Poverty Estimation Using Satellite Images: A Case Study in Sub-Saharan Africa

Christine Cepelak
christinecepelak@gmail.com

Janine De Vera
janinepdevera@gmail.com

Johannes Halkenhäußer
halkenjo@gmail.com

Abstract

Poverty estimation has crucial and wide-ranging applications, from policymaking to humanitarian interventions. Without accurate wealth statistics, countries lack the ability to identify vulnerable populations and fairly allocate their funds. Despite its importance, measuring and monitoring poverty is both a conceptually and operationally challenging undertaking. To contribute to the growing research body on bridging gaps in poverty statistics, we estimated several machine learning models that predict poverty through the use of daylight satellite images. Random Forest Regression, Support Vector Regression, Convolutional Neural Networks, and Transfer Learning were trained, validated, and tested using a dataset comprising four countries in Sub-Saharan Africa. Our findings indicate that Random Forest and Support Vector Regression perform best, with test R^2 of 61.8% and 60.8%, respectively. [1]

1. Introduction

Poverty is a multifaceted problem manifested by broad conditions such as malnutrition, homelessness, lack of access to clean water, and low educational achievement. It continues to be one the world's most pressing issues and it has only been exacerbated by the economic effects of the COVID-19 pandemic. The United Nations' Sustainable Development Agenda for 2015-2030 identifies poverty eradication as one of its top priorities – evidence that governments across the globe are committed to alleviating poverty.

It follows that poverty statistics are among the most important and most widely used data in the economic and policy research sphere. Practical and ground-level applications of poverty data include: the identification of vulnerable populations for humanitarian interventions, fair allocation of aid, and valuable direction for continued research and policy development. Poverty measurement is commonly the responsibility of national statistics offices, but the capacity to produce accurate and frequent information varies widely depending on the country. Further, economies that are able to produce these statistics often fall short of capturing the problem's granularity. Poverty estimates derived from on-the-ground surveys or censuses are typically quantified based on broad geographic clusters (e.g. in aggregated regional or provincial level), which presents problems of misrepresentation and unreliability. Without accurate, detailed, and up-to-date poverty statistics, countries lack information to meaningfully support their own communities, and development organizations risk unfair allocation of aid and other resources.

As the world wrestles with the impacts of Artificial Intelligence advancements and hyper connectivity on social media platforms, a basic foundation of governance is missing in many countries: accurate census data. Fortunately, recent years have brought major advancements in computer vision and machine learning techniques applied to geospatial research. Several studies have tackled the improvement of poverty statistics [2, 5, 7, 8] using satellite imagery. Our study contributes to existing literature by developing machine learning algorithms that are applied to several countries. The ultimate objective is to improve the accuracy of existing poverty estimates, and eventually produce up-to-date statistics for some of the most impoverished countries in the world where such information is unavailable.

2. Related Work

An internet search of the terms: *Satellite Images* and *Poverty* renders a multitude of studies, articles, and repositories. In 2015, the United Nations initiated a global poverty campaign which inspired a team of Stanford social and computer scientists to tackle poverty statistics gaps in African countries. They began with satellite images taken at night – attempting to use lights on the dark images to demonstrate the distribution of rich/poor populations.

However, they quickly realized that nighttime images only allowed a certain level of detail: they could not differentiate ”absolute poverty” (individuals living on less than 1.90 USD a day) from modest economic development. The Stanford team continued their research: now including daytime images. These provided more context: how far away

[1] GitHub respository: https://github.com/ccepelak/ML-SS22

Figure 1. Nighttime Satellite Imagery

was a home from the closest market, water source, etc. Daytime images were 81% more accurate at predicting poverty in places under the absolute poverty line (and 99 % more accurate in areas where incomes are less than half that.)

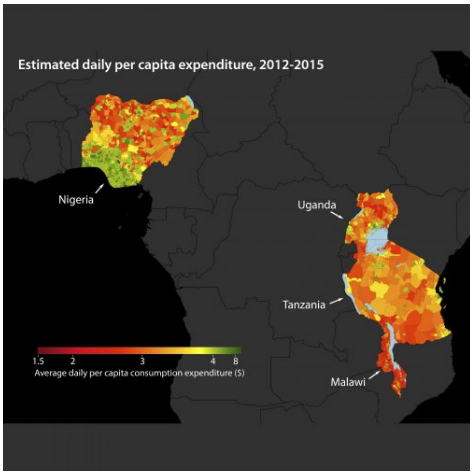
Estimated daily per capita expenditure, 2012-2015
Nigeria
Uganda
Tanzania
Malawi
1.5
2
3
4
8
Average daily per capita consumption expenditure ($)

Figure 2. Predictions from Standford's Study Combining Day Nighttime Satellite Images

This initial study focused on: Nigeria, Tanzania, Uganda, Malawi, Rwanda, and following projects followed suit, leveraging the growing resource of satellite images of African countries. Since then, there have been many contributions to this cause, with 2020 rendering two studies focused on poverty prediction in the Philippines and Rwanda, and another more recent study focused on applying CNN to 44,193 cities spanning Africa, South America, Asia, Europe, and the Caribbean.

3. Proposed Method

The study takes inspiration from previous applications of machine learning in poverty estimation, and implements a full machine learning workflow to measure wealth indices in four Sub-Saharan African countries: Ethiopia, Malawi, Mali, and Nigeria. The methodology is divided into two main segments - (i) data preparation, and (ii) model building.

3.1. Data Preparation

The main data set used in this study was constructed by merging satellite image data and geospatial wealth indices. Prior to combining the data sets, significant pre-processing had to be applied to in order to ensure optimal model performance.

Satellite data often comes in a set of thousands of images, with each image as a tensor of square pixels, and each pixel a three or four-dimensional array. The images for the four countries are composed of 256 x 256 pixels with RGBA (Red, Green, Blue, and Alpha for transparency) features. Each image therefore has 262,144 (256 x 256 x 4) different attributes. The first data pre-processing step is to resize the images from 256 x 256 pixels to 128 x 128 pixels, and drop the transparency trait alpha due to lack of variation. This reduces the features of the data by 62.5%, while still retaining a majority of the information contained in the images.

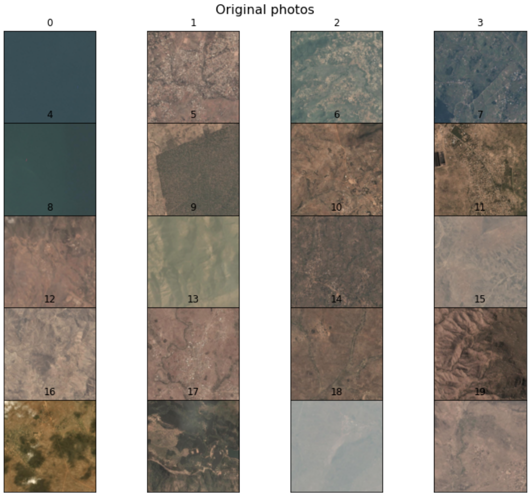

Figure 3. Sample Satellite Images of Malawi

Next, a Principal Component Analysis (PCA) trained on a random sub-sample of 100 images is applied to reduce dimensionality of the data into 30 components. This newly PCA-reduced data set is sorted such that only unique images remain. Using the indices of unique images, a subset of the resized satellite data is filtered to remove duplicated images. This is then merged with geotagged wealth indices per country using geolocations of image and survey results. Observations from all four countries are combined to generate the final data set, which is then divided into training (85%) and test (15%) data sets.

For certain models, another PCA is implemented to project the final data set into a lower dimensional space while at the same time maximizing the variance of the input data [10]. This PCA is separately trained from the first PCA that was used for image filtering. For the deep learning models, the training data is further split (with a 90:10 ratio) between training and validation sets, while the test set is left as is.

A schematic of the data pre-processing steps is presented in Figure 4 following.

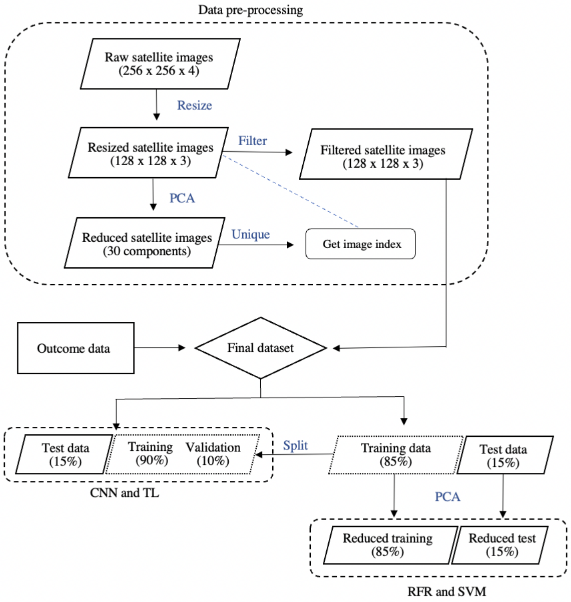

Figure 4. Data pre-processing flow chart

3.2. Model Building

For the model building experiments, five different models are compared. A linear regression model serves as the baseline against which performance of machine learning methods are assessed. Four machine learning regression algorithms were trained, validated, and tested: Random Forest (RF), Support Vector Machine (SVM), Convolutional Neural Network (CNN), and Transfer Learning (TL).

A. *Linear Regression:* A linear regression model minimises the Mean Squared Error (MSE) by finding an optimal combination of coefficients that are used to build a linear combination of input features. The linear regression is not adjusted in any way, giving an indication of how the most simple model fares.

B. *Random Forest Regression:* 51 Random Forest Regressors (RFR) with 4 folds each (204 fits) are trained using the final data set. A RFR grows individual Regression Trees given some parameter inputs (described in 4.4.C) that minimises the MSE using binary recursion and splitting the data along multiple decision nodes. The individual trees are ”pruned” (i.e. optimised in size to prevent overfitting) and combined into forests. Generally, larger forests allow for more robust estimators. By layering a 4-fold cross validation on top of each parameter combination, overfitting is again tested against [8] [10].

C. *Support Vector Regression:* 75 support vector regressors with 4 folds each (300 fits) were trained using the final data set. Support Vector Regression fits data into a hyperplane decision boundary in order to predict continuous outcomes. Parameter inputs (described in 4.4.D) are chosen with objective of fitting as many observations as possible within the hyperplane. Similar to the random forest regression, grid searching with a 4-fold cross validation is implemented.

D. *Convolutional Neural Network:* A Convolutional Neural Network (CNN) is commonly used for contextual data such as images. Unlike regular neural networks, CNNs use a convolutional and a pooling layer to extract information from a subspace of the image thus preserving some of the underlying structure of the image [11]. We construct a CNN from scratch with 3 convolutional layers using a leaky reLU activation function with 32, 64, and 128 filters respectively. After the convolutional layers the output is flattened and passed through a fully connected 32-node layer with linear activation function, to then finally predict the wealth index using a one-node linear output layer.

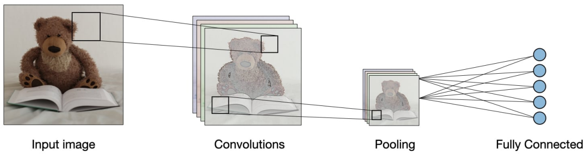

Figure 5. The essential architecture of a convolutional neural network [1].

E. *Transfer Learning:* Transfer Learning (TL) takes weights learned from training one model, and leverages results for a new, similar problem. We use the ResNet50V2 model that has been trained on ImageNet [6] and whose predecessor has previously performed well on satellite images [11]. After removing the original model's top layer, we again add a 32-node linear layer and a final linear output node.

4. Experiments

4.1. Data

A. *Ground truth data (wealth index)*: The main outcome variable for the study was collected from the Demographic and Health Surveys (DHS) Program. The DHS is regularly conducted in over 90 countries, with the purpose of gathering representative data on health and economic well-being of different populations.[3, 8, 10, 11]

The wealth index as reported in the DHS is calculated as the first principal component of several household attributes which are not direct measures of economic status. Examples of variables that are accounted for in the index are sources of drinking water, types of bathroom facility, ownership of various household appliances (e.g. refrigerator, television, telephone, etc.), and materials used for building houses. The indices are calculated based on pre-defined clusters that comprise several households. Cluster centroids are reported as the mean latitude and longitude of the households that belong to one cluster group.

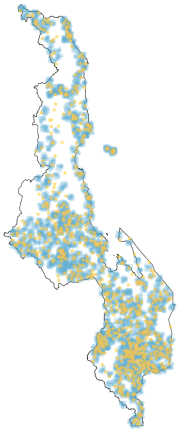

Figure 6. Overlap of satellite images and DHS wealth indices for Malawi. The blue areas represent centroids of satellite images while the yellow areas represent the cluster centroids of DHS indices.

B. *Satellite images*: The satellite images data set is from Planet Developer Resource Center's API, accessed through Kaggle [9]. Each image is made up of 256 x 256 pixels, with RGBA values. These images are matched to wealth indices based on the location of cluster centroids as defined in 4.1.A.

The spatial matching of wealth indices and images yielded both n:1 and 1:n correspondence. That is, there are cases where (i) several wealth indices are matched to one image and (ii) several images are matched to one wealth index. The full dataset contained a total of 33,390 matches.

Table 1 describes the main datasets used in this study. Information for all four countries used in the study.

Demograpic and Health Survey				
	Ethiopia	*Malawi*	*Mali*	*Nigeria*
Year collected	2019	2016	2018	2018
No. of clusters	305	850	379	1,400
No. of households	8,663	26,361	9,510	40,427
Daytime Satellite Imagery				
	Ethiopia	*Malawi*	*Mali*	*Nigeria*
Year collected	2015	2015	2015	2015
No. of images	8,587	12,700	12,800	11,535

Table 1. Summary of wealth index and satellite image datasets.

4.2. Software and Hardware

The machine learning workflow, including data preprocessing and all model building experiments, was executed in Python, using Jupyter Notebooks, Spyder, and Visual Studio Code as the integrated development environment.

The data pipeline was set up using local computing devices, with 2.6 GHz 6-Core Intel i7 and 3.2 GHz 8-core M1 processors. The baseline Linear Regression, Random Forest, and Support Vector Machine models were implemented on the same machines, while training for the Convolutional Neural Network and Transfer Learning models were conducted through Hertie GPU server. Specifications of the server are as follows:

GPU: 4 x NVIDIA A100 40GB HBM2
CPU: AMD EPYC 7742 (64 cores, Rome, 2.25 GHz)
RAM: 512 GB (8 x 64GB) ECC DDR4 3200 Mhz
SSD: 2,5" 3,8 TB U.2 NVMe TLC
Network: 1 x 10 GbE SFP+, 2x 1 GbE RJ45, 1 x IPMI Lan
OS: Ubuntu 20.04 LTS + Tensorflow, PyTorch and MxNet with Docker Container for Versions Management

4.3. Evaluation Method

As in related studies, the R^2 is used an evaluation metric to compare the performance of different models. The R^2

measures the variance in the dependent variable (wealth index) that is explained by the independent variables (satellite images features).

4.4. Experimental Details

PCA for dimensionality reduction

Prior to all model building experiments, a PCA was first applied to the final merged dataset containing all countries (see Figure 4). This reduced the size of the images down to 25 components. Dimensionality reduction was a necessary step to optimize model performance given the computational constraints of local machines. A random sub-sample of 10,000 images was used to train the PCA. Note that this is different from the first PCA in the data pre-processing step, which was trained specifically for removing duplicate images. After this step, all experiments (except CNN and Transfer Learning) are implemented using the same PCA-reduced training and test data set.

Figure 7 shows a representation of the 25 PCA components. It highlights how the models pick up patterns counter-intuitive to the human eye. Instead of assemblies of villages, bodies of water, etc., color patterns are identified by the model.

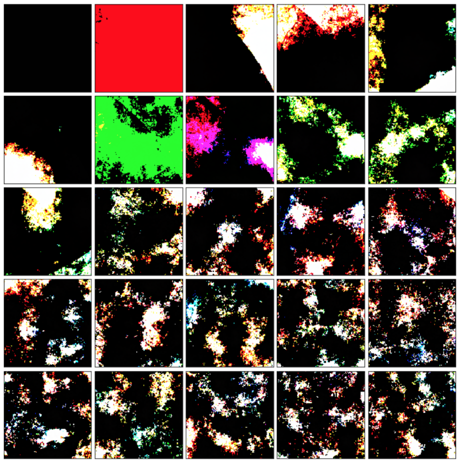

Figure 7. The representation of each of the 25 components produced by the PCA. They explain roughly 90% of variance within predicted value.

Training, Validation, Testing

This subsection details the experiments conducted for each model enumerated in section 3.2.

A. *Linear Regression*: A simple linear regression was estimated using the PCA-reduced training and test set for all countries. No additional parameters were included. This model serves as the baseline against which the succeeding models are initially compared.

B. *Random Forest Regression*: To account for the non-linearity of the data, a non-parametric RFR was used. All forests use bootstrapped samples, with the MSE as a splitting criterion. To tune the model and test multiple hyper-parameters, cross validation with a 4-fold grid search was implemented over all different combinations of the following parameters:

max_depth: The maximum tree depth determines how many levels the tree is allowed to have. The deeper the tree is allowed to grow, the larger the risk overfitting, as a deeper tree also leads to a higher segmentation of the outputs.

Range: [1, 2, 3, 4, 5, 10, 16, 18, 20, 22, 24, 26, 30, 40, 50, 100, 200]

max_features: During the training, the model is allowed to account for a varying number of features. This furthers de-correlation of the trees and thus allows building of a more robust forest. During the grid-search, allowing the full number of features, the square root of the number of features, and the log_2 of the number of features were passed as model parameters. With 25 samples, the $log_2(25) = 4.64$ is roughly $\sqrt{25} = 5$, so no difference can be expected. However, the distinction is kept in for future use should a higher number of PCA components be employed.

Range: $\left[\text{n\_features}, \log_2(\text{n\_features}), \sqrt{n\_features}\,\right]$

C. *Support Vector Regression*: The key characteristic of SVR is the ability to provide flexibility to determine how much error is acceptable in a model. It finds an appropriate hyperplane to fit into high dimensional data, with the objective of minimizing coefficient vectors. It does this by choosing the following parameters:

maximum_error (ϵ): A margin against which the absolute error of the model is compared. ϵ can be tuned is order to obtain the desired level of model accuracy.

Range: [0.1, 0.3, 0.5, 0.7, 0.9]

slack (C): C measures the model's tolerance of deviations from the margin ϵ. As C increases, the greater tolerance the model has for points that lie outside of the margin.

Range: [1, 10,100,1000, 10000]

kernel: The kernel or mathematical function used to transform input data can also be specified and tuned. This is commonly applied when finding a hyperplane in a higher dimensional space.

Range: ['poly', 'rbf', 'sigmoid']

D. *CNN*: There is no true best way to build a neural network. We chose a medium-size structure with three convolutional layers and 32, 64, and 128 filters respectively. The number of layers, their activation functions and other hyper-parameters can be adjusted to build a deeper network that allows more complexity. However, overfitting can be an issue here, which is why a validation set was used during training. After some experimenting with the batch-size, it was set to 75 which allowed enough samples to be seen simultaneously while not overflowing the memory and experimented with a learning rate of 0.001 and 0.0001. We report the learning rate of 0.0001 which prevented sudden spikes in loss during training. 50 epochs (8.23s p.epoch on the NVIDIA server) allowed for the network to learn well without over-fitting.

E. *Transfer Learning*: To allow for comparison with the CNN, the layers following the convolutional layers of ResNet, batch size, and epochs (50 epochs at 17.49s p.epoch on the NVIDIA server) are the same as the CNN. The learning rate was kept at 0.001. A recommended workflow is to first freeze the original model, train on the outputs, then unfreeze the entire model and train with a small learning rate for a short time [4]. Following this, we unfroze the ResNet's 50 layers after initial training and continued with a learning rate of $1^{-}5$ for 10 more epochs.

4.5. Results

A. *Linear Regression*: The baseline linear regression performed poorly in both data sets: resulting in an R^2 of 4.26% on the complete training set representing all countries. Its performance is even worse on the test set, evidence of its lack of robustness to new data.

B. *Random Forest Regression*: The random forest regression shows a significantly higher performance compared to the baseline. After training and hyper-parameter tuning, the best model yielded an training R^2 of 61.4% and test R^2 of 61.8% - close to the average model performance in related studies. Figure 8 shows the results of the grid search of RFR models using training data for all countries. The cross validation when testing forest parameters, bootstrapping of individual trees, and defining a maximum number of features to be considered allowed to find a robust forest that did not overfit. The standard deviation of the CV-scores for the best estimator was 0.6%. Hence, the test score lies within 2 standard deviations of the training-score of the best parameter.

C. *Support Vector Regression*: The grid search for the Support Vector Regression shows the highest test score for smaller values of hyperparameter ϵ and higher values of C. From Figure 9, the optimal values for ϵ lie around 0.3, and C between 1000 to 10,000 when using a polynomial kernel. After estimating the model using optimized parameters chosen by the grid search, we obtained a training R^2 of 62.8% and test R^2 of 60.8%.

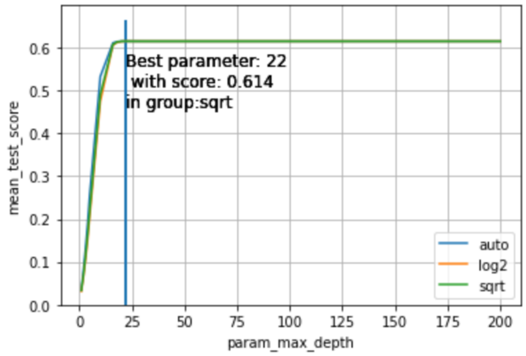

Figure 8. Random Forest Results for All Countries

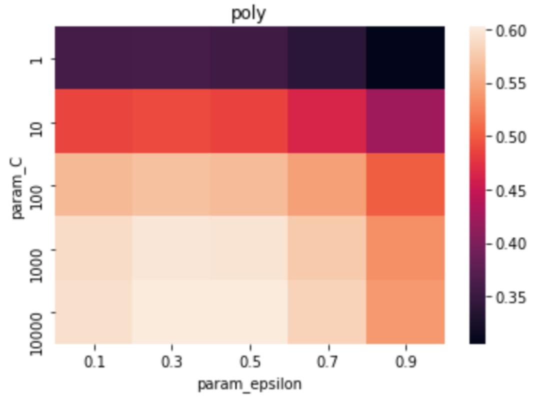

Figure 9. Support Vector Regression Grid Search Results

D. *CNN*: The two deep learning methods perform similarly, with the hard coded CNN performing better after 50 epochs, resulting in a training R^2 f 59.8% and 57.1% with the test set. Hence, there is little sign of overfitting. This can also be seen in the validation set R^2 10 and loss 14 are not diverging in the later epochs.

E. *Transfer Learning*: TL performs slightly worse than the CNN trained alone (R^2: 58.2% 56.4% Train/Test) after ten epochs of fine tuning. Not specifying an early-stopping criteria allows for better comparison: showing that after 50 epochs the pre-fine-tuning TL-model performs roughly two percentage points worse than the CNN. As fine tuning begins, the fit worsens and recovers after an epoch, then improving the fit and approaching closer to the performance of the CNN.

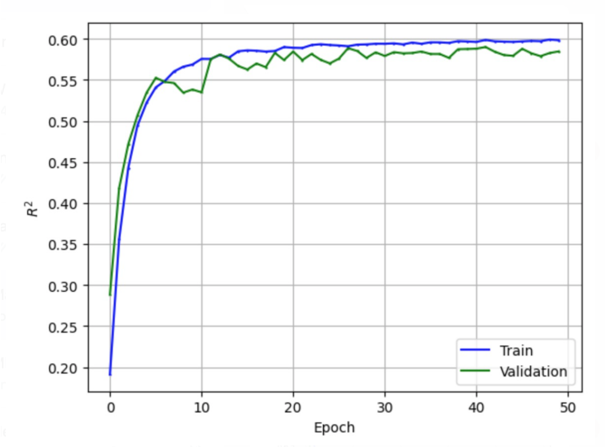

Figure 10. CNN Training Curve for Data from All Countries

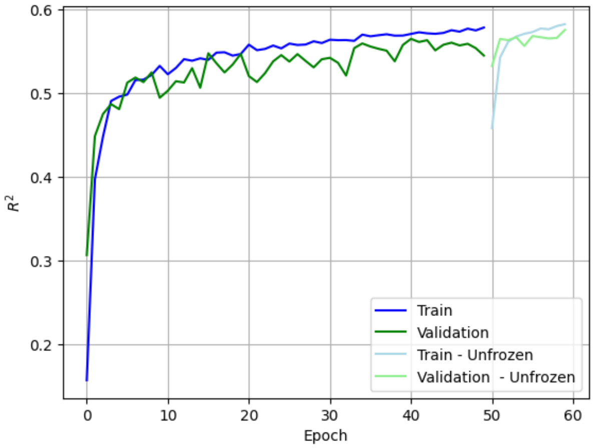

Figure 11. TL Training Curve for Data from All Countries

Method	Training R^2	Test R^2
Literature		65-70%
Linear Regression	4.26%	3.76%
RFR	61.4%	61.8%
SVR	62.8%	60.8%
CNN	59.8%	57.1%
Transfer Learning	58.2%	56.4%

Table 2. Comparison of test and training scores for each method

As expected, the baseline linear regression model performed the worst among five models in the study. This highlights the shortcomings of purely linear methods when applied to the complicated environment of computer vision.

Given the high dimensionality and non-linearity of the data, machine learning models significantly outperformed the baseline. Among the four machine learning regression algorithms, Random Forest and Support Vector Machine showed the highest performance, with R^2 greater than 60% for both training and test sets. In the case of the RFR, the fact that the test set marginally outperforms the training set is a sign that model has not overfit.

Against our expectation, the two deep learning methods (CNN and Transfer Learning) under performed compared to the Random Forest and Support Vector Regressions. This may be a consequence of training only in 50 epochs.

The following general improvements are necessary:

1. *PCA-dependency:* The results are highly dependent on the quality and effectiveness of the PCA. The PCA is trained only with a random sub-sample of the data (10,000). Increasing the number of samples to train the PCA (ideally, all) and the number of components will improve its ability to represent the data. Increasing the sample size should increase the effectiveness of each component and increasing the number of components will allow the models to train on more dimensions. The marginal variance explained decreases with each additional component, so adding more becomes insignificant after some time (Appendix: Figure 16).

2. *More precise image data:* Currently, multiple images can match a single household and one household can match multiple images (that are at least slightly different as ensured by filtering). Ensuring precise mapping of wealth indices to satellite images would reduce noise and increase the information passed to the models.

5. Analysis

The main objective of this project is to accurately predict poverty statistics in countries where data is sparse and provide more granular information for policy and humanitarian purposes. By comparing the ground truth wealth indices from DHS with the predictions from some of our machine learning models, we get an idea of how well the objectives of the study are met.

When comparing the predictions of the CNN against the ground truth (Appendix , we can see that while we roughly match the shape of the target distribution our predictions are more conservative than they ought to be. This is sensible given the high proportion of data in the poorer parts of the distribution. Further analysis using an activation heat map would allow us to which features of images drives the predictions.

Figure 12 visualizes the wealth indices for Malawi based on pre-defined clusters on the DHS survey, while Figure

13 shows the predictions from our best random forest regression model. Wealth indices produced from the machine learning model appear to be more granular in such a way that wealth predictions are more nuanced across the Malawi. For instance, in the middle of the country, there are a several "regions" that are significantly wealthier (and thus appear dark orange to red in the map) compared to its surroundings. The same area in the ground truth map collectively appears as relatively poor. Using the satellite images for estimation resolves this kind of aggregation of information.

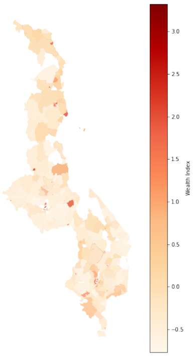

Figure 12. Malawi: Ground Truth Wealth Indices

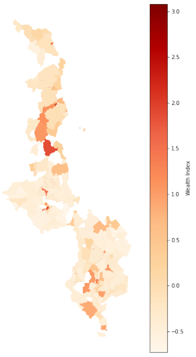

Figure 13. Malawi: Predicted Wealth Index

6. Conclusions

In this study, we implemented various machine learning models to estimate wealth indices in different geographic clusters of four Sub-Saharan African countries – Ethiopia, Malawi, Mali, and Nigeria. Our results confirm the viability of the methodology as a supplement to traditional data collection methods for poverty statistics. The best models (Random Forest and Support Vector Machine) achieved an R-squared of 61.8% and 60.8% for estimating asset-based wealth. Predicted wealth indices appear to be more nuanced than ground-truth data, an indication that satellite-based estimates are able to provide more accurate and granular information.

While we obtained model performance almost as good as previous studies with bigger scope, our study was limited by different factors, including the lack of stable and sufficient computing power to run and test deep learning models. To improve the study further, continued hyperparameter tuning can be conducted to improve the neural networks. The models can also be expanded to not only include images but other geospatial explanatory variables.

Ultimately, the application of machine learning techniques in poverty estimation has great potential to help governments, policymakers, and development institutions better understand the spatial characteristics of poverty and enable them to act accordingly.

7. Acknowledgements

Feedback & Guidance: Prof. Lynn Kaack for feedback on model performance and suggestions when faced with tricky obstacles, Eric Kolibacz for consistent ideas on other options to try when we experienced a roadblock with technical server issues, Prof. Will Lowe for advice on general improvement and developing efficient implementation of our work, and Prof. Slava Jankin for regular check in meetings to discuss our ideas and progress. *Thank you!*

Data Set: Planet Developer Resource Center generously shared satellite images for these countries on Kaggle.

Inspiration: Each of the studies noted in our earlier ”Related Work” section provided inspiration for not only the value of this work, but the opportunity for developers and data scientists to continue to make a difference in this space.

8. Contributions

While we worked in close collaboration, and each picked up tasks where needed, key contributions are: (A) image data pre-processing, and developing and implementing the machine learning models (RFR, SVR, CNN, TL) for Johannes; (B) outcome data pre-processing and developing visualizations for Janine; (C) and early research, results interpretation, and blog writing for Christine.

9. References

[1] S. Amidi and A. Amidi. Convolutional neural networks cheatsheet, 2019.

[2] B. Babenko, J. Hersh, D. Newhouse, A. Ramakrishnan, and T. Swartz. Poverty mapping using convolutional neural networks trained on high and medium resolution satellite images, with an application in mexico. *arXiv preprint arXiv:1711.06323*, 2017.

[3] X. Chen and W. D. Nordhaus. Using luminosity data as a proxy for economic statistics. *Proceedings of the National Academy of Sciences*, 108(21):8589–8594, 2011.

[4] F. Chollet. Keras documentation: Transfer learning amp; fine-tuning, 2020.

[5] R. Engstrom, J. S. Hersh, and D. L. Newhouse. Poverty from space: using high-resolution satellite imagery for estimating economic well-being. *World Bank Policy Research Working Paper*, (8284), 2017.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. *CoRR*, abs/1603.05027, 2016.

[7] N. Jean, M. Burke, M. Xie, W. M. Davis, D. B. Lobell, and S. Ermon. Combining satellite imagery and machine learning to predict poverty. *Science*, 353(6301):790–794, 2016.

[8] C. Ledesma, O. L. Garonita, L. J. Flores, I. Tingzon, and D. Dalisay. Interpretable poverty mapping using social media data, satellite images, and geospatial information. *arXiv preprint arXiv:2011.13563*, 2020.

[9] san__bt. Satellite images to predict poverty, Jan 2021.

[10] X. Zhao, B. Yu, Y. Liu, Z. Chen, Q. Li, C. Wang, and J. Wu. Estimation of poverty using random forest regression with multi-source data: A case study in bangladesh. *Remote Sensing*, 11(4):375, 2019.

[11] Z. Zhongming, L. Linong, Y. Xiaona, Z. Wangqiang, L. Wei, et al. Mapping poverty through data integration and artificial intelligence: A special supplement of the key indicators for asia and the pacific. 2020.

10. Appendix

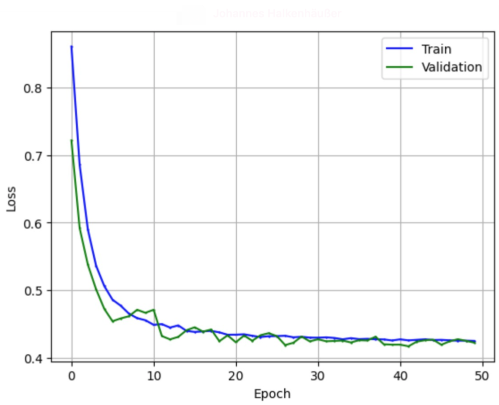

Figure 14. CNN Loss Results for All Countries

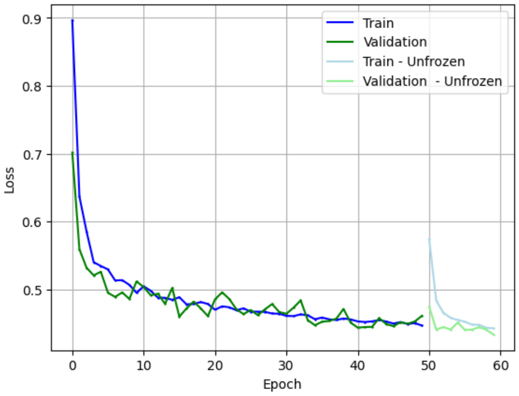

Figure 15. TL Loss Results for All Countries

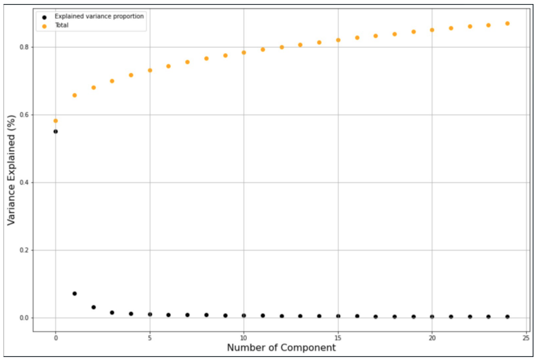

Figure 16. The marginal variance explained decreases for each component added.

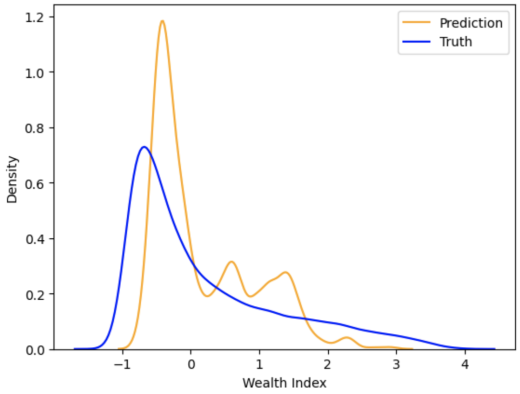

Figure 17. The kernel densities of the predictions of the CNN versus the ground truth.