ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

[1] Departamento de Engenharia de Computação e Sistemas Digitais - PCS

[2] Departamento de Engenharia de Sistemas Eletrônicos - PSI

CARLOS MASSAO OISHI GIUZIO[2]

IGOR VARELA ZOELLER[2]

RAFAEL ARAUJO COELHO[1]

Estimation of socioeconomic indicators through satellite imagery in the NEXUS area

SÃO PAULO - SP

2022

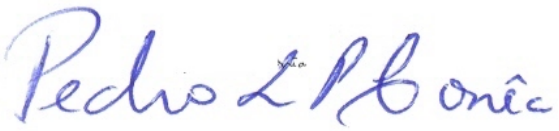
Estimation of socioeconomic indicators through satellite imagery in the NEXUS area

Work presented to the Polytechnic School of the University of São Paulo as a requirement to obtain a Bachelor's degree in Electrical Engineering.

Department of Electronic Systems Engineering and Department of Computer Engineering and Digital Systems

Advisor: Prof. Dr. Pedro Luiz Pizzigatti Corrêa

Co-Advisor: Dra. Marina Jeaneth Machicao Justo



Prof. Dr. Pedro Luiz Pizzigatti Corrêa (Advisor)

Escola Politécnica da USP

PhD. Marina Jeaneth Machicao Justo (Co-Advisor)

Escola Politécnica da USP

Cataloging in Publication

Estimation of socioeconomic indicators through satellite imagery in the NEXUS area / C. M. Oishi Giuzio, I. Varela Zoeller, R. Araujo Coelho -- São Paulo, 2022.
94 p.

Undergraduate Project - Polytechnic School of the University of São Paulo. Department of Electronic Systems Engineering and Department of Computer Engineering and Digital Systems

1.Deep Learning 2.Satellite images 3.Socioeconomic Indicators 5.NEXUS area I.University of São Paulo. Polytechnic School. Department of Electronic Systems Engineering and Department of Computer Engineering and Digital Systems II.Oishi Giuzio, Carlos Massao. III.Varela Zoeller, Igor IV.Araujo Coelho, Rafael

Acknowledgements

We thank all our families and friends for the wholehearted support during this project and all undergraduation years.

We thank Prof. Dr. Pedro Luiz Pizzigatti Corrêa and Dr. Marina Jeaneth Machicao Justo for accepting being our advisors and guiding us in this challenging and fulfilling project. We thank all the support, attention, inspiration and patience.

We thank NEXUS-PARSEC project and all its researchers for the support and materials provided, in specialty, Ali Ben Abbes, who helped us better understand the reference paper used in this project and the Deep Learning model pipeline, and Pedro Ribeiro de Andrade Neto, who gave us access to the main NEXUS area dataset.

Abstract

Parsec project aims to apply Data Science and Machine Learning techniques to applications for management and preservation of the world's biodiversity. In partnership with the University of São Paulo and the Brazilian National Institute for Space Research, INPE, a study is being conducted in the NEXUS area, a region of approximately 3.4 million square kilometers within the Brazilian territory, covering the São Francisco and Parnaíba River basins.This graduation project aids that study by proposing a methodology inspired by work done in the African continent in previous years, to estimate income, literacy and longevity indicators across this area, using publicly available satellite images collected with Google Earth Engine API. The contribution of this work is the creation and step-by-step documentation of a methodology for estimating indicators in the Brazilian territory through the training of Deep Learning models using satellite imagery, in addition to proposing a prototype of an interactive internet platform to the visualization of estimated indicators in the Brazilian map. The work also contributes with the creation of a new dataset, originated from merging a set containing more than 100 socioeconomic indicators provided by INPE and the administrative, geographic and geometric information of the Brazilian census sectors provided by IBGE. Deep Learning predictive models were trained using the backbone of a ResNet-18 to estimate socioeconomic indicators using multispectral and nightlight images. It was observed that multispectral experiments were appropriate for predicting the target values over the proposed dataset, while models using only nightlights performed below expectations due to data limitations. In view of the results obtained, it is understood that the methodology proposed by this work is suitable for the estimation of socioeconomic indicators in the Brazilian scenario.

Keywords: NEXUS area, Deep Learning, Satellite Images, Socioeconomic Indicators

Resumo

O projeto Parsec tem por objetivo aplicar técnicas de *Data Science* e Aprendizado de Máquina voltadas para aplicações de gerenciamento e preservação da biodiversidade mundial. Em parceria com a Universidade de São Paulo e o Instituto Nacional de Pesquisas Espaciais (INPE), está sendo conduzido um estudo na área NEXUS, uma região de aproximadamente 3,4 milhões de quilômetros quadrados dentro do território brasileiro, cobrindo as bacias hidrográficas do Rio São Francisco e Rio Parnaíba. O projeto de formatura auxilia neste estudo e propõe, inspirado em trabalhos semelhantes realizados no continente africano em anos anteriores, a estimativa de indicadores de renda, longevidade e alfabetização em toda esta área, utilizando imagens de satélite disponíveis publicamente pela API do Google Earth Engine. A contribuição deste trabalho é a criação e documentação passo a passo de uma metodologia para estimar indicadores no território brasileiro através do treinamento de modelos de *Deep Learning* utilizando imagens de satélite, além de propor um protótipo de uma plataforma interativa na Internet para a visualização dos indicadores estimados no mapa brasileiro. O trabalho também contribui com a criação de um novo conjunto de dados, originado da mescla de um conjunto contendo mais de 100 indicadores socioeconômicos fornecidos pelo INPE e as informações administrativas, geográficas e geométricas dos setores censitários brasileiros fornecidas pelo IBGE. Modelos de Deep Learning foram treinados usando o backbone de uma ResNet-18 para estimar indicadores socioeconômicos usando imagens multiespectral e noturna. Observou-se que os experimentos multiespectral eram apropriados para prever os valores alvo sobre o conjunto de dados proposto, enquanto os modelos usando apenas luzes noturnas desempenharam abaixo das expectativas devido a limitações de dados. Tendo em vista os resultados obtidos, entende-se que a metodologia proposta por este trabalho é adequada para a estimativa de indicadores socioeconômicos no cenário brasileiro.

Palavras Chave: área NEXUS, aprendizagem profunda, imagens de satélite, indicadores socioeconômicos

List of Figures

List of Tables

List of Equations

List of Abbreviations

AWS	Amazon Web Services
AZ	Available Zones
BA	Bahia
CNN	Convolutional Neural Networks
CSS	Cascading Style Sheets
CSV	Comma Separated Values
DHS	Demographic and Health Surveys
DMSP	Defense Meteorological Satellite Program
EPSG	European Petroleum Survey Group
FC	Fully Connected
FR	Functional Requirements
GDS	Geographical Data Science
GEE	Google Earth Engine
GHG	GreenHouse Gases
GIS	Geographic Information System
GPU	Graphical Processing Unit
GWP	Global Warming Potentials
HDI	Human Development Index
HTML	HyperText Markup Language
IBGE	Instituto Brasileiro de Geografia e Estatística
INPE	Instituto Nacional de Pesquisas Espaciais
JSON	JavaScript Object Notation
KNN	K-Nearest Neighbors
LSMS	Living Standards Measurement Study
MCTI	Ministério da Ciência, Tecnologia e Inovações
ML	Machine Learning
MS	Multi Spectral
MSE	Mean Square Error

MTTR	Mean Time To Repair
NCEI	National Center for Environmental Information
NFR	Non-Functional Requirements
NL	Night Light
NN	Neural Networks
NOAA	National Oceanic and Atmospheric Administration
ORCID	Open Researcher and Contributor ID
RGB	Red Green and Blue
RMSE	Root Mean Squared Error
RSS	Residual Sum of Squares
SFRB	São Francisco River Basin
SIRGAS	Sistema de Referência Geocêntrico para as Américas
TPS	Transactions Per Second
VIIRS	Visible Infrared Imaging Radiometer Suite
WGS	World Geodetic System
WKT	Well-Known Tex

Summary

1. Introduction

1.1. The PARSEC project

The PARSEC project (http://parsecproject.org/, BELMONT/FAPESP process 18/24017-3) seeks to manage and conserve global biodiversity through the application of Data Science and Artificial Intelligence (AI) techniques to analyze satellite imagery and socioeconomic indicators on a global scale. It is made up of 6 regions across the world: Australia, Brazil, Easter Africa, France, Japan and the United States of America.

The PARSEC project is designed to provide a unique opportunity for data and synthesis scientists to collaborate and exchange in real-time toward the goal of improving research outcomes, data sharing, and data reuse. It will also help pioneer new scientific and data science technologies aimed at improving both the management and conservation of global biodiversity.

It has two teams in this project with links between: a Synthesis Science team and a Data Science team. Our Synthesis Science team is employing artificial intelligence techniques to analyze satellite images and socio-economic information to better predict and mitigate the effect(s) of actions that potentially threaten the livelihoods and health of local (indigenous) communities. Like most researchers who investigate complex environmental problems, the team depends significantly on the availability of good, spatially dispersed, multidisciplinary, and time-series data.

The Data Science team of leading environmental data management professionals, data communities, society journals, and representatives of e-infrastructures for data attribution (e.g., DataCite and ORCID) will develop leading practices on data citation, attribution, credit, and reuse. As part of the integrated work with the synthesis-science team, the data-science team will provide a review of best practices for data management and stewardship using this effort as a case study of the wider scientific community to optimize data access and reuse. The team will also develop and implement a new tool to better track data usage and reuse for researchers.

1.2. The NEXUS project

The NEXUS project (http://nexus.ccst.inpe.br/, FAPESP process 17/22269-2), which is related to the Instituto Nacional de Pesquisas Espaciais (INPE), have the purpose to maintain and develop a sustainable future to the NEXUS area, a Brazilian region composed of Caatinga and Cerrado biomes, between the São Francisco River Basin (SFRB) and the Parnaíba Basin, a great place because of its agricultural and energetic relevance.

The project has 3 transversal phases in research and development, the first one produces socioeconomics, institutional and ambiental indicators, with the objective to reflect the actual situation of the area, aligning the national demands. With this, the project identified approximately 150 indicators divided into 8 categories, such as Forest conservation and biodiversity, Land degradation, Energy, GHG and BGW Cycles, Sustainable agricultural production, Water resources, Climatic risks and Socioeconomic.

The second phase has the objective of thinking about the future of the study area, through the creation of qualitative and quantitative cenarios. To do this, spatial projections are created based on the changes of land usage, the regional climate and its impacts.

The last phase is responsible for the activity of synthesis and analyses of the other two, capable of presenting paths that lead to the sustainable transition in the country.

1.3. The NEXUS-PARSEC project

Because of the alignment of both projects, together, they form the NEXUS-PARSEC project, a partnership between the Polytechnic school of the University of São Paulo and the INPE. The first is responsible for creating the study environment, producing and distributing the dataset containing the socioeconomic indicators used in this work. The second has the objective of maintaining this research and the data collected within it safe for the generations to come, assisting and facilitating the sharing of said results, creating a better space for new analyzes and researches with these estimations.

2. Motivation

Brazil is the fifth largest country in the world, with 8,5 million km², and the seventh most populous country, with 217 million people. It has a vast richness in natural resources, with 6 distinct biomes, Amazônia, Mata Atlântica, Cerrado, Caatinga, Pampa e Pantanal, and considered to be the most biologically diverse country in the world, with at least 100 thousand animal species and 43 thousand plant species [1], and this characteristics makes them the largest national economy in Latin America .

The NEXUS area is a region that englobes the most important watersheds of the country, the São Francisco River and Parnaíba basins and has 3,4 million km², representing around 40% of the country's total area [2]. It has 2513 municipalities, that represents 45,13% of total country value, and has 5 of 6 Brazilian biomes, not contemplating only the Pampa ecosystem. This territory (Figure 1) is the region of the Brazilian country with more influence in commercial agriculture and represents a rich diversity of natural resources, and therefore the need for preservation policies and sustainable development.

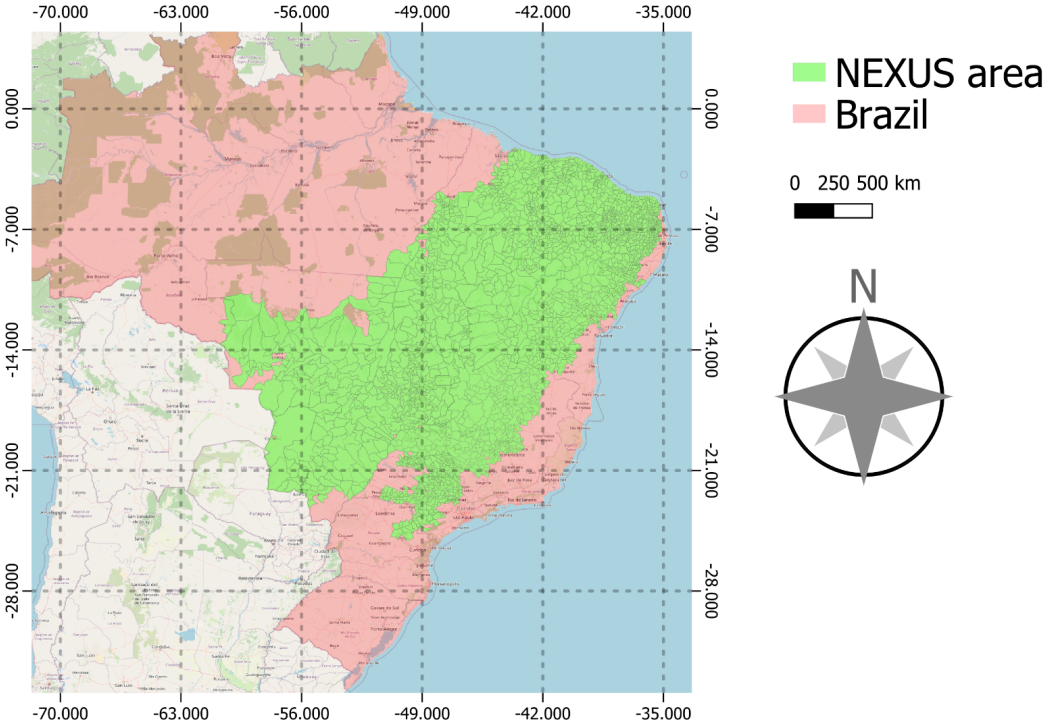


Figure 1 - Map of the NEXUS area.

For managing these resources, an important step is the frequent and in-depth monitoring of the country's socioeconomic and environmental indicators, which are useful for decision making. In Brazil, IBGE (Instituto Brasileiro de Geografia e Estatística), the Brazilian Institute of Geography and Statistics, is responsible for carrying out censuses to obtain social data used as a basis for calculating such indicators.

However, the accomplishment of these censuses becomes a challenge considering the large territorial expansion and the difficulty of accessing many of the country regions: census surveys can be expensive, time consuming, difficult to obtain and are collected every 10 years, not reflecting real-time reality [3].

Besides that, censuses can be affected by external factors, causing delays in their implementation and dissemination, as an example, the survey scheduled to take place in 2020 was postponed because of the COVID-19 pandemic and also in 2021 due to lack of budget, being carried out only in 2022, and municipalities received amounts of vaccines lower or higher than the total needed for their population due to outdated census information [4]. Another example is the lack of census agents due to the difficulty in hiring and retaining professionals in 2022, due to low remuneration and late payment of wages, according to [5], further delaying the completion of the collections of this research [6].

As alternatives for calculating indicators through ground surveying censuses, there are sources such as geo-located sensors, cell phone usage data, remote sensing imagery and aerial imagery that are created and broadcast daily and can be widely, low-cost and easily accessible [7]. In recent years, there has been a significant increase in the number of remote sensing dataset sources, such as Google Earth Engine (GEE), a free cloud platform and repository of petabytes of high resolution remotely-sensed data spanning over 40 years [7]. Such data is extensively being used to estimate social and environmental indicators through Deep Learning models with transfer learning, benefitting from large datasets, contextualized in the Big Data and Deep Learning evolution in the most recent years [8][9]. This allows the minimization of the previously mentioned limiting and delaying factors for collecting and calculating these indicators which can be estimated daily with lower costs than the resources needed for censuses.

Therefore, this motivates the employment of Deep Learning models to estimate socioeconomic indicators through satellite imagery for the NEXUS area, as done previously in similar works focused on African countries by Jean et al. (2016) [10] and Yeh et al. (2020) [11]. This methodology aims to increase census coverage, both spatially and temporally, by removing the territorial challenges, such as the need to physically be in the studied area, and by automating the analysis process.

3. Objectives

The project has the objective of proposing a Deep Learning model for estimating socioeconomic indicators within the NEXUS area. It is proposed that the model can estimate such indicators spatially, that is, it is trained with NEXUS area data and is able to generalize the results throughout the entire Brazilian territory, and temporally, i.e. the model is trained for the specific years when the data was collected and is able to generate values for different in the time.

The model is trained with satellite imagery comprised within the study region, which is collected from geospatial datasets, and socioeconomic surveys provided by the NEXUS-PARSEC project.

Another goal is to develop a visualization platform which allows users to explore the model predictions in different Brazilian regions, analyze the indicators distribution, request new predictions and download data.

The shared achieved results can also serve as a basis for new studies and research related to the Brazilian socioeconomic context. It will be useful for the PARSEC and NEXUS projects, providing relevant data for analysis on the Brazilian territory, taking into account the estimated socioeconomic indicators. One example is the study of the influence of protected areas in nearby municipalities compared to regions far from these locations.

4. Conceptual Aspects

4.1.Brazilian census

4.1.1.Administrative division

Brazil is divided into 5 main regions, which segment states based on their cardinal direction: South, Southeast, Midwest, North and Northeast. The highest hierarchical units in this division are the states previously mentioned [12], they count up to a total of 26 and each has a capital city. Brazil also has one Federal District, which is the center of Legislative, Judiciary and Executive powers, and also contains the country's capital, Brasilia. Next on the hierarchy are the municipalities, which count with administrative divisions called districts [12].

4.1.2.Census tracts

A tract is the lowest granularity defined for the Brazilian census [13]. The size and number of buildings are defined so that census agents are able to query the indicators in a reliable way. Each tract is strictly confined by the administrative divisions and is identified by what is called a "geocode" that reads as follows: the first two digits depict the state code, the next five digits indicate the municipality code, the following two digits come from the district code, then two digits indicating the subdistrict and the last four are inherent to the tract [13].

As of the year of 2010, IBGE describes three criteria to define the delimitation of tracts: (a) the number of households or establishments must be between 150 and 400 buildings for urban areas, while for rural areas this range is between 150 and 250; (b) the tract limits must must be strictly contained in the administrative divisions defined for that year and (c) it's limits should, when possible, be depicted by discernible landscape elements for easier identification [13].

The distinction of urban and rural areas is done either by recognition of each municipality's urban policy legislations, i.e. local government information provided on urban planning and zoning reports, when available and up-to-date; or by evaluating the morphology of the area, as well as identifying regions effectively urbanized and with lots smaller than the ordinary rural fractioning [13].

4.2.Machine Learning

Machine Learning (ML) is the study of computer algorithms that automatically improve through experience and the use of data [14], allowing systems to learn from problem-specific training data in order to automate the process of analytical model building and solve associated tasks [15].

4.3.K-Nearest Neighbors

K-Nearest Neighbors (KNN) is a supervised learning model that solves regression and classification problems by calculating the similarity between data points through some distance metric, such as euclidean distance. The value of K determines the number of neighbors that will be weighed to determine an output, e.g. in regression problems the output value is the average of the values of K nearest neighbors. It can also be called a “lazy learning” algorithm [16], since there is usually no optimization step and instead stores all the “training” data and utilizes them to calculate their similarity to the entry point during the prediction step.

4.4.Artificial Neural Networks and Deep Learning

Neural networks (NNs) are a subset of ML and are at the heart of Deep Learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal one another.

The neurons are represented by nodes, distributed in multiple layers that interconnect with each other, each with a certain weight associated and also what is called an activation function, both combined define how each node computes an output signal through the input it receives. The weights connected in a NN are coefficients that amplify or minimize the signal of a particular neuron in the network and are continuously learned, i.e. adjusted and fitted to the training data, through multiple iterations of the training process [15].

The behavior of a NN is shaped by its network architecture, which can be essentially defined through: number of neurons, number of layers and types of connections between layers. Figure 2 exemplifies a possible topology of a neural network.

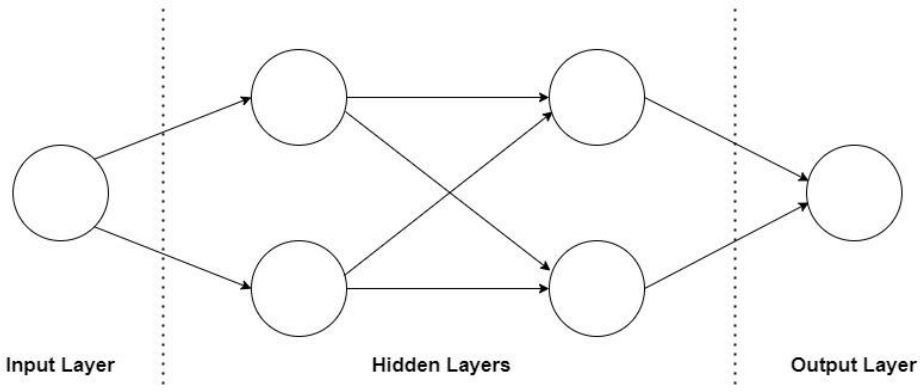


Figure 2 - An example of a neural network, specifically a multi-layered one of the feed-forward type. Each layer can have any number of neurons, and, in this case, are fully-connected.

Simple NNs architectures, such as the one from Figure 2, can also be called as *shallow* neural networks, named in contrast to the concept of Deep Learning models, which

count with a large number of hidden layers that compose a deeper, more complex architecture with nodes that are, in many cases, able to do more advanced transformations [15] (e.g. *convolution*, as seen in the next topic). The huge amount of layers and the more complex connections between them makes those systems capable of processing multiple parameters [17], and thus, benefiting from problems defined by large amounts of multidimensional data, such as audio or image [15].

4.5.Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are primarily used in the field of pattern recognition within images and represent Deep Learning models for data processing that have a known topology in the form of a grid. This is due to the fact that the model employs a mathematical operation called convolution, a specialized type of linear operation [18] which allows to encode image-specific features into the architecture, making the network more suited for image-focused tasks [19].

4.5.1.Convolution

Convolution is a simple linear transformation and its core operator is called a kernel: a small square matrix of weights with parameterizable size (e.g. 3x3). The kernel slides over points of the 2D input data matrix following a stride parameter, which determines how many points the kernel shifts in either dimension when sliding. The output is the weighted sums of each location. Thus, the kernel's weights are optimized during training to produce general features from local inputs, which yields great results for extracting information from images, since pixels are usually consistently spread and are influenced by neighboring pixels [20].

4.5.2.Pooling

Pooling is a nonlinear transformation layer utilized to gradually decrease the input dimension and uses a similar kernel operator as convolution [21]. The size of the output is also given by the stride parameter. The two most common pooling operations are Max Pooling, which extracts the maximum value inside the kernel; and Average Pooling, which averages the values in the kernel.

4.5.3.Architectures

A CNN has multiple architectures, which can diverge a lot from one another, but they are mostly based on 3 classes of layers, as Figure 3 shows. The input layer has the shape of the images fed to the network [17]. It can be, for example, a one-channel matrix (gray scale)

or a 3-channel matrix (RGB images); the feature-extraction layers are responsible for the mathematical operations (i.e. convolution, pooling, etc). Typically, in tasks regarding imagery, the first layers are responsible for extracting general information, such as edges and corners, while deeper convolutional layers can aggregate more and more complex features to the objects of interest [15]. Furthermore, those layers are usually a combination of a convolutional step that has its output passing through some activation function, followed by a pooling step (either maximum or average pool).

Finally, there are classification layers, which come down to a conventional dense network of fully-connected layers able to compute classification scores from high-order features extracted before.

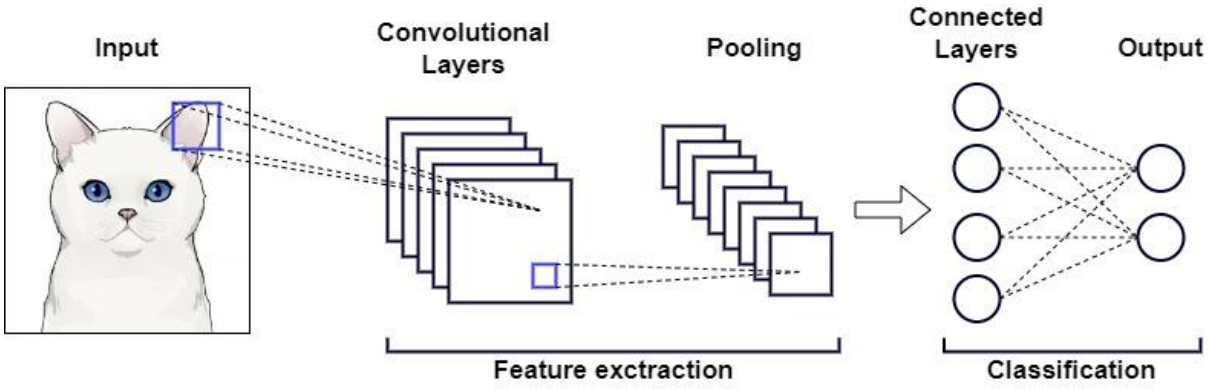


Figure 3 - Convolutional neural network architecture [17].

4.5.4.Deep Learning Hyperparameters

As any deep neural network, CNNs are composed of a list of hyperparameters that determine its design and the usage of training data, as those can be related to either the architecture or structure of the network, such as number and order of layers (e.g. convolutional, pooling or dense layers), number of neurons, connectivity patterns of dense layers, weight normalization, kernel size, dropout rate and activation function [22]. They can also be related to the training process, regarding learning rate, batch size and number of epochs. There is no clear directive on how to select these parameters [15], as the success of their usage can change greatly for different problems [23].

4.5.4.1.Input shape

The shape of the input layer defines the resolution of the images fed to the CNN and thus, is of extreme relevance to this work. It is important to note that this parameter is usually immutable, since most conventional convolutional networks have one or more fully connected layers, which rely on fixed input and output shapes; thus, altering the input layer implies retraining the model with changes in its architecture's layout. Early architectures were designed for square inputs between 224 × 244 and 299 × 299 pixels, as those resolutions

provide the best balance in computational efficiency and model performance [24]. Furthermore, experiments [24] show that architectural changes may introduce artifacts and can result in the reduction of more than 50% of the predictive performance.

4.6. Transfer Learning

Due to the large amount of time and processing power required to train a CNN model from scratch, a common approach is to take a previously trained CNN model that has performed well for a specific task and train it again on an additional set of data to solve a separate, but somewhat related, problem. This technique is called transfer learning [17]. Great examples to use transfer learning are image classification, using a widely trained model that already detects a lot of objects to classify a new one.

In other words, this approach repurposes a model's predictive capabilities. In that regard, the process will yield better results if the features previously learned are general and thus, suitable for both tasks, instead of only one in specific [25].

4.7. K-Fold Cross Validation

An alternative to increase model performance is to validate it, i.e. to estimate its performance on unseen data, on different samples of the dataset, this can be achieved by using a resampling method such as K-Fold Cross Validation.

The K signifies the total number of equally sized subsets (or folds) into which the available data is randomly split. From these subsets, one is held out to be a validation set and, in some cases, another is held to be used as a test set. The model is then trained on the remaining folds, this procedure is repeated K times so each subset is utilized to validate the model and the Mean Square Error (MSE) is calculated as the average MSE of the experiments.

The reason for utilizing a resampling method is that it portrays a more realistic overview of how the model would perform in a real scenario [26]. Another argument for K-Fold Cross Validation is that this method yields a good bias-variance trade-off since the majority of available data is being used as the training subset, thus reducing bias; while still keeping variance low, as each experimentation is different enough from each other to result in models less correlated between each other as opposed to using Leave One Out Cross Validation [26], which keeps only a single observation for validation. This balance also explains why it is common practice to utilize either K = 5 or K = 10, as those values have been shown to yield better validation estimates [26].

4.8.Standardization and Normalization

Before feeding inputs to the network, preprocessing the data is an important step if there are distinct measurement units being used. Besides, it prevents data from interfering with the backpropagation step by inflating gradient descent values as data in different scales may cause some weights to update faster than others [27].

Standardization means rescaling data so it falls in the range of mean zero and unitary standard variation. The resulting values, called standard scores or z-scores, are calculated by Equation 1.

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

Alternatively, there is min-max scaling, also called normalization in which data is restrained to a specific range (usually 0 to 1) by Equation 2. The main difference is that this approach does not guarantee unitary standard variation, since the objective is to transform any range of data a to b to the unitary one, making the statistics of the data change. One downside of this process is that outliers of data must be handled beforehand, to avoid maintaining the data less sparse, since to calculate this normalization needs the maximum and the minimum value of the dataset.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{2}$$

4.9.Ridge Regression

Linear regression models are used to estimate a target value y through n independent variables called features X_i, $i = 1, 2, \ldots, n$. The model uses $n + 1$ coefficients β_i, $i = 0, 1, 2, \ldots, n$, also called weights, to calculate a prediction $\hat{y}$ through Equation 3. The difference between the target value and the predicted one is called residual e, as shown with Equation 4 [26].

$$\hat{y} = \beta_0 + \sum_{i=1}^{n} \beta_i X_i \tag{3}$$

$$e = y - \hat{y} \tag{4}$$

The training procedure of a predictor with a training set of k samples is based on trying to optimize its weights in order to minimize a residual metric calculated over the $e_j, j = 1, 2, \ldots, k$ residuals, where j is the number of the sample. Generally, the residual metric used is the Residual Sum of Squares (RSS), described by Equation 5 [26].

$$RSS = \sum_{j=1}^{k} e_j^2 = \sum_{j=1}^{k} \left(y_j - \beta_0 + \sum_{i=1}^{n} \beta_i X_{ij} \right)^2 \quad (5)$$

Ridge Regression is another linear regression method which is very similar to the one already described, but it introduces a new residual metric by adding to the RSS a shrinkage penalty, $\alpha \sum_{i=1}^{n} \beta_i^2$, which is a term with a new hyperparameter alpha $\alpha \geq 0$, called tuning parameter. It is presented in Equation 6 [26].

$$\sum_{j=1}^{k} \left(y_j - \beta_0 + \sum_{i=1}^{n} \beta_i X_{ij} \right)^2 + \alpha \sum_{i=1}^{n} \beta_i^2 = RSS + \alpha \sum_{i=1}^{n} \beta_i^2 \quad (6)$$

This new term shrinks the estimated association of each variable with the response, i.e. when the coefficients are too small, the shrinkage penalty is also small, not affecting RSS value, however this factor grows when coefficients are bigger, causing RSS to increase. In other words, this forces coefficients to be smaller, leading to less variance by adding bias to the prediction, a fair trade-off [26].

The shrinkage is controlled by the tuning parameter: the higher its value, the more impactful it will be on the reduction of coefficients, this means that for $\alpha \to \infty \Rightarrow \beta_i \to 0, \forall i \neq 0$ (β_0 is not shrinked) [26]. This hyperparameter must be empirically tested in order to find the best possible configuration, being quite appropriate the use of k-fold cross validation to determine such value [28].

Ridge Regression is used for the analysis of multicollinearity in multiple regression data, when high intercorrelations occur between two or more independent variables of one model. It is most suitable when a data set contains a higher number of predictor variables than the number of observations (e.g. economic market or biological studies) [28].

This regression method strives on reducing the standard error by adding some bias in the estimates of itself, significantly increasing the reliability of the estimates.

4.10.Metrics for evaluation of regressive models

4.10.1.Pearson correlation coefficient

Pearson correlation coefficient, r^2, measures the linear correlation between two quantitative variables. It can be described as the ratio between the covariance of two variables and the product of their standard deviations and ranges from -1 to 1. The closer the coefficient is to -1, the closer the variables are to a perfect negative correlation, i.e. when one

increases, the other decreases. On the other hand, the closer the coefficient is to 1, the closer the variables are to a perfect positive correlation, which means that both increase or decrease jointly. Furthermore, if the coefficient is 0, the variables are linearly independent, but can still have a non-linear correlation [29].

4.10.2.Coefficient of determination

The coefficient of determination, R^2 or R2, measures how the variability in the target value is explained by a regression model, conceptually it ranges from 0 to 1, however it may give negative results when calculated by data science coding libraries due to the way it is implemented. The closer it is to 1, larger is the proportion of the target values' variability explained, while a R^2 approximate to 0 means the regressor is not able to explain much about that variability [26].

When it is in the latter situation, it indicates that either the model was not able to fit the data correctly or that the nature of the relation between the target value and the input features is not linear [26].

4.10.3.Mean squared error

Mean Squared Error (MSE) is one of the most commonly used measures to evaluate regression models. It is calculated as the mean value of the squared differences between the target and predicted values for all samples in the dataset being evaluated [26].

It measures how accurate the predictions of the model are over unseen data. The smaller test MSE (MSE over test data), the better the model for predicting new data [26].

4.10.4.Spearman rank correlation coefficient

A rank is the ordinal relationship between two variables: the first is either higher, lower or equal to the second. The Spearman rank correlation coefficient is a measure of a rank correlation, i.e. a statistic that measures the relationship between two ranks, indicating how similar they are to each other. This explains how well the relationship between two variables can be described using a monotonic function, that is the linear regression in this case [30].

It ranges from -1 to 1. The greater its absolute value, the greater the relationship between the features and the target value, while the sign shows how the features and the target are associated: it is positive if they are directly proportional, that means when the features increase, the target increases and vice versa, and it is negative if there is inverse proportionality, i.e. when the features increase, the target decreases and vice versa [30].

The Spearman rank correlation coefficient is mentioned as “rank” in this work.

4.11.Related works

Both articles mentioned next are related to the training of ML models which use satellite images to estimate socioeconomic indicators. They are used as the main reference base for carrying out the project.

4.11.1.Yeh et al., (2020)

Yeh et al. (2020) [11] aimed to build an accurate, inexpensive and scalable model capable of estimating socioeconomic indices through the analysis of high resolution satellite images, using a method with a good degree of accuracy based only on data available in the public domain.

To achieve this objective, a CNN is trained to predict a village-level and year-specific measure of wealth using both the low resolution nighttime and high resolution daytime images temporally and spatially matched as input, regarding more than half a million households living in 19669 villages across 23 African Countries, utilizing satellite imagery and census data gathered between 2009 and 2016.

The work is motivated by the following factors: (a) high cost for traditional census collecting methods, especially in developing countries, (b) low frequency of African households in well-being surveys (as shown in Figure 4) and (c) high coverage of satellite imagery at very low costs.

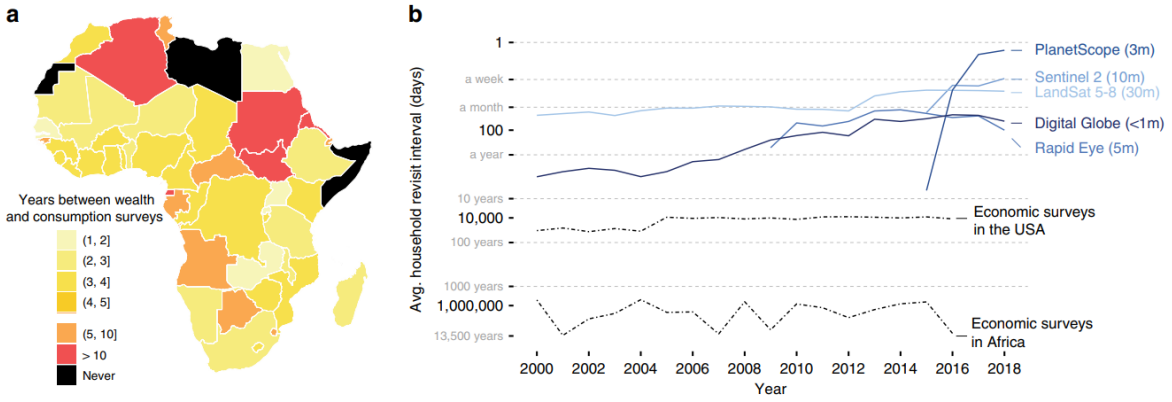


Figure 4 - Frequency of economic data from household surveys in African countries. [11]

The objective is to estimate wealth index through deep learning models trained with satellite imagery, since economic indicators help governmental livelihood programs and development and deployment of products, policies and services in parts of the developing world. The predictive performance contemplates both spatial and temporal variation, as the gathered data allows the distinction.

The input data comprehends: satellite imagery (LANDSAT), night light images and asset wealth surveys across 23 countries in Africa. About imagery, nighttime light (nightlight) images present coarse resolution (i.e. 1km/pixel), daylight images from private-sector providers are high resolution, using a scale of 30 meters/pixel at a resolution of 224 × 224 pixels, and they are processed into 3-year median composites, which clears satellite imagery and avoids seasonal or short-run distortion.

The entire MS network consists of two models trained separately - one for each type of input - and then joined in a final fully connected layer. Both models are ResNet-18 backbones, trained with transfer learning, for extracting common features such as textures, borderlines and patterns, one for LANDSAT imagery and another for nightlight, as shown by Figure 5. The fully connected layer is responsible for estimating the wealth index through the inputs of each convolutional model.

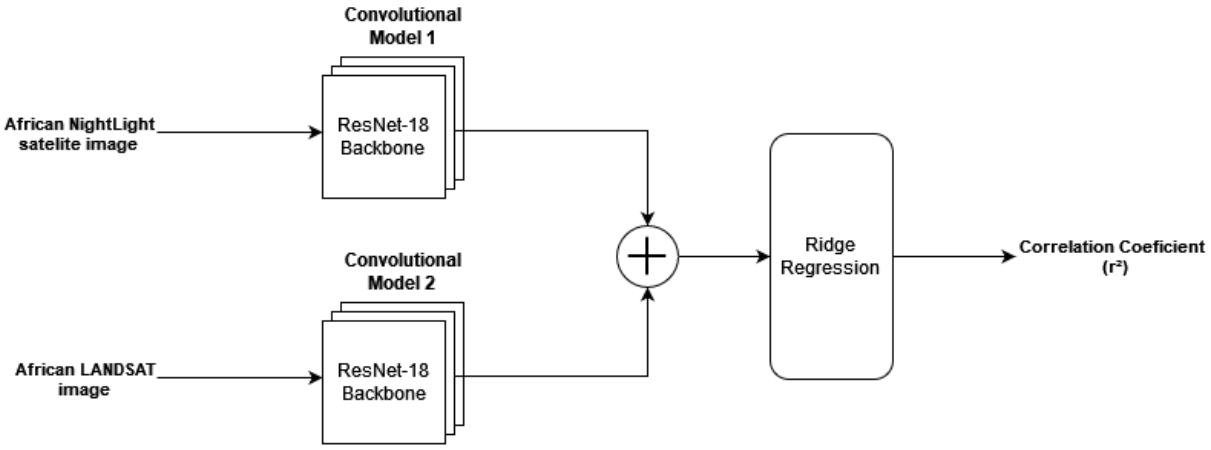


Figure 5 - Representation of Yeh et al. Deep Learning model. [11]

Finally, to test the model, both the actual wealth index and the predicted one generate a plane and the r^2 is calculated over a regression algorithm (Figure 6). The results show, at country level, that the precision is around 70% and never below 50%, therefore it is capable of differentiating wealth levels within countries. This model combining both image types has a good performance over time, indicating that probably only one of the image types (multispectral daytime) explains the time variation. The algorithm is fast and efficient, since the model was trained in 4 hours with a NVIDIA Titan X GPU and only 24 hours for imagery processing.

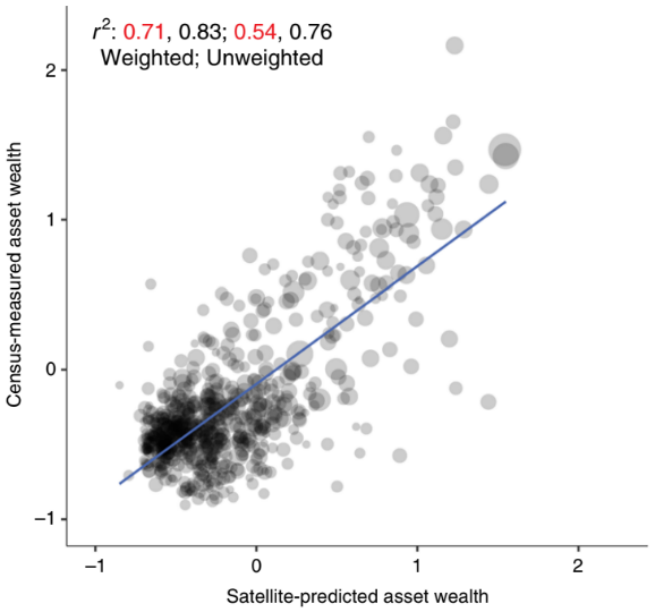


Figure 6 - Model validation through r² [11].

In this article, publicly available satellite images are used as a basis for training a machine learning model to estimate socioeconomic conditions in different regions of sub-Saharan Africa: the authors train a convolutional neural network (CNN) using daytime and night-light satellite imagery to estimate asset richness indicators, which is validated through Demographic and Health Surveys (DHS) ground-based data.

The model is able to explain, on average, 70% of the variation in wealth of goods measured in those ground surveys, a number higher than that achieved by models that took into account the use of cell phones or only satellite images of night lights.

This study is the main reference for this work.

4.11.2.Jean et al. (2016) [10].

The objective of the article was to build an accurate, inexpensive and scalable model capable of estimating socioeconomic indices through the analysis of high-resolution satellite images, using a method with a good degree of accuracy based only on data available in the public domain.

To achieve this objective, census data and satellite images referring to the territory of five African countries were used: Nigeria, Tanzania, Uganda, Malawi and Rwanda. The algorithm employed a trained convolutional neural network to identify aspects of the analyzed images and explain local variations of socioeconomic levels resulting in a workflow of satellite images as input and poverty rates indicators as output.

The data comprehends similar types as Yeh et al.: socioeconomic indicators referring to the degree of wealth and poverty of local populations, daytime satellite images and nightlights satellite images. Socioeconomic data were obtained through the World Bank

project of Living Standards Measurement Study (LSMS), converted into .csv format. Satellite images were obtained by geographic coordinates through the API of Planet and Google Static Maps. In addition, night light images were acquired by the National Oceanic and Atmospheric Administration (NOAA) interfaces in conjunction with the National Center for Environmental Information (NCEI).

The algorithm employed by the article was built in R language and works based on a work methodology divided into 4 distinct stages: data collection, feature extraction, model training and validation.

In the data collection stage, a study region is selected and subdivided into clusters, each one assigned a value of poverty rate. Each cluster corresponds approximately to an area measuring 10 km by 10 km. The algorithm acquires a minimum of 10 daytime satellite images per cluster and one image of night lights corresponding to each cluster.

Initially, features are extracted from daytime satellite images, so that the algorithm is able to recognize specific aspects of the images. Then, images of night lights corresponding to the same region are used so the algorithm filters and removes places with low light intensity. This occurs because it makes no sense to study regions without significant human presence, which could be identified by the absence of light in night photos. Then, with features extracted and images properly filtered, the algorithm trains the deep learning model with the census data and daytime satellite images.

The last step consists of validating the obtained results and refining the model. As a result, the algorithm was able to explain from 55% (R2 = 0.55) to 75% (R2 = 0.75) of the variation in wealth measured in the 5 countries studied, and 37% (R2 = 0.37) to 55% (R2 = 0.55) of the variation in domestic consumption, a satisfactory result that this work seeks to replicate for environmental indicators.

4.12.Distributed Architecture Systems

There are two opposite types of systems architecture: centralized systems, which run in a single machine, and decentralized (or distributed) systems.

Distributed systems consist of applications running in multiple software components, i.e. machines or, also called, nodes, allocated in different computers, minicomputers, mainframes or workstations. This kind of system is usually adopted considering its main benefits: scalability, performance, fault tolerance and cost-effectiveness [31].

The architectural style commonly employed with cloud computing services is the Object-Oriented Architecture, which is used in this work.

4.12.1.Main benefits

Consider a distributed computing system that receives requests through an API (Node 1), executes a logic in a server (Node 2) and stores the execution output in a database (Node 3). The following concepts take this system as a basis for discussion.

4.12.1.1.Scalability and Performance

If the application's TPS (Transactions Per Second) increases, exceeding the projected traffic limit, it is necessary to increase its capacity. Since each node runs a specific microtask separately, a good alternative is to increase the amount of Node 2 instances, i.e. scale horizontally. In this case, by adding a load balancer between Node 1 and the instances aiming to distribute the requests evenly, there will be concurrent executions, one for each request.

Regarding the same scenario, if it is necessary to decrease the latency for reading and writing into the database, along with increasing its storage capacity, the database machine may be updated to a better one, faster and larger (scale vertically).

In both cases, it is easy to add new hardware or to improve the existing ones, once each component works individually and is a black box to whichever else. Therefore, performance may be increased through parallelism, where each node simultaneously handles a task of the functionality (divide-and-conquer approach).

4.12.1.2.Fault Tolerance

Centralized applications may present a backup machine if the original server fails. A problem generally forces an analysis of the whole system in order to detect its cause and it might be necessary to rollback the entire code base to an older version.

On the other hand, for decentralized systems just like in the example, faults are generally concentrated into one or more nodes, which leads to localized unavailability instead of the system completely being off. For example, if an instance of Node 2 is unavailable, the application will still work with less capacity.

4.12.1.3.Cost-effectiveness

The initial cost of distributed architecture systems is higher when compared to centralized ones. However, the former architecture tends to be hard to scale horizontally and its mainframes will, eventually, need maintenance and replacement, i.e. vertical scaling. Horizontal scaling is much cheaper in the long term.

To achieve the same requirement, e.g. latency, throughput and availability, a distributed system will be, in many cases, more effective by using specific and low-cost

hardware. As already mentioned, cluster expansion and improvement is easier and more localized with a distributed architecture.

5. Technologies

5.1. AWS

Amazon Web Services (AWS) is the platform of Cloud Computing Services provided by Amazon that allows the use of web servers and database services. It is useful to store the project datasets and leverage model training, utilizing the computing power of available machines [32][33].

The project also aims to create a visualization platform for allowing users to request new predictions using AWS resources, such as AWS SageMaker, AWS S3 and DynamoDB.

AWS SageMaker is the environment of AWS that stores the project model and is capable of running it depending on the interface demand. AWS S3 is the main store service of the platform, used to store the images already estimated by the model. DynamoDB is the AWS main SQL database, able to store the images characteristics, such as its centroid, indicator estimations and IDs.

5.2. QGIS

QGIS is an open source Geographic Information System (GIS) multiplatform software for professional purposes [34]. A GIS is an application that analyzes and displays geographically referenced information [35] using data that is attached to a unique location. It is the tool utilized to merge and organize the raw shapefiles and also compute and fix geometries.

5.3. Google Earth Engine

Google Earth Engine is a cloud-based geospatial analysis platform that allows users to view and analyze satellite images, enabling remote sensing, epidemic prediction, natural resource management, among many other applications [36]. It is the source used in this project for acquiring satellite images.

5.4. Python 3

The entire project is developed and executed on Python 3, more specifically using sectionJupyter Notebooks to combine it with insightful text markings. The different file

versions are maintained on GitHub and all Python dependencies are listed in the next sections.

5.4.1.TensorFlow

TensorFlow is a free and open source library for machine learning [37]. Developed by Google, it can be used in a number of tasks, but has a particular focus on training and inferring deep neural networks.

5.4.2.Matplotlib

Matplotlib is a MATLAB based library for creating graphs and general data visualizations [38], made for and from the Python programming language and its NumPy math extension. It produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, Python/IPython shells, web application servers, and various graphical user interface toolkits.

5.4.3.Pandas

Pandas is a free open source software library created for the Python language for data manipulation and analysis [39], offering structures and operations for manipulating numerical tables and time series.

5.4.4.GeoPandas

GeoPandas is an open source project for working with geospatial data in Python [40]. It extends the datatypes used by pandas to allow spatial operations on geometric types. It also takes advantage of the matplotlib library to facilitate plotting.

5.4.5.Libpysal

Libpysal is a spatial analysis library that is particularly useful for Geographical Data Science (GDS) [41] due to its ability to compute spatial weight matrices through the existence of common boundaries. That makes it directly suitable to use with data described by polygons.

5.4.6.Scikit-learn

Scikit-learn is an open source machine learning library for Python, useful for predictive data analysis through its main applications: classification, regression, clustering, dimensionality reduction, model selection and data preprocessing [42].

6. Methodology

This work proposes a data science and deep learning approach. It is divided into 3 main tasks: data acquisition, estimation of indicators and development of an interactive platform.

In fact, the data acquisition includes the process of obtaining all the necessary data for the next steps: socio-economic indicators and satellite images, both representing the NEXUS area.

Estimation of indicators, inspired by Yeh et al. (2020) [11] methodology, describes the entire workflow of processing the Deep Learning model input data, the model training process and the results validation.

Development of an interactive platform discusses the creation of a web page to display the results obtained in this work and allow users to analyze model predictions.

6.1. Data acquisition

In this project there are two types of useful data: numerical data (indicators) and imagery (satellite images). This work uses a dataset of 150 indicators provided by INPE. This section discusses which of those indicators are useful for this project and how they are processed. Furthermore, this section presents the discussion that leads to the decision of creating a selection algorithm to divide the NEXUS area into clusters without overlapping images.

6.1.1. Social indicators

Social indicators are available and accessible thanks to the collaboration effort from INPE, which provides census data for the NEXUS area arranged in CSV format divided by their corresponding census tract and to years 2010 to 2015. The NEXUS project works on the identification of approximately 150 indicators. Therefore, a set of sustainability indexes is available, at Biome and Regional scales, whose territorial unit of analysis is at the municipality level. Besides that, those indicators are obtained by using spatially explicit data, remote sensing and the use of computational models.

This work focuses exclusively on the socioeconomic indicators in this dataset, besides, there is the need of reducing this huge amount of data to only a small number of relevant indicators. To do so, the experiments conducted are based on the Human Development Index (HDI), which is a well recognized criteria for studying and interpreting the socioeconomic development of communities, taking into account three indicators: economic

development, health and education [43]. Besides that, the experiments were also restricted only to data from 2010, for simplification.

The study conducted by Abreu et al. (2011) [44] argues that although HDI can be conducted on any scale, larger population scales mask the real Human Development in an area due to the mix of the reality of different territories and thus, those indicators should be collected on a smaller scale, which can better explain human interactions. Since both IBGE censitary dataset and INPE indicator dataset provide information on the level of census tracts, which is the smallest possible territorial division, this work is able to replicate the method applied by Abreu et al. (2011) [44] and calculate the three criteria established in this method as good approximations for HDI: income, literacy and longevity. Each of those criteria is calculated as follows [43]:

Income defines economic development and, since household income can be a good representation of the distribution of municipal income, for each census tract its per capita income is calculated as the division of the total nominal monthly income of responsible householders by the total resident population in the census tract. Then, income is obtained as in Equation 7.

$$Income = \frac{ln(PC) - ln(min)}{ln(max) - ln(min)} \quad (7)$$

Here, PC stands for the per capita income of a census tract, min = R$8.00 is the reference value for minimum income and max = R$4033.00, for maximum income [45].

Literacy is a proxy indicator for education as this dimension is obtained in HDI as the period in which the population is formed. Equation 8 is used, where p is the rate of alphabetization until 14 years old, and q is the rate of alphabetization after 14 years old.

$$Literacy = \frac{2p + q}{3} \quad (8)$$

Longevity represents the health dimension from HDI and is calculated from life expectancy at birth as shown by Equation 9, where LF stands for life expectancy in years, 85 is the value for maximum age and 25, for minimum age, according to Atlas Brazil [45].

$$Longevity = \frac{LF - 25}{85 - 25} \quad (9)$$

6.1.2. Segmentation of the study area into clusters

In this work, a cluster corresponds to a fraction of land that is described by the value of its social indicators and has a satellite image that represents it, with both being used as inputs to the model training, while, during prediction, the image is the input and the indicator is the output. To analyze the scale that will have to be used in the satellite imagery, it is

possible to compare the Brazilian case to work done by Yeh et al. (2020) [11], which utilizes a scale of 30 meters/pixel in 224 × 224 pixels images, covering approximately 45 km² of area in each image. The granularity of the wealth data is given at village level, whose extension is generally as big as the LANDSAT images used as inputs to the model in the reference study, while Brazilian indicators are provided in the granularity level of census tracts.

To take advantage of ResNet-18's backbone, it is ideal that the input shape of 224 × 224 pixels is maintained to avoid making changes to the network's architecture that can negatively affect the performance, as seen in section 4.5.4.1.

As a consequence of the granularity of data for the African study matching the area covered by an image in most cases, the distribution of clusters can also be at village level. The same cannot be said of the Brazilian case, since the census tracts span through a wide range of sizes throughout the NEXUS area.

In this case, each cluster will be formed by different amounts of tracts and the distribution of clusters to create the database for model training and validation takes additional effort as the collection of satellite images from the Google Earth Engine API must meet the three conditions described in the subtopics below to avoid model bias. Therefore, an analysis of the distribution of census tracts in the NEXUS area is necessary, exploring the extent and arrangement of urban and rural types, in order to determine the regions for obtaining images.

6.1.2.1. The amount of rural and urban images should appear in a meaningful proportion

Imbalanced datasets are a consequence of problems where data appears in an uneven distribution of classes. This condition can degrade model performance since it may not learn enough features from the minority class and thus, loses generalization capabilities.

However, it has been observed that in some domains, where data is naturally imbalanced, the class imbalance is not the only cause for reducing performance and might not impose such a big impact [46]. Besides, the influence of imbalance might start to become more noticeable in more extreme cases - e.g. when the ratio between classes is in the order of 1:1000 or more - and not so much in imbalance ratios ranging from 1:4 up to 1:100 [47][66]

6.1.2.2. Each image should represent only a single type (urban or rural)

To further reduce the impact of imbalance, which is highly expected in the NEXUS area, since most of it is covered by rural areas, it is best that the image depicts only one type of tract. This way, cluster data will show less variance.

6.1.2.3.The same region should not be depicted by more than one image

If multiple regions appear in more than one cluster, the model is exposed to bias by learning specific patterns from the duplicated data and loses generalization capacities to new data [48]. Besides that, there is the risk that correlated data is scattered between dataset folds, which further reduces the variance of the model, which leads to underfitting, and compromises model validation.

6.2.Estimation of indicators

This work proposes to leverage Machine Learning and Deep Learning technologies to extract information from census sectors in the NEXUS area, building on previous work done in Africa by replicating the model structure from Yeh et al. (2020) [11], which estimates well-being indicators from satellite imagery using Deep Learning. There are two models to be trained separately and one that is the combination of both: Nightlight model (NL), Multispectral model (MS) and the combined model (MS+NL).

The pipeline is based on the workflow proposed by that study, being divided into four main steps: Dataset creation, Data preparation, Dataset validation and Target estimation. Figure 7 represents how the pipeline is structured.

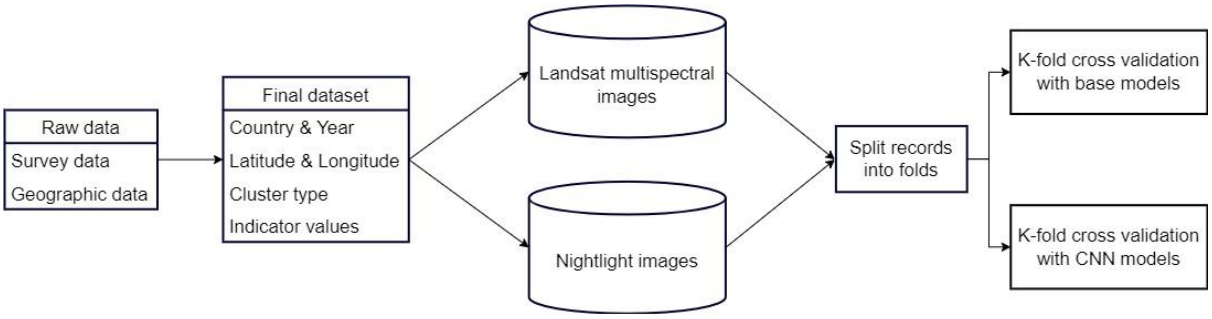


Figure 7 - Deep Learning pipeline.

6.2.1.Dataset creation

The first step of the pipeline is to gather the data to be used as the Deep Learning model input. This section describes the creation process of its components and how they are merged into a single final dataset.

6.2.1.1.Socioeconomic indicators construction

The socioeconomic indicators dataset samples represent clusters, where each is described by the following information: country, year, latitude, longitude, cluster type and indicator values.

The fields country and year are useful to identify the dataset if multiple countries or years combinations are being analyzed. Latitude and longitude are the centroid coordinates

of the cluster, which must be represented in the EPSG:4326 projection (WGS84 - World Geodetic System 1984) [49], because it is the default projection for downloading images with GEE [50]. The cluster type indicates whether the cluster is urban or rural, being useful for future steps in the pipeline, where the models' performance is analyzed over each type. Finally, each indicator value field represents the value of one indicator for the given cluster.

This dataset is the main input for the estimation algorithm, since it contains each cluster centroid coordinate and the year to be considered, which is useful to download the cluster image in the next pipeline step.

In order to construct it, it is necessary to define what are the clusters being used for the experiment and to obtain, clean and preprocess the indicators of each selected cluster, as discussed in section 6.1.

6.2.1.2.Remote sensing data

The satellite images are the remote sensing data used to estimate the indicators with the Deep Learning models. There are a total of nine bands, in which seven of them form a multispectral landsat image, used to train MS and MS+NL models, and the remaining two refer to nightlight images, for training NL and MS+NL models.

The seven bands of the multispectral landsat images are Red, Green, Blue, NIR, SWIR1, SWIR2 and TEMP1 [51]. A truecolor image (RGB) is represented by the bands Red, Green and Blue, where each is a component for each individual pixel in the image matrix. Visible light electromagnetic spectrum wavelength is in a range between 380 nm and 700 nm, while NIR, the Near-Infrared band, lies between 700 nm to 5000 nm [52]. Some of NIR subdivisions are SWIR1 and 2, which are Short-Wave Infrared bands defined between 1000 nm and 5000 nm. TEMP1 is a thermal band useful for measuring the surface temperature, allowing to differentiate clouds from soils, for instance [53].

NIR and, thereafter, SWIR wavelengths are absorbed by water thus it allows mapping water bodies, discriminating among different types of rock formations, soils and crops [54][55]. There are applications that combine these bands with Red as an alternative to RGB images for specific purposes, such as monitoring drainage and soil patterns and differentiating between soil and water [54][55]. Therefore, understanding the importance of each of the multispectral bands, it makes sense they are all used as an input by the Deep Learning models to extract relevant features of each cluster's geography.

The eighth band, DMSP (Defense Meteorological Satellite Program), refers to data collected by DMSP-OLS, the Defense Meteorological Satellite Program - Operation Linescan System, which produced eight global datasets of nighttime light images representing years from 1996 to 2012 [56]. This data is used as the nightlight image input to the models if the cluster is being analyzed in that specified time interval.

The ninth band is VIIRS (Visible Infrared Imaging Radiometer Suite), which is a high quality collection of global low-light images, presenting several technical improvements over DMSP-OLS data, such as in-flight calibration and wider dynamic range, however it is only available starting from 2012 [57]. Like the DMSP band, VIIRS can also be used by the model as input representing night light images, but only for clusters analyzed from 2012 onwards.

Note that a sample's year is either exclusively within the range of either DMSP or VIIRS, thus the band in which it is contained is used normally while the pixels of the other band are all set to zero, in order to not influence the training process by ensuring that the first convolutional layer is not trained with that missing data, not updating its weights during back-propagation [11]. This means that the band composed of 0 valued pixels is not considered by the model.

Nightlight data is important because it can present relevant features for detecting electric lighting present on the Earth's surface, human settlements and clouds, measuring the brightness of artificial lighting and mapping city boundaries [56][57].

Also based on the reference methodology proposed by Yeh et al. (2020) [11], images are a 3-year median composite created by taking the median of each cloud free pixel available during a period of 3 consecutive years. This is useful to gather clear satellite imagery, reducing cloud obstruction from imperfections in the cloud mask, and to minimize seasonal distortion and short-run variations, considering the target indicators being estimated tend to evolve slowly over time, just like the wealth index in the reference article.

Images are downloaded through GEE being centered on the given latitude and longitude using the socioeconomic indicators dataset created in the previous step, Socioeconomic indicators construction, with a resolution of 255 × 255 pixels and, then, center-cropped to 224 × 224. Each image band standardized to have mean equal to 0 and standard deviation of 1 across the entire dataset.

While images are downloaded, they are already merged with the socioeconomic indicators dataset to form a single final dataset, which contains all the information necessary to train the deep learning: country, year, cluster image of resolution 224 × 224 pixels and scale of 30 meters per pixel, cluster centroid coordinates, cluster type and the target values, i.e. the value of each indicator to be estimated by the Deep Learning models.

6.2.2.Models

This pipeline is responsible for training three different experiments: a model considering only nightlight images (NL), another only with multispectral images (MS) and, finally, a combination of both (MS+NL) as a third model. This final CNN model is shown in Figure 8, describing the high-level architecture of the neural network adopted.

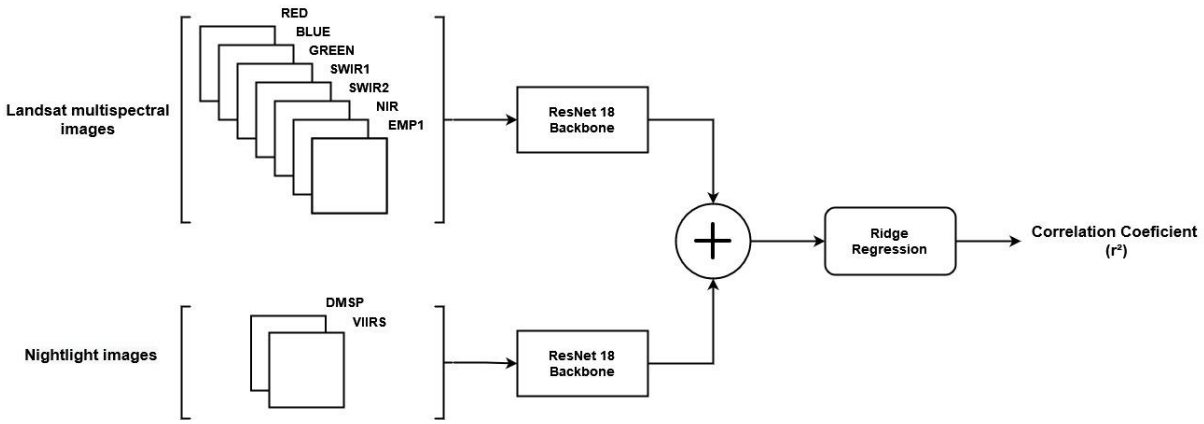


Figure 8 - MS+NL model high-level architecture.

The training procedure of NL and MS models is identical, taking NL as example: the final NL model is composed of two different models, trained separately. The first model is called ResNet-NL, while the second is Ridge-NL. Thus, the same goes for ResNet-MS and Ridge-MS, in the case of the MS model. The definitive models of this work are Ridge-NL, Ridge-MS and MS+NL.

Note that every model is only able to predict a single target value in the configuration being used, thus it is necessary to train versions of the models for each indicator. Given that, Table 1 represents all 9 models that will be trained if the experiment considers the 3 indicators (income, literacy and longevity) and the 3 definitive models (Ridge-NL, Ridge-MS and MS+NL). The following sections describe the processes of training and evaluating the models of a single indicator, since they are the same regardless of the target

	Model type		
Indicator	NL	MS	MS+NL
Income	Ridge-NL_income	Ridge-MS_income	MS+NL_income
Literacy	Ridge-NL_literacy	Ridge-MS_literacy	MS+NL_literacy
Longevity	Ridge-NL_longecity	Ridge-MS_longecity	MS+NL_longecity

Table 1 - Models to be trained if all indicators are considered.

6.2.2.1.ResNet-MS and ResNet-NL models

The ResNet-MS model uses a ResNet-18 v2 architecture whose first convolutional layer is modified to take the 7 MS bands as input, while its final layer is modified to output a single scalar value. The same goes for the ResNet-18 v2 used for the ResNet-NL model, however, since it presents a single band (either DMSP or VIIRS), there is no need to modify the input layer that accepts up to 3 bands, i.e. input shape of 224 × 224 × 3.

Knowing this modifications, it is important to summarize relevant shapes of the models: the input shape of the ResNet-MS model is 224 × 224 × 7, while the input shape of

the ResNet-NL model is $224 \times 224 \times 2$ (with one of the bands set to all zeros). The final hidden layer of each of those models is a fully connected layer that generates 512 features and the output layer transforms such values into a single scalar. This information is presented on Figure 9.

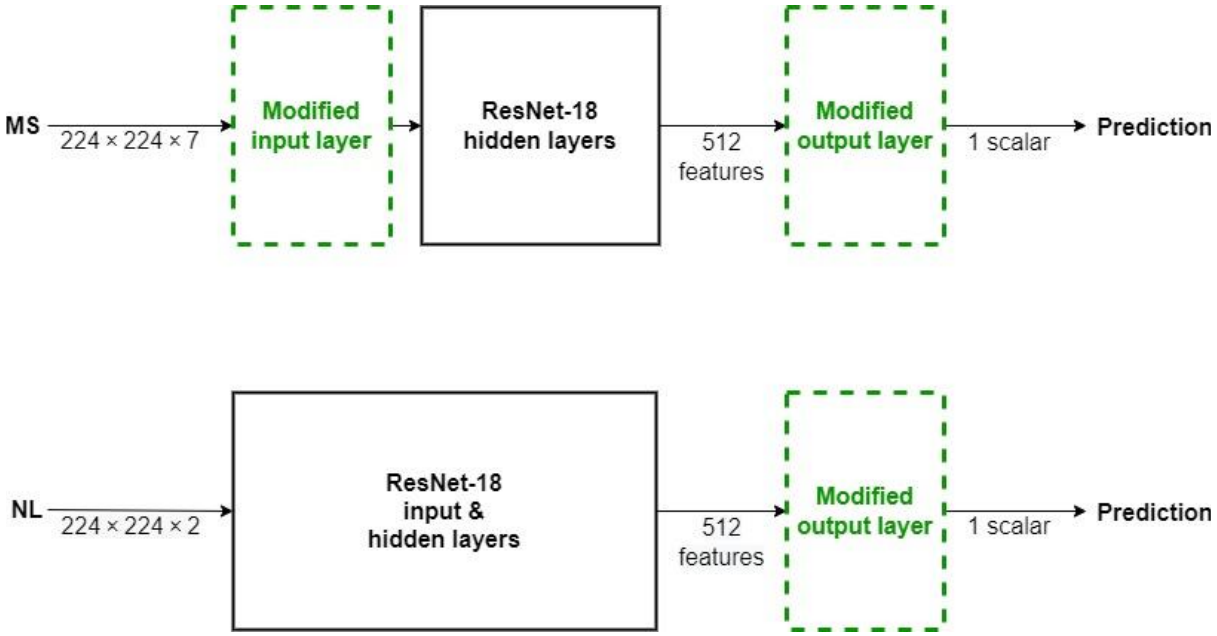


Figure 9 - Layer high-level representation of ResNet-MS and ResNet-NL CNN models.

These models are intermediary and their main purpose is to generate all the remaining ones: Ridge-MS, Ridge-NL and MS+NL.

6.2.2.2. Ridge-MS and Ridge-NL models

The Ridge-MS and Ridge-NL models are obtained from its ResNet respective models. Taking MS as an example, after ResNet-MS is trained, its modified output layer is discarded and the remaining layers, called ResNet-MS backbone (input layer and hidden layers), have their weights frozen. The 512 output features of that model already trained and fixed are used as input to a Ridge Regression model, which is trained to predict the target indicator. The same process is used to generate Ridge-NL, using the ResNet-NL backbone.

In order to run predictions with new data after training is complete, it will be necessary to input the images to the ResNet backbone and, then, its output features must be fed to the Ridge model.

Predicting over a large group of input features which might present multicollinearity, since they are all outcomes from an image processed by a CNN, may result in a high variance model. The Ridge Regression model is an appropriate solution because it has the characteristic of a good bias-variance trade-off, adding an acceptable amount of bias over the predictions to reduce variance [28].

Another reason for combining ResNets with Ridge Regressors is its efficiency, since it is extremely quick for training and estimating values, and shows great performance results, as demonstrated in previous Computer Vision works [58][59].

6.2.2.3.Combined Model

The combined model, MS+NL, joins the ResNet-MS backbone to the ResNet-NL backbone, which were trained separately before, by concatenating their final fully connected layers, i.e. the last hidden layer of the CNN, and training a Ridge Regression model on top of it, as shown in Figure 10. This means that the Ridge Regression inputs are the 1024 features (512 features of each model).

This model is expected to have a better predictive performance since it correlates the information contained in MS and NL bands and thus, the regressor has more descriptive features to estimate the final value.

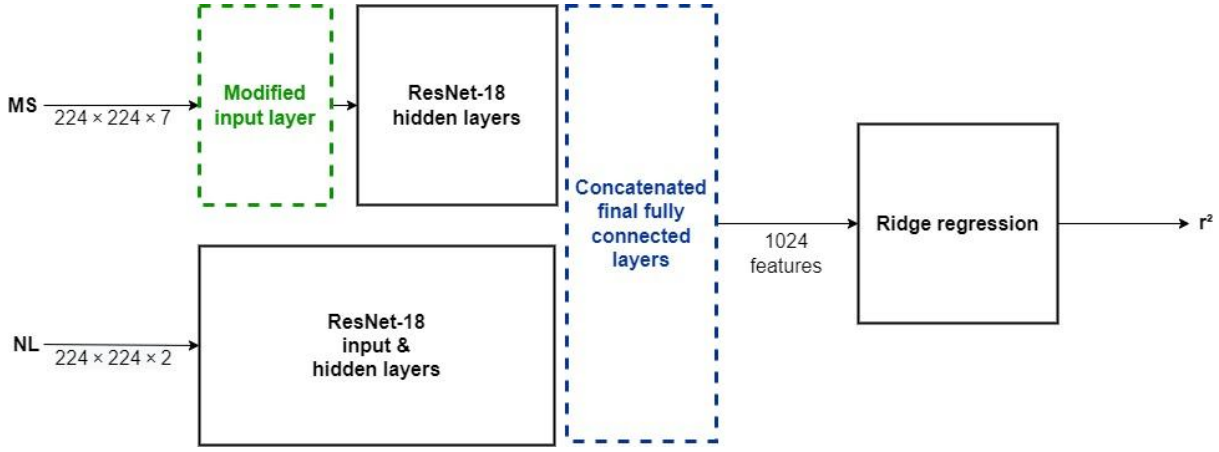


Figure 10 - Layer high-level representation of MS+NL model.

6.2.3.Data preprocessing

The data preparation is the next pipeline step in which the dataset is preprocessed by splitting data into folds in order to prepare it for the models' training.

The dataset is randomly and equally split into 5 folds: A, B, C, D and E. With those groups, 5 different sets of train, validation and test splits are created, where each set is labeled by the name of its test fold, according to Table 2.

	Split		
Set label	Train	Validation	Test
A	C, D, E	B	A
B	A, D, E	C	B
C	A, B, E	D	C
D	A, B, C	E	D
E	B, C, D	A	E

Table 2 - 5-fold Cross-Validation sets.

Although random, this selection must respect a criterion that two sufficiently close clusters must be allocated to different folds, which is useful to guarantee that there is no overlap of areas in the same fold. Even though this concern has already been addressed in section 6.1.2 when choosing clusters for training, this step is interesting to guarantee such a condition, in order to obtain generalizable models capable of estimating the target values in geographic regions not yet seen.

The ResNet-NL and ResNet-MS models are trained on a similar technique as K-Fold Cross-Validation, explored in section 4.7, using the train and validation splits from all sets from Table 1. The test split must exist and not be used in the training of ResNet models because it is later used to validate the MS+NL model. If this split is used to train the backbones and then for validation of the complete model, the results would not be consistent, as they would be biased.

Once the ResNet backbones are trained, the Ridge Regressions of the 3 models (NL, MS, MS+NL) are trained with 5-fold cross-validation, using the same 5 sets and training with the train and validation splits and validating with the test split.

6.2.4.Target estimation

With the dataset created and preprocessed into training, validation and test splits, it is possible to train the ResNet backbones and the Ridge regressions to estimate the target values, i.e. the proposed indicators.

6.2.4.1.Parameter initialization

The initialization methodology for the weights on the first layer of ResNet-MS depends on the bands: RGB bands have its weights initialized as the pre-trained weights on ImageNet, as a transfer learning approach since it is possible to take advantage of results already obtained with the original ResNet-18 architecture, while the remaining bands are

initialized as the mean of the RGB channels weights. Those weights are re-scaled by 3/7, where 7 is the total of bands. The same initialization for the ResNet-NL model uses He initialization (Kaiming initialization), which is an initialization method that considers the non-linearity of the model activation functions [60].

On the other hand, final layer weights of both models are initialized as random values from a standard normal distribution. The hidden layer weights are exactly the same as the original ResNet-18 v2 pre-trained with ImageNet.

6.2.4.2.Model training

Data augmentation is used in order to prevent overfitting: images are augmented by random horizontal and vertical flips and the brightness and contrast of MS bands are randomly adjusted, using changes of 0,5 standard deviation change and up to 25%, respectively.

The CNN models are trained with the Adam optimizer and a RMSE (Root Mean Squared Error) loss function. A batch size of 64 and the learning rate is decayed by a factor of 0.96 after each epoch. The default epochs are 200 for the trainings and each set used a different combination of hyperparameters, such as learning rate and the regularization penalty for the convolutional and fully connected layers, an advantage that the cross-validation brings to the experiments. Table 3 shows the different combinations used in this project for each set.

Set label	Learning Rate	FC Regularization	Convolutional Regularization
A	0.001	0.01	0.01
B	0.001	0.1	0.1
C	0.0001	1.0	1.0
D	0.0001	0.001	1.0
E	0.0001	0.001	0.001

Table 3 - Hyperparameters combinations for each set.

The ResNet-NL model is trained separately 5 times, one with each set from A, B, C, D and E, with 3 folds in the training split, 1 in the validation split and 1 in the test split, whose creation is described in section 6.2.3. The same goes for the ResNet-MS model.

Then, MS+NL combines both ResNet-NL and ResNet-MS backbones' final fully connected layers, freezes those model weights and trains a Ridge Regression over it. It is trained with 5-fold cross-validation making use of the 5 sets, where the model is trained on 4 folds and tested on the fifth. From the 5 model versions, the one with the highest coefficient

of determination (r^2) on the validation set across all epochs is used as the final MS+NL model for comparison. The same approach is applied to Ridge-MS and to Ridge-NL.

6.2.5. Performance comparison

Baseline models simpler than the Deep Learning ones, such as KNNs and Ridge Regressors, are trained for a multitude of experiments over the bands, in order to compare results and establish a minimum expected result.

The performances of those baseline models are compared to the 3 Deep Learning models before mentioned in this section. The comparison considers the r^2, R^2, MSE and rank.

6.3. Towards an interactive platform

The platform idea is inspired by the AdaptaBrasil MCTI system (https://sistema.adaptabrasil.mcti.gov.br/), which is an information system that allows the analysis of indices and indicators of risk of impacts of climate change in Brazil. Such a platform is managed by the Brazilian ministry of science, technology and innovations (Ministério da Ciência, Tecnologia e Inovações - MCTI) [61].

AdaptaBrasil MCTI main functionalities regarding data visualization are the possibility to display indicators and area names over the Brazilian map in different resolutions, i.e. geopolitical granularities, such as 5 geopolitical regions (North, Northeast, Central-West, Southeast and South), 27 Federative Units (26 states and a Federal District) and 5570 municipalities. Although it contains only Brazilian data, the displayed map allows zooming in and out and traveling through all over the world.

When an area of interest and an indicator are selected, AdaptaBrasil MCTI presents quantitative and qualitative information about it, for instance: an explanation of the indice and how it is calculated, a risk gradation and influencing factors. It also contains graphs for the analysis of the data distribution along the area, pessimist and optimistic projections of those values for future years and allows users to download the selected information as a images or text files, like Comma Separated Values (CSV) and JavaScript Object Notation (JSON).

This work aims to reproduce some of those functionalities and adapt others to achieve its main goals of sharing the results obtained with the model by creating an interactive web platform that allows users to visualize indicators of arbitrary regions for different years in real time and request new predictions to the model. It is expected that users are able to navigate through the Brazilian map and select regions of interest for its predicted indicators to be displayed, compare the model results with the real data used in the model training and download the predictions.

6.3.1.Functional Requirements

The functional requirements regards sharing and visualizing results obtained from the model trained to help users to comprehend the project goals and achievements and to assist new researchers to make use of this data. The following sections specify the platform functional requirements (FR).

6.3.1.1.Functional Requirements 1: Interactive map for navigation and visualization

The platform must display a map of Brazil, displaying the frontiers of each geospatial area for a given resolution (Region, Federative Unit or Municipality). It must be possible to visualize each area name by hovering the mouse over it and clicking on it for selection in order to display its information previously calculated by the model.

The user has to be able to zoom in and out and navigate through the world map, select the visualization style (Cartographic map or Satellite) and control the opacity of data being displayed over it.

6.3.1.2.FR2: Menu for visualization configurations and model requests

The platform must contain a menu for determining the predicted indicators to be displayed, similarly to a form: presenting text fields and dropdown lists for selecting prediction parameters (year of prediction, prediction resolution, indicator do be predicted). It must also allow the user to enter in the area of interest selection mode, where the map will stop displaying information of clicked areas and start to simply select the areas to be predicted.

Note that if the data is not ready to be displayed, i.e. preloaded in the platform, it will use the desired information in the form to request the model to predict those values and, then, display it.

6.3.1.3.FR3: Selection of prediction resolution and areas of interest

Indicator predictions are calculated over a cluster of area approximately equal to 45 km² and, by calculating the mean of all cluster indicators within a region, the indicator of that region is represented.

The selection of the areas of interest is one of the steps in the prediction configuration form and happens over the map displayed in the platform. The system has to allow users to configure what is the geospatial resolution of the predictions to be selected, the options are: 5 Regions, 27 Federative Units and 5570 Municipalities. The map shifts the displayed frontiers to the ones that define the configured resolution and the user is able to pick areas by clicking over them.

After the selection is finished and the data is displayed, it must be possible to change the resolution of visualization, still for the same options:: Regions, Federative Units and Municipalities.

6.3.1.4.FR4: Selection of year of prediction

Indicator predictions are calculated with clusters, which are satellite images taken in a specific year (or an average across adjacent dates to reduce noise and obstructions).

The system must allow users to select the year of the prediction to be displayed, making use of the temporal coverage capacity of the model. Some preloaded years are already going to be available in the platform memory to show the results instantly, but any other year may be selected, requiring the system to look for the information in a cache database or requesting the model to predict with clusters in that time stamp.

6.3.1.5.FR5: Selection of indicator to be predicted

The model is trained for predicting three indicators: income, literacy and longevity. The platform has to allow users to select which of those three is going to be displayed. It is possible to display just one at a time, to make the visualization clearer and more organized.

6.3.1.6.FR6: Comparison of the real indicator values with those predicted

The model is trained with real indicators from 2010 for the census tracts within the NEXUS area. The platform must enable users to toggle the exhibition of a layer over the map that displays the real indicator selected for each census tract. The opacity of the layer may be altered. This allows the comparison of the predicted results requested by the user to the real values.

6.3.1.7.FR7: Present graphs about the selected predictions

The system must contain a section for visualizing graphs regarding the predictions on the map. When clicking on a specific button in the menu, it is possible to access such a section where a histogram describes the indicator distribution.

6.3.1.8.FR8: Download selected predictions

The platform must allow users to download a CSV of the displayed information, containing the coordinates of each predicted cluster and the predicted indicator. This option should be displayed as a button in the same form to request a new prediction. This helps researchers to reproduce the work and consult and use the shared data for new studies.

6.3.2.Non-functional requirements

The platform is expected to be consulted by common users and researchers whenever they are interested, either to visualize data or download it for some related work. Besides that, this is the first version of development, with a reduced scope, probably requiring new features and adaptations in the future and the lifetime of the system is expected to be long. Given those arguments, the following sections discuss the main non-functional requirements (NFR) to meet such expectations.

6.3.2.1.Non-functional requirement 1: Medium availability

It is expected that the platform is available to be consulted at any time, since common users and researchers may need to access the data from all over the world in different time zones for different purposes. Besides that, their access is expected to be as a self service, not needing to contact any of the developers or administrators to use the platform. However, as it is a niche platform for a specific audience and which will not be disclosed, it is not expected to have many constant users, being idle for a long time.

Nevertheless, availability is an important non-functional requirement, allowing the system usage most of the time. Keeping a software system always available might be expensive and, as discussed, the platform does not need to be in such condition without exception, therefore guaranteeing medium availability is enough.

6.3.2.2.NFR 2: Medium scalability

As aforementioned, the system usage is not expected to be constant and the workloads may vary a lot: whenever researchers intend to use it, it might face high TPS (Transactions per Second), and most of the time it will be idle or underutilized. That said, it is important that the system is generally designed for low or no TPS situations, reducing resource usage and, consequently, keeping low costs, and, when necessary, it is able to scale horizontally to meet the demand due to a growing amount of requests.

Thus, medium scalability is relevant for the platform, considering there will be variations from no usage to a medium scale of requests.

6.3.2.3.NFR3: High maintainability and extensibility

The system is developed as a graduation work, keeping a small scope in a short period of time, however it is inserted in the context of a large research group project, Parsec. Thereafter, it is expected that it enables future growth through new requirements and features, ensures low mean time to repair (MTTR) to solve bugs or outages quickly, which can be achieved by allowing to repair or replace faulty or worn-out components without influencing other working elements.

Besides that, it is necessary to maximize the platform's useful lifetime, efficiency and reliability, so that it can be used consistently for a long time without the need for changes. All that said, high maintainability and extensibility are the most important non-functional requirements of the system.

6.3.3. Architecture discussion

Given the non-functional requirements discussed earlier, the application will benefit from an architectural design based on distributed system architecture, since, as presented in section 4.12, this type of architecture is scalable and performance driven, allows decoupling of modules and software components helping maintainability and extensibility and ensures cost-effectiveness.

This is also a good decision taking into account that cloud computing resources are appropriate for decentralized systems and the development group has credits provided by the PARSEC project to use in AWS, which is specialized in cloud computing services.

benefitting from AWS services appropriate for this usage. Also, the PARSEC project makes available credits on AWS for the use of its computing resources, which is yet another reason for developing the system in this environment.

Regarding NFR1 for availability, AWS AZs (Availability Zones) are fully isolated partitions of infrastructure within a region, such as Oregon and Northern Virginia regions, that contain 4 and 6 AZs respectively [62]. It is possible to partition applications across multiple AZs in the same region to better isolate any issues and achieve high availability [32].

Considering NFR2 about scalability, AWS allows provisioning the amount of resources that is actually needed and instantly scaling up or down along with the needs of the system [32], which can be achieved automatically with Auto Scaling [63]. This reduces costs and improves the ability to meet usage demands flexibly.

About NFR3: High maintainability and extensibility, the platform being designed as a distributed system guarantees the modularization of its software components. This can be achieved with AWS serverless applications to deploy each component separately, such as AWS Lambda [64], an event-driven compute service and AWS Fargate [65], a compute engine that works with containerized solutions without needing to manage infrastructure.

Those serverless resources are offered by AWS without managing servers for executing code, managing data and integrating applications. They feature automatic scaling, built-in high availability and a billing model to increase agility and optimize costs which allows payment only when resources are actually used [33].

7. Development

The development of the methodology discussed in section 6 is presented in this one. The entire process is described considering the technologies employed, files used and results obtained. The development model workflow (Figure 11) presents 13 items that refer either to a file, generally Jupyter Notebooks (.ipynb files) or to some tool, such as QGIS.

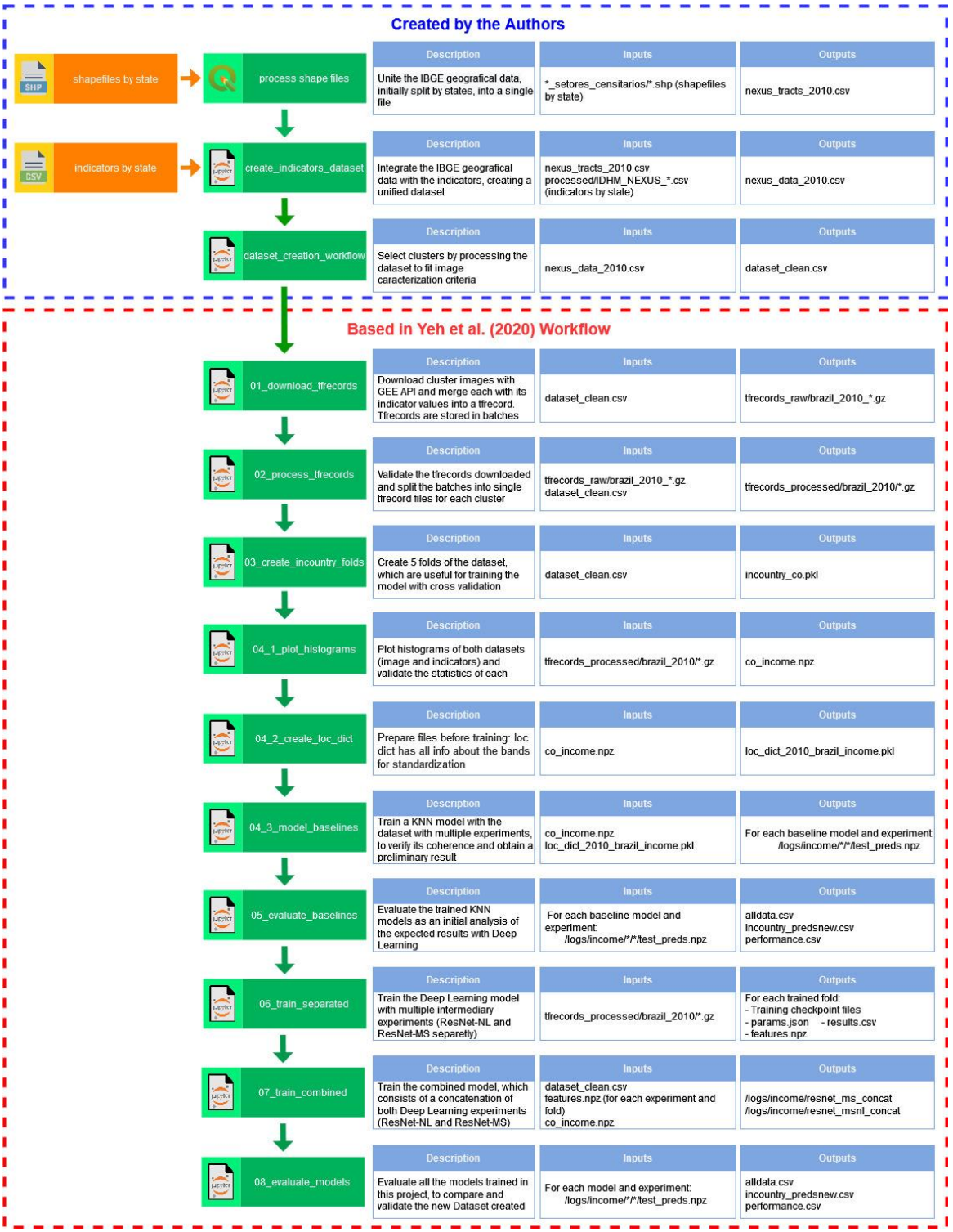


Figure 11 - Development model workflow.

7.1.Data acquisition

This section describes how all data necessary for starting the training pipeline is collected or generated: socioeconomic indicators and the selection of relevant clusters, which depends on processing based on geometric information from the Brazilian territory.

The steps described are the first three of the Model Workflow (Figure 11), in the "Created by the Authors" branch. All files are executed in Google Colab free tier environment.

7.1.1.Geometric information acquisition

Brazil's census conductor and public institute for federal administration, IBGE, publishes up-to-date information depicting the country's administrative divisions and also shapefiles (.shp files) describing the polygons that define the entire census tracts mesh represented by the SIRGAS 2000 coordinate reference system, the Brazilian Geodetic System [66]. For the year of 2010 - which belongs to the time window of the social indicators present in the project - the data is split between each of the 26 states plus the Federal District.

This database is loaded into QGIS as multiple vector layers that are then merged into a single one that represents all of Brazil. Since the process of merging can cause collision between polygons, the software's tools for geometry correction are also needed. Also, the coordinate reference system is reprojected from SIRGAS 2000 to WGS 84/Pseudo-Mercator (EPSG:3857), which is the same used by Google Maps [67].

The next step is keeping only the tracts that account for the NEXUS area, which can be easily done since the NEXUS project provides a list of all municipalities contained within, represented by their municipality code which is the field nexus_CD_GEOCMU of each IDHM_NEXUS CSV file, and the same code is given as metadata on IBGE's files (field CD_GEOCODM).

To acquire the geometric information, the polygons of the remaining tracts are converted to the Well-Known Text (WKT) representation format. Finally, the tract type (urban or rural), the WKT, municipality and tract codes are exported to a CSV file, nexus_tracts_2010.csv containing a total of 114244 census tracts. Its relevant fields are described by Table 4.

Field	Description	Example
CD_GEOCODI	Census sector geocode (15 digits)	270010205000001
CD_GEOCODM	Municipality geocode (7 digits)	2700102
NM_MUNICIP	Municipality name	ÁGUA BRANCA
NM_UF	Acronym of the UF to which the municipality belongs	AL
TYPE	Census tract general classification	“URBAN” or “RURAL”
WKT	Polygon of the census tract referenced to WGS 84 / Pseudo-Mercator	MultiPolygon (((-4223968.81988941 -1036056.26048122, …)))

Table 4 - Description of nexus_tracts_2010.csv fields.

7.1.2.Socioeconomic indicators acquisition

The database of socioeconomic and environmental indicators provided by the NEXUS project is reduced to only the three defined criteria: longevity, literacy and income, as discussed in the Methodology. The missing or incorrect values are deleted and the remaining are all max-min normalized to a range from 0 to 1, making sure each feature is equally important and transforming data to be an appropriate input to the Deep Learning models. This information is given in the granularity of census tracts, being the smallest possible territorial division.

In addition, IBGE provides the database of geographic information and classifications of the Brazilian mesh of sectors, which has already been processed in the previous topic to present only NEXUS census tracts information.

Both of those datasets were combined by sector code, making a single CSV file containing the indicators of each census tract along with its respective geometric information, called nexus_data_2010.csv. The resulting file presents 110214 samples with at all indicators available, therefore there are 4030 census sectors without information for the analysis. Table 5 describes the fields of this file.

Field	Description	Example
Year	The year of the given data. It is always 2010 in this experiment	2010
Cod_state	State geocode (2 digits)	11
Cod_municipality	Municipality geocode (7 digits)	1100304
Cod_sector	Census tract geocode (7 digits)	110030405000001
Income	Income indicator value	0.800035
Literacy	Literacy indicator value	0.800000
Longevity	Longevity indicator value	0.996329
Type	Census tract general classification	“URBAN” or “RURAL”
WKT	Polygon of the census tract referenced to WGS 84 / Pseudo-Mercator	MultiPolygon (((-6694800.91148638 -1429885.364, …)))

Table 5 - Description of nexus_data_2010.csv fields.

7.1.2.1.Analysis of the socio-economic indicators distribution

The indicators’ frequency distribution may be analyzed in order to understand how appropriate the data is for training a neural network, i.e. whether there is data bias, for instance. The histograms of the indicators are displayed by Figure 12.

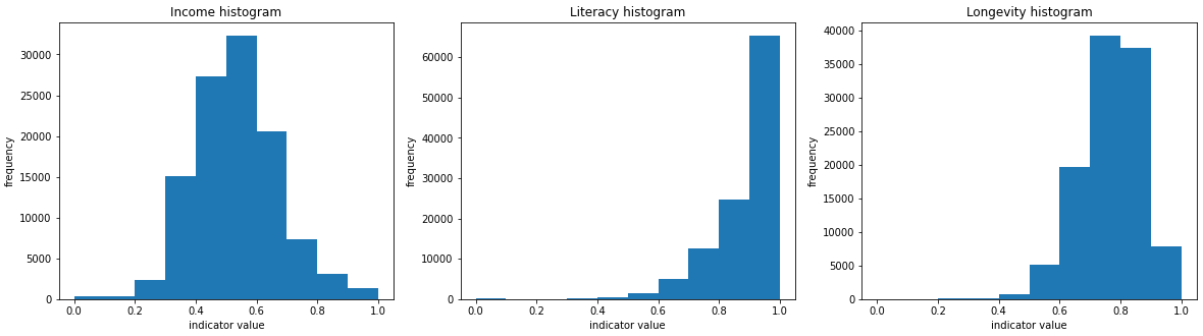


Figure 12 - Indicator histograms.

Note that the income graph is almost a symmetric histogram, very close to a normal curve (Figure 13), while the other two indicators are skewed left histograms. The second, third and fourth quartiles and the means of the targets are represented in Table 6 in order to explore quantitative values about these distributions.

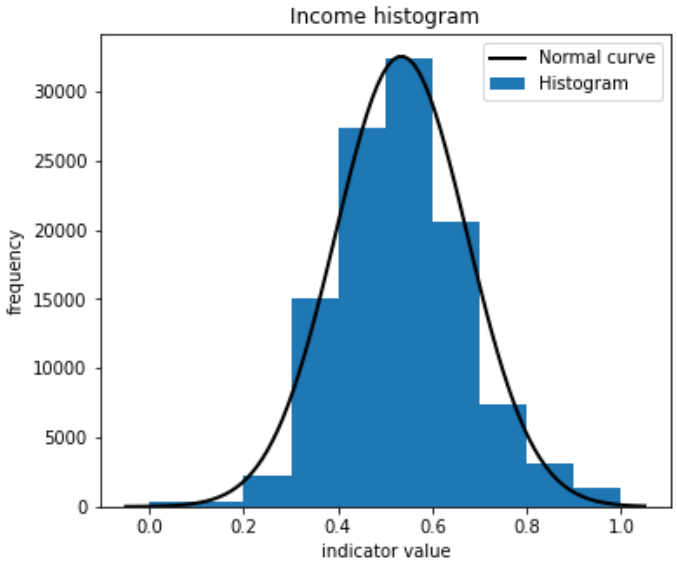


Figure 13 - Income histogram compared to a normal distribution.

	Quartiles			Mean
Indicator	25%	50%	75%	
Income	0.435424	0.531983	0.616460	0.534060
Literacy	0.836617	0.925862	0.967742	0.886870
Longevity	0.716667	0.766667	0.833333	0.765886

Table 6 - Statistic parameters of the target values.

Given this information, models trained with income might show greater results when compared to models trained to predict the other two indicators, taking into account that income distribution is closer to a standardized curve, which are widely used with neural networks to achieve good results, as discussed in section 4.8.

Knowing this, the intermediate results presented in the following sections are focused on the income experiments and the best models of each socioeconomic indicator will be compared to evaluate this assumption.

7.1.2.2.Analysis of the socio-economic indicators correlation

It is possible to visualize the distribution of indicators along the states of the NEXUS area, indicating the correlation between said indicators and demonstrating the relevance of the adopted granularity, i.e. a value for each census sector. Figure 14 exemplifies this by presenting indicator values throughout the state of Mato Grosso do Sul (MS).

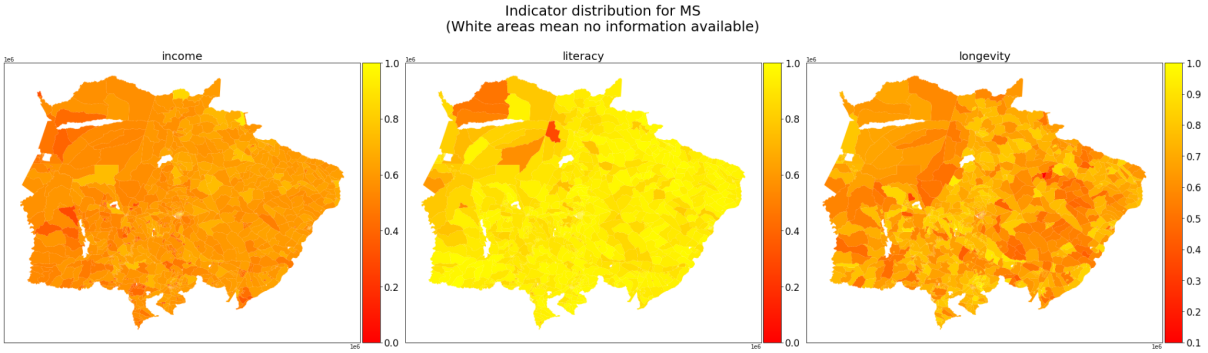


Figure 14 - Distribution of the HDI indicators along Mato Grosso do Sul.

From those graphs, it is possible to infer that in the same state, there are very different regions, as in the case of literacy, which is much lower in the northeast of MS compared to the other states. At the same time, the other indicators do not show the same behavior, which indicates that there is no direct correlation between them. Besides that, it is possible to see the missing data in some of the sectors, represented in white.

Figure 15 is a heatmap of the Pearson correlation coefficients between the indicators, two by two. The highest correlation presented is between income and literacy with a value of 0.68, while the remaining are much smaller. Given this result, it is understood that the features are not completely correlated and it still makes sense to create prediction models for each of them.

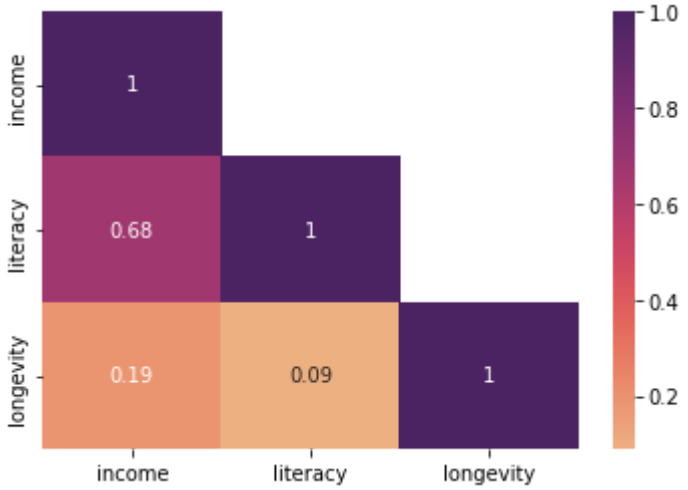


Figure 15 - Pearson correlation coefficients between target values, two by two.

7.1.3.Satellite imagery acquisition

In order to collect the satellite images, it is essential to define what are the clusters to be represented by the images. The criteria for this selection are discussed in section 6.1.2, focusing on avoiding image overlaps, reducing the chances of overfitting of the models to be trained.

Besides that, the selection must maintain a total number of images similar to that used by Yeh et al. (2020) [11] for two reasons. The first is to present the same statistical representativeness of the data splits while keeping the number of folds (k) during training. In other words, the total of folds is chosen considering that each split is large enough to be statistically representative of the entire dataset. When using the same approach of the reference article, it is important to try to utilize splits as representative as those used in that study for the same value of k, therefore, the total of samples in the dataset should be similar, i.e. around 19669 samples.

Another reason is to avoid overspending on image storage. When the dataset satellite images are downloaded, their storage can be costly and directly depend on the total memory used, so using an excessive number of images can greatly increase expenses.

7.1.3.1.Census tracts study

The study of the Brazilian census sectors is an important step to the project, in order to determine the clusters to generate the dataset of images and indicators that will be of use. Considering the criterias for the selection of those clusters, understanding the distribution of urban and rural census tracts throughout the NEXUS area may be useful to propose a selection methodology.

The file nexus_data_2010.csv generated in earlier sections contains the polygons and types of each tract, thus it is used as the basis to run analysis regarding the area and distribution of them. Figure 16 represents the distribution between urban and rural sector areas along the NEXUS area and also an area histogram. From the 110214 filtered samples, 75792 are urban tracts, representing an area of 38.9 thousand km² tracts, and the remaining 34422 are rural tracts, which is a total of 3.3 million km².

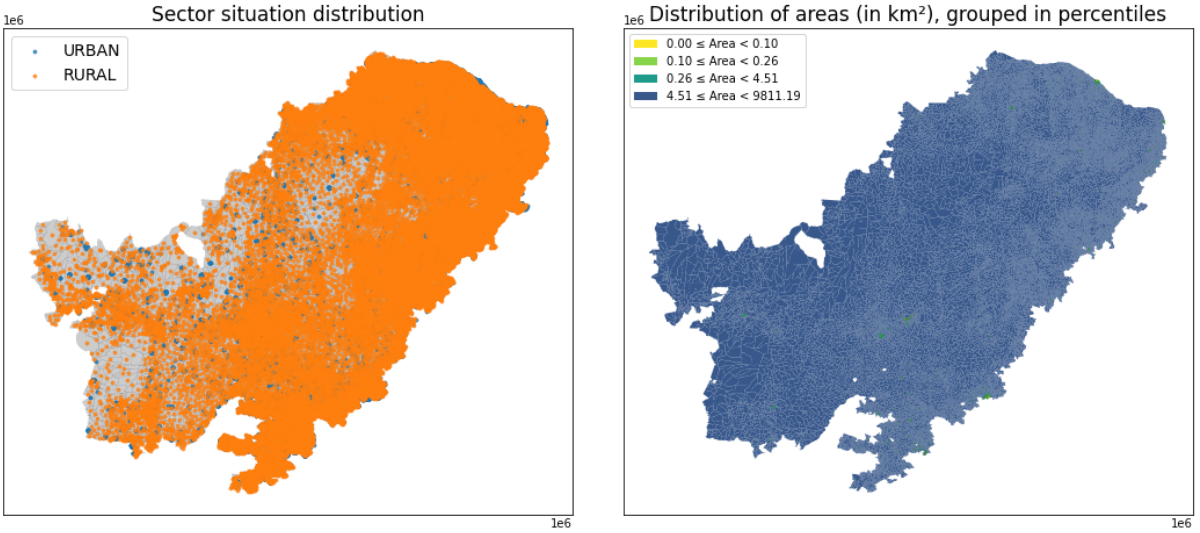


Figure 16 - NEXUS area census sectors. The left image shows the urban census tracts in blue and the rural ones in orange. On the right, there is a representation of the distribution of census tract areas, grouped in quartiles.

Although urban sectors represent 69% of the dataset samples, in the analysis figure the presence of orange regions is much more noticeable, i.e. rural sectors, meaning the latter are much more distributed throughout the NEXUS area, while urban sectors tend to be more concentrated. Besides that, the rural area is 98% of the entire NEXUS territory, which is reflected in the size of the sectors' areas, which are mostly between the highest quartile.

The distribution of urban and rural tracts can be better evaluated in Figure 17, which shows only the map of a state contained in the NEXUS area, Maranhão (MA). On the left image, it is clear that urban tracts are in great concentration in very small regions and rural sectors occupy the entire extension of the territory, presenting a much wider area. On the right image, there are some lighter regions, i.e. first quarters, indicating where some of these urban concentrations are located.

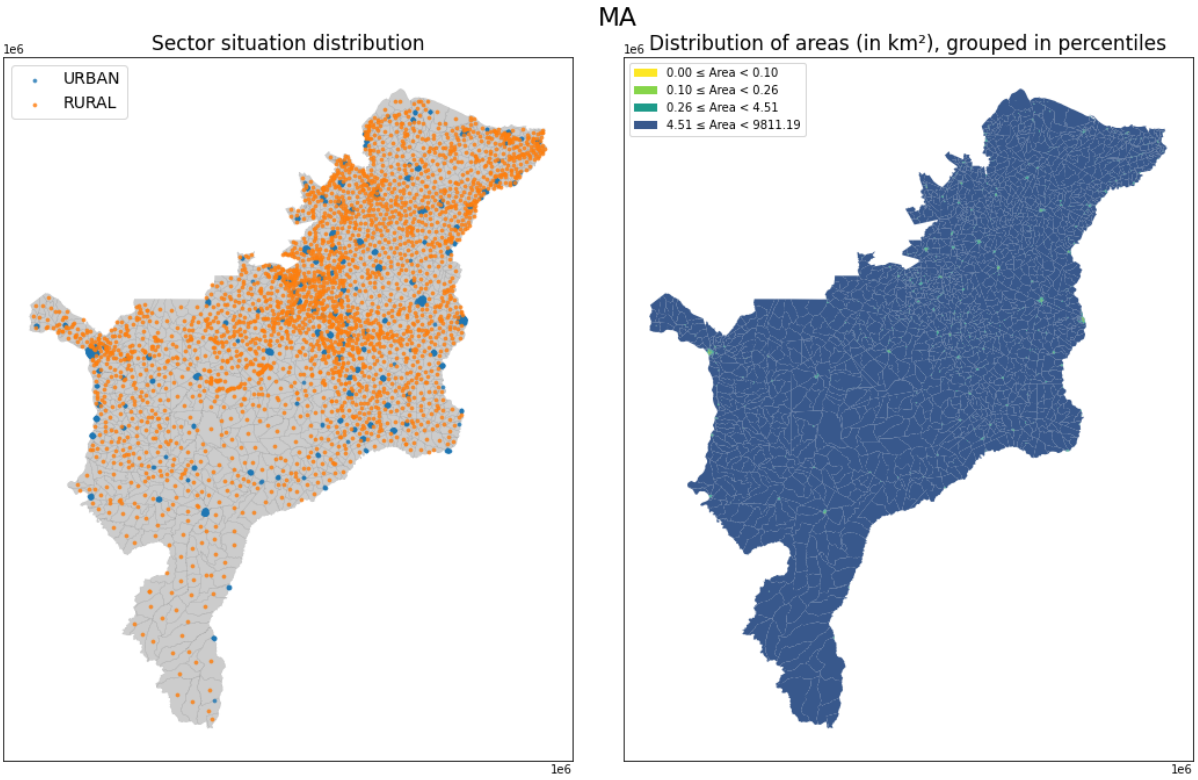


Figure 17 - MA census tracts distribution and areas.

It is important to analyze more deeply the statistical parameters of the tract areas in order to understand the expressiveness of the presented values. Figure 18 displays a boxplot of the tracts within NEXUS territory and another without the outliers, note that many of these extremely large areas are not common. The greater number of urban interferes in the analysis of quartiles disregarding outliers, which are better explored by plotting types separately on Figure 19.

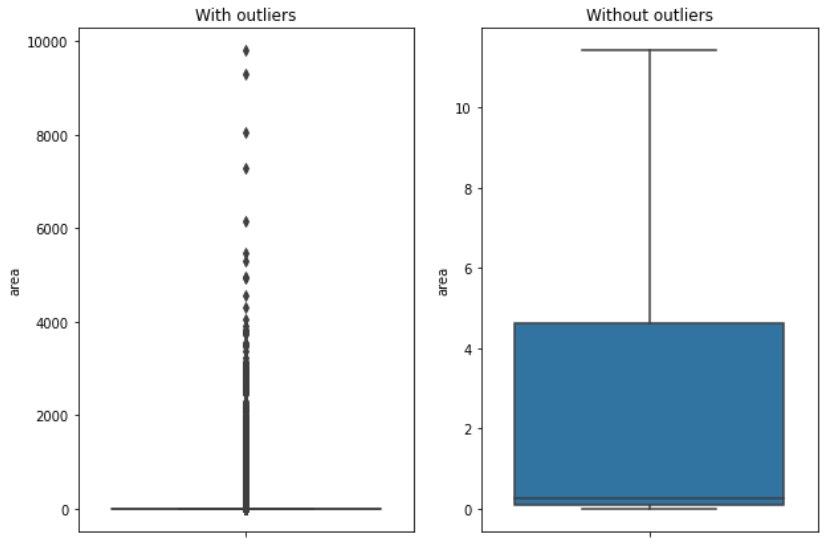


Figure 18 - Boxplots of the census tracts within the NEXUS area.

The mean area of the rural tracts is around 94 km², while, for the urban, it is close to 0.5 km². The boxplots indicate the difference between urban and rural areas that are significant in the dataset.

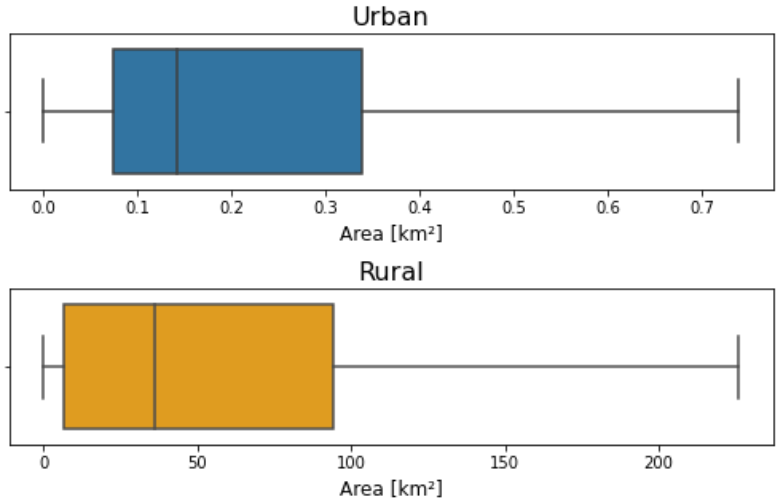


Figure 19 - Boxplot of census tract areas separated by type.

The satellite images to be collected have a scale of 30 meters/pixel and a resolution of 224 × 224 pixels, thus each covers an area of 45,2 km², approximately. This configuration presents no problem for selecting rural tracts as clusters, since they generally have an area greater than the coverage, therefore a cluster image would completely represent a census sector, maintaining the granularity of the collected data.

However, the same does not apply to urban sectors: because they are much smaller than the coverage area, it is not possible to select one to compose a cluster centered on it, since it would not occupy even 2% of the image, allowing the remaining area to be filled by other urban, rural sectors or regions not present in the study area. In this case, the criterion of clusters being represented mainly by only one type could be infringed.

This also imposes the problem that if careful selection of the position of urban clusters is not taken into account, the same sectors, i.e the same image information, may be present in multiple images, resulting in model bias. Thus, a strategy for selecting clusters must consider the points raised in this analysis.

7.1.3.2.Cluster selection

The analysis leads to the development of an algorithm to select a sample of clusters to compose the project's dataset, accounting for the strict criteria defined of not having a single region pertaining to more than one cluster and make each cluster represent only one type of tract, while still keeping a meaningful proportion of rural and urban areas.

The more straightforward approach is to sample NEXUS' area tracts randomly as candidates for cluster centroids and even though this method is able to keep the proportion of areas plausible, it makes the other criteria virtually impossible to achieve, as further discussed in section 7.1.3.2.1. Thus, a second, more elaborate method, was designed. This proposal aims to keep regions from overlapping by creating rural and urban groups and extracting the grid that best fits clusters for each group.

7.1.3.2.1.Method 1: Random Sampling

The first proposal of cluster selection is to take a random sample of tracts from each municipality and define each of those as the center of a cluster. Since the goal is to avoid overlapping regions, an heuristic method is in place and functions by mapping all tracts that are contained within clusters, as well as how many clusters they are inside, i.e. if it appears in more than one cluster, those clusters overlap. The algorithm does a certain number of iterations trying different samples in an attempt to find a good solution, i.e. lowest amount of overlaps.

It seems to be a good approach because it respects the criterion of maintaining the proportion between urban and rural regions in the dataset, when taking a sample that must keep the same properties of the population. In addition, since urban tracts tend to be concentrated, picking one randomly to be the center of a cluster may result in a cluster mainly urban, respecting the criteria proposed for that.

However, this method makes the problem of overlapping difficult to solve because, for many cases, the urban sectors are so close to each other that it is almost impossible to sample a good amount of random clusters not superimposed. An example of this problem can be seen in Figure 20, in which there is a large urban concentration in the center of the municipality.

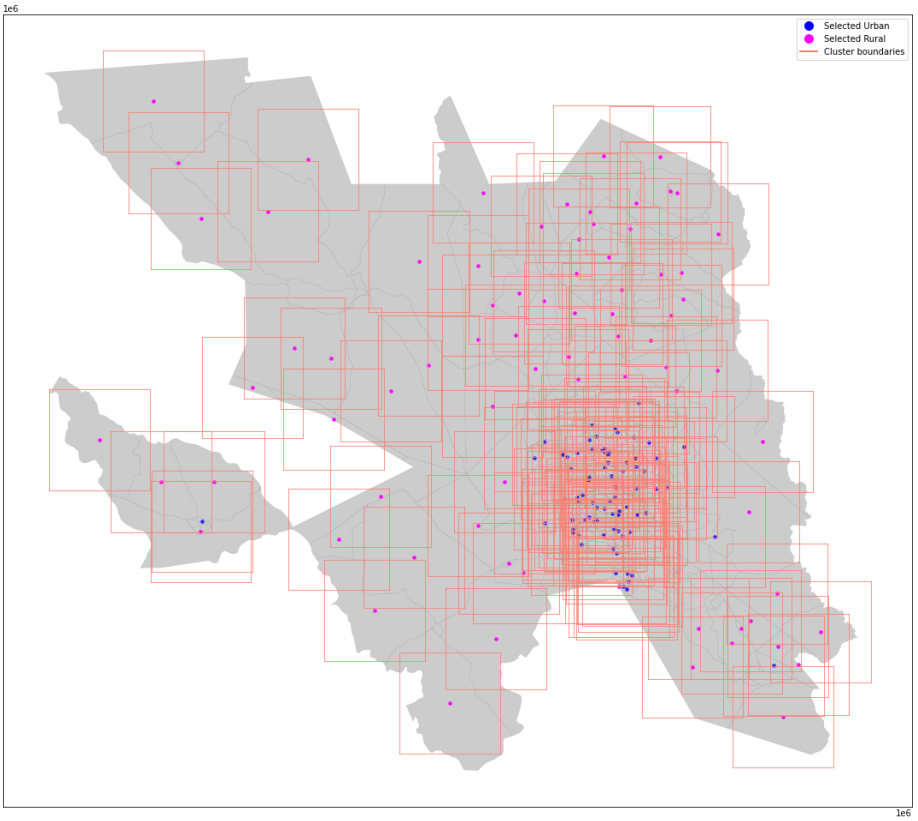


Figure 20 - Example of random cluster sampling method for Feira de Santana (BA) municipality.

In this figure, it is possible to notice again that the urban tracts (blue dots) are much more concentrated than rural ones (magenta dots), leading to the inevitable overlapping of clusters (orange squares).

Furthermore, the same problem also occurs for the rural sectors, because there is a large number of clusters being selected. It could only be solved by drastically reducing the number of clusters, however the resulting dataset would not have much data, being a bad outcome for use in deep learning algorithms.

7.1.3.2.2.Method 2: Neighboring Graphs

To solve the issue, the second proposed method segments urban and rural areas in groups and creates a grid of clusters for each group. The groups are created through an algorithm that computes a disconnected graph which maps the adjacency of tracts of the same type in the entirety of NEXUS area, as demonstrated in Figure 21 for the urban tracts.

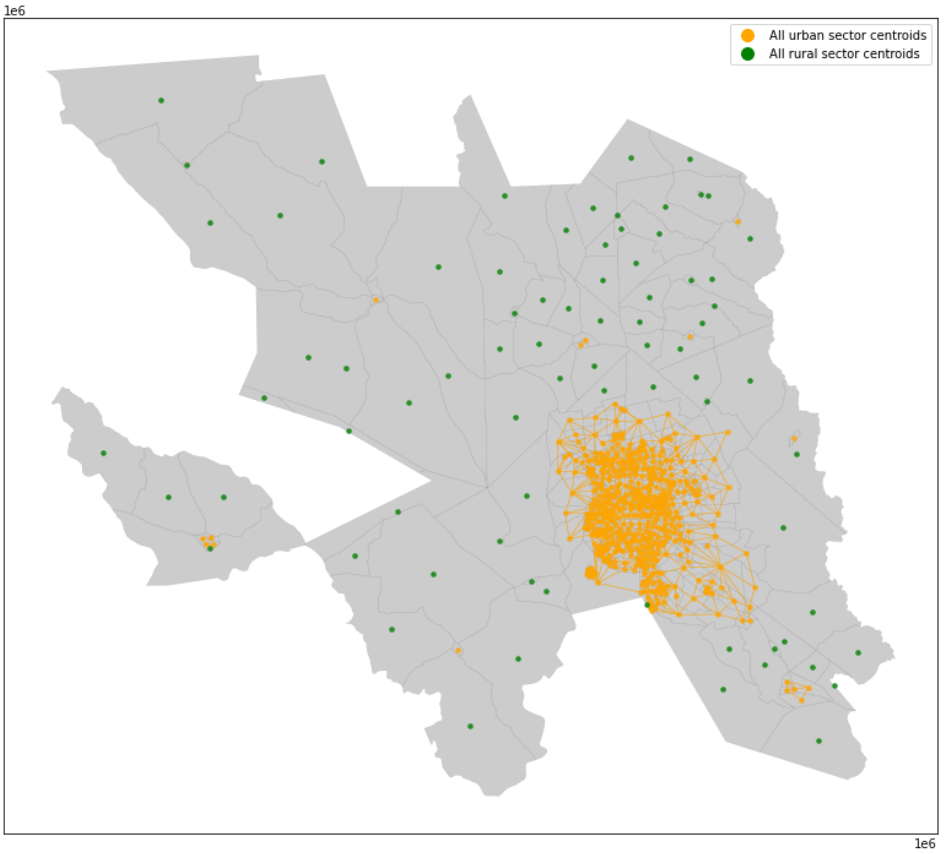


Figure 21 - Creation of urban groups through adjacency graphs for Feira de Santana (BA) municipality.

Each yellow dot represents the centroid of an urban tract and the yellow lines connect adjacent tracts, i.e. tracts that share at least one border. Those form a disconnected graph where centroids are its nodes and lines are the weights, which means all the urban neighborhoods. This disconnected graph may be detached as connected subgraphs, each representing a neighborhood. In Figure 21, there are nine urban neighborhoods.

Each subgraph is then merged to create a single polygon, called urban group, that represents the group and all possible clusters are fitted into this new form side-by-side, forming a grid. A cluster will fit the polygon if at least a specified threshold of its area is filled with tracts from a single type, in order to guarantee that all selected clusters mainly represent the type being selected (urban in this case). The result of this method can be seen in Figure 22.

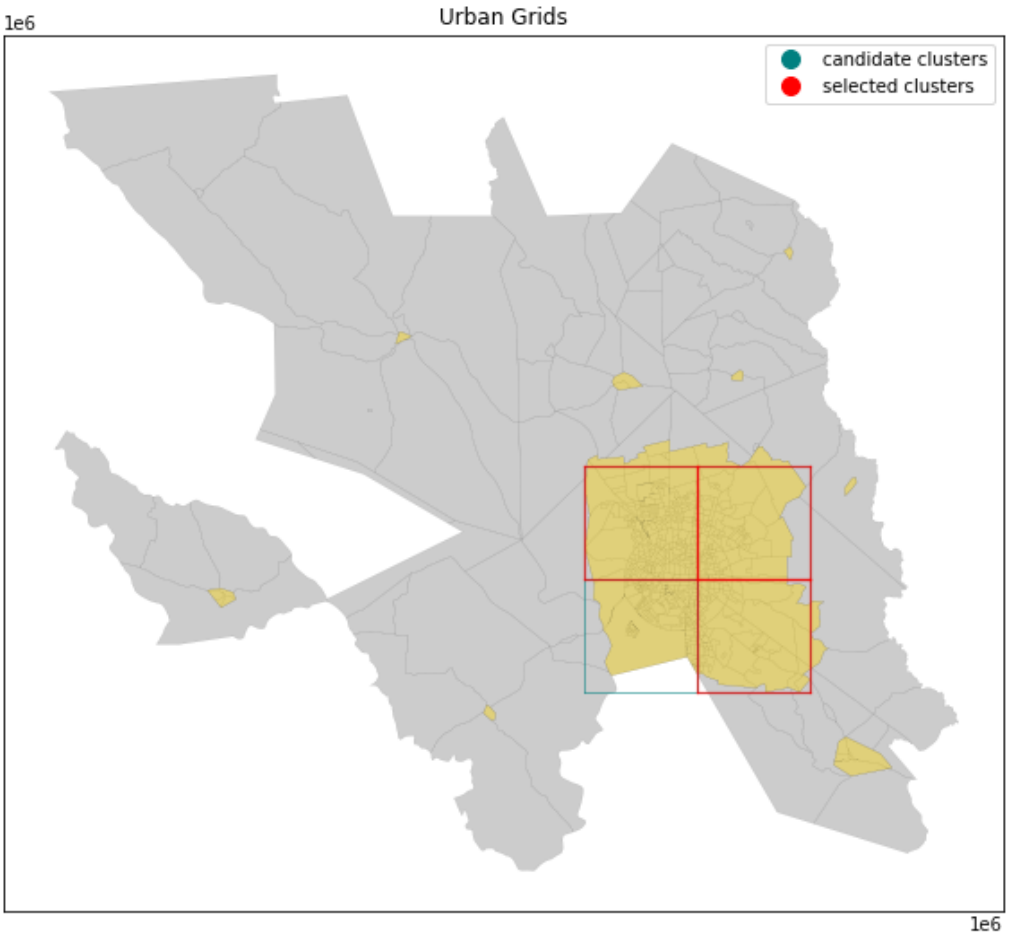


Figure 22 - Example of the Clustering method on urban groups for Feira de Santana (BA) municipality.

In the example, eight of the nine urban groups are not big enough to fit a single cluster considering the adopted threshold (adopted as 60% for urban selection). The biggest group fits four possible clusters, represented as green squares, and three of them are mainly composed of urban sectors regarding its area, i.e. at least 60% of each cluster area represents an urban region. All possible clusters that meet such criteria are then selected as a cluster for the dataset and, in the image, is represented as a red square.

Finally, for every cluster it is necessary to provide a value for each socioeconomic indicator. Since the granularity of the data is given for census tracts, which are generally smaller than the cluster area (except for some outlier rural tracts), it is possible to calculate the target value of the cluster by calculating the average of all tracts' targets within it.

This method is adequate since it solves the problem of having the same tract in more than one cluster. Besides that, since the clusters are linked to a grouping that represents either a rural or urban area, they depict only one type of tract.

7.1.3.2.3.Adopted proposal

The adopted proposal focuses on the second method, Neighboring Graphs, since it prevents any possibility of clusters overlapping and optimizes the selection of adjacent

regions, ensuring the generation of as many clusters as possible. This algorithm meets the proposed conditions for avoiding model bias.

As another example, the capital of Mato Grosso do Sul (MS), Campo Grande, is used to demonstrate the algorithm's applicability. Figure 23 presents the results for running Neighboring Graphs for its urban tracts: there are 7 urban clusters selected, with a threshold of 0,6. Figure 24 represents the same method for rural tracts, where there are 138 rural clusters selected, with a threshold of 0,9. The described thresholds were chosen empirically because they present a relevant amount of selected clusters while still maintaining the expressiveness of the type.

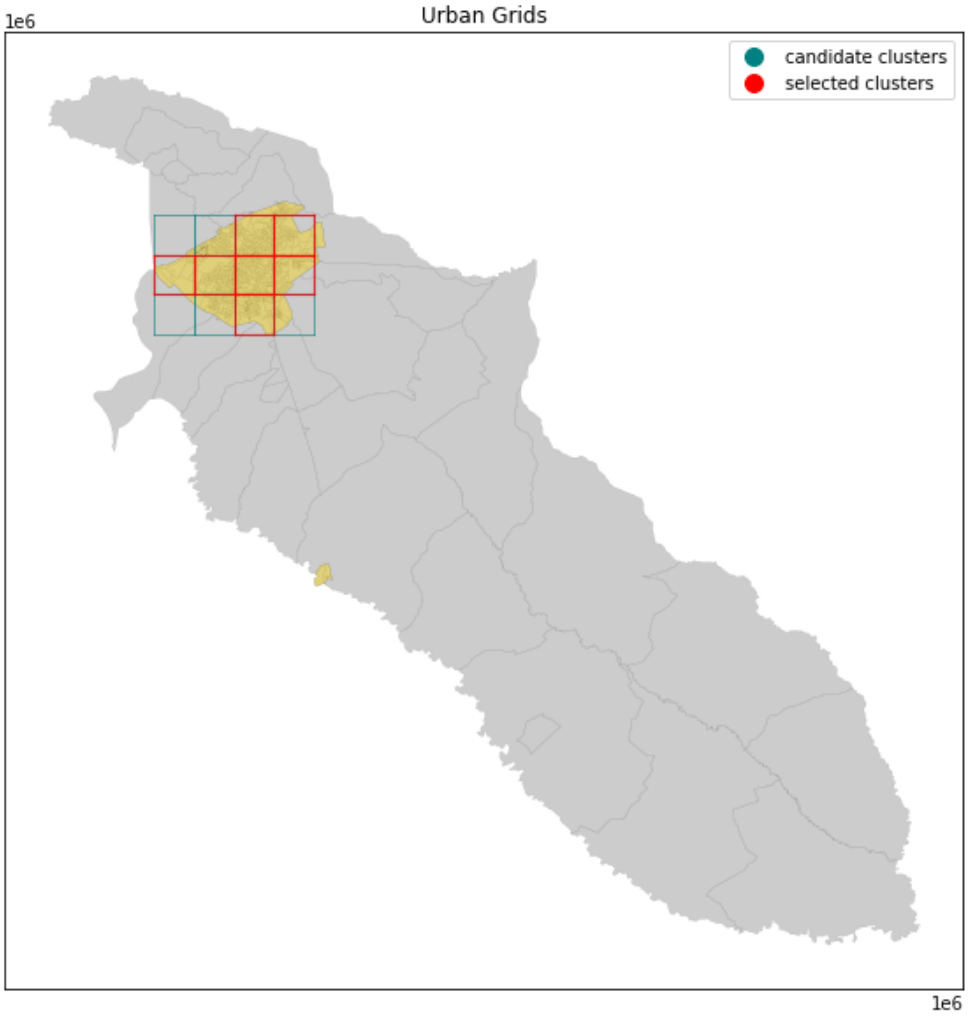


Figure 23 - Example of the clustering method on urban groups for Campo Grande (Mato Grosso do Sul) municipality.

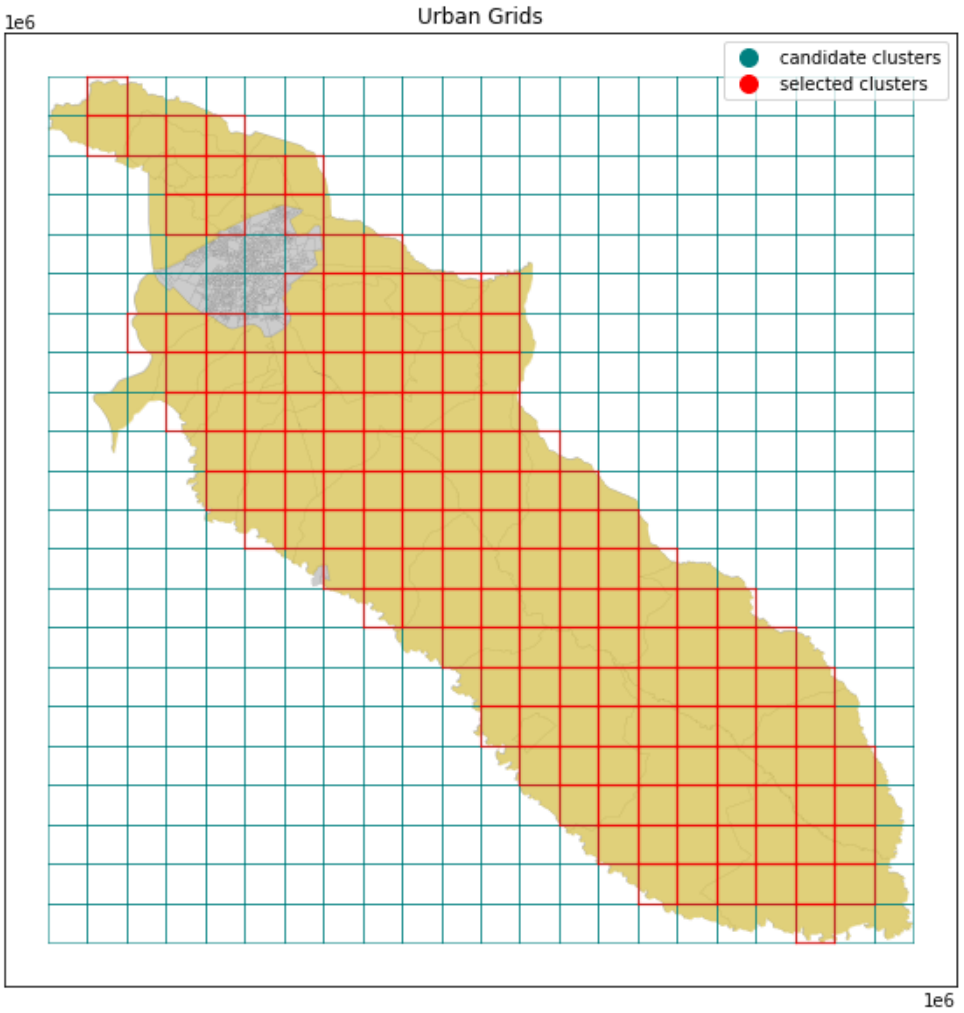


Figure 24 - Example of the clustering method on rural groups for Campo Grande (Mato Grosso do Sul) municipality.

The application of the algorithm for the entire NEXUS area results in the selection of 438 urban sectors for a threshold of 60% and 52631 for a threshold of 90%. The total area covered by all those clusters is around 2.38 million km², representing almost 69.6% of the entire NEXUS area territory, meaning the algorithm is able to cover a great part of the studied area. This dataset is stored as a CSV file named dataset_all_clusters.csv

It is important to reduce the amount of rural clusters in order to keep the proportion between urban and rural clusters to reduce the impact of class imbalance and to minimize costs regarding storage of the images. To this effect, rural clusters are undersampled, i.e. the clusters of the majority class are randomly eliminated to reduce class imbalance [46].

Taking into account the ratio between urban and rural areas in the NEXUS area, which is equal to 1.21%, and knowing that the total of urban sectors is 438, the rural sectors are undersampled to an amount of 20000. Thus, the resulting dataset consists of 20438

clusters, which results in the urban clusters composing 2.14% of the dataset and the ratio between types goes from 1:120 to approximately 1:46.

This amount is also similar to the total of images in the experiment executed by Yeh [11], allowing to achieve the same expressiveness of each split for 5-fold cross-validation.

This reduced dataset is, then, reprojected to EPSG:4326 projection (WGS84 - World Geodetic System 1984) [49], which is the default projection by GEE API [50], and stored as another CSV file, dataset_clean.csv, which is the main input of the estimation steps.

7.2.Socioeconomic indicators estimation

This section discusses the development of the methodology for estimating the estimators. The steps described are the ones from "Based in Yeh et al. (2020) Workflow" branch of the Model Workflow (Figure 11).

All notebooks are executed in AWS Sagemaker which allows centralizing all pipeline steps and data and guarantees great computational power with the machines used for processing: ml.m5.4xlarge for all files except when a GPU was needed for training Deep Learning models, requiring the usage of ml.g5.8xlarge.

7.2.1.Image download with GEE

The first step in estimating the socioeconomic indicators uses GEE API to download the clusters' images based on the geographical information in dataset_clean.csv, the images are collected and exported to a google drive as TFRecords, a format that stores data as binary strings for efficiently encoding long sequences, in batches.

The Jupyter Notebook responsible for downloading the images is 01_download_tfrecords.ipynb, which reads the dataset file, processes the year (2010) to obtain the 3-year composite from 2009 to 2011 and requests the download of the images through GEE API with the necessary parameters. Among the parameters, most have already been discussed earlier, such as cluster centroid, year composite, bands, projection, dimension, and scale. The remaining parameters are a storage path where batches will be exported to, a google drive, in this case, and the batch size, informed as 30, that determines the amount of samples in each Tfrecord file.

Each sample contains all the important data about an unique cluster: centroid coordinates, year, country, 9 image bands and socioeconomic indicator values.

This process execution takes about 8 hours for the given dataset containing the 20438 selected clusters. The downloaded 628 batches of TFRecords, each with 30 samples except for the last one with only 8, occupying 16 GB of memory.

Figure 25 represents the bands of a downloaded cluster image, whose income value is 0,481328, latitude is -13,248899 and longitude is -43.780254. Note that it is a rural cluster, and the NL band is completely null, since, probably, there is no light source.

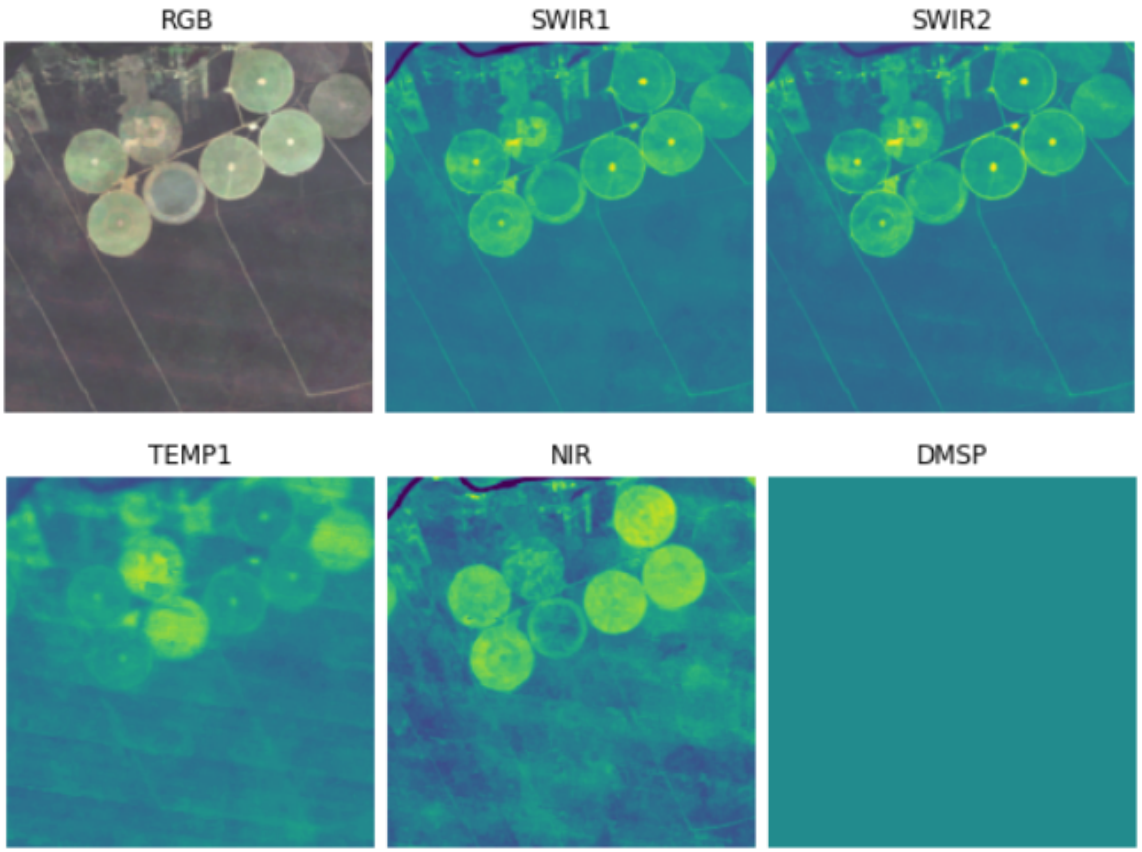


Figure 25 - Example of downloaded image bands.

By searching these coordinates on Google Maps, it is possible to check that the satellite image shown is in the same region, in Serra do Ramalho - BA. So this indicates the images downloaded correctly.

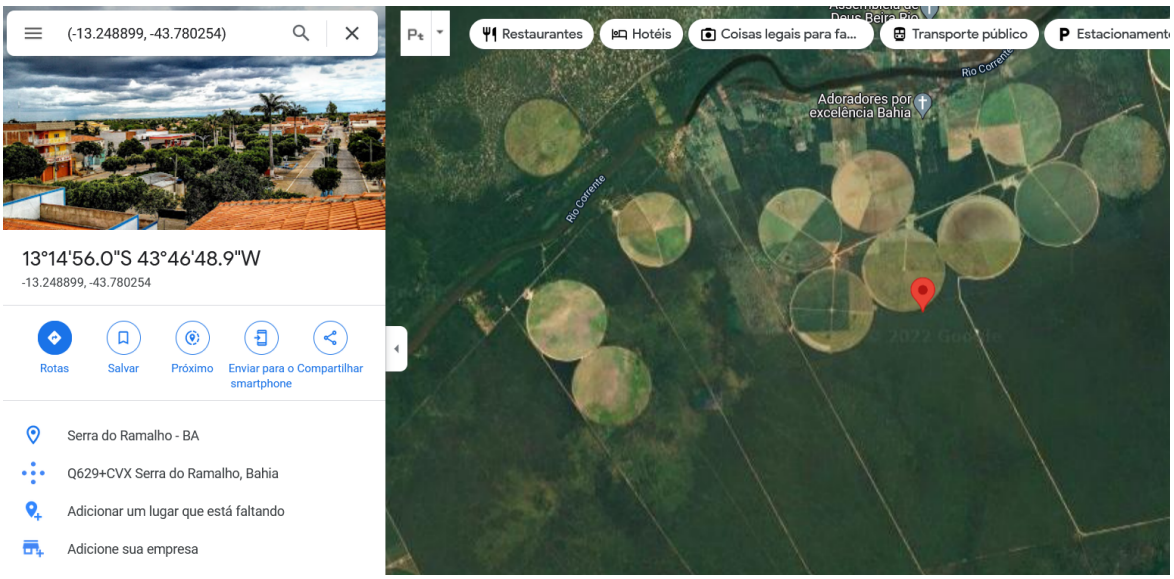


Figure 26 - Google Maps satellite image of the region used as example in Figure 25.

7.2.1.1.Process TFRecords

Process TFRecords is about validating by comparing it to the dataset_clean.csv and processing them by splitting batches into TFRecords of a single sample, resulting in 20438 TFRecords.

This step also calculates the means and the standard deviations of each image band of the dataset, crucial for the standardization of those values in training.

This is executed in the Jupyter Notebook 02_process_tfrecords.csv, whose main main parameters are the dataset_clean.csv and TFRecords downloaded in the first step. Its outputs are 20438 TFRecords of unique clusters, still occupying 16 GB. The execution takes around 40 minutes to read and rewrite the TFRecords, and another 40 to calculate the statistical parameters of the image bands, presented on Table 7.

Band	Mean	Standard Deviation
BLUE	0.04188	0.01797
GREEN	0.06712	0.02122
RED	0.07304	0.03323
SWIR1	0.22713	0.06981
SWIR2	0.12607	0.05633
TEMP1	297.94874	2.12837
NIR	0.24679	0.04157
DMSP	1.80547	16.17113
VIIRS	0.00000	0.00000

Table 7 - Statistical parameters of each image band across the entire TFRecords dataset.

7.2.1.2.Create Incountry Folds

The next step is to create the folds of the 5-fold cross-validation, which are used to form 5 sets of 3 splits (train, validation and test). The Jupyter Notebook 03_creatre_incountry_folds only needs the coordinates of each cluster centroid, present in dataset_clean.csv, in order to calculate the mean distance between two images borders within the same group.

That information is necessary to allocate clusters among 5 folds, called A, B, C, D and E, then create 5 sets, named according to which group is left to be used in the test split. Training splits contain 3 folds, and the remaining splits, only one each. Figure 27 displays how clusters are distributed among clusters throughout the NEXUS area.

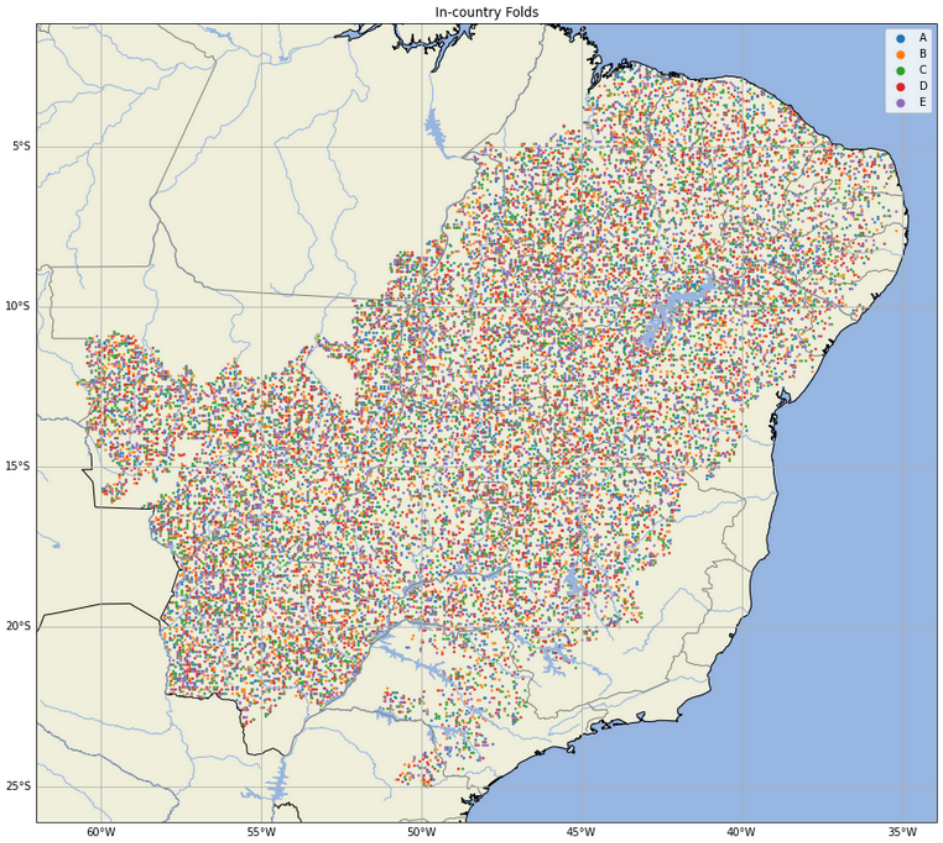


Figure 27 - Distribution of clusters into 5 folds across NEXUS area. Each sample represents the centroid of a cluster.

The output is a 1MB Pickle file, used to serialize Python object structures, called incountry_co.pkl, containing the indices of each set split and fold, which are used during training to actually create the folds for cross-validation. This code takes approximately 20 minutes to be executed.

7.2.1.3. Dataset Validation

In parallel with the creation of the folds, a data validation algorithm is executed to check if the data is coherent and suitable for being used as input to the Deep Learning models. This section approaches three validation processes: data histograms, process baseline parameters and baseline models.

7.2.1.3.1. Data histograms

The processed TFRecords' image bands are considered in this analysis. The Jupyter Notebook 04_1_plot_histograms.ipynb plots histograms of each band's pixel distribution, as shown in Figure 28. Note that there are only 8 bands represented, because VIIRS is not considered in the graph, since it only has data starting from 2012 and the dataset being used is from 2009 to 2011.

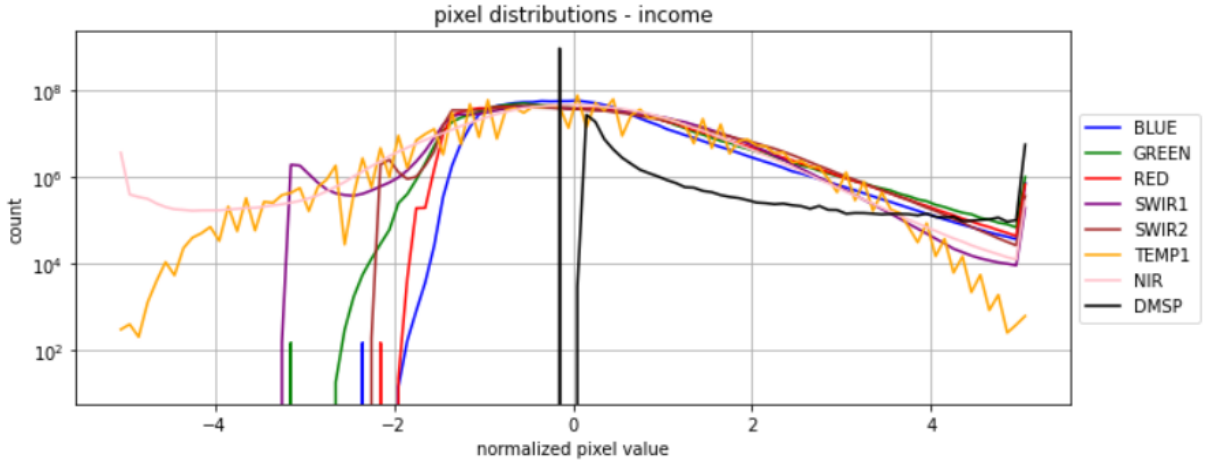


Figure 28 - Pixel distribution of each satellite image band.

Note that all bands have a wide range of pixel values in the histograms, except for the DMSP band with many null values. The low variety of values for such a band can be explained by the majority presence of images from rural regions in the dataset, which tend to be less illuminated than urban regions, and, as discussed during the selection of clusters in section 7.1.3.2.3, rural clusters are chosen with a threshold of 0.9, i.e. it is guaranteed that 90% of a rural cluster image represents a rural area, while only 60% of a urban cluster image is, for sure, urban, meaning that even urban clusters may present poor lighting.

This large amount of null pixels can negatively affect the training of neural networks related to NL images, since, as discussed in section 6.2.2.2, changing the pixel values of a band to null causes the ResNet not to consider that band during training. Therefore, the ResNet-NL network, to be trained with most DMSP images being mainly composed of null pixels, may not have its weights correctly optimized, presenting poor predictive performance.

This step also analyzes and plots the indicator's histogram in order to check if it is appropriate for the training, i.e. if there are any missing values and if it is biased. The outcome is identical to that obtained during section 7.1.2.

The output is a NumPy zipped file, co_income.npz (where income represents the name of the indicator being processed), containing the information needed to plot histograms again and cluster data, such as labels for each, centroid coordinates, year, the central pixel (center) and the mean of the DMSP (NL) band of each image. This 5 MB file is useful for the next step, which processes it to be used in baseline models.

The execution of this code takes around 40 minutes to read values from the TFRecords, process the output data and plot the histograms.

7.2.1.3.2. Process baseline parameters

The histogram data file, co_income.npz, is processed by the Jupyter Notebook 04_2_create_loc_dict.ipynb in order to prepare the input data to train baseline models.

The output of this step is another 1 MB Pickle file, loc_dict_2010_brazil_income.pkl, containing all the 20438 samples, but only considering scalar values about the nightlight images: its mean and center. Its structure is similar to a python dictionary, where latitude and longitude values form a tuple which is the key of each element, whose value is another dictionary.

The execution time to process the input file and create the new one is around 2 minutes.Since this code involves only the indicator dataset file, its running time is quick, with the resulting file having around 1 MB.

7.2.1.3.3. Baseline models

Experiments using baseline models are useful to validate if the TFRecords dataset is appropriate for the training with Deep Learning models and already estimate how well it is going to perform. The Jupyter Notebook 04_3_model_baselines.ipynb trains KNN models using the histogram data available in co_income.npz and loc_dict_2010_brazil_income.pkl.

Different experiments are executed by this step and they are all compared using r^2, R2, MSE and rank metrics. The experiments use two different models: KNN and Ridge Regression, and are trained over 6 different groups of data and some of its combinations: NL mean, NL center, NL hist, RGB hist and MS hist.

The NL mean data is the average of each pixel from the image DMSP band, thus it is a group of 20438 scalars. The NL center is similar, but it only uses the center pixel of each DMSP band. The remaining data groups, suffixed with “hist”, are the pixel histogram descriptions of the band.

This step executes 10 experiments 5 times, one for each set determined by incountry_co.pkl, which are described in Table 8.

This file outputs a test_preds.npz file of 100 KB for each experiment, storing the indicators prediction and validation metrics of said model. Since this model is simple, the execution time is no more than 20 minutes.

The Jupyter Notebook 05_evaluate_baselines is, then, used to evaluate the performance of each experiment by gathering the test_preds.npz files, calculating the evaluation metrics and creating CSV files that summarize the results obtained. These CSV files occupy around 4 MB of memory and the execution time is 1 minute.

Experiment name	Description	Input data
KNN NL center scalar	A one dimensional KNN over the DMSP band centers.	NL center
KNN NL mean scalar	A one dimensional KNN over the DMSP band means.	NL mean
KNN NL hist	A 102-dimensional KNN over the DMSP band pixel histogram	NL hist
Ridge NL center scalar	A Ridge Regression over the DMSP band centers.	NL center
Ridge NL mean scalar	A Ridge Regression over the DMSP band centers.	NL mean
Ridge NL hist	A Ridge Regression over the DMSP band pixel histogram	NL hist
Ridge RGB hist	A Ridge Regression over the pixel histograms of Red, Green and Blue bands.	RGB hist
Ridge RGB+NL hist	A Ridge Regression over the pixel histograms of Red, Green, Blue and DMSP bands.	NL hist RGB hist
Ridge MS hist	A Ridge Regression over the pixel histograms of Red, Green, Blue, NIR, SWIR1, SWIR2, TEMP1 bands.	MS hist
Ridge MS+NL hist	A Ridge Regression over the pixel histograms of Red, Green, Blue, NIR, SWIR1, SWIR2, TEMP1 and DMSP bands.	NL hist MS hist

Table 8 - Baseline model experiment descriptions.

The summary of each model's results is presented in Table 9. Note that the best model in Ridge MS+NL hist with the highest rank. For that experiment, the coefficient of determination is 0.32, representing a good result for a model as simple as Ridge Regression. Thus, it is expected that the performance of the Deep Learning model is greater than 0.32 and, therefore, satisfactory for this study.

Experiment	r^2	R2	MSE	rank
Ridge MS+NL hist	0.32390	0.32389	0.01687	0.63732
Ridge MS hist	0.31866	0.31862	0.01700	0.63291
Ridge RGB+NL hist	0.22525	0.22525	0.01933	0.53157
Ridge RGB hist	0.21615	0.21615	0.01956	0.52053
Ridge NL hist	0.01265	0.01263	0.02464	0.06251
KNN NL center scalar	0.00998	0.00837	0.02474	0.05500
KNN NL hist	0.00933	0.00832	0.02474	0.07183
Ridge NL mean scalar	0.00893	0.00893	0.02473	-0.00815
Ridge NL center scalar	0.00888	0.00888	0.02473	0.00092
KNN NL mean scalar	0.00673	0.00532	0.02482	0.05620

Table 9 - Baseline model results metrics.

7.2.1.4.Deep Learning models training

The Deep Learning models are trained using the 20438 TFRecords processed in earlier steps of the pipeline, running the experiments described in the methodology with two Jupyter Notebooks: 06_train_separate.ipynb and 07_train_combined.ipynb, both executed in an AWS Sagemaker GPU machine, ml.g5.8xlarge, which takes between 1 and 2 hours to train each set of data.

7.2.1.4.1.Separate models training

The first step is to run 06_train_separate.ipynb for generating the separate models, ResNet-NL and ResNet-MS, which are trained 5 times, one for each set. The outputs of this file are: 10 training checkpoints, 10 features.npz files, 10 CSV result files and 10 JSON training parameter files. The amount of file is due to the 2 experiments being executed for 5 sets each.

A training checkpoint is a set of files that, combined, keep the values of all trained weights in the CNN and the result of all epochs, also informing which was the one with the best validation performance. They are important for generating features.npz and reproducing training from an arbitrary point.

A feature.npz file contains the 512 features outputted from the last fully connected layer of the ResNet18 for each one of the 20438 images, occupying 200 MB of memory, approximately. This information is used during the training of the combined model: instead of using the trained ResNet-NL and ResNet-MS models and reprocessing the images, it is enough to simply inform the processed result of the images, since the weights are frozen and will not be changed.

A result.csv file is a table containing the metrics of each epoch of a single set of an experiment, indicating which had the best performance over the validation split, similar to the training checkpoint. It is useful for analyzing the training evolution during the experiment.

A params.json file contains all the parameters and environment configurations set for the experiment, such as hyperparameters, initial weights and initialization procedures, if a GPU was used for processing, if there was data augmentation, configurations for printing in the terminal, indicator name and split information. It is useful to identify how the training was conducted and the remaining files generated.

7.2.1.4.2.Combined model training

The next step is running 07_train_combined_model.ipynb for training the combined model and also Ridge-MS and Ridge-NL, using the features predicted by the trained ResNet-MS and ResNet-NL backbones. This file generates 100 KB prediction files,

preds_test.npz, of the best version of the 3 models, and ridge_weights.npz, which are 10 KB files containing the Ridge Rgression models' weights.

The training procedure of these models involves Leave-One-Group-Out cross-validation, which is helpful to heuristically determine the best tuning parameter (alpha), by picking the lowest MSE. Figure 29 shows the graphic for the MSE of each set over 14 experiments variating the value of alpha.

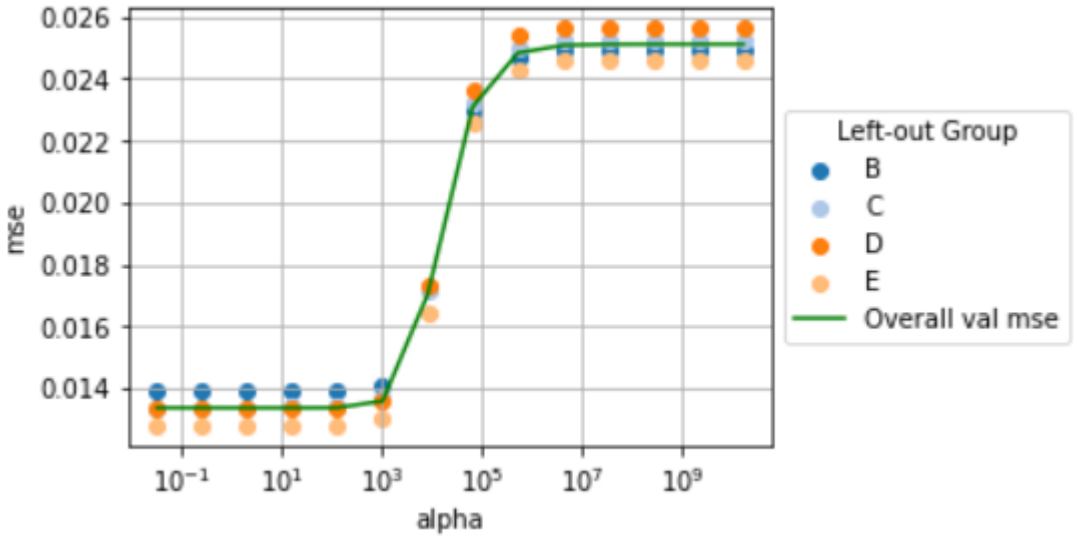


Figure 29 - Cross-validation to obtain the best Ridge Regression's tuning parameter value for MS+NL.

In addition to the training described in this pipeline, a new ResNet-MS model was trained, using 250 epochs instead of the 200 mainly used throughout this work, keeping the remaining hyperparameters of each set, trying to increase the overall score of the results obtained. From this, two new final models are created: new versions of Ridge-MS and MS+NL, both using that ResNet-MS backbone. Thus, there are a total of 5 models to be evaluated.

7.2.1.5.Deep Learning models evaluation

The Jupyter Notebook 08_evaluate_models.ipynb evaluates the 5 resulting models by comparing their results with the baseline models. The comparison takes into account the 4 proposed metrics, calculated with the preds_test.npz files.

Table 10 summarizes the metrics of all models trained in this work for the income indicator.

	Experiment	r^2	R2	mse	rank
Deep Learning models	MS+NL (250 epochs on ResNet-MS backbone)	0.42050	0.42025	0.01446	0.71550
	Ridge-MS (250 epochs on ResNet-MS backbone)	0.42000	0.41975	0.01448	0.71524
	MS+NL	0.41496	0.41420	0.01462	0.71119
	Ridge-MS	0.41457	0.41378	0.01463	0.71098
	Ridge-NL	0.00591	0.00591	0.02480	0.02300
Baseline models	Ridge MS+NL hist	0.32390	0.32389	0.01687	0.63732
	Ridge MS hist	0.31866	0.31862	0.01700	0.63291
	Ridge RGB+NL hist	0.22525	0.22525	0.01933	0.53157
	Ridge RGB hist	0.21615	0.21615	0.01956	0.52053
	Ridge NL hist	0.01265	0.01263	0.02464	0.06251
	KNN NL center scalar	0.00998	0.00837	0.02474	0.05500
	KNN NL hist	0.00933	0.00832	0.02474	0.07183
	Ridge NL mean scalar	0.00893	0.00893	0.02473	-0.00815
	Ridge NL center scalar	0.00888	0.00888	0.02473	0.00092
	KNN NL mean scalar	0.00673	0.00532	0.02482	0.05620

Table 10 - Comparison of experimented models' metrics.

The combined model clearly displays best results across all proposed metrics, as expected from inputting correlated MS+NL bands to the Ridge Regressor. Training both backbones with 200 epochs yielded a combined model with and r^2 of 0.41. Further experimentation showed that increasing epochs to 250 for ResNet-MS backbone generated even better results and the combined model trained with 250 epochs is the best overall, with a r^2 of 0.42.

Ridge-NL presents results even worse than the baseline models, being the worst model trained yet. As raised in section 7.2.1.3.1, this was a possibility taking into account the histogram of the DMSP band, which has many null pixels, causes model underfitting, which is demonstrated in Figure 30 by the fact that the MSE decays early during training (with the best result on epoch 43), and then fluctuates around 0.25 for the rest of the epochs. The results of baseline models are better probably because they use simpler features, such as the average, center and frequencies of the histogram of the DMSP band, which do not present mainly null values.

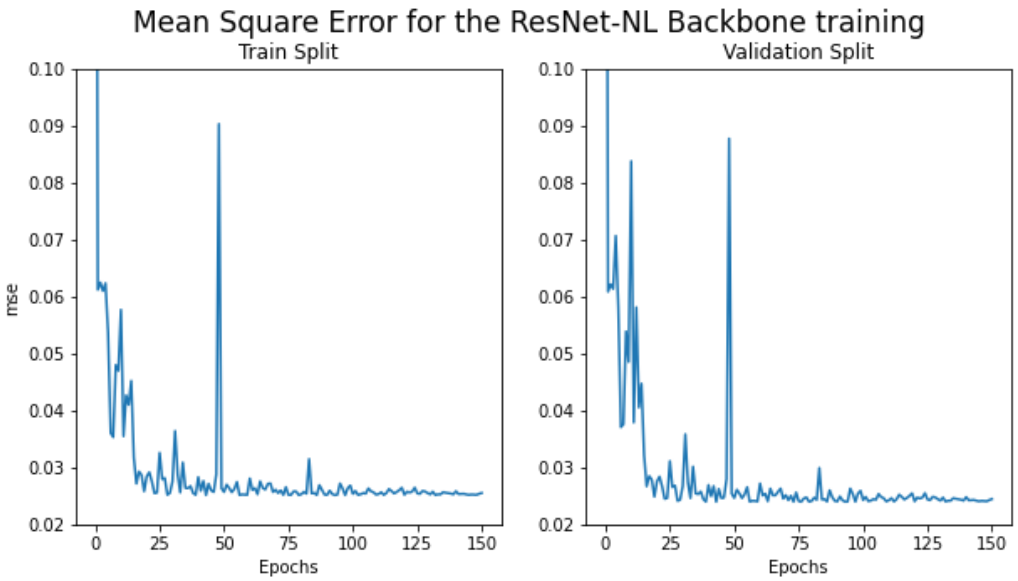


Figure 30 - Mean Square Error analysis for the ResNet-NL model.

Due to the low performance of the NL model, MS+NL and Ridge-MS results are very close, showing almost identical metrics. Given that, it is possible to infer that the NL images used do not relate to the income indicator, which may be caused due to the high occurrence of null pixels in the images, as previously seen in the histograms of the DMSP band.

Lastly, the results of deep learning models were significantly superior to those obtained for baseline models, as expected, since they consider many more features of the images using more complex models.

The distribution of predictions done by the best baseline model, Ridge-NL, Ridge-MS 250 and combined model are seen in Figure 31, as well as their corresponding fitted line of regression obtained from Ridge Regression. In the case of Ridge-NL it is noted that the distribution of predictions is largely centered approximately around 0.5 - which can be explained by the underfitting problem discussed before - and with some points fitting better to the identity line (y = x).

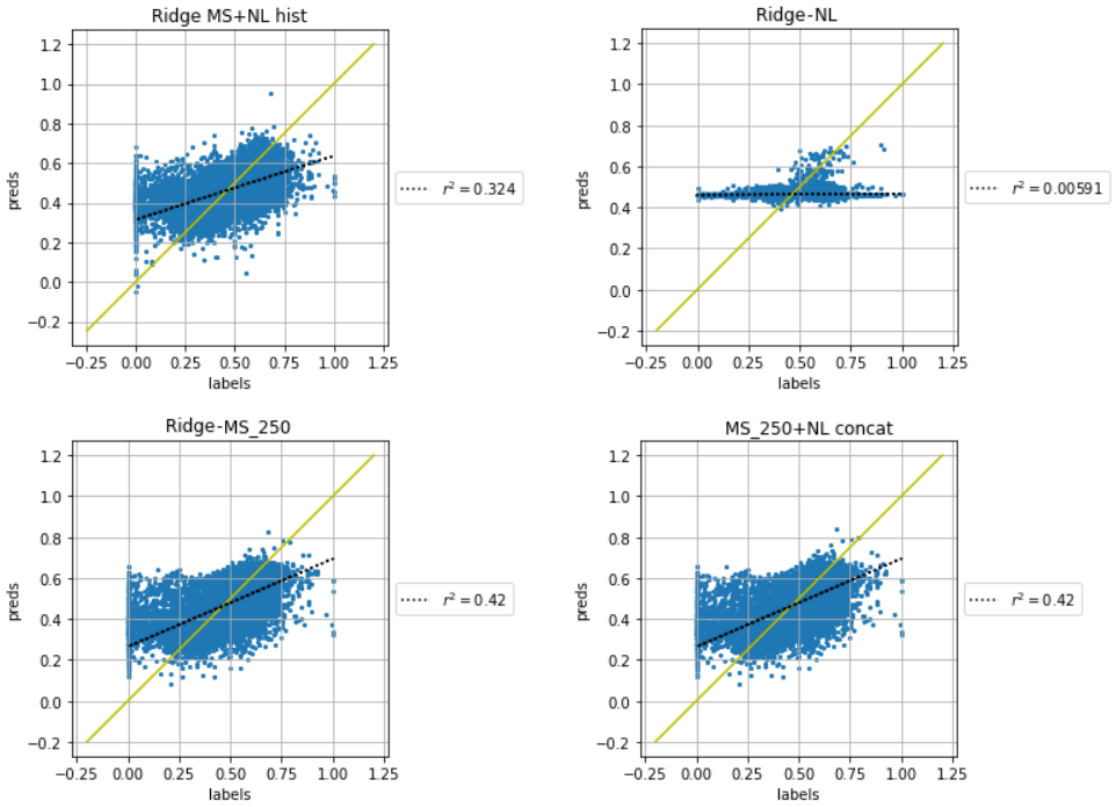


Figure 31 - Relation between target income and predicted income of the main models.

It is also possible to evaluate the prediction of indicators for both cluster types individually, as illustrated by Table 11. On models that use MS bands, the behavior perceived is that the r² for rural tracts is much higher than for urban tracts, which can be explained by the imbalance of both and the fact that the model learns much more features from the first type during training.

On the other hand, on the Ridge-NL model, urban tracts display a better r², since NL images from rural areas exhibit a majority of null pixels that lead to model underfitting, as previously discussed. Besides, on Figure 32, the distribution of predictions for each type are plotted and, for Ridge-NL, the points that best fit the identity line are all urban, further reinforcing the affirmation that urban nightlights provide better features for model training.

Experiment	r2		R2		mse		rank	
	Rural	Urban	Rural	Urban	Rural	Urban	Rural	Urban
MS+NL (250 epochs)	0.4171	0.2637	0.4170	0.1241	0.0146	0.0083	0.7127	0.6253
Ridge-MS (250 epochs)	0.4169	0.2499	0.4167	0.0934	0.0146	0.0086	0.7127	0.6103
Ridge MS+NL hist	0.3191	0.2069	0.3191	0.0945	0.0170	0.0086	0.6329	0.5326
Ridge-NL	0.0003	0.0445	-0.0001	-0.4559	0.0250	0.0139	0.0018	0.1743

Table 11 - Experiment results with Income indicator divided by cluster type.

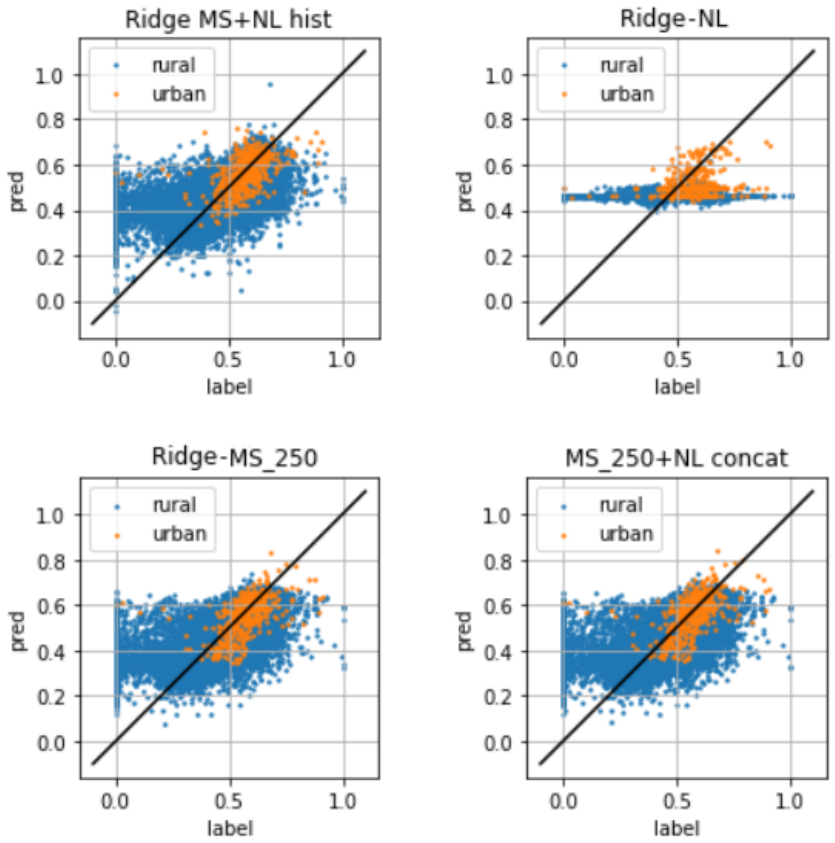


Figure 32 - Relation between target income and predicted income by cluster type.

Figure 33 plots the distribution of income predictions with relation to pixel average value on NL images. The first thing to note is that rural NL images are concentrated on zero pixel average while for urban tracts, are more dispersed.

Models that work exclusively with MS bands predict values heterogeneously distributed for both urban and rural tracts, while Ridge-NL display results concentrated from 0.4 to 0.5, which once again reinforces the assumption that rural NL images do not provide enough features to predict indicators and on the case of urban NL images, this effect is mitigated and the distribution is more similar to other models.

In the case of Ridge MS+NL it is noted that the performance is not affected by the undesired behavior seen in NL images, since it is very similar to the Ridge-MS graph.

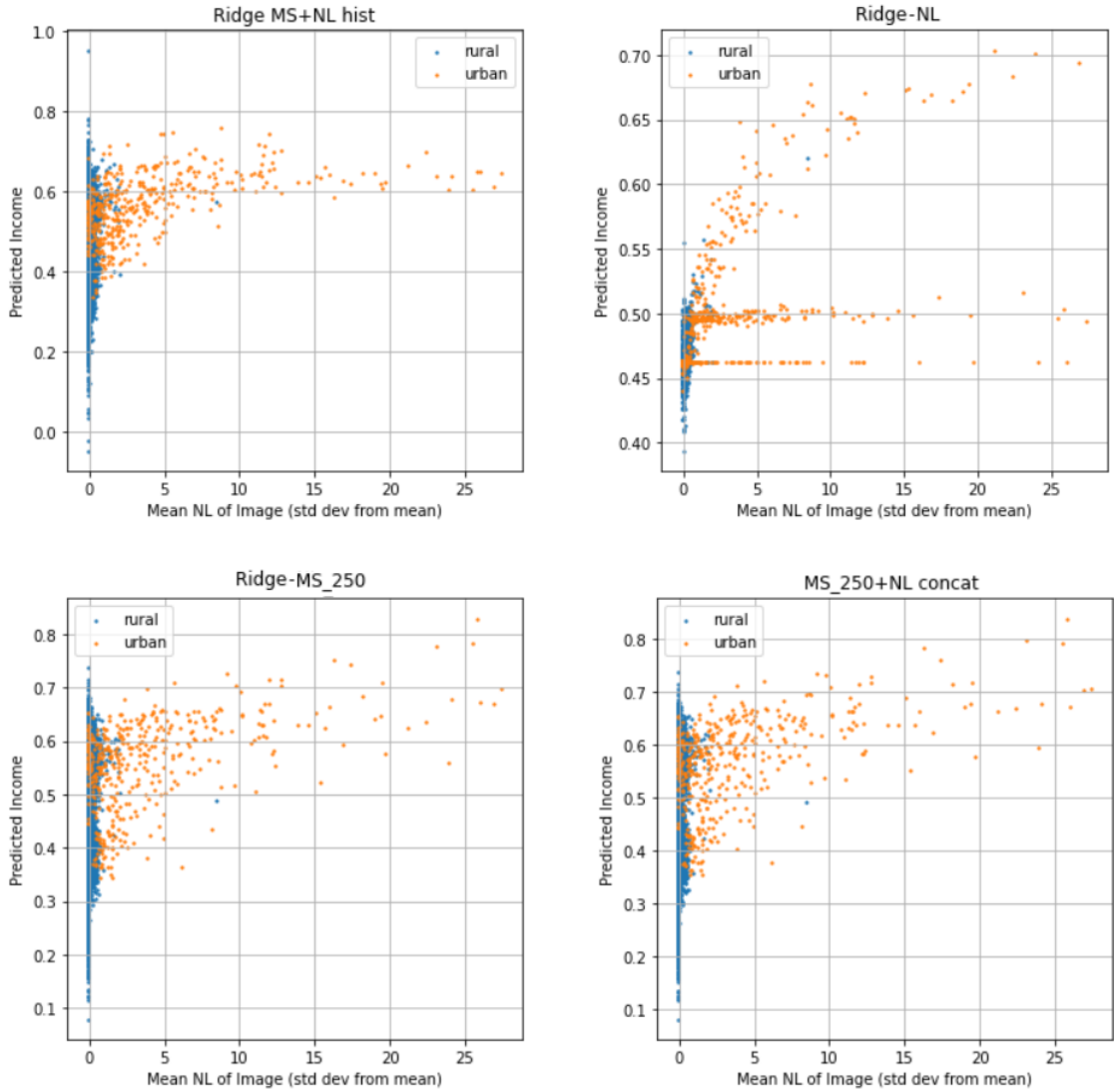


Figure 33 - Relation between income predictions and NL image pixel average divided by cluster type.

7.3.Platform

7.3.1.Architecture design

The architecture for the visualization platform consists of a system separated between a backend pipeline and a frontend web page in order to fulfill the functional and non-functional requirements raised in sections 6.3.1 and 6.3.2. This architecture describes the first version of a system to be implemented in the future.

The backend is responsible for downloading images through GEE API, running predictions over arbitrary data by processing the downloaded images with the best trained model of this study, storing predictions in a database and, finally, providing an API for the frontend to request predictions for arbitrary regions and years. If the predictions are already cached in the predictions database, they are returned, otherwise, the backend executes the prediction pipeline.

The backend architecture is distributed, making use of AWS serverless solutions to deploy its processing components, AWS API Gateway to create its API and DynamoDB and AWS S3 to store predictions and downloaded images for prediction, respectively. Its components and workflow are described by a provisional sequence diagram in Appendix A.

The frontend is responsible for allowing users to navigate through the interactive map of the Brazilian territory and the menu for selecting and configuring the platform's functionalities. The user requests for visualizing specific predictions not yet loaded into the frontend are requested to the backend through its API.

.

7.3.2.Implementation

A prototype of that platform is developed for this project. The page is hosted using GitHub Pages and is built from scratch using HTML, CSS and Javascript. The latter is required because the framework provided by Google, Earth Engine Apps [68], used to illustrate custom geometries and interact with Earth Map, is not able to communicate properly with the backend described on section 7.3.1.

Therefore a new App is in place, utilizing the functionalities provided by Google Earth API for Javascript and allowing the user to visualize and interact with a map of Brazil with NEXUS area highlighted. It is possible to change the resolution of visualization, i.e. select between states or municipalities, and then select the polygons on the map to see the predicted indicators as illustrated by the selected states colored in orange on Figure 34.

On the left-side menu, also shown on Figure 34, it is possible to select the year of the predictions and also one of the three indicators at a time. The application also disposes of charts to better understand indicators.

Figure 34 - Prototype of the main page of the website.

8. Test and evaluations

This section briefly discusses the evaluation of the other two indicators, longevity and literacy, as well as presenting an analysis of the geographic distribution of predicted indicators.

Similar to Table 9 for income, presented on session 7.2.1.5, there are tables for the results obtained for literacy and longevity on Appendix B. On both tables, it is noted that the results are a lot worse than income, with longevity being the worst of them. This is probably due to the fact that those indicators do not follow a normal curve, but instead are skewed left histograms, as already discussed on session 7.1.2.1.

There is also the hypothesis that those indicators may not be properly represented by satellite imagery. To investigate this possibility, it would be needed to apply different techniques for preprocessing available data, such as normalizing the data with some standardization method and increasing the dataset including other year surveys and experiment further.

To understand if there is any relation between model performance and the different regions that compose NEXUS area, the plots for Figures 35, 36 and 37 were created to illustrate income, longevity and literacy, respectively, estimated on the test split of the best evaluated set for the model. The plot depicts the absolute value of the residual, calculated by subtracting the predicted from the target value, for the predictions and it is possible to note that the residuals distribution is evenly spread across the entire area and, thus, there is no clear geographic impact on the model performance.

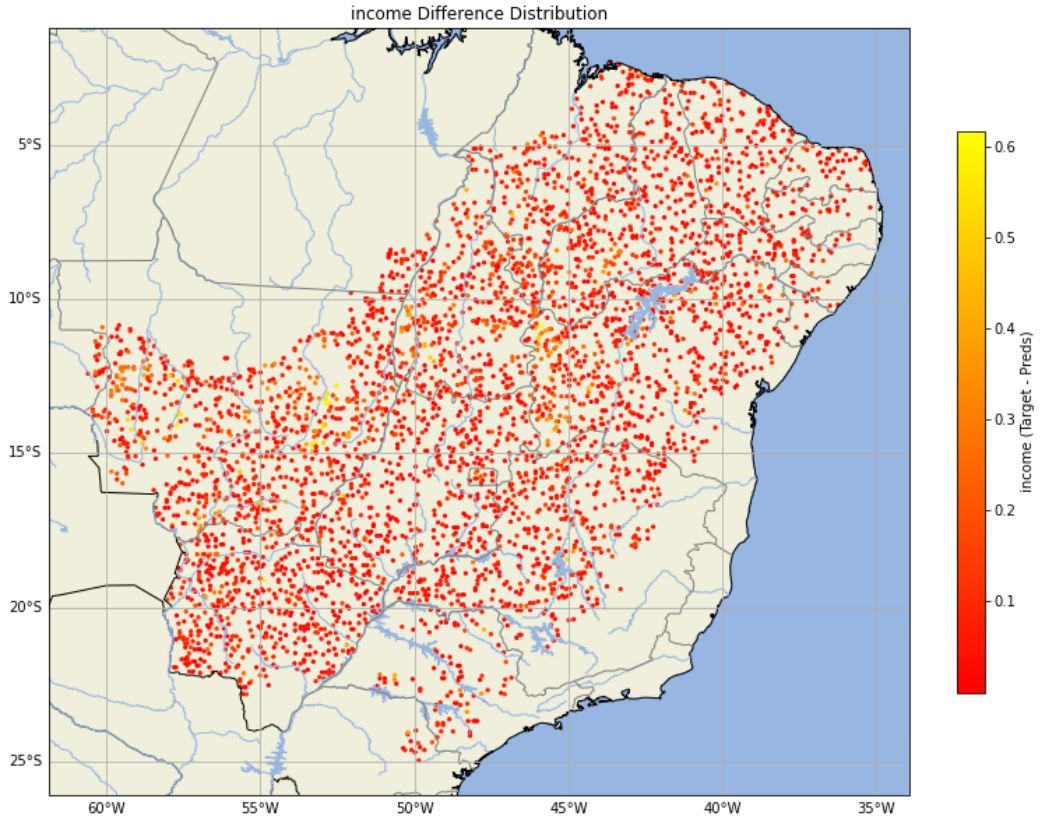


Figure 35 - Map of the Difference between the target and the predicted value for income indicator.

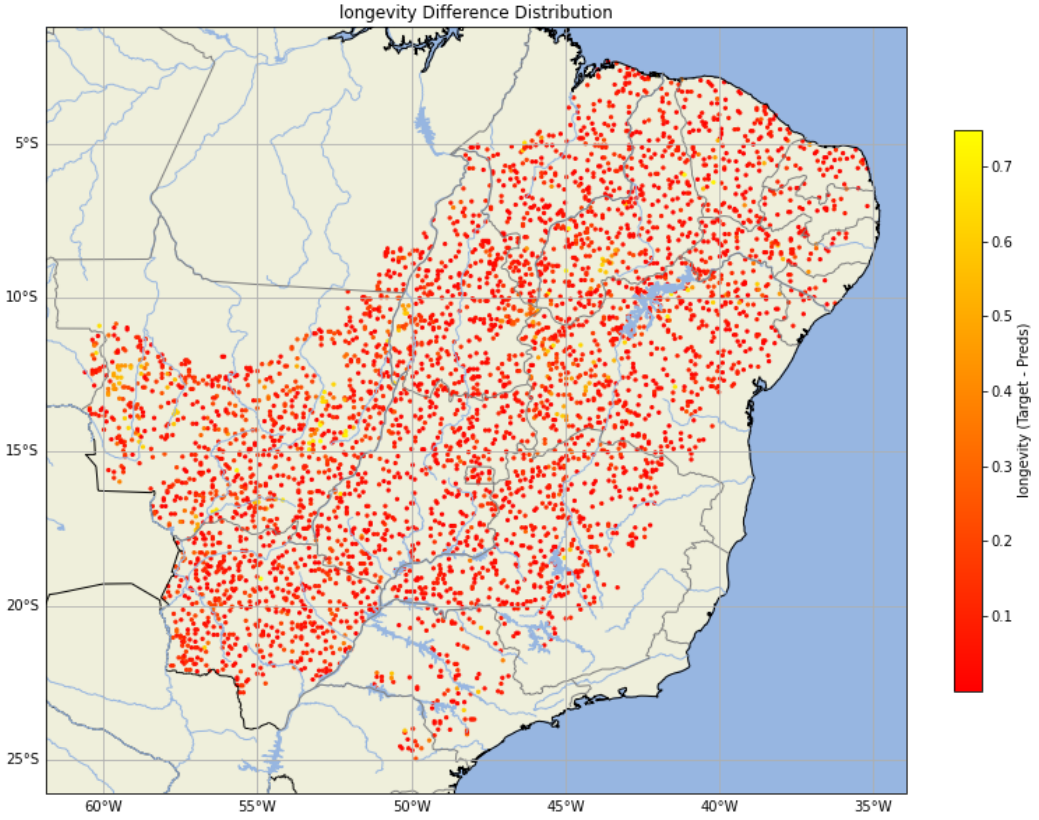


Figure 36 - Map of the Difference between the target and the predicted value for longevity indicator.

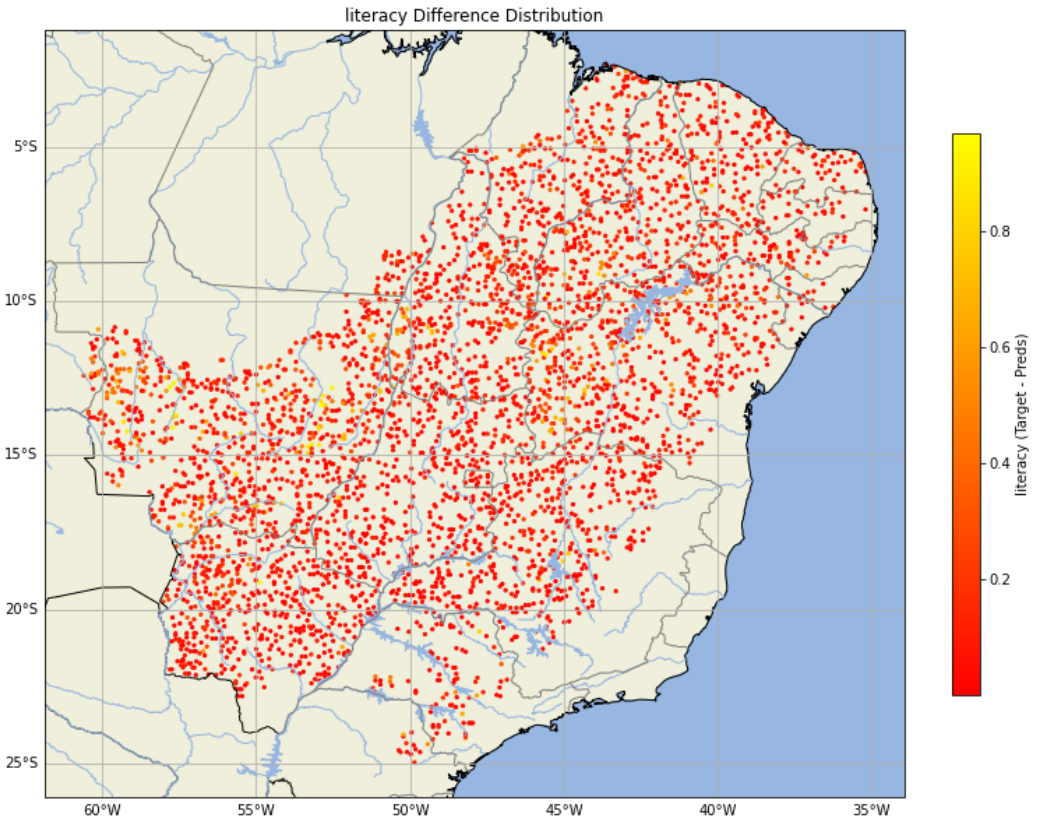


Figure 37 - Map of the Difference between the target and the predicted value for literacy indicator.

9. Final considerations

9.1. Conclusions

The use of remote sensing imagery is a largely available low-cost resource that is collected at very frequent rates, contrary to traditional census surveys, which take place every 10 years. Therefore, its use, combined with the always evolving area of Deep Learning, is an advantageous alternative to estimate socioeconomic indicators. This work starts from this premise to train Deep Learning models with LANDSAT and nightlight satellite images to estimate income, literacy and longevity indicators along the NEXUS area.

This work managed to reach its objective of proposing and applying a successful methodology of training Deep Neural Networks to estimate 3 indicators in the NEXUS area and also making the results available through a web platform. It demonstrated the models' capability of generalizing predictions spatially, however it was not possible to test their temporal generalization, due to the lack of data for the considered indicators in a wide range of years.

The models trained using ResNet-18 backbones and Ridge Regressors were evaluated for a multitude of experiments using MS and NL images and have yielded good results that prove that MS images are appropriate for predicting indicator values over the proposed dataset, achieving a correlation of up to 42% between the predicted and the target indicators. Models using only NL showed disappointing results, due to the characteristic of very low lit rural areas in Brazil. For this reason, NL provides little to no additional information to predictive models.

Furthermore, it is also evidenced that for the socioeconomic data in the proposed dataset, longevity and literacy indicators skew away from a normal curve and thus, cannot be as effectively estimated.

9.2. Contributions

This project demonstrates that it is possible to estimate socioeconomic indicators in Brazil using satellite images by creating and documenting a step-by-step methodology for training and evaluating CNNs for this task. The scripts developed for executing the methodology pipeline and the interactive platform source code are open source and can be easily adapted and expanded for use in other applications that involve visualizing data on maps.

Besides, the project provides a clear analysis on the peculiarities of the Brazilian scenario by illustrating the characteristics of socioeconomic indicators, as well as creating a new dataset of socioeconomic indicators and their relevant geographic metadata.

Lastly, a prototype of an interactive platform to visualize estimated indicators is presented and can be further improved for future works and studies.

9.3.Future work

Future work suggested is to expand the experiments performed to estimate a more diverse set of indicators throughout the Brazilian territory but also use.

The pipeline can also be adapted to work with data in a wider range of years, in order to evaluate the models' capability of temporal generalization, raising the need of collecting more data, from different census surveys, to train the models.

Regarding the proposed models' performances, there can be further investigation on NL bad results, attempting new training pipelines with different hyperparameters or techniques. In addition to that, running new experiments over literacy and longevity is also interesting, standardizing its values and increasing the dataset available.

The prototype of the web platform also needs to be improved as it reaches the proposed designed design and workflow of the distributed architecture backend.

References

[1] CDB. Brazil - Main Details from the Convention on Biological Diversity. Available at: <https://www.cbd.int/countries/profile/?country=br>. Accessed on December 6th, 2022.

[2] IBGE. São Paulo. Brasil em Síntese - Território. Available at: <https://brasilemsintese.ibge.gov.br/territorio.html>. Accessed on June 3rd, 2022.

[3] IBGE. São Paulo. Censo Demográfico. Available at: <https://www.ibge.gov.br/estatisticas/sociais/populacao/22827-censo-2020-censo4.html?=&t=o-que-e>. Accessed on June 4th, 2022.

[4] G1. Censo 2022 começa em todo o país. Available at: <https://g1.globo.com/economia/noticia/2022/08/01/censo-demografico-2022-comeca-nesta-segunda-feira.ghtml>. Accessed on November 15th, 2022.

[5] G1. Censo 2022: com menos da metade dos recenseadores necessários em campo, quase 64% da população já foi recenseada, diz IBGE. Available at: <https://g1.globo.com/economia/noticia/2022/11/01/censo-entrevistou-136-milhoes-de-pessoas-mais-de-60percent-da-populacao.ghtml>. Accessed on November 16th, 2022.

[6] G1. IBGE adia fim de coleta do Censo por falta de recenseadores. Available at: <https://g1.globo.com/economia/noticia/2022/10/03/ibge-adia-fim-de-coleta-do-censo-por-falta-de-recenseadores.ghtml>. Accessed no November 17th, 2022.

[7] TAMIMINIA, H.; SALEHI, B., et al. Google Earth Engine for geo-big data applications: A meta-analysis and systematic review. ISPRS J. Photogrammetry Remote Sens., v. 164, p. 152-170, 2020.

[8] XIE, M.; JEAN, N.; BURKE, M.; LOBELL, D.; ERMON, S. Transfer Learning from Deep Features for Remote Sensing and Poverty Mapping. 30th AAAI Conference on Artificial Intelligence, 2016.

[9] AYUSH, K.; UZKENT, B.; BURKE, M.; LOBELL, D.; ERMON, S. Generating Interpretable Poverty Maps using Object Detection in Satellite Images, p. 4410–4416, 2020.

[10] JEAN, N.; BURKE, M.; XIE, M.; DAVIS, W. M.; LOBELL, D. B.; ERMON, S. Combining satellite imagery and machine learning to predict poverty. Science, v. 353(6301), p. 790–794, 2016.

[11] YEH, C.; PEREZ, A.; DRISCOLL, A.; AZZARI, G.; TANG, Z.; LOBELL, D.; ERMON, S.; BURKE, M. Using publicly available satellite imagery and deep learning to understand economic well-being in Africa. Nature Communications, v. 11(1) , p. 1–11, 2020

[12] IBGE. Brazil's political and administrative divisions. Available at: <https://educa.ibge.gov.br/jovens/conheca-o-brasil/territorio/18310-divisao-politico-administrativa-e-regional.html>. Accessed on December 2nd, 2022.

[13] IBGE, Malha de Setores Censitários, Saiba mais - 2021, Malha Censitária. Avaiable at: <https://www.ibge.gov.br/geociencias/organizacao-do-territorio/malhas-territoriais/26565-malhas-de-setores-censitarios-divisoes-intramunicipais.html?=&t=saiba-mais-edicao>. Accessed on July 5th, 2022.

[14] MITCHELL, T. *Machine Learning*. New York: McGraw Hill, 1997. ISBN 0-07-042807-7. OCLC 36417892.

[15] JANIESCH, C.; Zschech, P.; Heinrich, K. Machine learning and deep learning. Electron Markets, n. 31, p. 685–695, 2021.

[16] IBM. What is the k-nearest neighbors algorithm?. Available at: <https://www.ibm.com/topics/knn>. Accessed on December 7th, 2022.

[17] PATTERSON, J.; GIBSON, A. Deep Learning: A Practitioner's Approach. O'Reilly Media, Inc., 2017.

[18] GOODFELLOW, I. et al. Deep Learning. MIT Press, 2016.

[19] O'SHEA, K.; NASH, R. An Introduction to Convolutional Neural Networks. 2015.

[20] TOWARDS DATA SCIENCE. Intuitively Understanding Convolutions for Deep Learning. Available at: <https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1>. Accessed on December 8th, 2022.

[21] PAPERSPACE. Pooling in Convolutional Neural Networks. Available at: <https://blog.paperspace.com/pooling-in-convolutional-neural-networks/>. Accessed on December 9th, 2022.

[22] ZHU, W.; YEH, W.; CHEN, J.; CHEN, D.; LI, A.; LIN, Y. Evolutionary Convolutional Neural Networks Using ABC. ICMLC '19: Proceedings of the 2019 11th International Conference on Machine Learning and Computing. p. 156-162, 2019.

[23] TUBA, E.; BAČANIN, N.; STRUMBERGUER, I.; TUBA, M. Convolutional neural networks hyperparameters tuning. In Artificial intelligence: theory and applications, pp. 65-84, Springer, Cham, 2021.

[24] RICHTER, M. L., et al. (Input) Size Matters for CNN Classifiers. International Conference on Artificial Neural Networks. Springer, Cham, 2021.

[25] YOSINSKI, J. et al. How transferable are features in deep neural networks?. Advances in neural information processing systems, n. 27, 2014.

[26] JAMES, G. et al. An Introduction to Statistical Learning with Applications in R. 1st ed. Springer, p. 440, 2013.

[27] SEBASTIANRASCHKA. Feature Scaling and Normalization. Available at: <https://sebastianraschka.com/Articles/2014_about_feature_scaling.html#about-standardization>. Accessed on December 5th, 2022.

[28] HOERL, A. E.; KENNARD R. W. Ridge Regression: Biased Estimation for Nonorthogonal Problems.Technometrics, v. 42, n. 1, p. 80-86, 2000.

[29] KIRCH, W. Pearson's Correlation Coefficient. Encyclopedia of Public Health. Springer, Dordrecht, 2008.

[30] SPEARMAN, C. The proof and measurement of association between two things. By C. Spearman, 1904. The American journal of psychology, n. 100 3-4, p. 441-71, 1987.

[31] PEACOCK R. Distributed architecture technologies. IT Professional, v. 2, n. 3, p. 58-60, May-June 2000.

[32] AWS. Global Infrastructure. Available at: <https://aws.amazon.com/about-aws/global-infrastructure/>. Accessed on October 7th, 2022

[33] AWS. Serverless Computing. Available at: <https://aws.amazon.com/serverless/, accessed on October 7th, 2022.

[34] QGIS. Sobre o QGIS. Available at: <https://qgis.org/pt_BR/site/about/index.html>. Accessed on September 14th, 2022.

[35] USGS. What is a geographic information system (GIS)?. Available at: <https://www.usgs.gov/gis>. Accessed on September 14th, 2022.

[36] GOOGLE. Google Earth Engine. Available at: <https://earthengine.google.com>. Accessed on September 14th, 2022.

[37] TENSORFLOW. TensorFlow Library. Available at: <https://www.tensorflow.org/>. Accessed on September 14th, 2022.

[38] MATPLOTLIB. Matplotlib Library. Available at: <https://matplotlib.org/>. Accessed on September 14th, 2022.

[39] PANDAS. Pandas Library. Available at: <https://pandas.pydata.org/>. Accessed on September 14th, 2022.

[40] GEOPANDAS. GeoPandas Library. Available at: <https://geopandas.org/en/stable/. Accessed on September 14th, 2022.

[41] PYSAL. Python Spatial Analysis Library. Available at: <https://pysal.org/libpysal/>. Accessed on September 14th, 2022.

[42] SCIKIT-LEARN. Scikit-learn Library. Available at: <https://scikit-learn.org/stable/>. Accessed on September 14th, 2022.

[43] MACHICAO, J. et al. Satellite images and deep learning for the prediction of socioeconomic indicators in the Vale do Ribeira. Anais do XVI Brazilian e-Science Workshop, Sociedade Brasileira de Computação, 2022.

[44] ABREU, M. V. S. et al. Methodological proposal for spatial calculation and analysis of the intra-urban HDI of Viçosa, Brazil. Revista Brasileira de Estudos de População 28, p. 169-186, 2011.

[45] PNUD. Índice de Desenvolvimento Humano Municipal -IDHM: Metodologia. (2012). Available at: <https://atlasbrasil.org.br/acervo/atlas>. Accessed on December 10th, 2022.

[46] He, H.; GARCIA, E. A. "Learning from Imbalanced Data". IEEE Transactions on Knowledge and Data Engineering, v. 21, n. 9, p. 1263-1284, Sept. 2009.

[47] KRAWCZYK, B. Learning from imbalanced data: open challenges and future directions. Prog Artif Intell, n. 5, p. 221–232, 2016.

[48] BARZ, B.; JOACHIM, D. Do we train on test data? purging cifar of near-duplicates. Journal of Imaging, v. 6(6), p. 41, 2020.

[49] EPSG.IO. EPSG:4326. Available at: <https://epsg.io/4326>. Accessed on June 26th, 2022.

[50] GOOGLE. Google Earth Engine Projections. Available at: <https://developers.google.com/earth-engine/guides/projections>. Accessed on December 10th, 2022.

[51] AZZARI, G.; LOBELL, D. B. Landsat-based classification in the cloud: an opportunity for a paradigm shift in land cover monitoring. Remote Sens. Environ., n. 202, p. 1–11, 2017.

[52] THE PHYSICS HYPERTEXTBOOK. Electromagnetic Spectrum. Available at: <https://physics.info/em-spectrum/>. Accessed on November 21st, 2022.

[53] EOS. Satellite Imagery And Spectral Band Combinations - Thermal Band. Available at: <https://eos.com/make-an-analysis/thermal/>. Accessed on November 20th, 2022.

[54] EOS. Satellite Imagery And Spectral Band Combinations - Shortwave Infrared. Available at: <https://eos.com/make-an-analysis/shortwave-infrared/>. Accessed on November 23rd, 2022.

[55] EOS. Satellite Imagery And Spectral Band Combinations - Land/Water. Available at: <https://eos.com/make-an-analysis/land-water/>. Accessed on November 25th, 2022.

[56] HSU, F.; BAUGH, K.; TILOTTAMA, G.; ZHIZHIM, M.; ELVIDGE, C. DMSP-OLS Radiance Calibrated Nighttime Lights Time Series with Intercalibration. Remote Sensing, n. 7, p. 1855-1876, 2015.

[57] ELVIDGE, C.; BAUGH, K.; ZHIZHIN, M.; Hsu, F.; Tilottama, G. VIIRS night-time lights. International Journal of Remote Sensing, n. 38, p. 1-20, 2020.

[58] ROLF, E.; PROCTOR, J.; CARLETON, T; et al. A generalizable and accessible approach to machine learning with global satellite imagery. Nat Commun, v. 12, p. 4392, 2021.

[59] ENDO, T.; MATSUMOTO, M. Aurora Image Classification with Deep Metric Learning. Sensors, v. 22, p. 6666, 2022.

[60] HE, K.; ZHANG, X.; REN, S.; SUN, J. Delving deep into rectifiers: Surpassing human level performance on imagenet classification. In Proc. 2015 IEEE International Conference on Computer Vision (ICCV), n. 15, p. 1026–1034, Washington, 2015.

[61] ADAPTA BRASIL. AdaptaBrasil MCTI Main page. Available at: <https://adaptabrasil.mcti.gov.br/>. Accessed on September 25th, 2022.

[62] AWS. Regions and Availability Zones. Available at: <https://aws.amazon.com/about-aws/global-infrastructure/regions_az/>. Accessed on October 7th, 2022.

[63] AWS. Auto Scaling. Available at: <https://aws.amazon.com/autoscaling/>. Accessed on October 7th, 2022.

[64] AWS. AWS Lambda. Available at: <https://aws.amazon.com/lambda/>. Accessed on October 7th, 2022.

[65] AWS. AWS Fargate. Available at: <https://aws.amazon.com/fargate/>. Accessed on October 7th, 2022.

[66] IBGE. São Paulo. Malha de Setores Censitários - 2010. Available at: <https://www.ibge.gov.br/geociencias/organizacao-do-territorio/malhas-territoriais/26565-malhas-de-setores-censitarios-divisoes-intramunicipais.html?edição=26589&t=saiba-mais-edição>. Accessed on June 10th, 2022.

[67] EPSG.IO. EPSG:3857. Available at: <https://epsg.io/3857>. Accessed on December 7th, 2022.

[68] GOOGLE. Earth Engine Apps. Available at: <https://www.earthengine.app>. Accessed on September 14th, 2022.

[69] PYTHON. Python programming language. Available at: <https://www.python.org/>. Accessed on September 14th, 2022.

[70] GOOGLE. Imbalanced data. Available at: <https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data>. Accessed on December 10th, 2022.

Appendix A - Backend sequence diagram

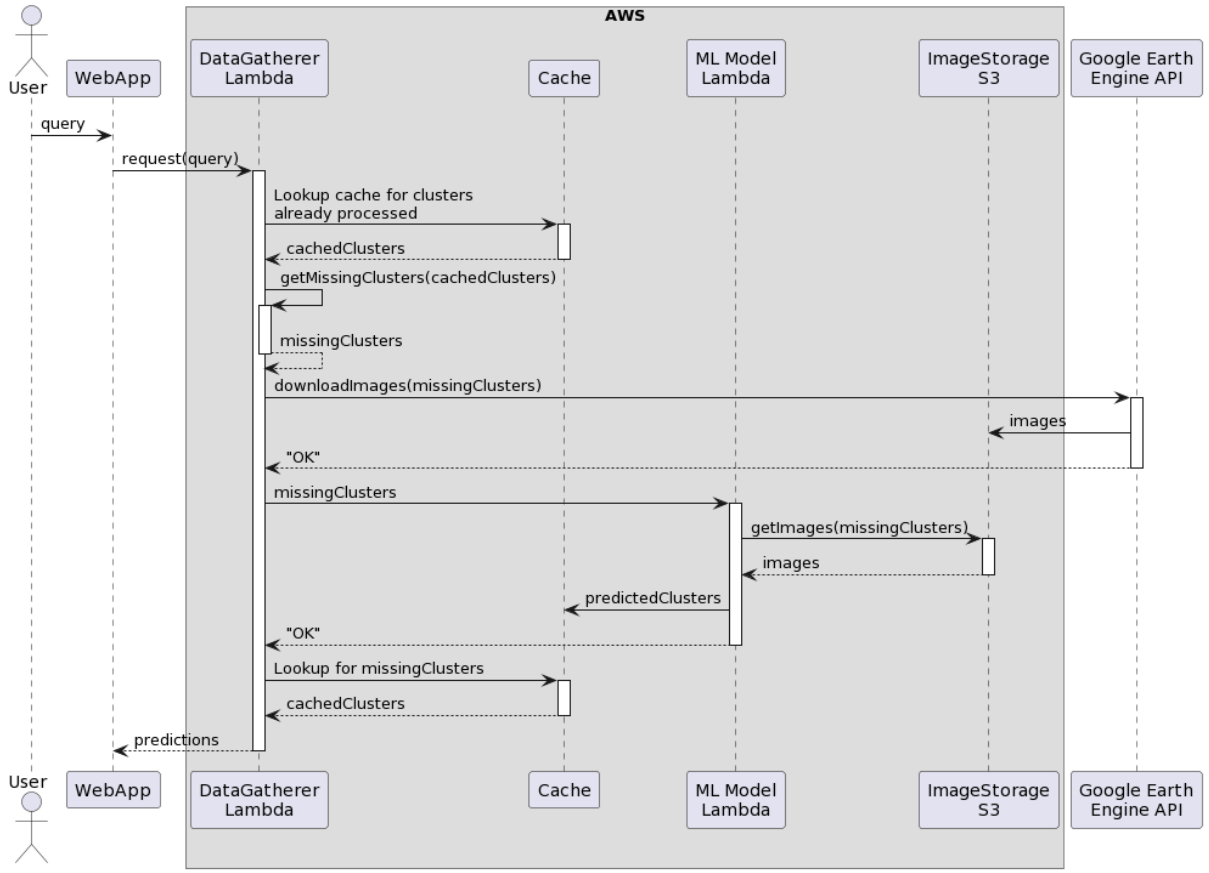


Figure 38 - Sequence diagram of the web application

Appendix B - Experiment results for literacy and longevity

	Experiment	r2	R2	mse	rank
Deep Learning models	MS+NL	0.24510	0.24364	0.03350	0.58801
	Ridge-MS	0.24503	0.24358	0.03351	0.58832
	Ridge-NL	0.20636	0.20627	0.03516	0.55688
Baseline models	Ridge MS+NL hist	0.20202	0.20193	0.03535	0.55035
	Ridge MS hist	0.14757	0.14755	0.03776	0.47371
	Ridge RGB+NL hist	0.14410	0.14409	0.03791	0.46590
	Ridge RGB hist	0.00920	0.00827	0.04393	0.02965
	Ridge NL hist	0.00786	0.00786	0.04395	0.02205
	KNN NL center scalar	0.00713	0.00604	0.04403	0.02327
	KNN NL hist	0.00502	0.00305	0.04416	0.03312
	Ridge NL mean scalar	0.00419	0.00419	0.04411	0.01700
	Ridge NL center scalar	0.00410	0.00410	0.04411	0.02085
	KNN NL mean scalar	0.00392	0.00392	0.04412	0.02051

Table 12 - Comparison of experimented models' metrics for Literacy indicator.

	Experiment	r2	R2	mse	rank
Deep Learning models	MS+NL	0.15937	0.15879	0.02331	0.37018
	Ridge-MS	0.15833	0.15771	0.02334	0.36786
	Ridge-NL	0.11151	0.11120	0.02463	0.30331
Baseline models	Ridge MS+NL hist	0.11000	0.10976	0.02467	0.29652
	Ridge MS hist	0.07860	0.07858	0.02554	0.24714
	Ridge RGB+NL hist	0.07538	0.07537	0.02563	0.23278
	Ridge RGB hist	0.01993	0.01989	0.02716	0.11532
	Ridge NL hist	0.01992	0.01981	0.02717	0.11136
	KNN NL center scalar	0.01107	0.01106	0.02741	0.10880
	KNN NL hist	0.00685	0.00682	0.02753	0.06599
	Ridge NL mean scalar	0.00542	0.00490	0.02758	0.03594
	Ridge NL center scalar	0.00098	0.00098	0.02769	0.03392
	KNN NL mean scalar	0.00092	0.00092	0.02769	0.05705

Table 13 - Comparison of experimented models' metrics for Longevity indicator.