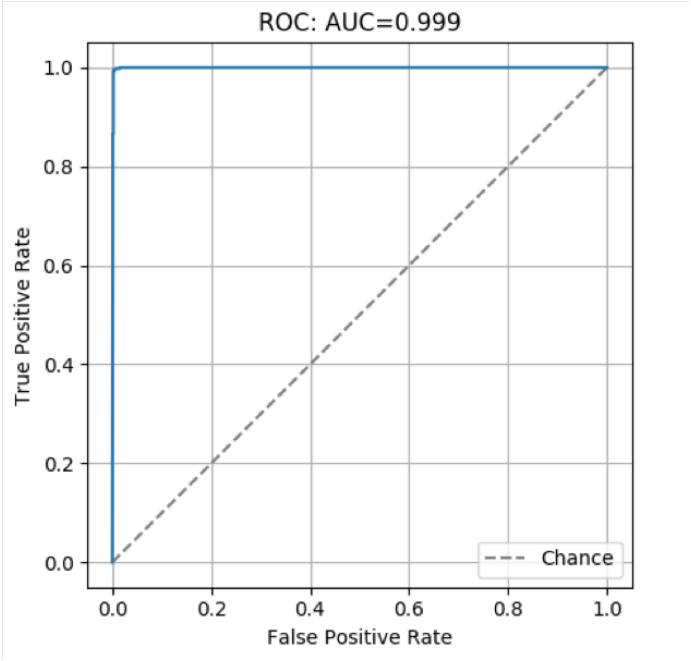

Figure 8: ROC Curve for CNN Model, Epoch = 100

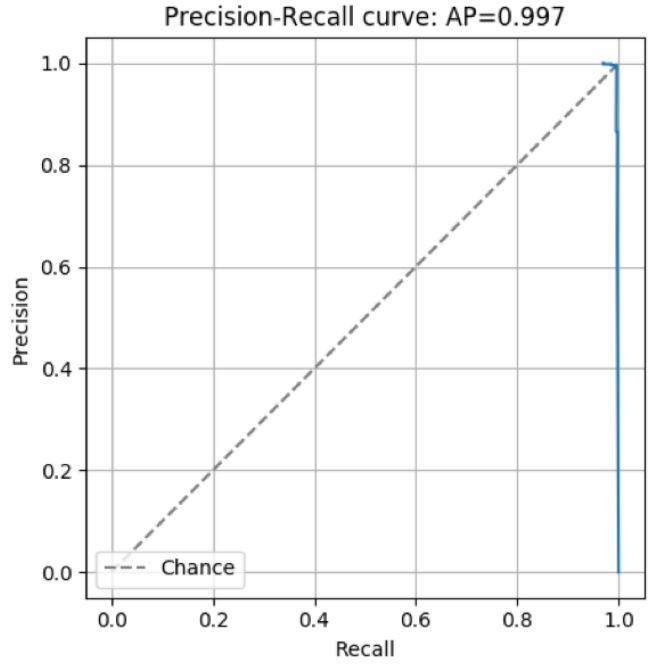

Figure 9: Precision Recall for CNN Model, Epoch = 100

5.4 CNN with Epoch 200

Performance of CNN model with epoch = 200.

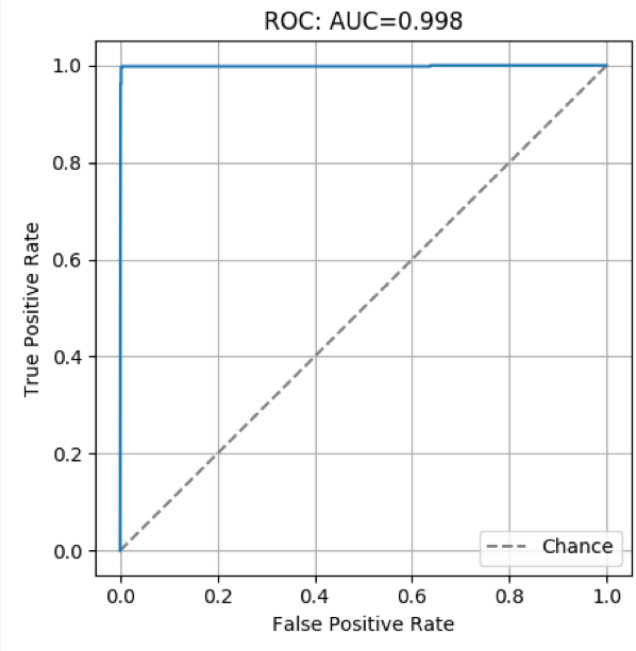

Figure 10: ROC Curve for CNN Model, Epoch = 200

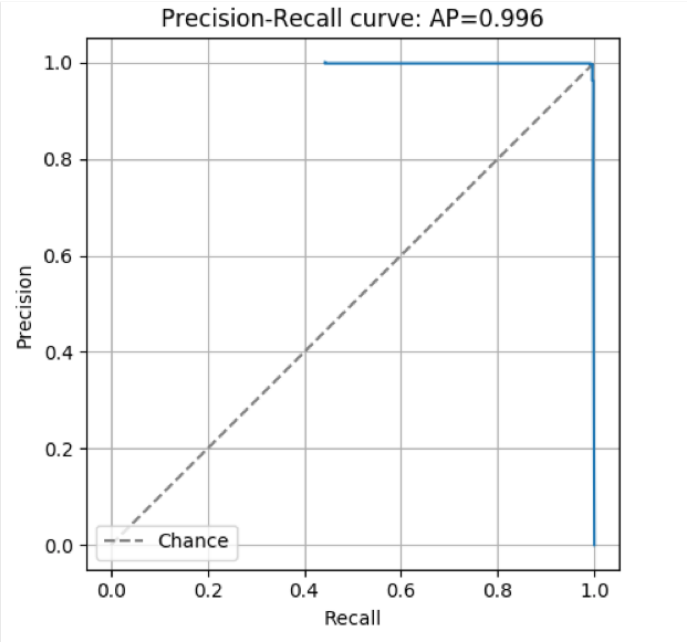

Figure 11: Precision Recall for CNN Model, Epoch = 200

5.5 Interpretation

The models with epoch 100 and 200 are similar in performance, and epoch 200 actually has marginally lower accuracy and average precision. However, based on testing data, the model with 200 epoch is our best model.

The model training history on training and validation data is listed below. We can observe that accuracy on the training is gradually increasing while plateauing on the validation set, while model loss is starting to increase on validation. As signs of overfitting emerged, we called epoch 200 our final model. [14]

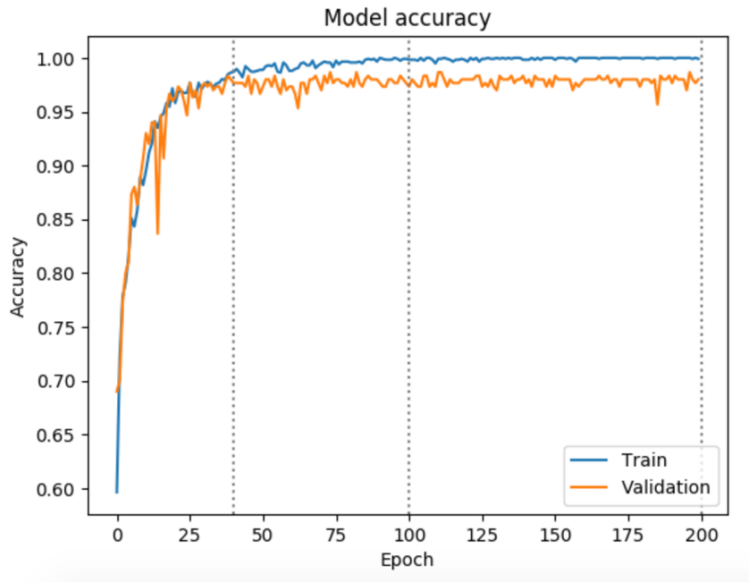

Figure 12: Accuracy During Training History

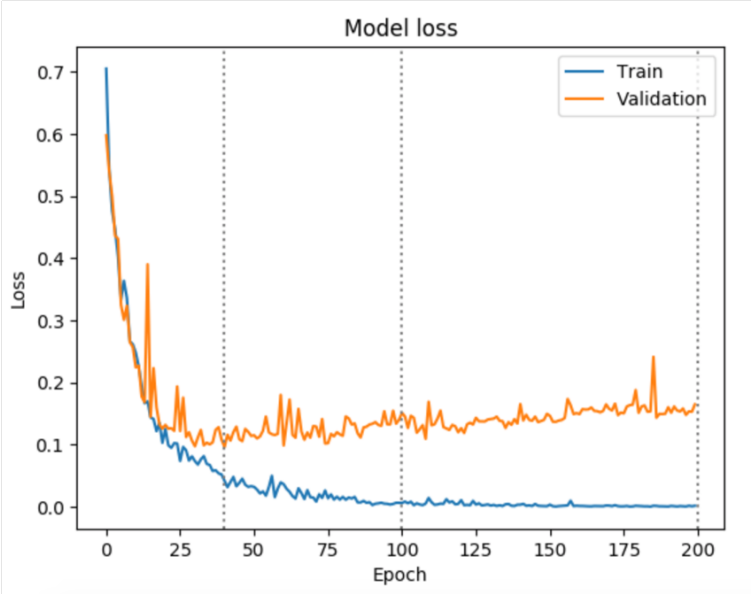

Figure 13: Model Loss During Training History

6 Conclusion

This project presented the challenge of creating an algorithm that can automatically detect solar panels in aerial imagery data using a training set of 1,500 101x101 pixel RGB colored images, 30% of which contained solar panels. Our steps included downloading and exploring the data, determining which features should be extracted each image, choosing a classification technique to apply to these features, evaluating the performance of our model so that we can improve upon it, and, finally, estimating test image labels to submit as our solution. Through this process, our best solution leveraged transfer learning on pre-trained VGG-16 specifications using augmented data, which achieved an AUC value of 0.998 on our private testing board.

This project had specific challenges based on the data provided, such as the variation in color for solar panels in different images, the number of solar panels changing by image, and having a limited training set to train our model on. We also had the added challenge of the long run-time given our model's complexity. We were able to solve these issues by choosing to use CNN, which was able to deal with the majority of data-related challenges, and solved the added challenge of run-time by porting the code over to Colaboratory Notebook on GPU which provided a significant speed boost. We hope this project offers a platform for decision makers to base decisions on regarding solar panel installation and upgrades, as well as gives future researchers a model to which they can further improve an algorithm. We suggest experimenting with more CNN algorithms to improve upon our solution, as well as suggesting that future solutions use, if possible, bigger or higher resolution datasets to train their model on.

7 Roles

Team H, The Knitting Club, consists of members Han Yan, Kate Coulter, Xuan Yu, Avani Gupta, and Danny Witt.

- Han Yan worked on and submitted code to Kaggle that improved upon our initial code's framework and also made it possible to quickly iterate our code by porting it over to GPU. She also translated the code into a readable workflow and wrote the methods and results sections of this report. Han also edited our code for readability.

- Kate Coulter managed the report writing, wrote drafts for each section and also edited this report for its final submission. Kate helped pre-process the data and also helped choose parameters used in our final submission code. Kate researched similar projects in image classification using machine learning techniques to understand what mistakes and challenges exist in the problem, thus ensuring our code accounted for special cases in each class so that the code was complex enough to correctly label these cases, but also not overly specific so that it didn't overfit to our datasets.

- Xuan Yu created the workflow for our initial code, which we used as the template for all improved submissions to Kaggle. Xuan chose to use a two-layer method, epochs =

200, and other final parameters that we discuss throughout this report and also directly contributed to our model's performance. He submitted and edited what would be our final submission for the competition.

- Avani Gupta provided background research support on the topic and models that have been used. Avani edited our final code to ensure its cleanliness and readability, and validated results of our model by creating the figures used in the results section of this report. She tested our final choice compared to other models we had to help decide which submission we should use as our final submission.

- Danny Witt contributed with background research support which largely framed the background section of this report. He also suggested different techniques that we could use in our own model that had been used in other high dimensional image data, largely relying on research from the health sector, and translating how those successful imaging techniques could be used in identifying solar array(s).

References

[1] Altman, N.S. An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician*, Volume 46, Issue 3, 175-185. Available: `https://www.tandfonline.com/doi/abs/10.1080/00031305.1992.10475879`. (27 Feb 2012).

[2] Bradbury, K., Saboo, R., Johnson, T.L., Malof, J.M., Devarajan, A., Zhang, Wu., Collins, L.M., Newell, R.G. Distributed solar photovoltaic array location and extent dataset for remote sensing object identification. *Scientific Data*, Volume 3, Article 160106. DOI: https://doi.org/10.1038/sdata.2016.106. (2016 Dec 6).

[3] Bradbury, K. Solar PV in Aerial Imagery: Identify solar arrays in aerial imagery data. (IDS705: Sp. 2019). *Kaggle*. Available: `https://www.kaggle.com/c/solar-pv-in-aerial-imagery/`. (2019 Feb 28).

[4] Chollet, F., et al. Keras. Available: `https://keras.io`. (2015).

[5] Esteva, A., Kuprel, B., Novoa, R., Ko, J., Swetter, S., Blau, H., Thrun, S. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, Volume 542, 115-118. DOI: https://doi.org/10.1038/nature21056. (2017 Jan 25).

[6] Jean, N., Burke, M., Xie, M., Davis, W.M., Lobell, D., Ermon, S. Combining satellite imagery and machine learning to predict poverty. *Science*, Volume 353, Issue 6301, 790-794. DOI: 10.1126/science.aaf7894. (2016 Aug 19).

[7] Magoulas G.D. Prentza A. "Machine Learning in Medical Applications". Lecture Notes in Computer Science. *Machine Learning and Its Applications*. Available: `https://doi.org/10.1007/3-540-44673-7_19`. (2001 Sep 20).

[8] Malof, J.M., Bradbury, K., Collins, L.M., Newell, R.G. Automatic detection of solarphotovoltaic arrays in high resolution aerial imagery. *Applied Energy*, Volume 183, 229-240. DOI: https://doi.org/10.1016/j.apenergy.2016.08.191. (2016 Dec 1).

[9] Mundy, J., Chellappa, R. Hoogs, A. "Session 2: Machine Learning from Image, Video, and Map Data." Challenges in Machine Generation of Analytic Products from Multi-Source Data: Proceedings of a Workshop. *National Academies of Sciences, Engineering, and Medicine*. Available: `https://doi.org/10.17226/24900`. (2017).

[10] New dataset developed at Duke will benefit solar energy growth. *The Energy Initiative at Duke University*. Available: `https://energy.duke.edu/news/making-solar-count-new-dataset-developed-duke-will-benefit-solar-energy-growth`. (2016 Dec 7).

[11] Pedregosa et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, Volume 12. Available: `https://arxiv.org/abs/1201.0490`. (2012 Jan 2).

[12] Razzak, M., Naz, S., Zaib, A. Deep Learning for Medical Image Processing: Overview, Challenges and Future. Available: `https://arxiv.org/pdf/1704.06825.pdf`. (2017 Apr 22).

[13] Ruder, S. An overview of gradient descent optimization algorithms. *Sebastian Ruder, Optimization*. Available: `https://arxiv.org/abs/1609.04747`. (2016 Jan 19).

[14] Simard, P.Y., Steinkraus, D., Platt, J.C. Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis. *International Conference on Document Analysis and Recognition*. Available: `http://cognitivemedium.com/assets/rmnist/Simard.pdf`. (2003).

[15] Simonyan, K. Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *International Conference on Learning Representations*. Available: `https://arxiv.org/pdf/1409.1556.pdf`. (2015).

[16] Uz, F.B., Salvaris, M., Grecoe, D. GPUs vs CPUs for deployment of deep learning models. *Microsoft Azure, Data Science*. Available: `https://azure.microsoft.com/en-us/blog/gpus-vs-cpus-for-deployment-of-deep-learning-models/`. (2018 Sep 11).

[17] Yu, J., Wang, Z., Majumdar, A., Rajagopal, R. DeepSolar: A Machine Learning Framework to Efficiently Construct a Solar Deployment Database in the United States. *Joule*, Volume 2, Issue 12, 2605-2617. DOI: https://doi.org/10.1016/j.joule.2018.11.021. (2018 Dec 19).

A Appendix

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
sequential_1 (Sequential)	(None, 1)	6423041

Total params: 21,137,729
Trainable params: 21,137,729
Non-trainable params: 0

Figure 14: Model Summary