Child Mortality Prediction in Africa using Low-Resolution Public Satellite Imagery

Project Category: Computer Vision
Project Mentor: Kevin Li

Matthew Kolodner
Department of Computer Science
Stanford University
mkolod@stanford.edu

Jack Michaels
Department of Computer Science
Stanford University
jackfm@stanford.edu

1 Introduction

In 2015, the United Nations released an agenda for reaching 17 sustainable development goals (SDGs) by 2030. Even though nearly half of that time has passed, progress towards these goals has been obstructed by a lack of widespread information surrounding the environmental and socioeconomic factors which plague developing countries. Current data is primarily collected through ground surveys; a particularly limited form of data collection given their diminished temporal and spatial capabilities.

To compensate for this, and aid the UN in reaching their SGDs by 2030, our algorithm takes as input a public, low-resolution image to fill the temporal and spatial data gap. We then use linear classification, KNN, and deep learning models to output predicted child mortality rates, of which get classified into a variable number of buckets representing different levels of socioeconomic status. The said dataset contributes towards SGD3: Good Health and Well-Being, which we believe will provide broader information on poverty and human livelihood to policy makers to better inform their decisions. Additionally, we hope that the techniques used in this paper will inspire or be useful in the creation of much-needed datasets for other SGDs to achieve the looming 2030 deadline.

2 Related Work

Extensive data accumulation is difficult in certain developing countries. Many machine learning papers therefore mirror our usage of satellite imagery with a convoluted neural network to solve SGDs [1] [2] [3]. While our project used a prepared dataset from SustainBench, a 2016 paper had to use noisy proxies for training data, correcting the noise through pre-trained CNN models (ImageNet) and regularization [3]. Contrast this to a trivial method, such as the 2019 paper which by hand classified about 500 images and then rotated them by 90º, 180º, and 270º to generate data for their CNN algorithm [2]. Many state-of-the-art algorithms involve algorithmic layering; a paper on HFI estimation created separate CNNs for different region types culminating in layer-wise relevance propagation [1] or a slavery classifier which inputted their two-stage object Faster-RCNN into a GoogLeNet CNN classifier, the effect of which reduced commission errors [2]. In comparison to our project, the underlying CNN and data classification remains ubiquitous throughout, though we lack this multilayering of pre-processing and post-processing algorithms which are present in the papers.

While deep learning CNN algorithms are the traditional option for image interpretation, they require large datasets. As an alternative, versions of Artificial Neural Networks have been used to interpret skin conditions [4] and astronomy data [5]. The 2016 dermatology paper compared between kNN, decision trees, and ANN, with ANN performing marginally better [4], reflecting similarities between their findings and ours even through the classification differences. While ANN achieves high accuracy in both papers, higher resolution imagery may make ANN algorithms impossible due to unfeasible training time/number of parameters and a CNN algorithm like ours should be used [5].

3 Dataset and Features

Our test and training dataset comes from SustainBench and is composed of satellite imagery paired with child mortality labelling via DHS surveys [6]. Daytime satellite imagery was provided via the Landsat 8 satellite which contains a native 30m/pixel resolution and an image size of 255x255 pixels. Landsat 8 collects a broad range of imagery data via seven bands, going in order from blue, green, red, shortwave infrared 1, shortwave infrared 2, thermal, to near infrared. When approaching our modeling task, we only used the first three bands, RGB. From the selected countries Albania, Armenia, and Angola we have 1862 satellite images, of which 300 images were set aside and split evenly for the validation and test set with the rest of the images being placed in the training set.

While the majority of image preprocessing was performed by SustainBench benchmark, re-labelling was done to better suit the needs of our algorithm. Such relabelling included surjectively mapping SustainBench labels (representing the number of children that died in a given region) to a number of quantiles dependent on the desired resolution of our labels. More specifically, the distribution of mortality rates ranged between 0 to 80 per image, prompting experimentation with mapping the labels to 4 quantile classes (intervals of 20) and 8 quantile classes (intervals of 10). We evaluate the performance of both these low and high resolution labels. Since we want to determine areas at risk of high child mortality, creating this mapping was critical since predicting a quantile for this risk is just as useful as a regression task for mortality.

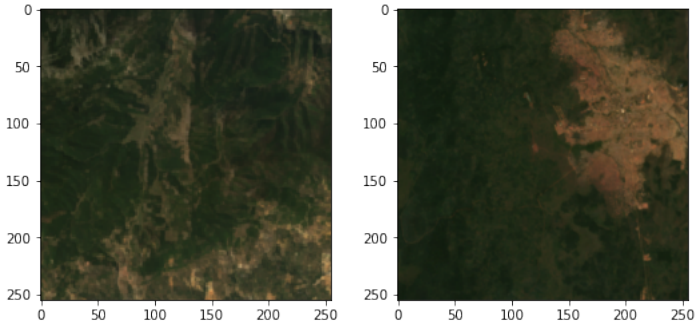

Figure 1: Left: Image w/ label 11 deaths per 1000; Right: Image w/ label 27 deaths per 1000

4 Methods

For our project, we have three primary models we used for learning. More specifically, we used a linear classifier baseline model, a K-Nearest-Neighbors (KNN) model, and a ResNet18 model in order to predict child mortality into one of the quantified buckets. We also deployed saliency mapping to further interpret each of the models.

4.1 Linear Classifier Baseline

Multiclass classification involves the classification of the response variable y into any one of i distinct values. Classification on some data x involves using the hypothesis function described in eq. (1). Applying this sigmoid function will specifically place the predicted results within a range of 0 to 1. Furthermore, since we are in a multi-class setting, for each training example j we measure the loss using eq. (1) over i classes where $\hat{y^{(j)}}_i$ represents the predicted probability (result of the hypothesis function) of class i being the correct class and $y_i^{(j)}$ represents 1 or 0 depending on whether class i is the correct prediction for the training example j.

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}} \quad L(\theta) = -\sum_{i=1}^{n} y_i^{(j)} \log(\hat{y^{(j)}}_i) \tag{1}$$

4.2 K-Nearest Neighbor

The K-nearest neighbors algorithm seeks to classify a given example by calculating the k-closest training examples to a given test example and assigning that test example a class based on the most common class among the k neighbors. For this algorithm we observe fast train time since all we have to do is provide data to the model. However, we observe longer test time since the distance between each test example and every train example needs to be computed. To measure the distance between two given points, we use the L_2 distance, defined in eq. (2) where p and q are both datapoints. We trained this model using [7].

$$L2 = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{2}$$

4.3 ResNet18

A ResNet18 is an 18-layer model that utilizes 1) convolutional neural networks and 2) skip connections. Convolutional neural networks are a class of machine learning models that perform well on image tasks. By convolving eq. (3) $n \times n$ filters g containing weights with input images f, these models perform well at extracting spatial dependencies within the data. Furthermore, the number of filters convolved with each image can transform the number of channels within the intermediate layers of the model, and the sizes of the filter and choice of padding/stride can transform the height and width of the intermediate layers.

ResNet18 is a unique model in the sense that it was the first model to introduce the concept of skip connections. To combat the saturation problem of very deep models, where the performance starts to decrease with the size of the model, the ResNet18 model adds the outputs of one layer to the inputs of later layers in the model, effectively allowing the model to learn to skip any layers in between. This allows larger, more deep models to be effective. Our ResNet18 model was training using the same loss function in eq. (1) using [8]

$$f[x, y] * g[x, y] = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f[i, j] \times g[x - i, y - j] \tag{3}$$

4.4 Saliency Mapping

Finally, we deploy saliency mapping to interpret the results of the model. Saliency is a visualization technique that can be used to determine, given an image, which parts of the image are contributing towards the most likely prediction. This is accomplished by running a forward pass of a trained model on an example image, getting the most probable class from the image, and then running backpropagation on this most probable output to compute the gradients of the input image. Since there are multiple input channels, the maximum is taken among all the channels for the visualization, where brighter pixels indicate more importance in the overall class prediction.

5 Experiments/Results/Discussion

5.1 Evaluation Metrics

Considering that we are in the mulit-class classification domain, the primary metrics we used for evaluation are accuracy, macro-precision, macro-recall, and macro-f1 score. Accuracy is a measure of the percentage of correct classifications. Precision is computed by calculating the percentage of positives that were correct positives for each class. Recall is the percentage of total true positives actually found by the model. Finally, the F-1 score is a harmonic mean of the precision and recall. We calculate the macro-averaged values for each of these three metrics, meaning we compute the scores for each of the classes and average over all of them.

Model	# of Quantiles	Accuracy ↑	Precision ↑	Recall ↑	F1-Score ↑
Linear Classifier	4	0.767	0.37	0.395	0.382
KNN	4	0.600	0.316	0.343	0.307
ResNet18	4	**0.813**	0.402	0.439	0.414
Linear Classifier	8	0.427	0.273	0.250	0.252
KNN	8	0.440	0.240	0.375	0.281
ResNet18	8	0.600	**0.498**	**0.574**	**0.477**

Table 1: Quantile and Model Test Performances

5.2 Model Training

For each model, we fine-tuned the parameters for optimal performance. For our KNN-model, we found that the optimal n_neighbors value was 16 for both quantile experiments. We also found that 0.001 was the best learning rate across all of the deep models, achieving the highest accuracy. We selected a batch size of 16 across all models, achieving the best trade-off between performance and runtime. We also loaded the ResNet18 model using pretraining from the ImageNet dataset, yielding better results than without this pretraining. In order to prevent overfitting, we implemented L2 Regularization with a regularization strength of 0.001 in order to penalize very large weights.

5.3 Model Results

Looking at the results of the model in table 1, we see that generally the ResNet18 model performs the best, outperforming the corresponding linear classifier and KNN models in every metric. This result is generally expected, as the ResNet18 model is the best performing on the ImageNet benchmark out of the three while also having more parameters and spatial understanding. While we do observe a relatively high accuracy for the 4-quantile ResNet18 model, we see that this performance quickly falls when looking at the remaining metrics. In comparison, we see the gap in performance between accuracy and the remaining metrics in the 8-quantile ResNet model to be much smaller. Surprisingly, this leads us to see that the higher-resolution labels have better performance than the more generalized 4-quantile model. One potential reason for this could be that these three metrics are averaged over each class, meaning that the 4-quantile model only performs very strongly on one class, with its performance on the other classes being significantly worse. Thus, the high accuracy for the 4-quantile model is likely a result of significantly more 0 labels in the test dataset than the other classes. In comparison, through its higher label resolution, the 8-quantile model likely is less prone to these inbalanced labels in the test set.

5.4 Confusion Matrices

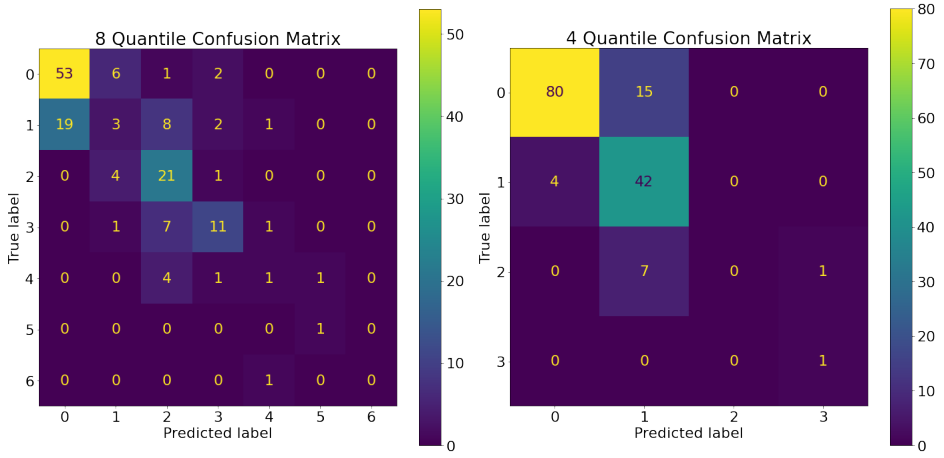

Figure 2: Confusion Matrices for ResNet18 Model

fig. 2 resembles the prior conclusions drawn. We see that the 8-Quantile model struggles the most with overpredicting on buckets 0 and 2 while the 4-quantile model has difficulty with false positives along the 1 bucket. This makes sense given how we quantized the labels in the dataset. Unlike the 8-Quantile model, we see that 4-quantile model doesn't have an issue with overpredicting on the 0 bucket, but rather struggles in underpredicting it. As discussed earlier, these difficulties with certain quantiles is likely a result of inbalance within the dataset.

5.5 Saliency Mapping

Saliency mapping is largely useful for providing us with a means for interpreting what aspects of an image are important. In section 5.5, we see two example saliency maps for the same ground truth point. On the left saliency map, we see that the 4-quantile model appears to identify some dark characteristic of the input image and predict an incorrect quantile of 3 instead of 2. On the right saliency map, we see a more noisy, uniform saliency mapping throughout the image, yet we see that the 8-quantile model is able to correctly label the 5 quantile for the image. This relatively noisy saliency map poses some concern for what the model might actually be learning from the images. The differences in saliency mappings between the two differently quantized models on the same test example highlight both the difficulty of the problem of connecting satellite image with a metric such as child mortality and the importance of the number of quantiles hyperparameter relative to the performance and behavior of the model. This distinction in behavior between the differently-quantiled models is observed across all the test-images for the test dataset.

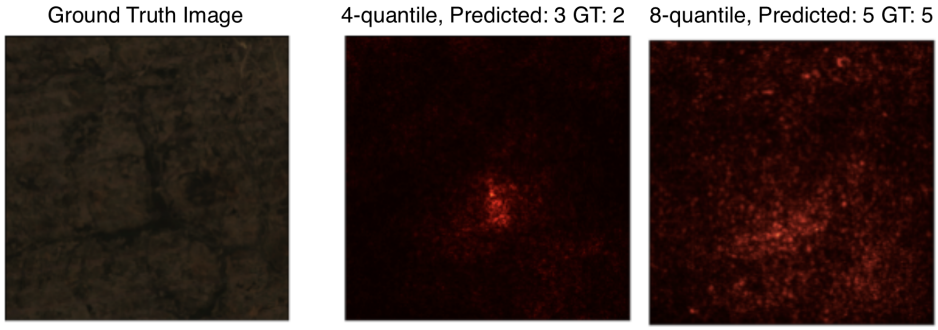

Figure 3: Saliency Results for ResNet18 Model

6 Conclusion/Future Work

Overall, from our experiments, we found that the ResNet18 model performed best. More specifically, considering the overall improvement in $\frac{3}{4}$ of the metrics that the 8-quantile ResNet18 had over the 4-quantile Resnet18 and the higher label resolution, the 8-quantile Resnet18 model overall had the best performance. However, looking at the various metrics, confusion matrices, and saliency maps we see that this model still has lots of room for improvement. More specifically, we see that the model has more difficulty predicting less common labels, as the dataset is inbalanced. Nevertheless, we believe our work acts as a strong initial step towards poverty mapping through low-resolution satellite imagery. One future step for this project given more time and compute power is to run more experiments with deeper and more complex models (i.e. ResNet101, Vision Transformers) and also run experiments with a different model settings, such as with a different combination of input channel bands beyond just RGB. Additionally, we would also like to explore upsampling and data augmentation to try to address the issue with the inbalanced dataset.

7 Contributions

Matthew Kolodner - Training and fine-tuning models, paper writeup, experimental approach
Jack Michaels - Data pre-processing, paper writeup, experimental approach

References

[1] Neil H Carter Patricia W Keys, Elizabeth A Barnes. A machine-learning approach to human footprint index estimation with applications to sustainable development. volume 16. Environmental Research, 2021.

[2] Doreen S. Boyd Xiaodong Li Jessica Wardlaw Giles M. Foody, Feng Ling. Earth observation and machine learning to meet sustainable development goal 8.7: Mapping sites associated with slavery from space. volume 11, no. 3, page 266. Remote Sensing, 2019.

[3] Michael Xie W. Matthew Davis David B. Lobell Stefano Ermon Neal Jean, Marshall Burke. Combining satellite imagery and machine learning to predict poverty. volume 353, pages 790–794. Science, 2016.

[4] Varun Saboo Vinayshekhar Bannihatti Kumar, Sujay S Kumar. Dermatological disease detection using image processing and machine learning. In *Third International Conference on Artificial Intelligence and Pattern Recognition*, pages 1–6. IEEE, 2016.

[5] Robert J. Brunner Nicholas M. Ball. Data mining and machine learning in astronomy. volume 19, pages 1049–1106. World Scientific Publishing Company, 2009.

[6] Christopher Yeh. Dhs survey-based datasets. SustainBench.

[7] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.