Estimating Poverty from Aerial Images Using Convolutional Neural Networks Coupled with Statistical Regression Modelling

Vongani Maluleke

Supervisor: Dr Sebnem Er

Co-Supervisor: Dr Quentin Williams

A Minor Dissertation submitted to the Faculty of Science, University of the Cape Town, Cape Town, in partial fulfilment of the requirements for the Master of Science in Advanced Analytics.

Cape Town, October 2019

Dedication

Ndzi khensa Hosi, hikuva "lexi xi nga heriku xa hlola" – Tsonga Proverb.

[Everything has an end]

Acknowledgement

I would like to firstly thank God for his favour and grace upon my life. Thank you for giving me the knowledge and strength to achieve my goals.

I would like to further thank my siblings, Siphiwe Maluleke, Dzunisani Maluleke, Makhensa Maluleke and Mikateko Maluleke for the moral support that has got me this far. Your encouragement, support and love has been very impactful in my life. To my niece, Zoe Maluleke, you've brought happiness to my life during a time I needed it the most.

To my grandmother, Norah Mabaso, everything I am today is a result of you making and selling homemade decoration cloths to send my mother to school. I thank you for all the sacrifices you've made that has had a generational ripple effect. The little you did back then is more than enough today.

To my father, Samson Maluleke, thank you supporting my academic ambitions and for being a good example of how education, discipline and hard work can help secure your future.

A special thank you to my mother, Nonhlanhla Jeanneth Maluleke, who has always motivated me to be the best I can be and to never settle when it comes to my dreams. From my first day of school you've always been my number one cheerleader. Today I can proudly tell you that all the sacrifices you've made were not in vain.

I would like to also thank the Meraka Institute at CSIR, for the endless academic and

financial support that they have given me. I've academically grown all thanks to the institution and all the international and local conferences that they have supported me to attend and present my work. This has had an incredible impact in my life and I will always be grateful.

I would like to also thank National Income Dynamics Study (NIDS) for granting me access to their datasets, DataFirst for allowing me to use their facilities to securely analyse the NIDS data, the National Geo-spatial Information (NGI) and with the assistance of Dr Konrad Wessels for giving me aerial images to use for my research and the Deep learning Indaba for the $1000 Google Cloud Platform (GCP) credits to train my models. This research wouldn't have been possible without the respective roles you played in this research.

Finally I would like to thank my supervisor Dr Sebnem Er and my co-supervisor Dr Quentin Williams for believing in me and academically supporting me through my Masters. You created a space for me to freely learn and explore the world of Statistics and Artificial Intelligence with no boundaries and encouraged my way of thinking. You've played a key role in this achievement and I thank you for all the contribution you've made.

Abstract

Policy makers and the government rely heavily on survey data when making policy-related decisions. Survey data is labour intensive, costly and time consuming, hence it cannot be frequently or extensively collected. The main aim of this research is to demonstrate how Convolutional Neural Network (CNN) coupled with statistical regression modelling can be used to estimate poverty from aerial images supplemented with national household survey data. This provides a more frequent and automated method for updating data that can be used for policy making. This aerial poverty estimation approach is executed in two phases; aerial classification and detection phase and poverty modelling phase. The aerial classification and detection phase use CNN to perform settlement typology classification of the aerial images into three broad geotype classes namely; urban, rural and farm. This is then followed by object detection to detect three broad dwelling type classes in the aerial images namely; brick house, traditional house, and informal settlement. Mask Region-based Convolutional Neural Network (Mask R-CNN) model with a resnet101 CNN backbone model is used to perform this task. The second phase, poverty modelling phase, involves using NIDS data to compute the poverty measure Sen-Shorrocks-Thon (SST) index. This is followed by using regression models to model the poverty measure using aggregated results from the aerial classification and detection phase. The study area for this research is Kwa-Zulu Natal (KZN), South Africa. However, this approach can be extended to other provinces in South Africa, by retraining the models on data associated with the location in question.

Contents

List of Figures

List of Tables

List Of Abbreviations

AAR Average Accuracy Rate

CI Confidence Interval

CNN Convolutional Neural Network

CV Cross Validation

DHS Demographic and Health Survey

FPL Food Poverty Line

GDP Gross Domestic Product

GCP Google Cloud Platform

ILSVRC ImageNet Large-Scale Visual Recognition Challenge

IoU Intersection over Union

KZN Kwa-Zulu Natal

LBPL Lower Bound Poverty Line

Mask R-CNN Mask Region-based Convolutional Neural Network

mAP mean Average Precision

MSE Mean Sqaured Error

NDP National Development Plan

NGI National Geo-spatial Information

NIDS National Income Dynamics Study

OLS Ordinary Least Squares

PMT Proxy-Means Test

PPI Poverty Probability Index

PCR Principle Component Regression

PWT Penn World Table

RSE Residual Square Error

SALDRU Southern Africa Labour and Development Research Unit

SST Sen-Shorrocks-Thon

UBPL Upper Bound Poverty Line

Notation and Nomenclature

$\boldsymbol{A}$: Matrix

A_{ij}: element ij in the Matrix $\boldsymbol{A}$

$\boldsymbol{a}$: Column vector

a_i: ith element in the vector $\boldsymbol{a}$

a: Scalar value

Chapter 1

Introduction

1.1 Motivation

Poverty is a very complex phenomenon that requires extensive research other than just analysing survey data. In this research, Convolutional Neural Network (CNN) in the field of computer vision is utilized to offer a different approach into estimating poverty. The main aim of this research is to demonstrate how CNN coupled with statistical regression modelling can be applied to estimate poverty from aerial images supplemented with national household survey data.

1.2 Objectives and Significance

This research will explore the use of CNN models to identify dwelling types and geo-types from aerial images to estimate the Sen-Shorrocks-Thon (SST), poverty rate. The end result of this research will assist policy makers to efficiently and frequently monitor the impact of the implemented policies. This will further help them adapt existing policies that are aimed to contribute to poverty alleviation as planned in the National Development Plan (NDP), which is the South African government policy document drafted by the National Planning Commission with the aim to eliminate

poverty and reduce inequality by 2030, nationally.

The objectives of this research are to:

- Create a CNN model that classifies three broad geo-types (urban, farm, and rural) from aerial images.
- Create a CNN model that detects three broad dwelling types (brick house, traditional house, and informal settlement) from aerial images
- Create a statistical regression model that estimates the poverty rate from the identified dwelling types and geo-types in the aerial images.
- Create a statistical regression model to understand the relationship between dwelling types, geo-types and the SST poverty rate in, KwaZulu-Natal, South Africa, which can further be extended to other provinces.

The significant contributions of this research are:

- Providing a CNN model that identifies the dwelling types (namely; brick house, traditional house and informal settlement) and geo-types (namely; urban, rural and farm) found in an aerial image. This will help policy makers understand the geo-spatial distribution of dwelling types and geo-types. This can further be used in the other fields such as town and rural planning and remote sensing.
- Providing a statistical regression model that estimates the SST poverty rate of a certain location from an aerial image. This is significant because policy makers will be able to perform on-demand assessment of the implemented policies to alleviate poverty.
- Providing a statistical regression model that understands the relationship between dwelling types, geo-types and the poverty rate in KwaZulu-Natal, South Africa, which can be subsequently used by policy makers to identify areas that

need more attention than others in terms of implementing policies to alleviate poverty.

1.3 Layout

This dissertation discusses the application of CNNs in the computer vision field coupled with a statistical regression model to estimate the poverty rate, SST, from aerial images supplemented with survey data.

Chapter 2 introduces poverty and how it is measured. It further also discusses how expenditure or income can be adjusted for different household compositions to make direct poverty comparisons.

Chapter 3 provides a review of the literature for previous research that has been done on estimating poverty using different forms of data and approaches. It also provides a detailed review for computer vision-based approaches for estimating poverty-related indices.

Chapter 4 describes the datasets used in this dissertation and the roles they play in the overall approach.

Chapter 5 presents the method design for estimating poverty from aerial images and also discusses the methods used that form the building blocks of the method design.

Chapter 6 provides a description of the environment set-up used to implement the research, the implemented data preparation and the evaluation metrics used to assess the performance of the models.

Chapter 7 provides the results obtained from the classification of geo-types, detection of dwelling types and the statistical regression results.

Chapter 8 provides the dissertation overview conclusion and summary of the results.

Chapter 9 provides recommended future work that can be done to improve the results.

Chapter 2

Poverty

2.1 Definition

Poverty is a complex, non-discriminatory, multi-dimensional phenomenon that constitutes of multiple deprivation aspects (Narayan et al., 1999). It is influenced by many factors, making it complex but not impossible to perform studies on. This is due to the many ways of measuring and defining poverty.

There are two broad poverty study categories, which are objective and subjective poverty studies. Objective poverty studies involve using data collected from a direct observation of the subject matter, for example household demographics (Instituto Nacional de Estadstica, 2017). Subjective poverty studies are on data collected from the perception of the subject matter's standard of living (Instituto Nacional de Estadstica, 2017). This research will implement an objective poverty study because observable and measurable factors are going to be used to measure poverty.

2.2 Measurement of Poverty

2.2.1 Household Adjustment

Poverty can be measured at either an individual or household level. The latter level is preferred, because individual income and expenditure is often pooled within the household (Poobalan et al., 2007). Households differ in both size and demographic composition, this results in difficulty of making straight forward poverty comparisons (Bhorat et al., 2001). This problem can be solved by normalising the households in two approaches. The first approach of household normalisation involves dividing the total income or expenditure by the number of household members (Woolard et al., 1999; Bhorat et al., 2001). This approach is not efficient because it does not allow for economies of scale within the households.

The other household normalisation approach involves using households equivalence scales to adjust the household income or expenditure. This approach allows us to make direct comparisons between households of different sizes and demographic compositions (Bhorat et al., 2001). Household equivalence scale is a measure of the cost of living of a household of a given size and demographic composition relative to the cost of living of a reference household, which is usually a single adult, when both households attain the same level of standard of living (Lewbel and Pendakur, 2006).

There are two types of adjustments that can be made under the second household normalisation approach. The first type involves multiplying household income or expenditure by a factor to allow economies of scale. The second adjustment type involves applying different weights to household members based on whether they are categorised as a child or an adult household member (Bhorat et al., 2001). This adjustment takes into account the different consumption requirements of households with different demographic composition (Bhorat et al., 2001).

Once the necessary household income or expenditure adjustments have been made,

different poverty measures can now be considered to perform direct poverty comparisons. The following section will discuss in detail the different poverty measures including the poverty measure that will be used in this research.

2.2.2 Poverty Measures

Poverty measures can be regarded as the guiding tools of how poverty is monitored, analysed and assessed in a country. This is because policy makers structure poverty reduction aimed policies based on how poverty is defined and measured. Poverty measures can be broadly divided into two categories, ends measure and means measure. Ends poverty measures are measures that use a desired outcome or end to categorise entities as being in poverty, for example nutritional status (Lok-Dessalien, 2000). Means poverty measures are measures that considers inputs to achieve the eradication of poverty as proxy measures of poverty, for example income and asset ownership (Lok-Dessalien, 2000).

Mean poverty measures are preferred because the data used to compute them is easier to obtain. Furthermore, data used on ends poverty measures is not suitable for monitoring poverty over a short to medium time because the change overtime is relatively slow (Lok-Dessalien, 2000).

The South African government, policy makers and National Development Plan (NDP) use poverty lines to set and monitor poverty alleviation targets. A poverty line is an index that is used to divide the population into two groups based on some measure such as welfare (Bhorat et al., 2001). Individuals or households that fall below the poverty line are considered poor, while those that fall above the poverty line are not.

South Africa has three types of national poverty lines that were first published in 2012 by an organisation called Statistics South Africa and have been recalculated and republished numerous times thereafter by the same organisation. The latest published poverty lines were released in 2017 and contain 2006-2017 poverty line estimates, see

appendix A.1 for the latest poverty line table. The three types of national poverty lines are namely Food Poverty Line (FPL), Lower Bound Poverty Line (LBPL) and Upper Bound Poverty Line (UBPL) and were calculated by using the cost-of-basic-needs approach, which is the most commonly used approach (Statistics South Africa, 2017). This approach estimates the daily cost of acquiring sufficient food to obtain adequate nutrition per person, and the cost of other essentials, for example, shelter and clothing (Haughton, 2009).

These poverty lines were introduced with the intention of enabling the country to measure and monitor poverty at different levels (Statistics South Africa, 2017). The FPL is the monetary value below which individuals or households (entities) cannot afford purchasing or consuming adequate food to obtain minimum per capita per day energy required for adequate health. Both the LBPL and UBPL are derived by incorporating FPL and non-food items. The difference between LBPL and UBPL is that entities at LBPL cannot afford both food and non-food items and hence end up sacrificing food for essential non-food items. The entities at UBPL can afford purchasing both food and non-food items (Statistics South Africa, 2017).

The poverty line is an imperfect construct because it makes the crude assumption that entities who are one unit below the poverty line is living in poverty while entities who are one unit above the poverty line is not (Bhorat et al., 2001). Furthermore, Beckerman (1984) has also argued that it is not sensible to define poverty by using some minimum level when people continue to survive below it. For the purpose of analysis, one frequently has to draw a line somewhere to further understand the nature of poverty as argued by Bhorat et al. (2001).

The three types of poverty lines can be used to calculate commonly used poverty measures such as the head count index and the poverty gap index. Head count index is the proportion of people who fall below the poverty line (Haughton, 2009). This index represents the incidence of poverty as it measures the proportion of poor people

(P_0) and is calculated as follows:

$$P_0 = \frac{1}{N} \sum_{i=1}^{N} I(y_i < z) \tag{2.1}$$

where N is the total population (or sample), and $I(\cdot)$ is an indicator function that returns one when the income or expenditure of the ith entities (y_i) is less than the poverty line (z) and returns zero otherwise. The head count index is popularly used because it is easy to understand and construct (Haughton, 2009). However, it does not take into account the intensity of poverty nor does it indicate the degree of how poor the poor are.

The poverty gap index is a measure of the average poverty gap in the population, expressed as the percentage of the poverty line (Haughton, 2009). Poverty gap (g_i) represents the depth of poverty, as it is the extent to which entities lie below the poverty line and is calculated as:

$$g_i = (z - y_i) \cdot I(y_i < z) \tag{2.2}$$

where z is the poverty line, y_i is the ith household's expenditure or income and $I(\cdot)$ is the indicator function as explained above. From Eq.(2.2), entities that are not poor will have a zero poverty gap while poor entities will have a poverty gap greater than zero. The poverty gap index (P_1) is calculated as:

$$P_1 = \frac{1}{N} \sum_{i=1}^{N} \frac{g_i}{z} \tag{2.3}$$

where the mean of poverty gap Eq.(2.2), with respect to all the entities is taken and divided by the poverty line to obtain the mean poverty gap as a proportion of the poverty line.

These poverty measures only capture one aspect of poverty, which is misleading and results in false conclusions (Poobalan et al., 2007). It is highly recommended to use a composite poverty measure because it provides a robust view of poverty (Poobalan et al., 2007). Sen (1976) proposed a poverty measure that takes into account the population size of the poor, income or expenditure shortfall relative to the poverty line and the degree of inequality. This measure is called the Sen index, denoted P_S, and is defined as:

$$P_S = P_0(P_1^P + (1 - P_1^P)G^P) \tag{2.4}$$

where P_0 is the head count index, P_1^P is the poverty gap index for the poor population, and G^P is the Gini coefficient of the poverty gap ratios also of only the poor population.

Gini coefficient is a measure of inequality for a given distribution and is based on the Lorenz curve (Haughton, 2009). It ranges between zero and one and is often used to measure wealth inequality, where zero indicates perfect wealth equality and one indicates perfect wealth inequality.

Shorrocks (1995) later proposed a modified version of Sen index which was then firstly modified by Thon (1979) to become the Sen-Shorrocks-Thon (SST) index (Haughton, 2009). This poverty measure takes into account the depth, incidence and inequality of poverty.

The SST index is given by Eq.(2.5), which is the product of the head count index (P_0), the poverty gap index (P_1^P) in the poor population, and the Gini coefficient of the poverty gaps ($\hat{G}^P$) (i.e. Gini coefficient of the g_i's given by Eq.(2.2)) of the entire population:

$$P_{SST} = P_0 P_1^P (1 + \hat{G}^P) \tag{2.5}$$

The SST index is a more appealing version of the Sen index because decomposing the SST index allows for analysis of source of changes in poverty overtime (Haughton, 2009). This decomposition enables us to know over a certain amount of time:

- if the number of poor people increased,
- if the poor have become poorer, and
- if the inequality among the poor improved.

The SST index can be expressed in logarithmic terms as follows:

$$\triangle lnP_{SST} = \triangle lnP_0 + \triangle lnP_1^P + \triangle ln(1 + \hat{G}^P) \tag{2.6}$$

which is interpreted as the percentage change in SST index, which is equal to the sum of the percentage change in the depth, incidence and inequality of poverty.

In this research, the SST index will be used for measuring poverty in a location and will be focussing on point poverty estimation.

Chapter 3

Literature Review

The South African government has compiled a plan called the National Development Plan to combat poverty and reduce inequality in South Africa by 2030 (National Planning Commission, 2011). Policy makers rely heavily on survey data because it has been the main source of monitoring the development and growth of the country. Due to limited resources, it is not always frequently and extensively done to a level where the data properly reflects the true livelihood of individuals in South Africa.

There is a faster alternative for measuring poverty using nationally-representative survey data, which is less expensive and burdensome to execute, called the Proxy-Means Test (PMT). PMT is a scorecard or formula that uses 10 to 30 survey questions to estimate the probability of a household being poor (Kshirsagar et al., 2017). This involves selecting a subset of variables from the survey data, training a model using the selected subset of variables to estimate poverty at household level, translate the model into a PMT scorecard, and validate the PMT results using out-of-sample data and real world field tests (Kshirsagar et al., 2017).

PMT is a traditional approach that is constructed by using stepwise logistic regression, which has high variability in the model selection step, resulting in a high prediction error (Brown et al., 2016). Kshirsagar et al. (2017) introduced a novel approach, called

the Poverty Probability Index (PPI), to construct PMT using machine learning methods to perform variable selection and regression to overcome the shortcomings of the traditional approach. Using Zambia household survey data, they demonstrated that their approach performs variable selection better than the traditional approach and hence estimates the PMT scorecards with a better predictive accuracy rate (Kshirsagar et al., 2017).

Over the past few years, researchers have been combining different data sources with survey data to enhance the prediction of poverty in a certain region using computational methods. Blumenstock et al. (2015) used anonymised data from Rwanda's largest mobile phone network to predict the subscribers' poverty status and further create a geographic distribution of wealth in Rwanda. They demonstrated that an individual's historic records of mobile phone usage can be utilised to infer the individual's socio-economic status and also accurately reconstruct the distribution of wealth in the Rwanda.

Blumenstock et al. (2015)'s approach involves combining feature engineering and feature selection, where they used the wealth index to measure poverty. They trained an elastic net model using 5 fold cross-validation and found a strong correlation of $r = 0.916$ between district level wealth predicted by mobile phone data and average wealth of households from Demographic and Health Survey (DHS) data. Hence concluded that their approach can be used to approximate national wealth distribution in Rwanda. However, the reliance of anonymised phone usage data makes it an infeasible approach because some network providers have strict rules about sharing their subscribers' phone usage data.

Researchers (Elvidge et al., 1997; Xi Chena and William D. Nordhausa, 2011; Xie et al., 2016), have investigated the use of night-time luminosity, which is the night-time light intensities, as a proxy for economic indicators. Elvidge et al. (1997) found that there is a strong correlation between Gross Domestic Product (GDP) and luminosity data at a country level and their linear regression model estimating GDP using

luminosity had an R^2 of 0.97. This indicates that luminosity of an area explain 97% of the variability of GDP.

Xi Chena and William D. Nordhausa (2011) demonstrated how much luminosity data contributes to the construction of the true GDP measures by using the weight fraction on luminosity (θ_i) of their luminosity-output proxy to measure the contribution. They found that θ_i for countries with grade D is approximately 30% while for grade $A-C$ countries is less than 3%, where the grades are defined by Penn World Table (PWT). PWT is a grading system that involves assigning qualitative grades from A to D to countries based on several criteria. Their results indicate that luminosity data adds considerable value to countries with a PWT grade D than to countries with grade $A-C$.

Recently, many researchers have attempted to use machine learning techniques to measure poverty. Xie et al. (2016) used night time light intensities as a proxy to predict and map poverty at a country or continental level. They implemented transfer learning by training a fully convolutional CNN model to predict night time light intensity from the day time satellite images and simultaneously train another model that captures the effect of features on the satellite image to predict poverty. They obtained that daylight satellite images can be utilised to make relatively accurate spatial economic status predictions across Nigeria, Uganda, Malawi, and Tanzania (Jean et al., 2016; Xie et al., 2016). Xie et al. (2016)'s transfer learning approach of estimating poverty has outperformed models that are trained using passively collected data such as cellphone meta-data, because aerial images provide additional information that they leveraged on and used as a proxy for poverty. The results of research has ignited a field of potential ways of using satellite images to supplement survey data to perform studies that were data restricted due to poor available data.

Head et al. (2017) explored the extent in which Xie et al. (2016)'s approach to estimate poverty can be used to estimate a broader set of socio-economic indicators in different regions. Head et al. (2017) demonstrated that Xie et al. (2016) approach

can be generalised to other countries and continents. However they found that the performance of the model is sensitive to the model hyperparameters used. Furthermore, Head et al. (2017) found that this approach does not generate a model that accurately estimates other socio-economic indicators such as access to water, electrification, education level and child weight-for-height-index, as it estimates asset-based wealth indexes such as the GDP.

Engstrom et al. (2017) further investigated the use of night time lights and high resolution satellite images to estimate poverty. They found that features from satellite images explains approximately 60% of the variation of poverty and a model built using night time lights explained 15% of the variation of poverty (Engstrom et al., 2017). They also found that built-up area and roof type have a strong correlation with welfare. These satellite object features were identified using deep learning-based CNN and classification of spectral and textural traits.

Recently, Gebru et al. (2017) explored the use of survey data, presidential election voting data and collection of cars observed in an American neighbourhood using Google street view to estimate the neighbourhoods demographics. Their findings suggest that demographic, socio-economic features and voting patterns can be extrapolated from the type of cars observed in neighbourhoods using Google street view (Gebru et al., 2017). This research will be following a similar paradigm where a combination of different data sources, specifically South African aerial images and survey data are used to estimate poverty, by coupling CNN and statistical regression models.

Chapter 4

Data Description

The data that will be used in this research comes from two different data sources, which are aerial photographs and survey data. The data sources are of the same context and are concurrent. This chapter will discuss the two different data sources in detail and the role(s) that they play in the research.

4.1 Survey Data

South Africa is a developing country that has made substantial infrastructural development over the past few years, which can be used as an indicator for economic development (Queiroz and Gautam, 1992). Policy makers and the government rely heavily on survey data to make well informed decisions and plans, especially when it comes to measuring and monitoring poverty. Contrary to the beneficial state of survey data, conducting surveys is time-consuming, labour intensive and expensive, hence it is not frequently or extensively conducted.

Surveys are an important source of supporting information in South Africa that assist policy makers with well informed decisions and plans. A survey refers to selecting a relatively large sample of individuals from a pre-determined population of interest, followed by collecting relevant data in a standardised form from the individuals found

in the pre-determined population (Kelley et al., 2003). A standardised data collection process involves no manipulations or controlling of conditions and applying the same process to all survey participants (Kelley et al., 2003). The collected data is then used by researchers to make inference on the entire population (Kelley et al., 2003).

Cross-sectional and longitudinal surveys are two types of basic survey methods implemented (Babbie, 1990). Cross-sectional surveys are used to gather data on a population at a single point in time, while longitudinal surveys are used to gather data over a period of time (Babbie, 1990). There are three main longitudinal survey study types, namely; trend, cohort and panel studies (Babbie, 1990).

Trend studies are conducted on a particular population by repeatedly sampling (e.g. Sample from grade 12 learners in 2019, 2020, ...) and observing any changes that occur in the population over the period of study. (Babbie, 1990). Usually, trend studies are conducted over a long period of time and the samples (samples of grade 12 learners in 2018, followed a sample of grade 12 learners in 2019 and so on) are from the same population (e.g. grade 12 learners) but are not composed of the same individuals (Babbie, 1990). For example, performing multiple market targeted surveys by sampling from the same population. This is very similar to Cohort studies, the only difference is that the samples come from the same population of interest. For example, performing a survey on 2019 grade 12 students and then 2020 grade 12 students from the same high school.

Panel studies on the other hand, conduct studies on the same sample of individuals, which help researchers understand the changes in the population over time (Babbie, 1990). Despite this benefit, panel studies tend to be time-consuming, expensive and have a high attrition rate, which is defined as the drop out rate of respondents from a study (Babbie, 1990).

The common survey data collection methods include face-to-face interviews, telephone interviews and postal questionnaires (Kelley et al., 2003). These methods have their

weakness and strengths which make them suited for different use-cases. Face-to-face interviews involve the researcher physically interacting with the survey respondent while the other two methods use the telephone and cold mail, respectively. These two methods are cheap to conduct but usually have low response rates compared to the face-to-face interviews (Kelley et al., 2003). Majority of the surveys commissioned by the South African government are conducted face-to-face and some have been made publicly available for research purposes.

4.1.1 National Income Dynamic Survey (NIDS) data

The survey data that will be used in this research is the NIDS wave 3 (public and secured) from South Africa (Southern Africa Labour and Development Research Unit, 2016) and the main focus for this research will be KZN households. Public NIDS data is available and open to the public for research use, while secured NIDS data contains private household information such as their geo-coordinates. Secured NIDS data is only available to individuals granted access by DataFirst for a limited period and can only be viewed from the DataFirst labs located in the Economics building at the University of Cape Town.

The survey data was provided by DataFirst [1] and permission to access the secured data was granted for this research. NIDS was conducted by the Southern Africa Labour and Development Research Unit (SALDRU) to give a national representation of South Africa's residents in terms of nutrition and health, education, quality of life and economic activity (Chinhema et al., 2016).

NIDS is an income, expenditure, household and individual longitudinal survey. It has a national coverage and the lowest level of geographic aggregation is district municipality. The survey was designed with the sole purpose of analysing the effect of

[1] DataFirst is a research data service dedicated to opening up African survey and administrative micro-data to researchers and policy analysts.

the dimensions of the well-being of South Africans overtime (Chinhema et al., 2016). The survey data was collected via face-to-face questionnaires and so far the following NIDS waves have been conducted:

- Wave 1: Feb\2008 - Dec\2008
- Wave 2: May\2010 - Sept\2011
- Wave 3: May\2012 - Dec\2012
- Wave 4: Sept\2014 - Aug\2015

The survey data will perform two roles in this research, which are to:

- assist in manually labelling dwelling types and geo-types found in the aerial images. This will help ensure accuracy and consistency in aerial image labels.
- compute the actual poverty rate, SST index, for data preparation.
- train a statistical regression model to estimate SST index.

Missing Data

In this research, only households that had been successfully interviewed are going to be used and the rest will be discarded, as recommended by Chinhema et al. (2016). The survey data contain a variable called *interview outcome*. The variable was used to obtain households that were successfully interviewed. The number and the proportion of successful and unsuccessful households interviewed are shown in Table 4.1. Wave 1 has a 100% interview success rate because it is the first survey conducted and the attrition rate is 0 and the next wave experienced a decrease which was then followed by a steady increase in wave 3 and wave 4.

Variables

The household variables obtained, shown in Table 4.2, from NIDS wave 3 that will be used in this research are: *w3_expf, w3_hhsizer, w3_h_dwltyp, w3_geo2011, w3_prov2011, w3_dc2011, w3_gps_e,* and *w3_gps_s* (Southern Africa Labour and Development Re-

search Unit, 2016). The *w3_h_dwltyp* variable is the household dwelling type variable, which is categorical with 11 classes and have been reduced to three broad categories namely; *brick house, traditional house,* and *informal settlement*, and then converted into unordered integer values 0, 1, 2. *w3_geo2011* is the household *Geo-type* variable, another categorical variable with three classes namely; *urban, rural* and *farm* that are also converted into unordered integer values 0, 1, 2.

Table 4.1: Number and proportion of successful and unsuccessful households interviewed

Wave (Year)	Successfully Interviewed	Unsuccessfully Interviewed
Wave 1 (2008)	7296 (100 %)	0 (0 %)
Wave 2 (2010-2011)	6728 (73.7 %)	2399 (26.3 %)
Wave 3 (2012)	8033 (78.6 %)	2186 (21.4 %)
Wave 4 (2014-2015)	9620 (80.9 %)	2275 (19.1 %)

Table 4.2: NIDS wave 3 (2012) Variable Description

Variable	Description	Type
w3_expf	Household monthly food expenditure	Continuous
w3_hhsizer	number of household members	Integer
w3_h_dwltyp	Household dwelling type	Categorical
w3_geo2011	Household geo-type	Categorical
w3_prov2011	Household province	Categorical
w3_dc2011	household district	Categorical
w3_gps_e	Household longitude	Continuous
w3_gps_s	Household latitude	Continuous

The *w3_expf* and *w3_hhsizer* variables are *monthly food expenditure* and *number of household members* that are continuous and integer values greater than zero, respectively. These variables were used to derive a new monthly food expenditure variable that has been adjusted for the different household sizes. This new variable is called *monthly food expenditure per household member* and will be used in the computation

of the poverty measure, SST index.

w3_prov2011, w3_dc2011, w3_gps_e, w3_gps_s are the location variables (household province, district, longitude, and latitude), which play an important role, as they are used to create a link between the number of dwelling types found within each of the three geo-types and the computed poverty measure from NIDS. Furthermore, these variables can be used to observe SST index and perform poverty estimations at different granular levels.

The actual SST index is computed at topological area code level using Eq.(2.5) and NIDS wave 3 survey data. The number of dwelling types found within each of the geo-types are aggregated and standardised by dividing by the number of households found within the particular location. Table 4.3 represents the variables that are used in this research with the associated value ranges.

Table 4.3: Variables used in poverty modelling.

Variable	Type	Range
Actual SST	Response	0-1
urban brick house	Independent	0-1
urban traditional house	Independent	0-1
urban informal settlement	Independent	0-1
rural brick house	Independent	0-1
rural traditional house	Independent	0-1
rural informal settlement	Independent	0-1
farm brick house	Independent	0-1
farm traditional house	Independent	0-1
farm informal settlement	Independent	0-1

4.2 Aerial Photographs

Aerial photographs are digital images taken from an aircraft, helicopter, or any other flying object. We used aerial photographs because they are less expensive than satellite images and can be readily taken on demand, such as when there is need to estimate poverty in a particular region.

The aerial photographs for the period 2013 were provided by the NGI, which is South Africa's national mapping organisation. These aerial photographs are vertical aircraft-taken photographs with a contact size of 23x23cm for the year 2013 of Kwa-Zulu Natal (KZN), South Africa. The photographs are orthophotos, as they have been geometrically corrected to remove topographic displacement and enlarged to a scale of 1:10 000. This image rectification allows for true distance measurements between features to be performed in the same manner as a map, as the ground truth is well represented.

Each photograph is a national map series identifiable by an unique seven or eight alphanumeric sequence of values, which will refer to as topological code. Since the research focus is on Kwa-Zulu Natal (KZN), see Appendix B Figure B.1 for the map sheet layout with topological area codes. The first four digits represent the combination of latitude and longitude coordinates of the top left corner of the degrees square. A degrees square is a single orthophoto grid cell where each corner is defined by the latitude and longitude coordinates as illustrated in Figure 4.1 (a). For example, in Figure 4.1(a) the top corner latitude and longitude coordinates are 29° and 31°, respectively and so the first four values of the unique topological code for photographs found within the degrees square is given as 2931. To obtain the rest of the topological code values, each degrees square is divided twice into four quadrants of size $30' \times 30'$ and $15' \times 15'$, respectively. These quadrants are represented by A-D as shown in Figure 4.1(b-c) and forms the fifth and sixth alphanumeric value of the topological code. Images found in the shaded cell in Figure 4.1 (c) will have a topological code

starting with 2931CA. Finally, each quadrant is further divided into 25 cells each of size $3' \times 3'$ represented by 1-25, which forms the last or last two values of the topological code. For an example, an image in the shaded cell in Figure 4.1 (d) will have a topological code of 2931CA1.

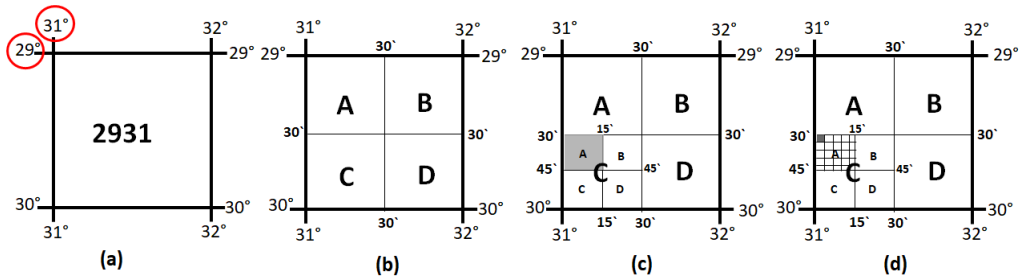
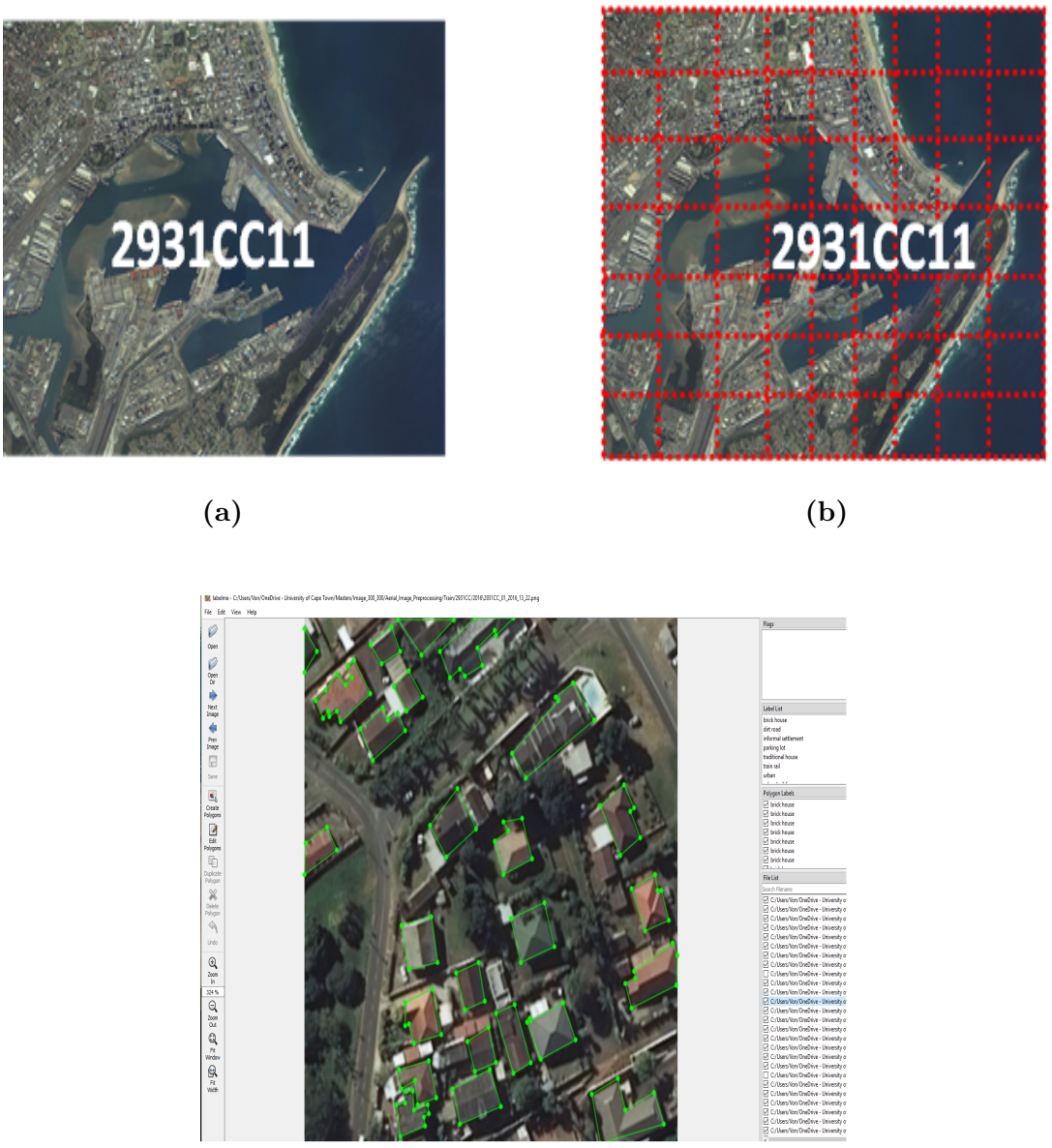


Figure 4.1: Illustration of how a topological code is assigned to a orthophoto using a grid of latitude and longitude (a) Combine latitude and longitude coordinates of the top-left corner of the degrees square, to obtain 2931 (b) Divide into four quadrants each of size $30' \times 30'$, to obtain 2931(A-D) (c) Further divide into four quadrants each of size $15' \times 15'$, to obtain 2931C(A-D) (d) Further divide into 25 components each of size $3' \times 3'$, to obtain 2931CA(1-25)

The images provided have a resolution of 34000x34000 and for classifying and detecting dwelling types and geo-types, these images were cropped into 300x300 resolution images. This was implemented by overlaying a grid with 300x300 sized cells as shown in Figure 4.2 (b). For the purpose of dwelling type object detection, the cropped aerial images were manually labelled using labelme.py, which is a python package that is used to draw polygon shapes around objects (Tzutalin, 2015). Figure 4.2 (c) illustrates how the images were labelled. As it can be seen, a polygon is drawn around each object and also each object is given label. The final output is a json file with the coordinates of each polygon drawn and corresponding label. Simultaneously, the geo-type of each image was also recorded for the purpose of performing geo-type image classification. For accurate and consistent labelling, NIDS data was used to assist with the labelling. This was done by mapping each topological code with the households' geo-type given in the survey data using the latitude and longitude coordinates.



(a) **(b)**

(c)

Figure 4.2: Illustration of how the images were cropped into smaller images and labelled using labelme.py (a)Original image of 2931CC11 topological code (b) Grid overlaid image of 2931CC11 topological code (c) 300×300 labelled image from 2931CC11 cropped aerial image.

Chapter 5

Method

This chapter gives a detailed description of the method design and the two phases involved, which are namely the aerial classification and detection phase, and the poverty modelling phase. This is then followed by a detailed discussion of Convolutional Neural Network (CNN), which will be used in the aerial classification and detection phase to extract information from aerial images. Lastly, multiple linear, ridge, lasso, and compositional statistical regression methods are discussed in detail as they are used in the poverty modelling phase.

5.1 Method Design

Estimating poverty will be implemented in two phases, as shown in the method design in Figure 5.1. These phases are namely; (i) the aerial classification and detection phase, and (ii) the poverty modelling phase. This research will focus on the KwaZulu Natal (KZN) province in South Africa because there is more household variety in terms of dwelling types and geo-types compared to other provinces found in the National Income Dynamics Study (NIDS) data.

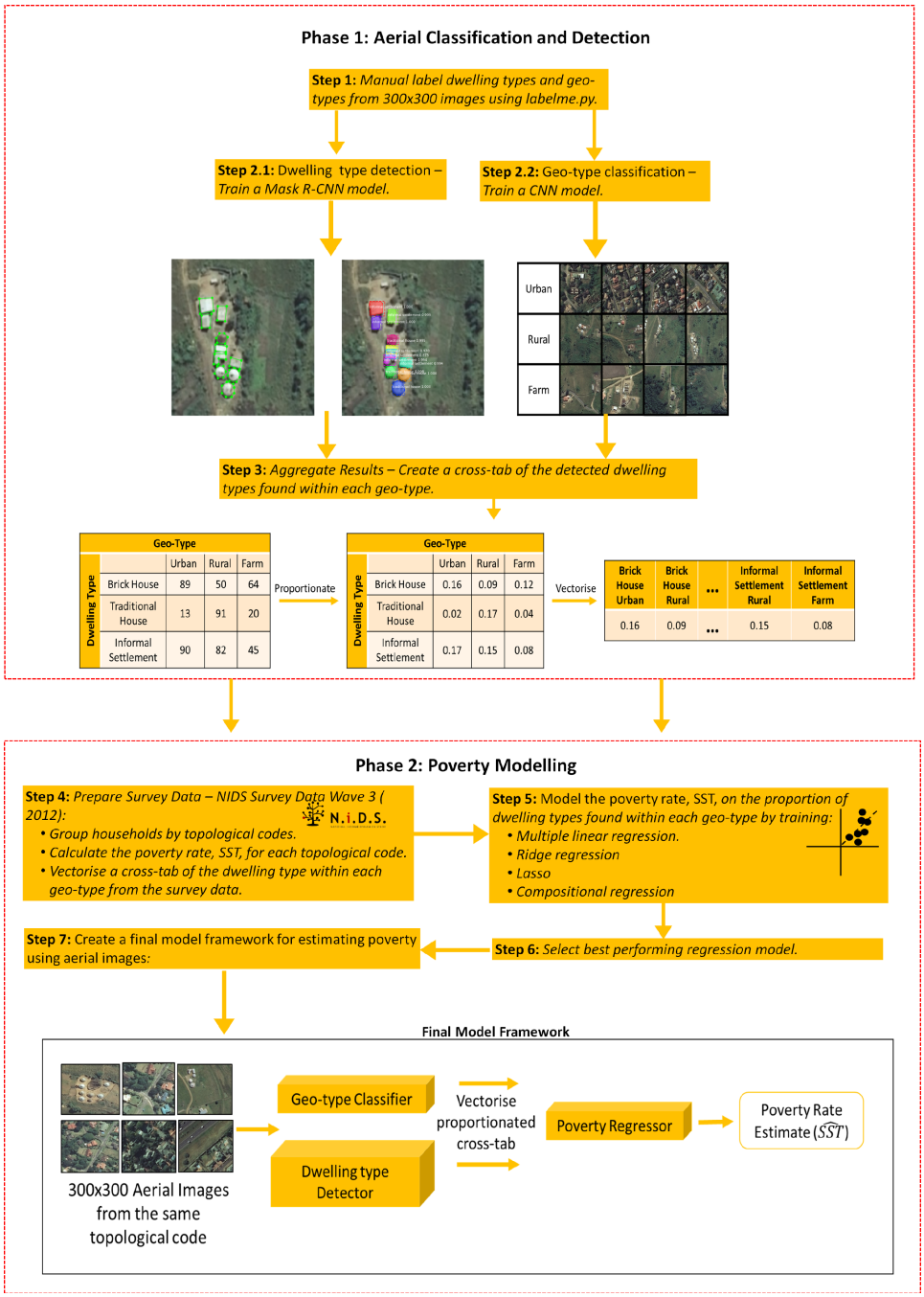


Figure 5.1: Method design for poverty estimation from aerial images. There are two phases involved namely; the aerial classification and detection phase (Phase 1), and the poverty modelling phase (Phase 2). Phase 1 involves the use of Convolutional Neural Network (CNN) models to detect dwelling types and classify geo-types from aerial images. The identified dwelling types and geo-types are aggregated, processed and then passed on to Phase 2. Phase 2 involves training regression models to estimate the poverty rate, Sen-Shorrocks-Thon (SST) index, where the best performing model is selected and used in the final aerial poverty estimation model.

5.1.1 Phase 1: Aerial Classification and Detection

The first step in aerial classification and detection phase involves manually labelling the dwelling types (brick house, traditional house, and informal settlement) and geo-types (urban, rural, farm) using labelme.py. These manually labelled images are then used in step 2.1 and step 2.2 to train a dwelling type detection model and a geo-type classification model, respectively, as seen in Figure 5.1. To perform dwelling type detection, an instance segmentation model, Mask R-CNN, is trained using the aerial images as input and the associated object coordinates and class labels that was manually labelled as output.

Geo-type classification of the aerial images is done by considering three different CNN models, namely; 3-layered CNN model, VGG19 and InceptionV4. These models were chosen because they have obtained very high accuracy rates on the Imagenet challenge and have different model architecture complexities (Alom et al., 2018; Szegedy et al., 2016). Amongst the three models, InceptionV4 model has the highest level of architecture complexity and three layered CNN model has the lowest and will be used as the base-line model because of it's basic and simple architecture.

These models will be later discussed in more detail. The 3-layered CNN model will be used as the base-line model and the best performing model will be selected to perform the geo-type classification in the final aerial poverty estimation model.

The final step in this phase involves using the trained detection and classification models to create a cross-tab of the different dwelling types found with each geo-type for each topological code. Each topological code cross-tab is proportioned and vectorised as shown in step 3 Figure 5.1. These vectorised topological code cross-tab will be used later in the final aerial poverty estimation model to estimate poverty in Phase 2.

5.1.2 Phase 2: Poverty Modelling

The poverty modelling phase involves training four different regression models using NIDS survey data wave 3 (2012) and selecting the best performing model to use in the final model framework, which will be later discussed in more detail. To train the regression models, the data is firstly prepared by grouping all the households by topological codes. This is then followed by using the derived variable, *monthly food expenditure per household member*, to compute the SST index for each topological code and to vectorise a proportioned cross-tab of the dwelling types found with in each geo-type for each topological code using the survey data.

The prepared survey data is then passed into step 5, as shown in Figure 5.1, to train the regression models. These models take the computed SST index as the dependent variable and the vectorised cross-tab elements as the independent variables. The best performing model is then selected and used in the final model framework to estimate poverty from aerial. The final model framework takes 300×300 aerial images from same topological code and pass it through trained dwelling type detector and geo-type classifier, where the output is aggregated into a vectorised proportional cross-tab, as shown in step 3, Figure 5.1. The aggregated results are then passed through a trained regression model that outputs the poverty rate estimate, $\hat{SST}$, for the topological code associated with the aerial images used.

Next the techniques implemented to execute the two phases, CNN and statistical regression models, are introduced and discussed in detail respectively.

5.2 Convolutional Neural Network

Convolutional Neural Network (CNN) is a specialised kind of neural network that uses convolution, a mathematical operation, in at least one layer in place of the matrix multiplication (Goodfellow et al., 2016). CNN models are used for processing

data that has a known grid-like topology structure, such as image (including spatial dependent images), time-series, audio and text data (Goodfellow et al., 2016).

Generally, convolution is a mathematical operation, denoted with an asterisk ($*$), implemented on two functions of real numbered ($\mathbb{R}$) arguments (Goodfellow et al., 2016). It represents an input signal passing through a linear time-invariant filter. The convolution output of an input signal x with a filter w, both defined over a continuous domain space in one-dimension, $s(t)$, is defined as

$$s(t) = (x * w)(t) = \int x(a)w(t-a)da \tag{5.1}$$

where t and a are index integer values of the input signal (x) and filter (w), respectively. When x and w are only defined on a discrete domain space, then the convolution output is defined as

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{5.2}$$

In CNN, convolution can be extended to multidimensional applications, where both the input signal and filter are multidimensional arrays known as tensors. For example, given a two-dimensional image ($\boldsymbol{I}$) as the input signal then a two-dimensional filter ($\boldsymbol{F}$), will be used and the convolution output will be defined as:

$$\boldsymbol{S}(i,j) = (\boldsymbol{I} * \boldsymbol{F})(i,j) = \sum_{m}\sum_{n} \boldsymbol{I}(m,n) * \boldsymbol{F}(i-m, j-n) \tag{5.3}$$

where i and j are the convolution output ($\boldsymbol{S}$) indexes and m and n are the input signal ($\boldsymbol{I}$) indexes.

The commutative property of convolution allows Eq.(5.3) to be also defined as:

$$\boldsymbol{S}(i,j) = (\boldsymbol{I} * \boldsymbol{F})(i,j) = \sum_m \sum_n \boldsymbol{I}(i-m, j-n) * \boldsymbol{F}(m,n) \tag{5.4}$$

which is preferred to Eq.(5.3) because it is more efficient and ideal to implement in machine learning since the range of m and n is smaller compared to the range found in Eq.(5.3) where the filter $\boldsymbol{F}$ has been flipped relative to the input signal $\boldsymbol{I}$.

Discrete convolution is equivalent to performing a dot product of the filter $\boldsymbol{F}$ and a sub image, of the same dimension, centred at ij to produce an ij value of $\boldsymbol{S}$. In machine learning, a similar operation called cross-correlation is implemented, where the filter $\boldsymbol{F}$ is not flipped and this is defined as:

$$\boldsymbol{S}(i,j) = (\boldsymbol{I} * \boldsymbol{F})(i,j) = \sum_m \sum_n \boldsymbol{I}(i+m, j+n) * \boldsymbol{F}(m,n) \tag{5.5}$$

Figure 5.2 gives an example of discrete convolution operation with and without a flipped filter. Figure 5.2 (b) uses Eq.(5.5) and Figure 5.2(c) uses Eq.(5.4).

In the context of CNN, the first function in Eq.5.5 is referred to as the input, denoted as $\boldsymbol{I}$, and the second function is the filter or kernel, denoted as $\boldsymbol{F}$. The output of convolving these two functions is K feature maps, as the input data is mapped to the feature space.

Furthermore, convolution in CNN provides means of working with input of different sizes and also improves the machine learning via sparse interactions, parameter sharing and equivariant representations (Feng et al., 2016).

Sparse interaction, also known as sparse connectivity or sparse weights is observed in CNN when a filter smaller than the input is used. This facilitates the detection of small, meaningful features using small filter hence fewer parameters are stored. Con-

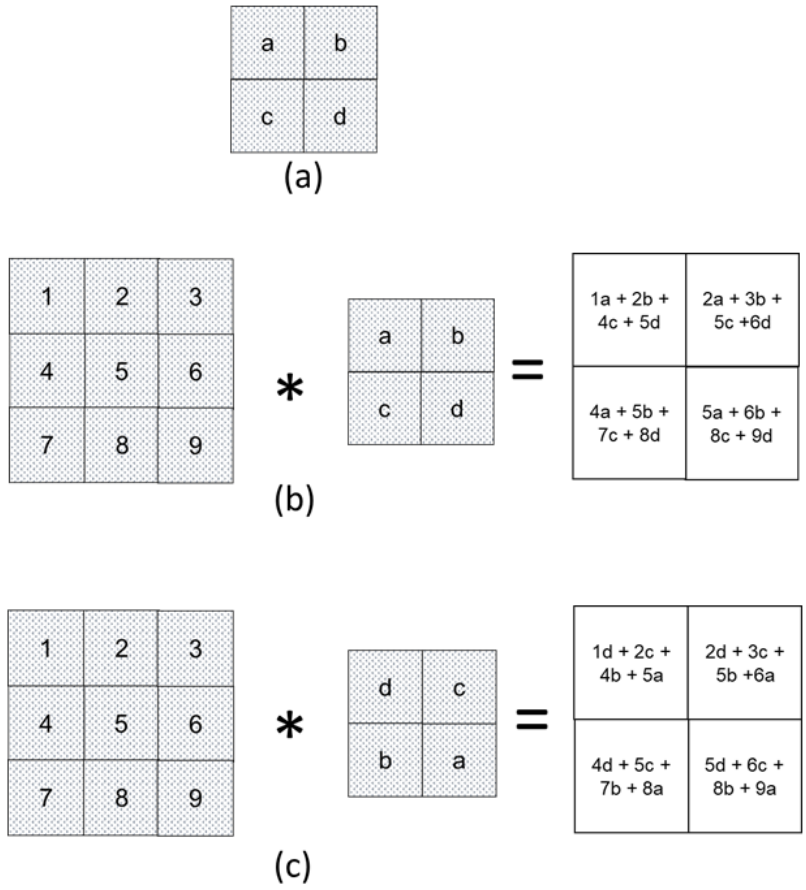


Figure 5.2: Demonstration of the discrete convolution operation in 2-D where (a) 2×2 filter (b) convolution without flipped filter (cross-correlation) example (c) convolution with flipped filter example

sequently, this reduces the storage requirements and improve the statistical efficiency of the model (Goodfellow et al., 2016).

Parameter sharing results in the same impact as in sparse interaction. By definition, parameter sharing is when the same parameter is used by more than one function in a model (Feng et al., 2016). Even though this does not decrease the runtime of the model, it does reduce the storage requirements by storing fewer parameters (Goodfellow et al., 2016). In CNN, the whole filter is used at each position of the input, hence the model learns only one set of parameters because the network has tied weights. This means weights applied to one input are tied back to the values of weights applied elsewhere in the network (Goodfellow et al., 2016).

The nature of parameter sharing in CNN causes the convolutional layer to have an

equivariance to translation property. A function is equivariant if the output changes the same way the input does (Reisert and Burkhardt, 2007). When processing images, convolution creates a 2-D feature map displaying where certain features appear in the input. Consequently, moving an object in the input results in the output representation moving in the same amount. There are some transformations that convolution is not naturally equivariant to, such as image scaling and rotation. In this case other techniques can be used to accommodate these transformations (Goodfellow et al., 2016).

Next the structure of a standard CNN model is discussed in more detail and the focus will be on types of layers namely convolutional, pooling and fully connected layers

5.2.1 Convolutional Neural Network Structure

A standard CNN structure consists of three types of layers, this is shown in Figure 5.3. These are convolutional, pooling and fully connected layers, which will be discussed in detail. From Figure 5.3, the dimensions are superscripted with l, which denotes the layer number and since the convolutional layer is the first layer, l is set to 1 ($l = 1$). The Convolutional layer and Pooling layer are sometimes considered as one single layer in CNN, because in some conventions a layer is considered one if it has parameters to learn, in which the pooling layer does not have (Goodfellow et al., 2016). CNN involves two broad processes, which are feature extraction followed by classification as seen in Figure 5.3. Together the convolutional and pooling layers perform feature extraction while the fully connected layer performs classification (Lecun et al., 2010).

Convolutional Layer

The convolutional layer involves simultaneously performing several convolutions to produce a set of linear activations from an $n_H^{[l-1]} \times n_W^{[l-1]} \times n_C^{[l-1]}$ image, where $n_H^{[l-1]}$ and $n_W^{[l-1]}$ is the height and width dimensions of the image, respectively, and $n_C^{[l-1]}$ is the number of channels the image contain (Goodfellow et al., 2016). The Convolutional

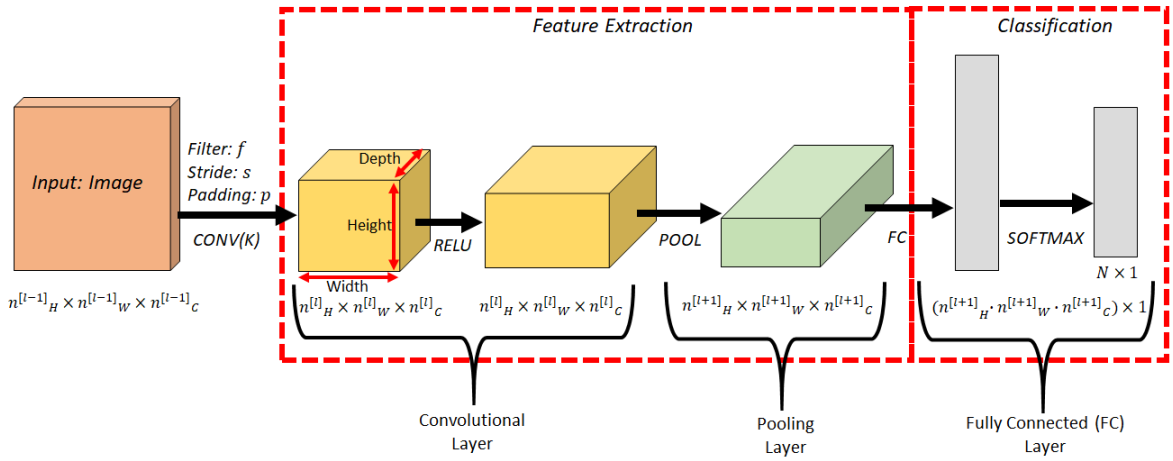


Figure 5.3: Convolutional Neural Network Structure

layer can either take images or feature maps as input depending on the network layer. In the first network layer, images are input and there after feature maps become input to the Convolutional layers.

The convolution of the image is implemented by using K filters of size $f_H \times f_W \times n_C^{[l]}$, where f_H and f_W are the height and width dimensions of the filters (for $f_H = f_W = f$), which is smaller than that of the image and $n_C^{[l]}$ is equal to the number of channels found in the image input ($n_C^{[l-1]}$). The dimension of the output is $(n_H^{[l]} - f + 1) \times (n_W^{[l]} - f + 1) \times K$, which is smaller than the dimension of the input. The size of the filter, f, is one of the three hyperparameters found in Convolutional layer (Liew et al., 2016). Moreover, instead of using researcher formulated filters, each element in a filter can be treated as a model parameter that can be learnt.

Convolving on an image using a filter for edge detection shrinks the output and loses some edge information. This problem can be avoided by padding the image input. Generally, padding refers to appending p rows and p columns around the borders of a matrix, where p is the amount of padding (Kong and Lucey, 2017). Padding is applied to all the channels of the input image as shown in Figure 5.4, all the 3 channels of the image are padded by zero.

Padding images helps preserve the original input size. The dimension of an output

from convolving a padded image is $(n_H^{[l]} + 2p - f + 1) \times (n_W^{[l]} + 2p - f + 1) \times K$. The amount of padding (p) needed is determined by setting the height and width dimensions of the original input equal to the corresponding output dimensions and then solving for p, as shown by Eq.(5.6). There are two types of padding convolutions, which are Valid and Same Convolutions. Valid Convolution refers to convolving an image with no padding ($p = 0$) while Same Convolution refers to convolving a padded image ($p \geq 1$). Most commonly used type of padding is padding an image with the number zero as shown in Figure 5.4, this is called zero padding (Goodfellow et al., 2016).

$$\begin{aligned} n + 2p - f + 1 &= n \\ 2p - f + 1 &= 0 \\ p &= \frac{f-1}{2} \end{aligned} \tag{5.6}$$

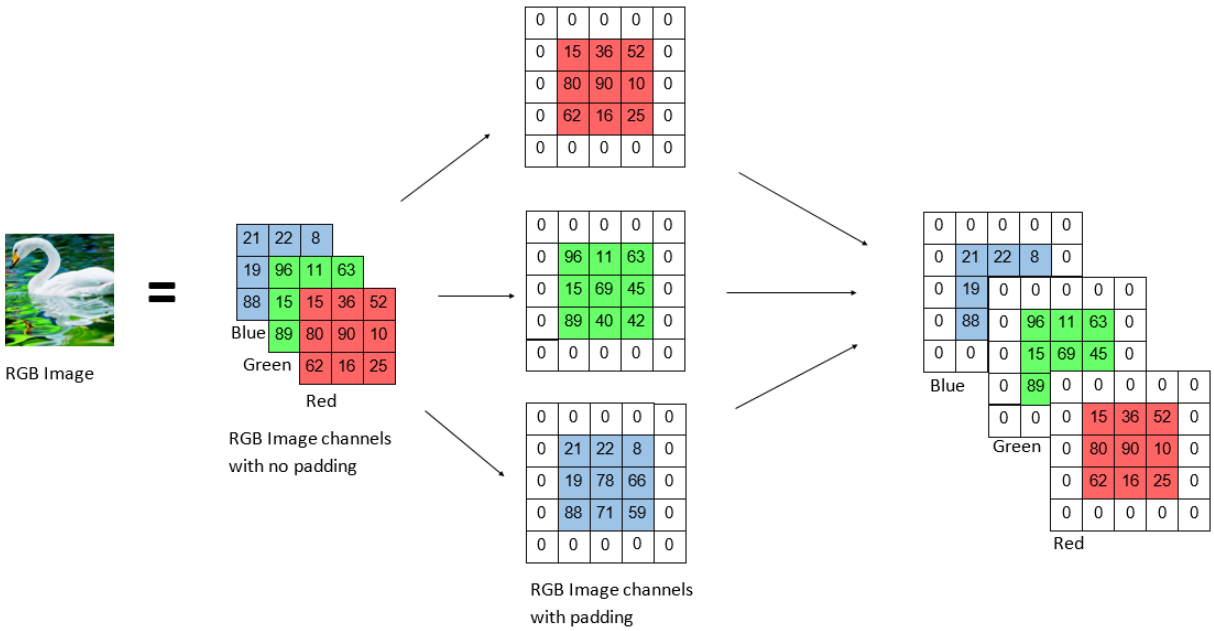


Figure 5.4: Illustration of zero padding ($p > 0$) an RGB image, which has three channels that also get padded

Stride is another hyperparameter in the convolutional layer and is used to reduce the number of model parameters to learn by reducing the overlap of receptive fields and spatial dimensions (Kong and Lucey, 2017). Stride, denoted as s, refers to the

amount of horizontal or vertical pixel steps a filter makes when convolving an image input (Goodfellow et al., 2016).

The convolutional layer is also called the detector stage as this is where edges (vertical/horizontal) of the image are detected. This is achieved by running each linear activation, obtained by convolving an image, through a non-linear activation function ($\theta(\cdot)$) (He et al., 2015). The commonly used non-linear activation functions are sigmoid, tanh, and rectified linear unit (ReLu), given by Eq.(5.7), Eq.(5.8), and Eq.(5.9), respectively (Krizhevsky et al., 2012). ReLu is more popularly used because of the advantages it offers such as low computational complexity, which has been shown by Krizhevsky et al. (2012).

$$Sigmoid: \quad \theta(x) = \frac{1}{(1+e^{-x})} = \frac{e^{x}}{e^{x}+1} \tag{5.7}$$

$$Tanh: \quad \theta(x) = tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{5.8}$$

$$ReLu: \quad \theta(x) = max(0,x) \tag{5.9}$$

The output of the convolutional layer with stride s and padding p is K feature maps of size $(\frac{n_H^{[l-1]}+2p-f}{s}+1) \times (\frac{n_W^{[l-1]}+2p-f}{s}+1)$, given by Eq.(5.10), where $\boldsymbol{I}$ is the input matrix that is convoluted $(*)$ with the kth filter matrix $\boldsymbol{F}_k^{[l]}$. This is then followed by adding a bias value, denoted as $b_k^{[l]}$, to the resultant matrix, corresponding to the kth filter, and the application of a non-linear function $\theta(\cdot)$. The ReLu function, defined by Eq.(5.9), is commonly used in image classification because of its ability to maintain the scale of output values.

$$\boldsymbol{C}_k^{[l]} = \theta(\boldsymbol{I} * \boldsymbol{F}_k^{[l]} + b_k^{[l]}), \quad for \quad k = 1, \ldots, K \tag{5.10}$$

Pooling Layer

The pooling layer involves applying a pooling function to the convolutional layer output, using a window with pre-set hyperparameters (stride, padding, and window sizes). This process reduces the spatial dimensions of the output while preserving important input image information. This also reduces the computational complexity of the model. Max Pooling and Average Pooling are the commonly used pooling functions. Max Pooling involves returning the maximum output within the neighbour of the window used, as show in Figure 5.5, while Average Pooling returns the average, as shown in Figure 5.6 (Gu et al., 2017). The choice of pooling function used is dependent on the type of dataset and features used, this has been demonstrated by Boureau et al. (2010) and Lee et al. (2018).

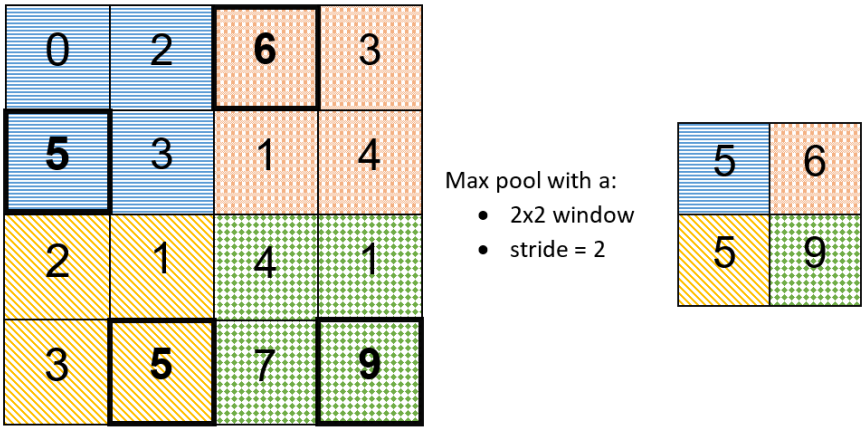


Figure 5.5: Illustration of Max Pooling a 4×4 feature map using a 2×2 window with a stride of 2, which results in a 2×2 feature map. For each 2×2 window, the max value in the window is extracted to form a cell in the new 2×2 feature map.

Pooling is used to further modify the output of this standard CNN layer by making it approximately invariant to small input translations to produce distortion-invariant features (Goodfellow et al., 2016). Invariance to translation refers to performing small translations on the input resulting in no changes to most of the pooled outputs. This is a useful property in cases where there is more importance in determining the presence

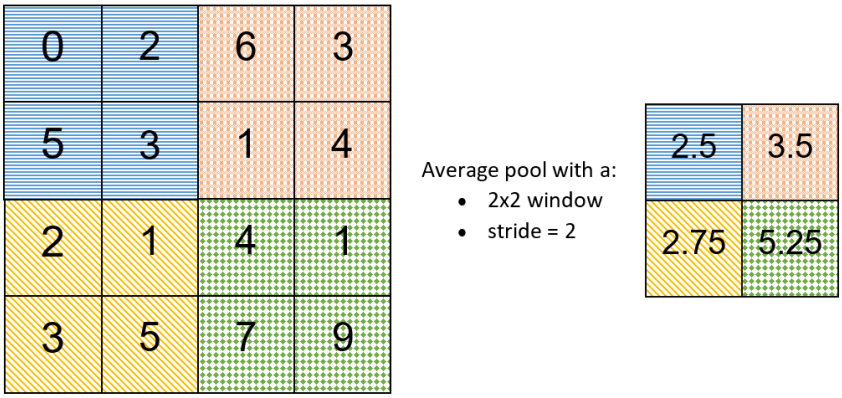


Figure 5.6: Illustration of Average Pooling a 4×4 feature map using a 2×2 window with a stride of 2, which results in a 2×2 feature map. For each 2×2 window, the average window value is extracted to form a cell in the new 2×2 feature map.

of a feature than its location (Goodfellow et al., 2016).

Fully Connected Layer

The fully connected layer is the final layer in a standard CNN. This is where classification of the original image input takes place, as shown in Figure 5.3. In this layer, the output of the pooling function is vectorised and then concatenated to form a vector of the feature map, denoted as $\boldsymbol{z}$, of size $n_H^{[l+1]} \times n_w^{[l+1]} \times K$ as input, where $n_H^{[l+1]}$ is the height, $n_W^{[l+1]}$ is the width and K is the number of channels of the pooling function output. The elements in the vector are then treated as neurons that are in one layer connected to every neuron in another layer. The same matrix multiplication principle applied in the traditional neural network is applied in this layer. Using the SoftMax function, defined in Eq.(5.11), the fully connected layer output is given by Eq.(5.12), where $\boldsymbol{W}$ is a matrix of weights and $\boldsymbol{b}$ is a vector of biases. The output, $\hat{\boldsymbol{y}} \in [0, 1]$, is an N dimensional vector, where N is the number of classes and each element in the

vector is the probability of classifying the original input as the corresponding class.

$$\theta(x_i) = \frac{e^{x_i}}{\sum_{n=1}^{N} e^{x_n}}, \quad for \quad i = 1, \ldots, N \tag{5.11}$$

$$\hat{\boldsymbol{y}} = \theta(\boldsymbol{W} \times \boldsymbol{z} + \boldsymbol{b}) \tag{5.12}$$

5.2.2 Convolutional Neural Network Learning Algorithm

CNN models contain model parameters and hyperparameters. Model parameters are learnt during the training phase of the model, which involves determining the best model parameters that fits the data well by using an optimization procedure, this is called backpropagation.

Backpropagation involves three processes namely; forward propagation, backward propagation and model parameter(s) update. Before implementing the three processes, all the biases are initialised to a value close to zero while other parameters are randomly sampled from a uniform or Gaussian distribution (Efron and Hastie, 2016). Backpropagation pseudo algorithm is described in Algorithm 1 (Raul, 1996).

Algorithm 1: Backpropagation Algorithm

Input: Training data
Output: Optimised model parameter(s)
Randomly initialise model parameter(s)
while *(iteration* $\geq$ *threshold* ***or*** $\Delta error \leq 1 \times 10^{-10}$*)* **do**
    Forward propagation
    Backward propagation
    Update model parameter(s)
end
return Updated model parameter(s)

Forward propagation

Forward propagation is the first process in the Backpropagation Algorithm given in Algorithm 1. It involves using the initialised model parameters to train a CNN model by propagating the image input through the model to generate predicted output Eq.(5.12). This is then followed by the computation of the loss function, given by

$$\mathcal{L} = -\sum_{i=1}^{N} y_i log(\hat{y}_i) \tag{5.13}$$

where y_i is the ith actual class output and $\hat{y}_i$ is the ith predicted class output defined by Eq.(5.12).

Backward propagation

The next process in Algorithm 1 is the backward propagation, which involves propagating the loss function backward through the model. This is done by using the chain rule to compute the derivatives of the loss function ($\mathcal{L}$) given by Eq.(5.13), with respect to the model parameters ($\boldsymbol{F}$, $\boldsymbol{b}$,$\boldsymbol{W}$, and b_k).

Model Parameter Update

The final process in Algorithm 1 involves updating the model parameters with the help of an optimiser based on the loss function Eq.(5.13). Optimisers assist in finding optimised model parameter values that minimise the loss function $\mathcal{L}(\boldsymbol{\theta})$, where $\boldsymbol{\theta} = [\boldsymbol{F}, \boldsymbol{b}, \boldsymbol{W}, b_k]$.

Gradient descent is a very popular optimization algorithm and most commonly used in neural networks (Ruder, 2016). It minimises the loss function $\mathcal{L}(\boldsymbol{\theta})$ by updating model parameter(s) in the opposite direction of the loss function gradient $\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ with respect to the model parameter(s). This is because the gradient indicates the direction of increase and so the opposite direction minimises the loss function. The size of steps taken to obtain a (local) minimum of the loss function is determined

by the hyperparameter learning rate $\eta > 0$ (Ruder, 2016). This hyperparameter affects the convergence to the minimum because when the learning rate is too small convergence becomes very slow and when its too large it causes the loss function to fluctuate around the minimum or probably diverge (Ruder, 2016).

There are three variants of gradient descent algorithm, which are namely batch gradient descent, stochastic gradient descent, and mini-batch gradient descent (Ruder, 2016). These variants differ in terms of how the data is introduced to the algorithm to compute the gradient of the loss function and hence further differ in terms of time complexity and accuracy.

Batch gradient descent involves updating the model parameter(s) by computing the gradient of the loss function with respect to each model parameter using the entire data (Ruder, 2016). In cases where a large dataset is used, this variant optimises very slowly because the whole dataset is used to make one parameter update. The slow pace of batch gradient descent guarantees that for convex surfaces there will be convergence to a global minimum and for non-convex surfaces there will convergence to a local minimum (Ruder, 2016). The update of the model parameters in batch gradient descent is defined as (Ruder, 2016):

$$\boldsymbol{\theta}^* = \boldsymbol{\theta} - \eta \cdot \frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \tag{5.14}$$

The shortcoming of batch gradient descent is that it performs redundant computation for large datasets because before each parameter update the same data is used to recompute the gradients. Stochastic gradient descent overcomes this shortcoming by updating parameters for each observation (x_i, y_i) chosen uniformly at random from the training set (Efron and Hastie, 2016). This is performed as (Ruder, 2016):

$$\boldsymbol{\theta}^* = \boldsymbol{\theta} - \eta \cdot \frac{\partial \mathcal{L}(\boldsymbol{\theta}; x_i, y_i)}{\partial \boldsymbol{\theta}} \tag{5.15}$$

The frequent parameter updates with high variances in stochastic gradient descent causes the loss function to fluctuate a lot (Ruder, 2016). However, this enables the optimiser to escape local minima traps by quickly moving to improved minima (Efron and Hastie, 2016). It has been observed that stochastic gradient descent displays similar convergence behaviour as batch gradient descent when the learning rate is small enough (Ruder, 2016).

Mini-batch gradient descent overcomes shortcomings experienced by both batch and stochastic gradient descent by performing model parameter update on each mini-batch of size n. This enables mini-batch gradient descent to result in a more stabilised convergence because the variance experienced by the model parameter(s) when updated is reduced and hence the fluctuations of the loss function experienced in stochastic gradient descent is also reduced (Ruder, 2016).

The batch size n varies for different applications. From the given definition of mini-batch gradient descent, batch and stochastic gradient descent can be further defined as special types of mini-batch gradient descent with a batch size of N (entire dataset size) and one, respectively. Parameter update in mini-batch gradient descent is given as (Ruder, 2016):

$$\boldsymbol{\theta}^* = \boldsymbol{\theta} - \eta \cdot \frac{\partial \mathcal{L}(\boldsymbol{\theta}; x_{(i:i+n)}, y_{(i:i+n)})}{\partial \boldsymbol{\theta}} \tag{5.16}$$

This research will use stochastic gradient descent to perform model parameter updates because the randomisation help to escape local minima traps and it is computationally cheaper because it takes a small amount of time to run the algorithm (Abu-Mostafa et al., 2012). This amount of time is called runtime complexity, which is essentially the amount of time it takes to perform the elementary operations of the algorithm. Using the above CNN structure, model parameters updates will be computed as (Zhifei

Zhang, 2016):

$$\boldsymbol{W}^* = \boldsymbol{W} - \eta \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}} \tag{5.17}$$

$$\boldsymbol{b}^* = \boldsymbol{b} - \eta \frac{\partial \mathcal{L}}{\partial \boldsymbol{b}} \tag{5.18}$$

$$\boldsymbol{F}^* = \boldsymbol{F} - \eta \frac{\partial \mathcal{L}}{\partial \boldsymbol{F}} \tag{5.19}$$

$$b_k^* = b_k - \eta \frac{\partial \mathcal{L}}{\partial b_k} \tag{5.20}$$

Finally, once the model parameters have been updated, the 3 processes are repeated until a condition is met. This condition can either be until the maximum number of iterations is reached or until convergence is reached. Convergence occurs when the $loss > previous\_loss - tolerance\_value$, where the tolerance value is a stopping criteria used to determine the allowance of how much the loss should decrease by for convergence to be achieved.

CNN can be used to perform different computer vision tasks. However, this research will only focus on image classification and instance segmentation, which are used for image recognition and object detection, respectively.

5.2.3 Image Classification

Image classification is a computer vision task used to perform image recognition by taking an image as input and outputting a single class label or a distribution of probabilities of the image belonging to a particular class. Using the standard structure of

CNN, one is equipped to build a simple or complex CNN model that performs image recognition. However, there are already existing CNN model architectures that have been shown to perform very well that one can use (Gu et al., 2017).

In this research, a 3-layered CNN model will be built from scratch and used as the base-line model to assess the performance of the existing complex and deeper models namely VGG19 and InceptionV4, to perform geo-type classification. These models will be discussed in more detail below.

Base-line Convolutional Neural Network Model: 3-layered Convolutional Neural Network

The 3 layered CNN model was built from scratch and will be used as the base-line model to assist in benchmarking the performance of the VGG19 and InceptionV4 model. It contains three convolutional layers each followed by the max-pooling layer to perform feature extraction, then followed by a single fully connected layer to perform classification, see Appendix C Figure C.1 for model architecture. The relu activation function is applied to each convolved feature map, which is an output of the convolution layers, to increase non-linearity in the network. This model was trained using stochastic gradient descent and it took approximately 10 hours to complete training on a Google Cloud Platform (GCP) virtual machine instance with 8 GPUs and memory of 104GB. The specification of the environment that the model was trained in will be later discussed is more detail.

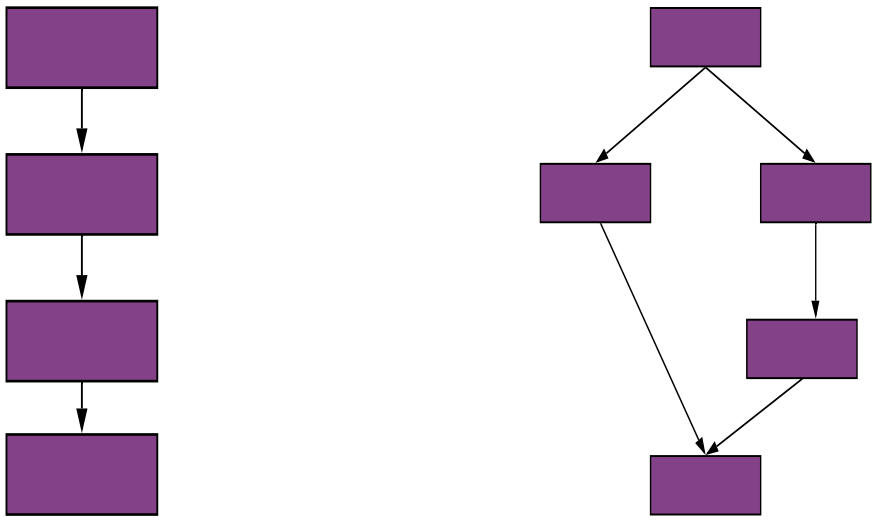
VGG19

The VGG19 is one of the variations of the VGGNet models introduced by Simonyan and Zisserman (2015) that has 19 weight layers and was a runner-up in the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) 2014 (Simonyan and Zisserman, 2015). It is made up of 16 convolutional layers each with a 3×3 sized filter with stride and padding of one, five max pooling layers of size 2×2 with a stride of one

and no padding and three fully connected layers, see Appendix C Figure C.2 for the model architecture (Simonyan and Zisserman, 2015). The introduction of VGGNet introduced the notion of increasing the depth of a CNN model to produce state-of-the art results for image recognition tasks and classification and localisation tasks (Simonyan and Zisserman, 2015). However, this results in increased computational cost and number of parameters.

InceptionV4

InceptionV4 is a variant of a pure inception model with no residual connection and was introduced by Szegedy et al. (2017). An inception model is a CNN model with a sparsely connected architecture instead of a densely connected architecture, as seen in Figure 5.7 . InceptionV4 originated from GoogLeNet, also known as InceptionV1, which is the winner of the ILSVRC 2014 Szegedy et al. (2017). It achieved state-of-the-art results by increasing the depth and width of the network while keeping the computational budget (processing time) constant Szegedy et al. (2014). Compared to its predecessors, InceptionV4 is a more simplified and unified network, equipped with more inception modules (Szegedy et al., 2017).



(a) Densely connected architecture **(b)** Sparsely connected architecture

Figure 5.7: Convolutional Neural Network (CNN) model architectures. Adapted from

The architecture of the InceptionV4 model consists of a stem module and three incep-

tion module classes namely; Inception-A, Inception-B and Inception-C, as shown in Appendix C Figure C.3. The Inception modules are stacked multiple times onto each other and in a single module there is convolution of the feature map from the previous layer in parallel. Stem module consists of the convolution, pooling, and normalization operations one would find in a CNN model, as shown in appendix C.4 (Szegedy et al., 2017).

The inception module classes in InceptionV4 differ in size and structure and have been devised to work with sparse CNN, which can be seen in Appendix C Figure C.4. After inception module class A and B, a reduction module is introduced to reduce the dimension of the inception module class output. Global average pooling is introduced after the last inception module to minimise overfitting by reducing the number parameter in the model (Szegedy et al., 2017).

5.2.4 Instance Segmentation

Instance segmentation involves the identification of each object instance of each pixel for every known object found within an image. The goal of instance segmentation is essentially to perform object detection which looks at producing masks on the different instance of the detected objects. In this research dwelling type detection on aerial images is implemented by using Mask R-CNN, which will be discussed in detail below.

Mask Region-based Convolutional Neural Network (Mask R-CNN)

Mask R-CNN is a type of instance segmentation CNN model that is used to perform instance object detection by identifying the outlines of objects at pixel level (He et al., 2017). It is essentially a combination of object detection and semantic segmentation, where an additional branch in Faster R-CNN is added to generate a segmentation output as seen in Figure 5.8. According to He et al. (2017), it currently outperforms all existing single-model entries on every task. Furthermore, it is easy to train and to

generalise to other tasks (He et al., 2017).

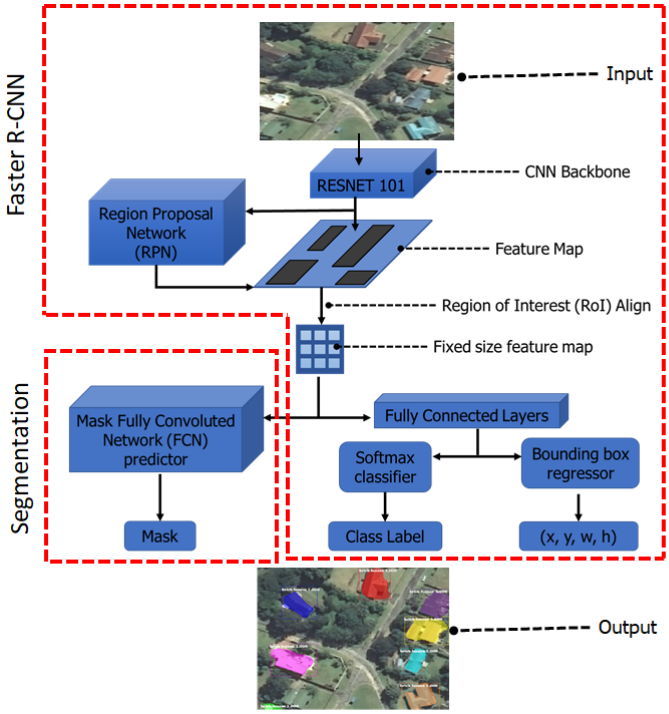


Figure 5.8: Model architecture of Mask R-CNN with Resnet101 as the CNN backbone. Mask R-CNN is an extension of faster R-CNN with a segmentation branch and RoIAlign for accurate mapping between original image and proposal.

Firstly, an image is passed through a CNN backbone model where low-level features (edges of the object) and high-level features (physical features of the object) are extracted. A CNN backbone model in Mask R-CNN is the CNN model that is used to perform feature extraction. This backbone network model outputs a feature map which becomes the input for proceeding stages. The feature map from the backbone model is then passed through a region proposal network (RPN) where the network uses a sliding window to scan the feature map with the purpose of finding regions that are likely to contain an object called proposals (regions of interest). These regions that are scanned to find proposals are called anchors, which are overlapping boxes of different aspect ratios and sizes distributed all over the feature map (He et al., 2017).

Once the proposals have been generated by the RPN, each proposal is passed into a Faster R-CNN to perform classification and localisation to output a class label and a four-valued bounding box list (x= x-coordinate of the bounding box center, y=

y-coordinate of the bounding box center, w=width of the bounding box, h = height of the bounding box), respectively. Simultaneously, a Fully Convolutional Network (FCN) is applied to each proposal to predict the mask of the detected object instance. This mask is the binary encoding of the spatial layout of the object, where pixels that belong to the object are encoded with the value one and zero otherwise (He et al., 2017).

Faster R-CNN has a misalignment problem between the original image and feature map caused by quantisation in RoIPool that Mask R-CNN overcame by introducing RoIAlign (He et al., 2017). The misalignment in Faster R-CNN is observed when the feature map is resized into a fixed size (He et al., 2017). RoIAlign allows for accurate mapping of the proposals from the original image onto a feature map (He et al., 2017).

The loss function of the Mask R-CNN is defined on each proposal during training as the combination of the log loss of classification ($\mathcal{L}_{cls}$), localisation ($\mathcal{L}_{bbox}$), and segmentation mask ($\mathcal{L}_{mask}$) and is given by (He et al., 2017):

$$\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{bbox} + \mathcal{L}_{mask} \tag{5.21}$$

$$\mathcal{L}_{cls} = -log(p, u), \quad p = (p_1, \ldots, p_K) \quad and \quad u = 1, \ldots, K \tag{5.22}$$

$$\mathcal{L}_{bbox} = \sum_{i \in \{x,y,w,h\}} Smooth_{L1}(t_i^u - v_i) \tag{5.23}$$

$$Smooth_{L1}(x) = \begin{cases} 0.5x^2, & \text{if } |x| < 1. \\ |x| - 0.5, & \text{otherwise.} \end{cases} \tag{5.24}$$

where $\mathcal{L}_{cls}$ is the negative log loss of the detected object belonging to the predicted

class, denoted by p, given that the ground truth class is $u = 1, \ldots, K$. $\mathcal{L}_{bbox}$ is a smoothed $L1$ for the bounding box and $\mathcal{L}_{mask}$ is the average cross entropy loss for the masks, where t_i^u is the predicted bounding box of object class u and v_i is the corresponding ground truth bounding box. Both t_i^u and v_i are given as a list of the height(h), width (w) and x, y coordinates of the center of the bounding box. A model that performs well has a loss that is close to zero, because as the predicted probability increases, the loss tends to zero due to the fact that the negative log of a big number produces a small number and vice versa. It should be noted that the loss is the negative log of the predicted probability of the actual class and the loss applies to the distributions of the actual class and the predicted probability of the actual class.

Once the CNN models have been successfully trained, the predicted output of the aerial images generated by the CNN models will be used as input in a statistical regression method to estimate poverty. Next is a detailed discussion of multiple linear regression, ridge regression and lasso, which are statistical regression methods that will be used to implement Phase 2, the poverty modelling phase, as shown in Figure 5.1.

5.3 Statistical Regression Methods

5.3.1 Regression

Regression is a statistical tool used for identification and characterization of functional relationships between the dependent variable (y) and the independent variables ($x_1, x_2, \ldots, x_p$) (Chatterjee et al., 2000). The true relationship between the independent variable(s) and the dependent variable is approximated by the model

$$y = f(x_1, x_2, \ldots, x_p) + \epsilon \tag{5.25}$$

where $f(x_1, x_2, \ldots, x_p)$ is the functional relationship between the dependent and independent variable(s) and ϵ is the model error that represents the model approximation discrepancy, which is the difference between the observed output and the expected output (Chatterjee et al., 2000).

Regression can be categorised into different variants based on three criteria. The first criteria is based on the number of dependent variables and the categories are namely univariate and multivariate regression (Chatterjee et al., 2000). Univariate regression involves one dependent variable while multivariate regression involves two or more dependent variables.

The second criteria based on the number of independent variables categorises regression into simple and multiple regression, where simple regression involves one independent variable while multiple regression involves two or more independent variables (Hastie et al., 2009).

Finally, the last criteria is based on functional form of the relationship between independent and dependent variables and the categories are linear and non-linear regression. Linear regression involves linear relationship between the dependent and independent variables while non-linear regression involves a non-linear relationship such as polynomial regression (James et al., 2013).

Univariate multiple linear regression will be implemented in this research to estimating poverty from aerial images. Multiple linear regression, ridge regression, lasso and compositional regression models will be considered and compared and the best performing model will be used to model poverty based on how well the models fit the data.

5.3.2 Multiple Linear Regression

Multiple linear regression is a statistical technique used to model the relationship between two or more continuous or categorical independent variables and a continuous dependent variable. Essentially, multiple linear regression fits a line through a multi-dimensional space to model the data (James et al., 2013). The population multiple linear regression model is defined as

$$y_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{i,j} + \epsilon_i \quad for \quad i = 1, \cdots, N \tag{5.26}$$

where y_i is the dependent variable for the ith observation, x_{ij} is the jth independent variable for the ith observation, where $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, p$, β_j is the jth unknown parameter, where $j = 1, 2, \ldots, p$ and ϵ_i is the deviation between the ith observed value and ith expected value, which is based on the whole population, for $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, p$. The matrix form of Eq.(5.26) is given by

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{5.27}$$

where $\boldsymbol{y}$, $\boldsymbol{\beta} = [\beta_0, \ldots, \beta_p]$, and $\boldsymbol{\epsilon}$ are vectors of $N \times 1$, $(p+1) \times 1$, and $N \times 1$, respectively, and $\boldsymbol{X}$ is an $N \times (p+1)$ matrix.

Multiple linear regression can be used to identify and understand the effects that the independent variables have on the dependent variable. This technique can also be used to forecast the changes of the dependent variable with respect to the change in the independent variables. Furthermore, it can also be used to predict trends and future values of the dependent variable by obtaining point estimates (James et al., 2013).

There are assumptions that comes with multiple linear regression models, which are

as follows (Wooldridge, 2014):

- **MLR1**: The model is linear in parameters, i.e. there is linear relationship between the dependent and independent variables.
- **MLR2**: The errors ($\boldsymbol{\epsilon}$) are normally distributed.
- **MLR3**: The $\boldsymbol{\epsilon}$ are homoskedastic, which means that the error has the same variance given any independent variable.
- **MLR4**: There is no perfect multicollinearity between independent variables.

The model error $\boldsymbol{\epsilon}$ cannot be observed and so hypothetical values ($\tilde{\boldsymbol{\beta}}$) of model parameters ($\boldsymbol{\beta}$) that can be used to compute fitted values ($\tilde{\boldsymbol{y}}$). Eq.(5.28) are introduced to compute observable error estimates called residuals ($\boldsymbol{e}$) given by Eq.(5.29). Each ith observation from the data will have a corresponding residual (e_i) based on $\tilde{\boldsymbol{\beta}}$.

$$\tilde{\boldsymbol{y}} = \boldsymbol{X}\tilde{\boldsymbol{\beta}} \tag{5.28}$$

$$\boldsymbol{e} = \boldsymbol{y} - \boldsymbol{X}\tilde{\boldsymbol{\beta}} \tag{5.29}$$

The model parameters, $\boldsymbol{\beta}$, are estimated by using Ordinary Least Squares (OLS) estimation, which involves minimising the residual sum of squares (RSS) given by Eq.(5.30). RSS is computed by taking the sum of squared differences between the true output values ($\boldsymbol{y}$) and the fitted values ($\tilde{\boldsymbol{y}}$), which can be used as a measure of the overall model fit.

$$RSS(\tilde{\boldsymbol{\beta}}) = \boldsymbol{e}^T\boldsymbol{e} = (\boldsymbol{y} - \boldsymbol{X}\tilde{\boldsymbol{\beta}})^T(\boldsymbol{y} - \boldsymbol{X}\tilde{\boldsymbol{\beta}}) \tag{5.30}$$

The OLS estimator ($\hat{\boldsymbol{\beta}}^{\boldsymbol{OLS}}$) of $\boldsymbol{\beta}$ is defined as (James et al., 2013):

$$\hat{\boldsymbol{\beta}}^{OLS} = \underset{\boldsymbol{\beta}}{argmin} \quad RSS(\boldsymbol{\beta}) \tag{5.31}$$

and is computed as

$$\hat{\boldsymbol{\beta}}^{OLS} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{5.32}$$

Multiple linear regression experiences low bias but suffers from high variance. In cases where the number of independent variables (p) is greater than the number of observations (N), the OLS estimator fails to obtain a solution. Furthermore, in the presence of multicollinearity among the independent variables, the OLS estimator is unstable (Al-Hassan, 2010). This means the high correlation between the independent variables will result in a model with very large standard errors of the estimates and hence large $\boldsymbol{\beta}$ coefficients, which might be of implausible magnitude and/or sign, will be required to obtain statistical significant $\boldsymbol{\beta}$ coefficients (El-Fallah and El-Sallam, 2010).

Other regression models can be used to overcome these disadvantages by introducing some sort of penalty to Eq.(5.30) such as ridge and lasso regression. These models are discussed in detail in the next two sections.

5.3.3 Ridge Regression

Ridge regression is an extension of the multiple linear regression that is used for analysing multiple regression data suffering from multicollinearity, which is when the independent variables are collinear. This was first introduced by Hoerl and Kennard (1970) to improve the instability of the OLS estimator.

Ridge regression is also known as a shrinkage method that shrinks the parameters towards zero by imposing an L_2 penalty, which is given by $\sum_{j=1}^{p} \beta_j^2$, on their size (James et al., 2013). It assigns low weights to variables that explain less variance (James et al., 2013). The parameters in ridge regression are estimated by minimising the Penalised RSS defined as

$$PRSS(\lambda) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta} \tag{5.33}$$

where $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_p]$ and $\lambda \geq 0$ is the tuning parameter, which controls the amount of shrinkage observed. Different λ results in different ridge regression coefficients, hence cross validation is used to find the optimal λ that minimises the prediction error (James et al., 2013). Cross validation is a robust approach used to assess the predictive values of a model on an unseen dataset to avoid overfitting (van Houwelingen and Sauerbrei, 2013).

Minimising Eq.(5.33) with respect to the model parameters, results in the following ridge regression parameter estimator (Hastie et al., 2009)

$$\hat{\boldsymbol{\beta}}_{\boldsymbol{\lambda}}^{ridge} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{5.34}$$

The intercept parameter term in ridge regression, denoted as β_0, is not penalised because the main aim in ridge regression is to shrink only the predictor variables $(\beta_1, \beta_1, \cdots, \beta_p)$. The β_0 ridge parameter estimator is given by

$$\hat{\beta}_0^{ridge} = \bar{y} = \sum_{i=1}^{N} \frac{y_i}{N} \tag{5.35}$$

When λ is equal to zero the shrinking penalty has no effect and the ridge regression parameter estimator is equal to the OLS estimator. In cases where λ tends to infinity

the penalty effect increases and the ridge estimate tends to zero.

Figure 5.9 provides a graphical demonstration of the ridge parameter estimation process which is defined by minimising RSS restricted by a constraint function such as in Eq.(5.33). The circle represents the constraint function for a model with two parameters (β_1, and β_2) defined by $\beta_1^2 + \beta_2^2 \leq t$, and the red ellipses represents the contour lines of the residual sum of square (RSS) where it is minimised at point $\hat{\boldsymbol{\beta}}^{OLS}$, OLS estimator. The ridge parameter estimator ($\boldsymbol{\beta}_\lambda^{ridge}$) is given by the point where the circle and the ellipse intersect (James et al., 2013).

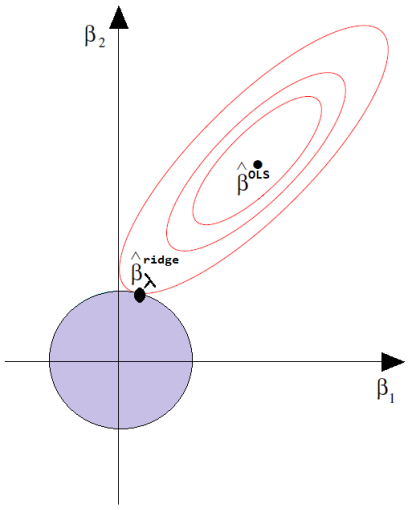


Figure 5.9: Pictorial demonstration of the estimation process of the ridge parameter estimates, where the ridge parameter estimator ($\boldsymbol{\beta}_\lambda^{ridge}$) is given by the point where the circle and ellipse intersect. The circle represents the constraint function for a ridge model with two parameters defined by $\beta_1^2 + \beta_2^2 \leq t$ and the ellipse represents the contour lines of the RSS. Adapted from (James et al., 2013).

Shrinkage methods result in models that have low prediction error but performs poorly in terms of offering clear and easily interpretable models. Furthermore, shrinking the parameters by a penalty introduces bias into the model but reduces the sample variance and hence resulting in a smaller MSE than the OLS estimator. Next, Lasso, another shrinkage model is discussed, which overcomes the disadvantage of the ridge model of not being easily interpretable.

5.3.4 Lasso

Lasso was proposed by Tibshirani (1996) as another extension of multiple linear regression that imposes an L_1 penalty on RSS to continuously shrink the regression coefficients towards or to exactly zero, excluding the intercept term (β_0), which is not penalised. The L_1 penalty is given by $\sum_{j=1}^{p} |\beta_j| = ||\boldsymbol{\beta}||_1$, whereas the parameters in lasso regression are estimated by minimising the Penalised RSS defined as:

$$PRSS(\lambda) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) + \lambda||\boldsymbol{\beta}||_1 \tag{5.36}$$

where $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_p]$ and $\lambda \geq 0$ is the tuning parameter that is pre-determined. Optimal λ can be determined by performing CV based on a range of λ values (James et al., 2013). Sufficiently large values of λ results in some parameter estimates being shrunk to zero (James et al., 2013). This is illustrated in Figure 5.10, where the diamond represents the constraint function in Eq.(5.36) for a lasso model with two parameters (β_1, and β_2) defined by $|\beta_1| + |\beta_2| \leq t$. The red ellipses represents the contour lines of the RSS where it is minimised at point $\hat{\beta}^{OLS}$ and the lasso estimate is given by the point where the ellipse and diamond intersect.

Lasso is less flexible compared to the multiple linear regression because it is more restrictive in estimating the model parameters ($\beta_1, \ldots, \beta_p$) and sets some of them to zero. However, it is more interpretable in comparison to both multiple linear regression and ridge regression because the final lasso model will consist of the dependent variable and associated with a small subset of the independent variables, which are the ones with non-zero parameter estimates (James et al., 2013). Consequently, Lasso's shrinking property overcomes the disadvantage of the ridge regression of not having easily interpretable models (Hastie et al., 2009).

Lasso might appear as an improved version of the ridge regression because it performs shrinkage and feature selection, but neither of them dominate each other universally.

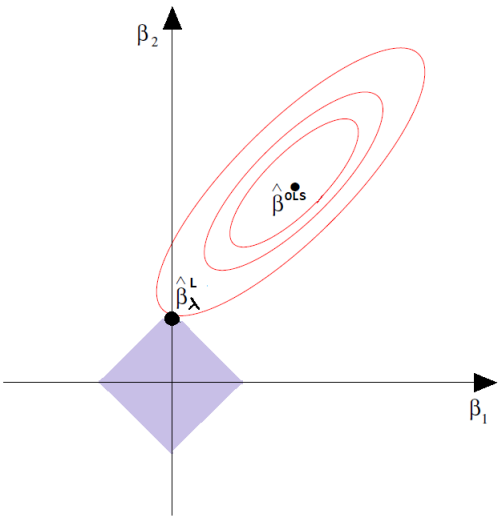


Figure 5.10: Pictorial demonstration of the estimation process of the lasso parameter estimates where the lasso parameter estimator (β_λ^L) is given by the point where the circle and ellipse intersect. The circle represents the constraint function for a lasso model with two parameters defined by $|\beta_1| + |\beta_2| \leq t$ and the ellipse represents the contour lines of the RSS. Adapted from (James et al., 2013).

They are both shrinkage methods that result in models that exhibit less sampling variability and lower prediction error as compared to the multiple linear regression and this is all dependent on the dataset used (James et al., 2013). These regression methods are unable to handle multicollinearity in the data that may exist. There are dimension reduction regression methods that can be used to overcome multicollinearity that behaves similarly to ridge regression, such as PCR (James et al., 2013), which will be discussed in more detail in the next section as part of the compositional regression method.

5.3.5 Compositional Regression

Compositional regression involves modelling compositional data, which is a type of data whose vector elements sum to one (Tsagris, 2015). Mathematically, this is defined as (Tsagris, 2015)

$$\mathbb{S}^d = \{(x_1, \ldots, x_p)^T | x_i \geq 0, \sum_{i=1}^{p} x_i = 1\} \tag{5.37}$$

where $\mathbb{S}^d$ is the sample space of the compositional data (also known as the simplex) (Tsagris, 2015), p is the number of independent variables and $d = p - 1$.

There are many different techniques of modelling compositional data for different cases where the compositional data is either on the dependent or independent variable side (Aitchison, 1983; Egozcue et al., 2003; Gueorguieva and Zelterman, 2008; Wang and Tenenhaus, 2010; Hron and Thompson, 2012; Wang and Guan, 2013; Tsagris, 2015, 2018). In this research, the compositional data is on the independent variables side and hence univariate regression with an $\alpha-$transformation (Tsagris et al., 2011), proposed by Tsagris (2015) as $\alpha-$PCR, will be the compositional regression technique discussed in this section.

$\alpha-$PCR involves applying $\alpha-$transformation to the compositional data and then performing PCR (Tsagris, 2015). This regression technique naturally handles zero values in the data and hence no imputations is necessary because of the $\alpha-$transformation (Tsagris, 2015). Furthermore, PCR overcomes multicollinearity problems that may exist, due to it dimension reduction nature. The $\alpha-$transformation and PCR techniques are discussed in more detail below.

$\alpha-$Transformation

$\alpha-$Transformation is a power transformation that is data driven and has one free power parameter, similar to the Box-Cox transformation, developed by Tsagris et al. (2011).

The $\alpha-$transformation on the independent variables, proposed by Tsagris et al. (2011), was derived from the compositional data power transformation defined by Aitchison (2003) as

$$\boldsymbol{u}_\alpha = \left(\frac{x_1^\alpha}{\sum_{j=1}^p x_j^\alpha}, \ldots, \frac{x_p^\alpha}{\sum_{j=1}^p x_j^\alpha}\right)^T \tag{5.38}$$

where $\boldsymbol{x} \in \mathbb{S}^d$ as defined in Eq.5.37.

Tsagris et al. (2011) defined $\alpha-$transformation on the independent variables as

$$\boldsymbol{z}_\alpha = \frac{1}{\alpha}\boldsymbol{H}(p\boldsymbol{u}_\alpha - \mathbf{1}) \tag{5.39}$$

where p is the number of independent variables, α is the free power parameter and $\boldsymbol{H}$ is a $d \times p$ Helmert sub-matrix (Lancaster, 1965), which is an orthogonal matrix without the first row. Eq.5.39 is essentially a linear transformation of Eq.5.38 (Tsagris, 2015).

Principle Component Regression (PCR)

PCR is a dimension reduction regression method that involves the use of principle component analysis, which is a dimension reduction technique (James et al., 2013). In PCR the principle components obtained are used to fit a regression model using least squares in replacement of the original independent variables from the full dataset (James et al., 2013).

The PCR algorithm incorporated with the $\alpha-$transformation can be summarised as follows (Tsagris, 2015):

1. $\alpha-$transformation: Select an α value to apply $\alpha-$transformation on the compositional data, which is on the independent variables (denoted as $\boldsymbol{X}$) side, to obtain $\boldsymbol{Z}_\alpha$, given by Eq.5.39.

2. Eigen analysis: Calculate the principle components $(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_M)$ of $\boldsymbol{Z}_\alpha$, which is given by

$$\boldsymbol{S} = \boldsymbol{Z}_\alpha \boldsymbol{V} \tag{5.40}$$

   where $\boldsymbol{V}$ is a matrix of eigenvectors obtained from the eigen decomposition of $\boldsymbol{Z}_\alpha^T\, \boldsymbol{Z}_\alpha$.

3. Regression analysis: Regress $\boldsymbol{y}$ on $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_M$ for $M \leq p$, denoted as $\hat{\boldsymbol{\theta}}_m$ for $m = 1, \ldots, M$, where M represents the selected number of principle components. Since the principle components $(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_M)$ are orthogonal, the regression model can be represented as the sum of the individual regressions

$$\boldsymbol{y}^{\hat{PCR}}{}_{(M)} = \hat{\theta}_0 \mathbf{1} + \sum_{m=1}^{M} \hat{\boldsymbol{\theta}}_m \boldsymbol{s}_m$$

$$\boldsymbol{y}^{\hat{PCR}}{}_{(M)} = \hat{\theta}_0 \mathbf{1} + \boldsymbol{S}\hat{\boldsymbol{\theta}} \tag{5.41}$$

where $\hat{\theta}_0$ is the intercept coefficient estimate and $\hat{\boldsymbol{\theta}}_m = < \boldsymbol{s}_m, \boldsymbol{y} > / < \boldsymbol{s}_m, \boldsymbol{s}_m >$ (James et al., 2013).

From Eq.5.40,

$$\boldsymbol{y}^{\hat{PCR}}{}_{(M)} = \hat{\theta}_0 \mathbf{1} + \boldsymbol{Z}_\alpha \boldsymbol{V}\hat{\boldsymbol{\theta}} \tag{5.42}$$

$$\boldsymbol{y}^{\hat{PCR}}{}_{(M)} = \hat{\boldsymbol{B}}_0{}^{PCR} + \boldsymbol{Z}_\alpha \hat{\boldsymbol{\beta}}^{PCR}_{(M)} \tag{5.43}$$

where $\hat{\boldsymbol{B}}_0{}^{PCR}$ is the estimated coefficient for the intercept term and $\hat{\boldsymbol{\beta}}^{PCR}_{(M)}$ is the estimated regression coefficients for the M principle components (excluding the intercept) and is respectively given by

$$\hat{\boldsymbol{B}}_0{}^{PCR} = \bar{y} \tag{5.44}$$

$$\hat{\boldsymbol{B}}^{PCR}_{(M)} = \boldsymbol{V}\hat{\boldsymbol{\theta}} = \boldsymbol{V}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{S}^T\boldsymbol{y} \tag{5.45}$$

Any value of $M \geq p$ can be used, however, if $M = p$ then $\hat{\boldsymbol{B}}^{PCR} = \hat{\boldsymbol{B}}^{OLS}$ because the columns of $\boldsymbol{S}$ will be spanning the column space of $\boldsymbol{Z}_\alpha$ (James et al., 2013).

Moreover, if $M > p$ then the $(p - M)$ principle components that explain the least variation in the data will be discarded (James et al., 2013).This corresponds to the $(p - M)$ components with the smallest eigenvalue (James et al., 2013).

The optimal α value used in the $\alpha-$transformation and optimal M value for the number of principle components are selected by using CV to find the optimal results that minimises the MSE of the regression model predictions (Tsagris, 2015).

$\alpha-$PCR overcomes the biggest drawback that multiple linear regression, ridge, and lasso cannot handle, which is multicollinearity. This property makes it seem more superior than the other regression methods, however, PCR has two major drawbacks that makes it less favourable. The first drawback is that the principle components calculated in PCR are all individually linear combinations of the original independent variables, hence regressing using principle components results in coefficient estimates corresponding to each principle component that cannot be easily interpreted (Hastie et al., 2009). Consequently, this research will focus on the predictive performance of the $\alpha-$PCR when comparing the overall performance of the different regression methods in Chapter 7.

The second PCR drawback is the fact that the dependent variable is not considered when selecting which and how many principle components to keep or discard. Consequently, some variables that are dropped might be statistically significant and some variables that are kept might be insignificant (Hastie et al., 2009). In this research, regression performance metrics will be used to determine the optimal regression method that fits the data well, which will be discussed in the next chapter.

Next, the implementation of the above techniques and methods are further discussed to understand the required environment set-up, data preparation and model evaluation metrics used in this research.

Chapter 6

Implementation

In this chapter, the practical implementation of computer vision models in the aerial classification and detection phase and statistical regression models in the poverty modelling phase is discussed in detail. This includes the environment set-up, data preparation and the corresponding model evaluation metrics.

6.1 Environment Set-up

The aerial classification and detection phase and the poverty modelling phase were conducted on a virtual machine instance created on the GCP with specifications given in Table 6.1. Python and R programming languages were used to implement phase one and phase two, respectively.

Table 6.1: Google Cloud Platform computing environment specifications

Specification	Description
Operating System	Ubuntu 16.04.5 LTS
GPU	8 NVIDIA Tesla K80 GPUs
CPU	16 vCPu
Memory	104 GB
Region	us-west1 (Oregon)
Zone	us-west1-b

6.2 Aerial classification and detection phase

This section provides a description of the additional data preparation done to cater for the different implemented CNN models and the evaluation metrics used to assess the performance of all the implemented models in the aerial classification and detection phase.

6.2.1 Data Preparation

Two datasets were created for this phase from the aerial image dataset provided by the NGI, for the purpose of geo-type classification and dwelling type detection. Each of the datasets contain 5174 images, which are linked to 130 topological codes that will be considered in this research. It should be noted that a single topological code is linked to multiple aerial images as explained in Chapter 4 Figure 4.2.

The images were appropriately manually labelled by the author and this was a time-consuming and tedious task that took over 2 months to complete. There are more efficient methods of labelling the data such as semi-supervised learning where one trains a model using a small sample of labelled data to predict labels for unlabelled data (Hamidreza et al., 2017). However, due to limited computing resources, manually labelling the data was sufficient for this research.

Both datasets were randomly split into 70% (3626) training data and 30%(1548) testing data. K fold Cross Validation (CV) over 5 model runs was done on the 70% training data to assess the performance of a model to generalise to unseen data. Additionally, it was used to assist in selecting a model that will perform best on the unseen data. The 30% testing data was used to further evaluate the performance of the model that will be selected from the K fold CV exercise.

Table 6.2: Number of training and testing observations found in the three broad geo-types classes; urban, rural, and farm. The classes are relatively balanced in terms of class labels in both the training and testing data

Geo-type Data	Urban	Rural	Farm
Geo-type Train Data	1180	1234	1212
Geo-type Test Data	504	524	520

Figure 6.1: Sample of the classification dataset depicting the three broad geo-type classes (Urban, Rural, Farm), which was manually labelled using the survey data to achieve maximum labelling consistency.

The classification dataset was created by manually labelling the geo-types of the aerial images. To achieve consistent labelling and minimise human labelling error, the NIDS data was used to obtain the geo-type for each topological code, which was then mapped back to the aerial images using the corresponding topological code. The random split of the full dataset into training and testing data resulted in relatively balanced data class labels of the geo-types. Table 6.2 shows the relative balance of the class labels in the geo-type data as Wei and Dunbrack (2013) demonstrated that balanced data positively affects the performances of a classification model. Figure 6.1

depicts a sample of the classification dataset and the three broad geo-type classes. It can be seen that without the use of NIDS data, labelling rural and farm geo-type classes would have been difficult because of the visually high similarities between the landscapes of the two classes.

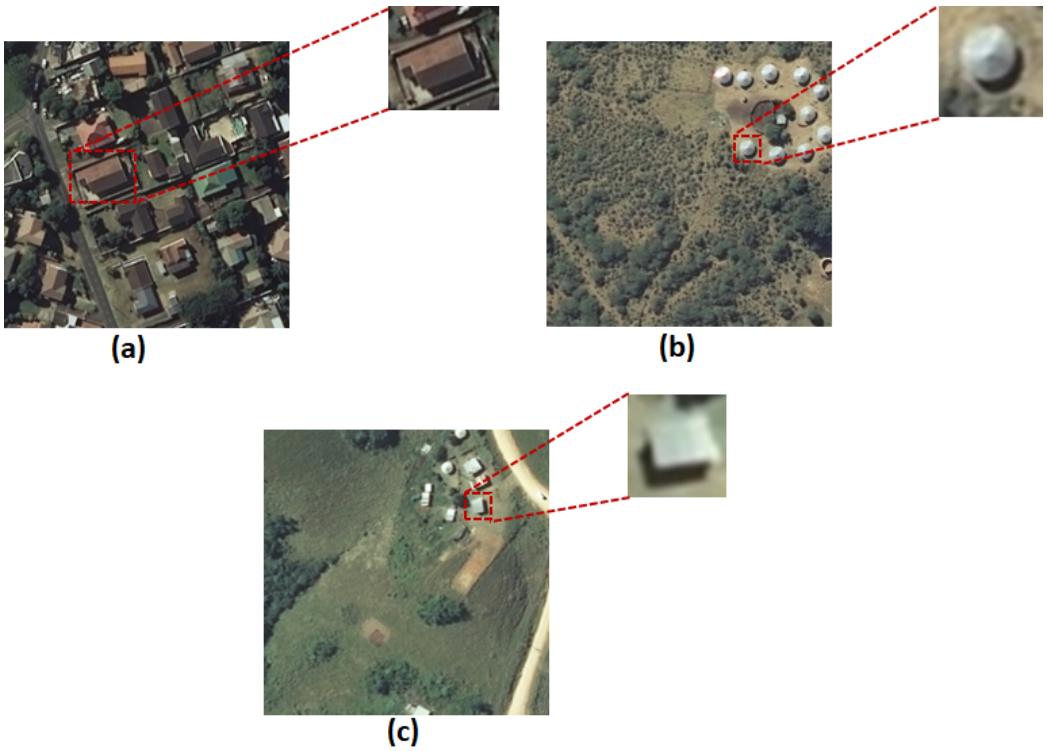


Figure 6.2: Sample of the detection dataset depicting the three broad dwelling type classes where each aerial image has an enlarged dwelling type class image on the top right corner showing (a) brick house with a polygon-shaped roof (b) traditional house with a circular-shaped roof (c) informal settlement with a small four sided polygon-shaped roof.

The dwelling type classes are visually differentiable, which made the labelling process less tedious. From Figure 6.2, it can be seen that from an aerial view brick houses tend to have polygon-shaped roofs, traditional houses tend to have circular-shaped roofs and informal settlements have relatively small four sided polygon-shaped roofs.

Next a detailed description of metrics used to evaluate the geo-type classification models is provided. Followed by the dwelling-type model evaluation metrics.

6.2.2 Evaluation Metrics For Geo-type Classification Models

The classification of geo-types is a multi-class classification problem where each image can only belong to one class of $K = 3$ non-overlapping classes (Sokolova and Lapalme, 2009). Two popularly used classifier performance metrics will be considered namely; average accuracy rate and categorical cross entropy loss (Sokolova and Lapalme, 2009). For each metric, confidence intervals will then be used to compare the performance of the models.

Average Accuracy Rate

Average accuracy rate is a version of the accuracy rate, which is a metric that measures the effectiveness of a classifier in terms of making correct classifications with macro-averaging applied to it (Sokolova and Lapalme, 2009). Macro-averaging involves averaging the same performance metric of the three classes (urban, rural, and farm) (Sokolova and Lapalme, 2009). To compute the accuracy rate, there are four measures that must computed first, namely:

- True positives (TP): Number of outcomes where the classifier correctly predicted a positive class.
- True negatives (TN): Number of outcomes where the classifier correctly predicted a negative class.
- False positives (FP): Number of outcomes where the classifier incorrectly predicted a positive class.
- False negatives (FN): Number of outcomes where the classifier incorrectly predicted a negative class.

These measures enables us to compute the accuracy rate as follows

$$Accuracy\ Rate = \frac{TP + TN}{TP + FN + FP + TN} \tag{6.1}$$

where the sum of TP and TN gives the total number correct classifications made and the sum of TP, TN, FP, and FN gives the total number correct and incorrect classifications. Eq 6.1 is easily applicable in binary classification problems, where $K = 2$. However, in multi-class problems, where $K > 2$, the effectiveness of the classifier is measured by the average accuracy rate, denoted AAR, of the classifier over the K classes and is defined as (Sokolova and Lapalme, 2009):

$$AAR = \frac{\sum_{k=1}^{K} \frac{TP_k + TN_k}{TP_k + FN_k + FP_k + TN_k}}{K} \tag{6.2}$$

where $\frac{TP_k + TN_k}{TP_k + FN_k + FP_k + TN_k}$ is the accuracy rate obtained from correctly classifying class k over the other classes (Sokolova and Lapalme, 2009).

Categorical Cross Entropy Loss

Categorical cross entropy (CE) loss is a log loss used to evaluate the performance of mutli-class classification tasks. It is defined as (Gomez, 2018):

$$CE = -\sum_{k=1}^{K} y_k log(p_k) \tag{6.3}$$

where y_k is the true label for class k and p_k is the predicted probability of belonging to the kth class. The loss increases as the predicted probability (p_k) diverges from the true label. A low loss is desired as this indicates that the model has predicted the probability of belonging to the correct class with high confidence (Gomez, 2018).

Confidence Interval

Confidence interval is an interval that provides a range of values, whereby there is (1-α) $\times$100% probability that the true unknown value lies with the given range (James et al., 2013). The value of alpha represents the significance level and it is popularly

0.05 (95% confidence interval) (Petty, 2012). It is defined as (Petty, 2012):

$$[\bar{x} - t_{\frac{\alpha}{2}} \frac{s}{\sqrt{n}}, \bar{x} + t_{\frac{\alpha}{2}} \frac{s}{\sqrt{n}}] \tag{6.4}$$

where $\bar{x}$ is the mean value of the sample measurement, $t_{\frac{\alpha}{2}}$ is a critical value of a student t-distribution with $n-1$ degrees of freedom at a confidence level of (1-α)$\times$ 100%, while n and s represent the sample size and the sample standard deviation of the data, respectively.

In this research, confidence intervals will be used to provide a range of average accuracy rate and categorical cross entropy loss for each of the geo-type classification models. Computing the confidence interval for each of these models will involve setting n to 50 because over the 5 model runs 10 fold CV was implemented which resulted in 50 models being trained and hence obtaining 50 model performance measurements. It should be noted that a small number of model runs was selected due to limited computing resources, because a high number of model runs or K folds results in increased computing time.

6.2.3 Evaluation Metrics For Dwelling Type Detection Model

Evaluation of object detection models, in this case Mask R-CNN, involves taking into account three tasks, classification, localisation and segmentation. These three tasks and the overall detection performance of the model can be assessed by using the log-loss and mean Average Precision (mAP).

Log Loss

The log-loss ($\mathcal{L} > 0$) of the mask R-CNN is given by

$$\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{bbox} + \mathcal{L}_{mask} \tag{6.5}$$

which has been explained in detail in chapter 5. A low log-loss is desired and it indicates that the model performs classification, localisation and segmentation tasks well.

Intersection over Union

IoU is a similarity metric for finite sample sets. It is also known as the Jaccard index and ranges between zero and one. In the context of object detection, it measures the accuracy of the detections made and is defined as the ratio between the intersection($\bigcap$) and the union ($\bigcup$) of the ground truth and predicted bounding boxes, denoted by B_{gt} and B_p, respectively, as shown in Figure 6.3.

$$IoU = \frac{Area \quad of \quad B_p \bigcap B_{gt}}{Area \quad of \quad B_p \bigcup B_{gt}} =$$

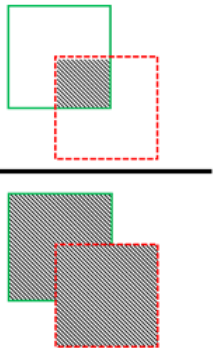


Figure 6.3: Equation of the IoU metric, where $B_p \bigcap B_{gt}$ denotes the intersection between the predicted and the ground truth bounding boxes and $B_p \bigcup B_{gt}$ denotes the union. Visual demonstration of computing the IoU metric in an object detection context, where the area of the small shaded square in the numerator represents the area of the intersection between the actual and predicted bounding boxes, and the area of the two shaded squares in the denominator represents the area of the union of the actual and predicted bounding boxes (Rezatofighi et al., 2019).

A high detection accuracy is associated with a high IoU score, which means that the predicted bounding box (B_p) heavily overlaps the ground truth bounding box (B_{gt}).

This can be seen in Figure 6.4, where the more B_p (represented by red dotted box) overlaps B_{gt} (represented by green solid box), the higher the IoU score and detection accuracy becomes and vice versa (Rezatofighi et al., 2019).

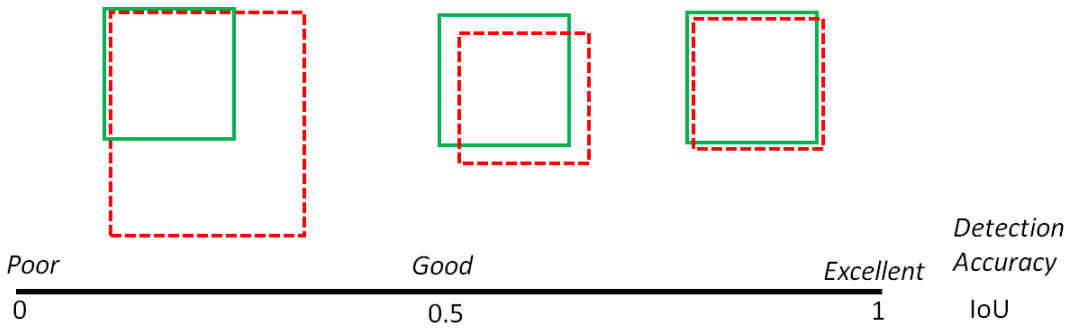


Figure 6.4: Various examples of IoU scores associated with poor to excellent detection accuracy scores, where the red dotted boxes and green solid boxes represents B_p and B_{gt}, respectively. The more B_p overlaps the B_{gt} the better the detection accuracy.

In object detection, a detection is considered correct if the associated IoU score is greater than the pre-determined IoU threshold (p), which is commonly 0.5 and 0.7. This is also known as a true positive (TP) detection. When the IoU score is less than p then the detection is a false positive (FP).

Mean Average Precision

Mean average precision ($mAP \in [0, 1]$) is derived from the precision metric given by

$$Precision = \frac{TP}{TP + FP} \tag{6.6}$$

where TP is the true positive detection and FP is the false positive detection. In an object detection setting, average precision is computed by using the TP and FP that is calculated using the intersection over union (IoU) metric as explained above, for each of the K classes. This is then averaged over the K classes to obtain mAP. A high mAP indicates that the model is performing well in terms of making detections.

6.3 Poverty modelling phase

This section provides data preparation involved in the poverty modelling phase and the evaluation metrics used to assess performance of the models.

6.3.1 Data Preparation

The NIDS wave 3 survey data has 2340 KZN households and after data cleaning there were 2001 households left. From the 2001 households, only 130 topological codes could be extracted from the data using the household geo-coordinates variables, which links back to the aerial images. For this phase, the topological codes were randomly split into 70/30% as shown in Table 6.3.

Table 6.3: Topological area code split into 70% training and 30% testing data.

Dataset	Training	Testing
NIDS Wave3	91 (70%)	39 (30%)

The best performing model will be selected based on the performance on the testing data using metrics which will be discussed next. Cross Validation (CV) was considered for model performance comparison but since the number of topological codes were dictated by the location of the households in the survey data, the training and testing approach is sufficient to analyse the performance of the regression models. However, to obtain optimal tuning parameters for ridge and lasso regression models, CV will be used.

6.3.2 Evaluation Metrics

The output of the regression models is an estimate of the poverty rate, SST index, which is continuous and ranges between zero and one. To assess the performance of the models of this nature, one can look at how well the model fits the data, the

relationship between the independent and response variable and how well the model estimates the response variable.

Residual Square Error

Residual Square Error (RSE) is a performance metric that measures the lack of fit by estimating the average deviation between the estimated response ($\hat{y}$) and the actual response (y) for N observations (James et al., 2013). It is measured in the units of the response variable in absolute terms and is computed as

$$RSE = \sqrt{\frac{1}{N-p-1}\sum_{N}^{i=1}(y_i - \hat{y}_i)^2} = \sqrt{\frac{1}{N-p-1}RSS} \tag{6.7}$$

where RSS is the residual sum of squares and measures the variation in the actual response variable not explained by the independent variables in the regression model.

In the context of this research, RSE represents the average deviation of the estimated SST poverty rate from the actual SST poverty rate. If the estimated response values ($\hat{y}_i$ for $i = 1, \ldots, N$) are close to the actual response values (y_i for $i = 1, \ldots, N$) then the RSE will be close to zero, which suggests that model fits data well (James et al., 2013). The main drawback is that RSE does not have an upper limit since it is measured in the units of the response variable in absolute terms. Therefore, one should also consider other metrics such as R^2 and r, which will be discussed next.

R^2 and r

R^2 is another measure of fit metric which is used to analyse the performance of the model based on the linear relationship between the response variable and independent variables. It measures the proportion of the variance of the response variable that can be explained by the independent variables, i.e. captured by the regression model. This metric is independent of the scale of the response variable and take on values

within the range of zero and one. It is computed by

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y}_i)^2} = 1 - \frac{RSS}{TSS} \tag{6.8}$$

where TSS is the total sum of squares, which measures the total variance inherent to the actual response variable (James et al., 2013).

R^2 values close to one indicate that a high proportion of variability of the response variable has been explained and captured by the variation in the independent variables in the regression model, whereas, an R^2 close to zero indicates that the regression model is unable to sufficiently capture and explain the variability of the response variable. In this research, R^2 is used to measure the proportion (%) of variation of the SST poverty rate explained by the *dwelling-type-geo-type* variables.

It is not always straight forward if one has obtained a good RSE or R^2 value. This is the main reason for evaluating the performance of a model by using multiple metrics. In this case, R^2 provides additional insight especially since R^2 has a interpretability advantage over RSE.

r is an alternative performance metric that adds more dimensionality in terms of how well the model estimates the response. It is defined as the correlation between the actual and estimated response values, which essentially measures the linear relationship between the response variable and the independent variables, same as the R^2. In fact, $r^2 = Cor(\hat{y}, y)^2$ is approximately equal to R^2, which is computed as

$$r = Cor(\hat{y}, y) = \frac{\sum_{i=1}^{N}(\hat{y}_i - \bar{\hat{y}})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(\hat{y}_i - \bar{\hat{y}})^2}\sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}} \tag{6.9}$$

where $\bar{y}$ and $\bar{\hat{y}}$ are the mean of the actual and the estimated response variables, respectively (James et al., 2013).

Next, results of the trained models are analysed and discussed by assessing the performance of the models using their respective evaluation metrics discussed in this chapter.

Chapter 7

Results

This chapter discusses the performance of the Phase 1 geo-type classification and dwelling type detection models and the Phase 2 regression models. The respective evaluation metrics described in Chapter 6 are used to analyse the performance of the models and to further identify the best performing models to be used in this research.

7.1 Phase 1 Results

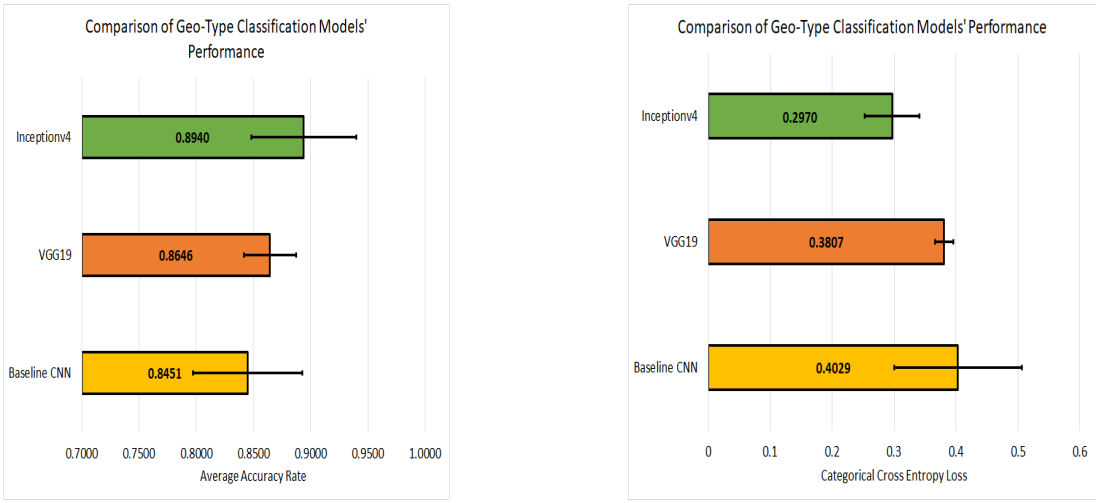
Overall the models in Phase 1 performed significantly well on the 5174 labelled images linked to the topological codes. Geo-type classification results were obtained by using 10 fold CV over 5 model runs and for each model run, the same folds were used in all three models to allow for direct model performance comparison.

From Figure 7.1, it can be seen that the 95% Confidence Intervals (CIs) of the different models slightly overlap each other. The CI values have been further summarised in Table 7.1 and it shown that Average Accuracy Rate (AAR) and categorical cross entropy loss distribution of the baseline CNN model is [0.7976, 0.8927] and [0.2998, 0.5059], respectively, whereas for the VGG19 model is [0.8416, 0.8876] and [0.3661, 0.3954], and for the InceptionV4 model is found within [0.8480 , 0.94] and [0.2530, 0.3410], respectively. The test AAR of the InceptionV4 model is 0.8596, which is

the highest and followed by the VGG19 model with 0.8432 and then by the baseline CNN model with 0.8197 as shown in Table 7.1. It is also observed from Table 7.1 and Figure 7.1 that the InceptionV4 model overall outperformed the VGG19 and baseline CNN models. Henceforth, InceptionV4 will be used as part of the final model and will be our main focus.

Table 7.1: Performance Results of the Geo-Type Classification Models

Model	95% CI AAR	95% CI Loss	Test AAR
Base-line CNN Model	0.8451 ± 0.0476	0.4029 ± 0.103	0.8197
VGG19	0.8646 ± 0.0230	0.3807 ± 0.015	0.8432
InceptionV4	$\mathbf{0.8940 \pm 0.046}$	$\mathbf{0.2970 \pm 0.044}$	**0.8596**



(a) Average accuracy rate **(b)** Categorical Cross Entropy Loss

Figure 7.1: Model performance comparison using performance metrics obtained from 10 fold Cross Validation (CV) over 5 model runs of InceptionV4, VGG19 and Baseline Convolutional Neural Network (CNN). The error bars represents 95% Confidence Interval. InceptionV4 is the best performing model followed by the VGG19 and baseline CNN model, respectively.

The AAR of the InceptionV4 model indicates that the model has correctly classified 89.40% of the geo-type images. The low categorical cross entropy loss of the InceptionV4 model of 0.2970 indicates that there is minimal divergence between the

predicted classification probabilities made by the geo-type classification model and the actual class labels.

Due to limited resources, only one instance segmentation model was considered to perform dwelling type detection. The mask R-CNN with an Resnet101 backbone was chosen because it was successfully applied by CrowdAI to execute a mapping challenge that involved the use of aerial images to build missing maps (Mohanty, 2018). When the initiation started their baseline model achieved a mAP of 0.697 and after more training they finally achieved a mAP of 0.938, both measures were computed at an IoU threshold of 0.5.

The dwelling type detection model also performed relatively well. The model achieved a log-loss of 0.839 which is the sum of the class label loss, bounding box loss and mask loss. From Table 7.2, the class label loss is the lowest loss with a loss of 0.219, followed by the bounding box and mask loss with a loss of 0.220 and 0.401, respectively. These three losses each indicate that there is minimal divergence between the predicted probabilities and actual values of the class labels, bounding box tuple and binary mask, respectively. The model achieved a mAP of 0.665 at an IoU threshold of 0.5. This indicates that the model can sufficiently detect the three different dwelling types from aerial images.

Table 7.2: Performance Results of the Dwelling Type Detection Model

Metric	**Dwelling Type Detection Model**
Log loss	0.839
Class label loss	0.219
Bounding Box loss	0.220
Mask loss	0.401
mAP (@ IOU=0.5)	0.665

Now that the classification and detection models have been identified and trained. The Next phase involves identifying the best performing regression model for the purpose of modelling the poverty rate, SST. Household survey data is used in Phase

2 to compute the actual SST values for each topological code and the corresponding independent variables as discussed in Chapter 4.

7.2 Phase 2 Results

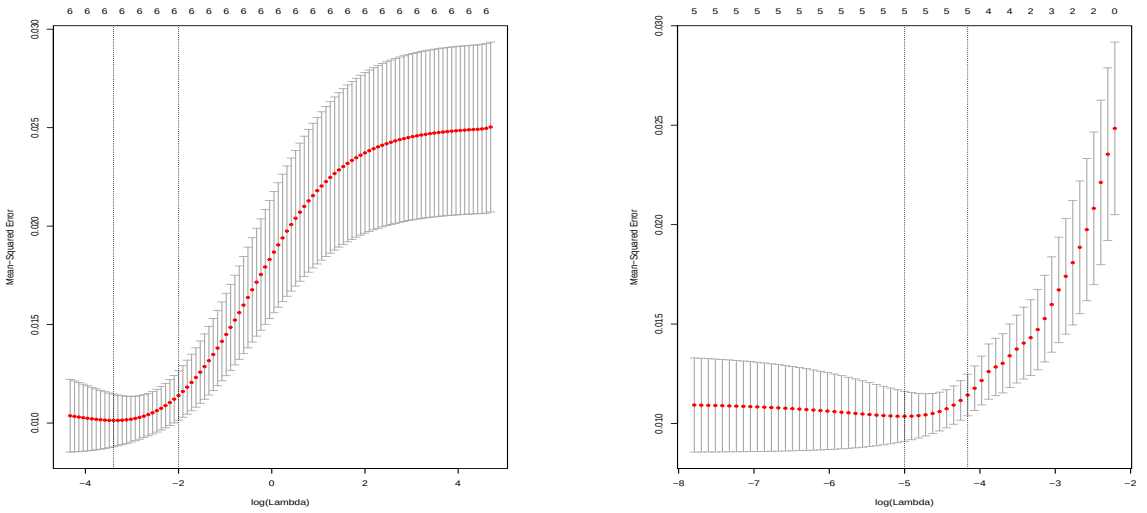
The performance of the regression models was analysed by considering three performance metrics namely; R^2, r and RSE. These metrics were used to compare and select the best performing regression model, which will be later discussed in more detail. The metrics are good performance metrics because they consider how well the different models fit the data based on the relationship between the response variable (SST) and independent variables (proportion of dwelling types found within each geo-type) and overall prediction capability. These variables and metrics have been explained more in detail in Chapter 4 and Chapter 6, respectively.

The data variables used to execute Phase 2 was given and described in Chapter 4 Table 4.3. It should be noted that the survey data used in this research has been assessed and it has been found that the data meets all the multiple linear regression assumptions listed in Chapter 5, see Appendix D for the assessment results.

From the multiple linear regression assumptions assessment, it is observed that a multiple linear regression model fitted using all the independent variables has an F-statistic of 13.08 with a p-value of 2.63e-09, see Table D.1 in Appendix D for the model summary output. The F-statistic is computed to test the null hypothesis that all the coefficients are equal to zero ($H_0 : \beta_1 = \beta_w = \cdots = \beta_p = 0$) against the alternative hypothesis that at least one independent variable is non-zero (H_a : at least one β_j is non-zero for $j = 1, \ldots, 9$) (James et al., 2013). The p-value associated to the F-statistic of the above regression model is less than the significance level of 0.05, hence we reject H_0 and conclude that there is strong evidence that at least one of the proportion of the dwelling types found within certain geo-types affects the level of poverty in an area.

Stepwise regression was used to perform variable selection from the above multiple linear regression model to identify the significant independent variables, which are the variables that affects the level of poverty in an area. Six out of nine variables were identified to be significant using stepwise regression, which are namely; $Brick\_house\_rural$, $Brick\_house\_urban$, $Brick\_house\_farms$, $Traditional\_house\_rural$, $Informal\_settlement\_rural$, and $Traditional\_house\_farms$. In this research, these independent variables will be used to model poverty using multiple linear regression, ridge and lasso models.

The performance of the multiple linear regression, ridge, lasso and compositional regression ($\alpha-$PCR) on the training and testing data has been summarised in Table 7.3. The optimal tuning parameters for ridge (λ_{ridge}) and lasso (λ_{lasso}) models were obtained by using 10 fold CV. These optimal tuning parameters corresponds to the minimum CV error of the models. Figure 7.2 depicts the CV Mean Sqaured Error (MSE_λ) over different values of log λ for both ridge and lasso regression models. The optimal ridge regression model has $\lambda_{ridge} = 0.0335$ and lasso regression has $\lambda_{lasso} = 0.0067$, these optimal values corresponds to the minimum mean square error values of 0.0101 and 0.0103, respectively.



(a) Ridge regression CV plots **(b)** Lasso regression CV plots

Figure 7.2: Cross-validation (CV) Mean Sqaured Error (MSE_λ) against log λ for ridge and lasso regression. The optimal tuning parameters for ridge (λ_{ridge}) and lasso (λ_{lasso}) models corresponds to the minimum cross-validation error of the models shown by the dotted lines.

The optimal α value and M number of principle components in $\alpha-$PCR were obtained to be 0.9 and 7 principle components, respectively. These optimal parameters were obtained by using 10 fold CV and corresponds to the minimum CV mean squared prediction error(MSPE) of 0.008750554. Figure 7.3 illustrates all the possible optimal combination of the parameters that minimise CV MSPE using a contour plot. The colour of the contour plot represents the CV MSPE values obtained for a given α and M number of principle components. Generally, it is observed that CV MSPE becomes smaller as M and α increase and the optimal parameters are found on the top right corner of the contour plot, which corresponds to the minimum CV MSPE.

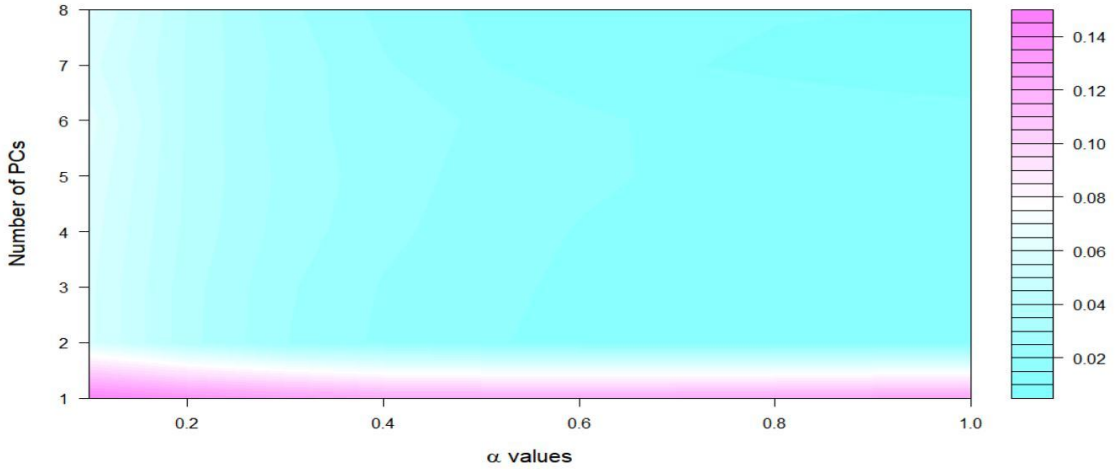


Figure 7.3: Contour plot of the Cross Validation (CV) mean squared prediction error(MSPE) over a range of α values and M number of principle components.

Table 7.3: Performance Metrics Results of Optimised Statistical Regression Models for Training and Testing Data

Term	**Multiple Linear Regression**	**Ridge** ($\lambda = 0.0335$)	**Lasso** ($\lambda = 0.0067$)	**Compositional Regression** ($\alpha-$PCR)
Train: RSE	0.0741	0.0806	0.0798	0.0750
Train: R^2	0.7360	0.6973	0.7002	0.7298
Train: r	0.8579	0.8351	0.8368	0.8543
Train: RSS	0.2308	0.2725	0.2674	0.2362
Test: RSE	0.0729	0.0600	0.0579	0.0562
Test: R^2	0.4546	0.5191	0.5768	0.6051
Test: r	0.6742	0.7205	0.7594	0.7779
Test: RSS	0.2232	0.1512	0.1410	0.1329

On the train dataset, it is observed in Table 7.3 that multiple linear regression has the highest R^2 of 0.736, which means that the proportion of $Brick\_house\_rural$, $Brick\_house\_urban$, $Brick\_house\_farms$, $Traditional\_house\ \_rural$, $Informal\_settlement\_rural$, and $Traditional\ \_house\_farms$ explain 73.60% of the variation in the SST index. Whereas the ridge, lasso and $\alpha-$PCR regression models have an R^2 of 0.6973, 0.7002 and 0.7298, respectively. In the case of the test data set, the $\alpha-$PCR regression model has the highest R^2 of 0.6051 followed by the lasso regression model with an R^2 of 0.5768, then ridge regression model with 0.5191 and multiple linear regression has the lowest R^2 of 0.4546. The drop of R^2 from train to test in multiple linear regression is due the instability of the OLS estimator as discussed in Chapter 5.

A positive correlation (r) between the estimated SST and actual SST is obtained for all regression models, as shown in Figure 7.4 and Table 7.3, which indicates that the regression models all have a strong ability to estimate SST from the classification of geo-types and detection of dwelling types from aerial images. On the train data set, all of regression models have very high r of above 0.8. The multiple linear regression model has an r value of 0.8579, followed by the $\alpha-$PCR model with 0.8543, then lasso model with 0.8368 and the ridge model has the lowest value of 0.8351. However, on the test data the order is changed and the $\alpha-$PCR model has the highest r of 0.7779 and multiple linear regression model has lowest r of 0.6742.

The performance of the models can also be observed in Figure 7.4 where on the training data set, the data points lie relatively close to the 45° line for all three models resulting in high r values. In the test data set case, multiple linear regression has data points slightly further away from the 45° line. On the train data set, the models tend to over-estimate (data points lie below the line) the SST index when there is low poverty (low SST rate) and under-estimate estimate (data points lie above the line) when there is high poverty (high SST rate).

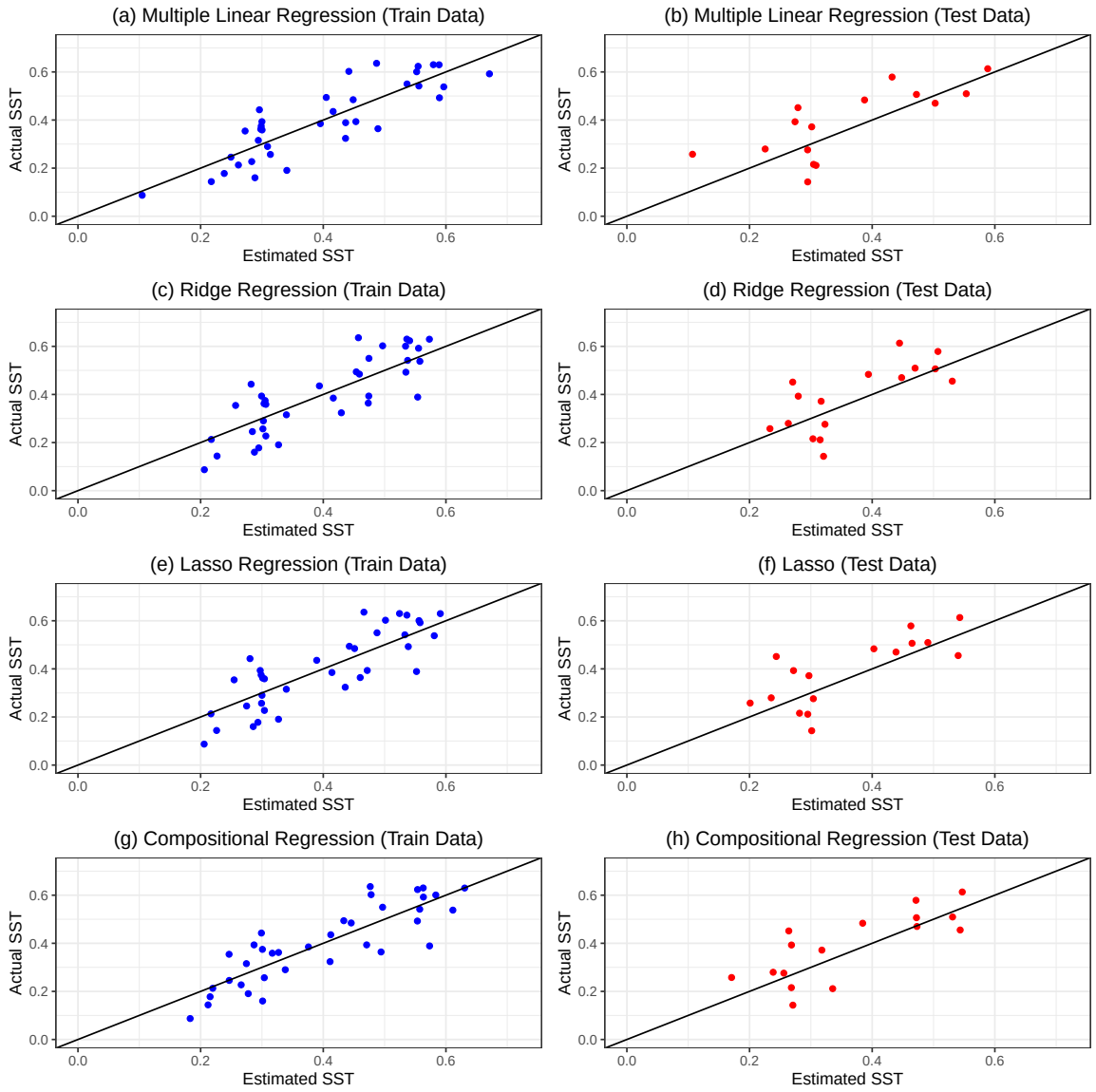


Figure 7.4: Actual SST index vs estimated SST index of multiple linear regression, ridge, lasso and compositional regression ($\alpha-$PCR) models. On the train data set, the models tend to over-estimate (data points lie below the line) the SST index when there is low poverty (low SST index) and under-estimate estimate (data points lie above the line) when there is high poverty (high SST index)

On all the plots in Figure 7.4, there appears to be two primary clusters of topological code areas on the SST index band. Areas found on the lower end of the SST index band are areas that have lower incidence and depth of poverty and inequality. These areas essentially have fewer households living below the poverty line and the households that do live below the poverty line are slightly living below it. The second cluster is on the upper end of the SST index band and is made up of many households extremely living below the poverty line.

Overall, from Figure 7.4 it can be seen that all the regression models performed relatively well against each other. The multiple linear regression model outperformed the ridge, lasso, and compositional regression ($\alpha-$PCR) models on all the performance metrics analysed in the train data set case. However, in the test data set case, the $\alpha-$PCR model with $\alpha = 0.9$ and $M = 7$, outperformed the multiple linear regression, ridge ($\lambda_{ridge} = 0.0335$) and lasso ($\lambda_{ridge} = 0.0067$) models. Therefore the $\alpha-$PCR regression model will be selected to be used in the final model to estimate the SST poverty rate.

As mentioned in Chapter 5, Section 5.3.5, the coefficient estimates corresponding to the principle components produced by $\alpha-$PCR cannot be easily interpreted and hence analysis of estimated coefficients will be focussed on the multiple linear regression, ridge and lasso models to provide insight on the effect of the independent variables.

The estimated coefficients of the multiple linear regression, ridge and lasso models have been reported in Table 7.4. The *Intercept* term found in Table 7.4 represents the SST index value that one would estimate if all the independent variables, which are proportion of dwelling types found within each geo-type, were equal to zero. The *Intercept* term for multiple linear regression, ridge, and lasso is obtained as 0.3914, 0.4149 and 0.3041, respectively. These values are all above zero which suggests that an area has an inherent poverty rate that's based on the location only and other factors either drive the rate up or down.

In the rural areas of KZN, estimated coefficients of the models suggests that the proportion of brick and traditional houses has a positive effect on the SST index, whereas, the proportion of informal settlements has a negative effect on the SST index. This can be collaborated by the survey data, in which majority of households living in brick houses and traditional households in the rural areas of KZN are living below the poverty line regardless of the employment status of the head of the house. Therefore, a proportional increase in brick or traditional houses results in a higher SST index and the opposite effect is observed with informal settlements.

Table 7.4: Estimated Coefficients and Performance Results of the Statistical Regression Model

Term	**Multiple Linear Regression**	**Ridge** ($\lambda = 0.0335$)	**Lasso** ($\lambda = 0.0067$)
Intercept	0.3914	0.4149	0.3041
Brick_house_Rural	0.1135	0.0777	0.3214
Traditional_house_Rural	0.0085	0.0262	0.1982
Informal_settlement_Rural	-0.0449	-0.0432	-1.3123
Brick_house_Urban	-0.0150	-0.0682	0
Brick_house_Farms	-0.0482	-0.0458	-0.4048
Traditional_house_Farms	0.0354	0.0209	1.0875

According to the multiple linear regression and ridge models, the proportion of brick houses in urban areas have a negative effect on the SST index, as one might expect. This means that for each proportional increase in brick houses in an KZN urban area, the SST index becomes higher. It is also observed from Table 7.4 that the *Brick_house_urban* coefficient estimate of the lasso model is equal to zero because unlike the other regression models, lasso is a shrinkage method that performs variable selection by shrinking coefficients to zero.

Furthermore, the models' estimated coefficients further suggest that the proportion of brick houses in farms have a negative effect on the SST index and the proportion of traditional houses has a positive effect. This means that in farm areas of KZN, an proportional increase in brick houses results in a lower SST index whereas a proportional increase in traditional house results in a higher SST index. From the survey data it is observed in both urban and farm geo-types that the majority of urban brick house and informal settlement residents live above the poverty line while the majority of traditional house residents live below it.

7.3 Poverty Distribution Mapping

The distribution of poverty can now be mapped using the final aerial poverty estimation model framework in Figure 7.5. The InceptionV4, Mask R-CNN, and compositional regression ($\alpha-$PCR) models have been selected to perform geo-type classification, dwelling type detection and poverty modelling, respectively, in the final aerial poverty estimation model.

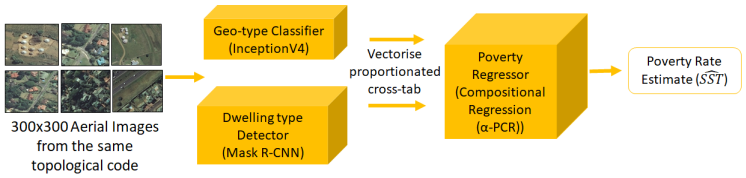
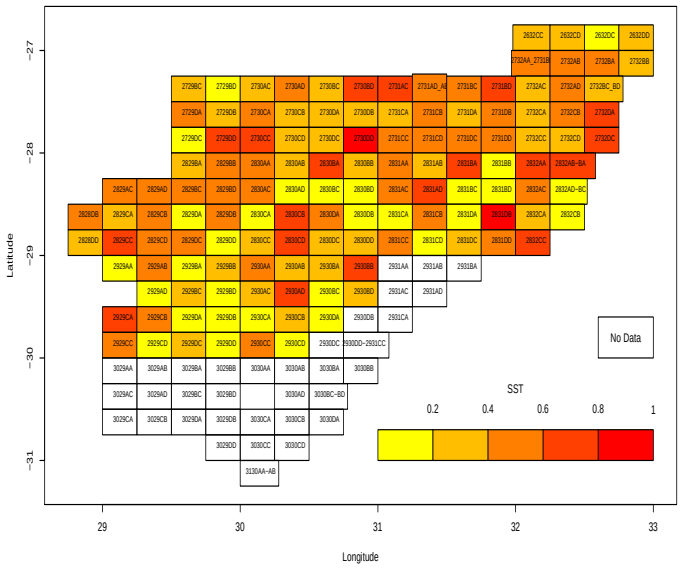


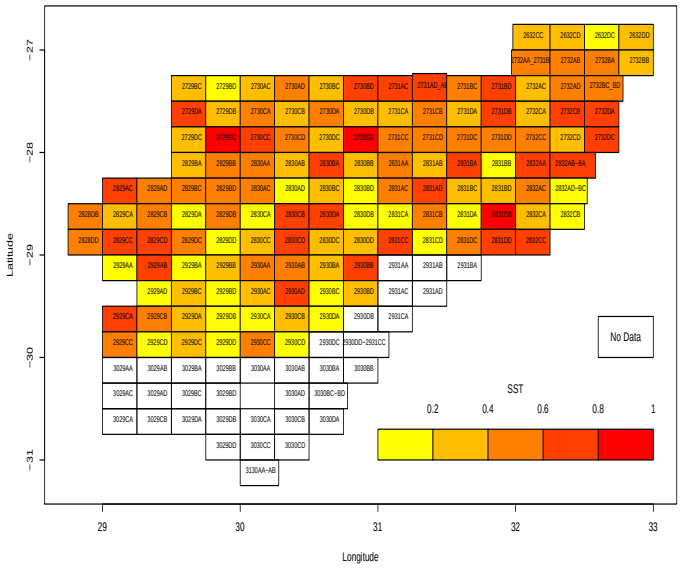
Figure 7.5: Final aerial poverty estimation model with trained InceptionV4, Mask R-CNN, and compositional regression ($\alpha-$PCR) models to perform geo-type classification, dwelling type detection and poverty modelling, respectively.

The trained InceptionV4 model for geo-type classification, mask R-CNN for dwelling type detection and compositional regression ($\alpha-$PCR) for poverty modelling was used to make estimates of topological codes in KZN, where aerial images were provided. From Figure 7.6, it can be seen that the final model is able to produce a reasonably similar distribution of poverty in KZN to the actual map obtained using NIDS wave 3 survey data (Southern Africa Labour and Development Research Unit, 2016).

The colour gradient of yellow to dark red is utilised to display the intensity of poverty in area. Yellow indicates that the location has a low SST index and a dark red indicates a high SST index. It appears that there tends to be high poverty rates in the northern areas of the KZN province and low rates as you tend towards the southern of the area.



(a) Estimated SST index



(b) Actual SST index

Figure 7.6: Sen-Shorrocks-Thon (SST) index distribution map in KwaZulu-Natal where (a) is the estimated distribution and (b) is the actual distribution obtained from the National Income Dynamics Study (NIDS) Wave3 (2012) data.

Chapter 8

Conclusion

This chapter concludes the implemented research by providing an overview of the research and the summary of findings.

This research presented an approach to estimate poverty from aerial images by using Convolutional Neural Network (CNN) coupled with a statistical regression model. This approach involves classifying and detecting the geo-types and dwelling-types, respectively, from aerial images and then aggregating the results to estimate the poverty rate, Sen-Shorrocks-Thon (SST) index, which is a composite poverty measure that captures a robust view of poverty. The application of this approach was demonstrated by using aerial images of Kwa-Zulu Natal (KZN), South Africa. This approach displays great potential in becoming a supplementary tool that the government can use to efficiently measure, monitor and analyse the poverty rate as they implement policies targeted to alleviate poverty.

All four research objectives mentioned in Chapter 1 were achieved with satisfactory results. Geo-type classification and dwelling-type CNN models were successfully created. Moreover, a statistical regression model that estimates the SST index from the identified geo-types and dwelling types was also successfully created, which further provided insight of the relationship between the identified dwelling types within

within each geo-type and the SST index. Next, a summary of the research findings are provided.

Overall, the Phase 1 geo-type classification models performed well. A three-layered CNN model was used as the base-line model to assess the performance of the VGG19 and InceptionV4 models which have more complex and deeper model architectures. Both models outperformed the base-line model which suggests that these deeper models outperforms small and simple models. Marginal difference in the performance of the VGG19 and InceptionV4 was observed, however, InceptionV4 achieved better results than the VGG19. The dwelling type detection model achieved satisfactory results by obtaining an mAP of 0.665 at an IoU threshold of 0.5 and a log-loss of 0.839.

In Phase 2, compositional regression ($\alpha-$PCR) model, with $\alpha = 0.9$ and $M = 7$ principle components, marginally outperformed the multiple linear regression model, ridge and lasso model by achieving a R^2 of 0.6051, RSS of 0.1329, r of 0.7779 and RSE of 0.0562 using the test data.

The trained InceptionV4, Mask R-CNN and compositional regression ($\alpha-$PCR) models have been selected to form part of the final model to estimate the poverty rate, SST index. The selection was done on the basis of models' performance in the respective tasks. Using the final model to estimate poverty, it was observed that the northern areas of KZN experience high poverty levels as compared to the southern areas.

Chapter 9

Recommendations and Further Studies

The main drawbacks of this poverty estimation approach is the amount of labelling time required and the amount of time spent to train the Convolutional Neural Network (CNN) models. Manually labelling the images requires a lot of time that one needs to plan ahead for. There are other labelling methods that can be explored that are more efficient and faster than manual labelling. There is semi-supervised learning where one trains a model using a small sample of labelled data to predict labels for unlabelled data (Hamidreza et al., 2017). Due to limited resources, this was implemented, however, it is recommended if one wants to efficiently label a lot of images.

Using Google Cloud Platform (GCP) assisted with reducing time spent to train the models, as we were able to utilise 8 GPUs to obtain good results for our research. It is highly recommended to use GCP to train computer vision models.

Finding aerial images that were collected concurrently with the survey data, took more time than expected. We were unsuccessful in obtaining aerial images that were exactly collected the same time as the survey data. There is a few months difference

between the collection dates for the two data sources. It is recommended to use survey data that is closely concurrent with the aerial images, because of the unknown embedded error that the model may inherently have.

The overall results can possibly be improved by considering other features that can be detected or identified using computer vision such as mines, police stations, hospitals, water sources, road type, etc. This study was done for one year, further studies can be performed on the change in poverty levels overtime using aerial images.

Bibliography

Abu-Mostafa, Y. S., Magdon-Ismail, M., and Lin, H.-T. (2012). *Learning from data*. AMLbook.com.

Aitchison, J. (1983). Principal component analysis of compositional data. *Biometrika*.

Aitchison, J. (2003). The statistical analysis of compositional data.

Al-Hassan, Y. (2010). Performance of a new ridge regression estimator. *Journal of the Association of Arab Universities forBasic and Applied Sciences*.

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Hasan, M., Esesn, B. C. V., Awwal, A. A. S., and Asari, V. K. (2018). The history began from alexnet: A comprehensive survey on deep learning approaches. *CoRR*, abs/1803.01164.

Babbie, E. R. (1990). *Survey Research Methods*. Wadsworth Publishing Company, 2 edition.

Beckerman, W. (1984). Measuring poverty in rich and poor countries. *Carnegie Conference Paper No. 3*, Cape Town: South African Labour and Development Research Unit (SALDRU).

Bhorat, H., Leibbrandt, M., Maziya, M., van der Berg, S., and Woolard, I. (2001). *Fighting Poverty-Labour Markets and Inequality in South Africa*. UCT Press.

Blumenstock, J., Cadamuro, G., and On, R. (2015). Predicting poverty and wealth from mobile phone metadata. *Science*, 350(6264):1073–1076.

Boureau, Y. L., Ponce, J., and LeCun, Y. (2010). A theoretical analysis of feature pooling in visual recognition. *27th International Conference on Machine Learning*.

Brown, C., Ravallion, M., and Van De Walle, D. (2016). A poor means test? econometric targeting in africa. *Technical Report , National Bureau of Economic Research*.

Campbell, J. (2002). *Introduction to Remote Sensing*. Taylor & Francis.

Chatterjee, S., Hadi, A., and Price, B. (2000). *Regression analysis by example*. John Wiley & Sons, Inc.

Chinhema, M., Brophy, T., Brown, M., Leibbrandt, M., Mlatsheni, C., and Woolard, I. (2016). National Income Dynamics Study Wave 4 User Manual. Cape Town: Southern Africa Labour and Development Research Unit.

Craig, B. J. (2007). Online satellite and aerial images : issues and analysis. *North Dakota Law Review*, 83:547.

Efron, B. and Hastie, T. (2016). *Computer age statistical inference*. Cambridge University Press.

Egozcue, J., Pawlowsky-Glahn, V., Mateu-Figueras, G., and Barcelo-Vidal, C. (2003). Isometric logratio transformations for compositional data analysis. *Mathematical Geology*.

El-Fallah, M. and El-Sallam, A. (2010). Estimation methods for multicollinearity proplem combined with high leverage data points. *Journal of Mathematics and Statistics*.

Elvidge, C. D., Baugh, K. E., Kihn, E. A., Kroehl, H. W., Davis, E. R., and Davis, C. W. (1997). Relation between satellite observed visible-near infrared emissions, population, economic activity and electric power consumption. *International Journal of Remote Sensing*, 18(6):1373–1379.

Engstrom, R., Hersh, J., and Newhouse, D. (2017). Poverty from Space: Using High Resolution Satellite Imagery for Estimating Economic Well-being and Geographic Targeting. *World Bank*, (December):1–48.

Feng, M., Xiang, B., Glass, M. R., Wang, L., and Zhou, B. (2016). Applying deep learning to answer selection: A study and an open task. *2015 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2015 - Proceedings*, pages 813–820.

Ferreira, M. D., Corrêa, D. C., Nonato, L. G., and de Mello, R. F. (2018). Designing architectures of convolutional neural networks to solve practical problems. *Expert Systems with Applications*, 94:205–217.

Fu, G., Liu, C., Zhou, R., Sun, T., and Zhang, Q. (2017). Classification for high resolution remote sensing imagery using a fully convolutional network. *Remote Sensing*, 9(5):1–21.

Gebru, T., Krause, J., Wang, Y., Chen, D., Deng, J., Aiden, E. L., and Fei-Fei, L. (2017). Using Deep Learning and Google Street View to Estimate the Demographic Makeup of the US. *PNAS*, 114(50):13108–13113.

Ghadban, K. and Mohamed, I. (2016). Multicollinearity and a ridge parameter estimation approach. *Journal of Modern Applied Statistical Methods*.

Gomez, R. (2018). Understanding categorical cross-entropy loss, binary cross-entropy loss, softmax loss, logistic loss, focal loss and all those confusing names.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., and Chen, T. (2017). Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377.

Gueorguieva, R., R. R. and Zelterman, D. (2008). Dirichlet component regression and its applications to psychiatric data. *Computational statistics & data analysis*.

Hamidreza, A., Shakarian, P., and J. E. Kelly, S. (2017). Semi-supervised learning for detecting human trafficking. *Security Informatics*.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning*. Springer.

Haughton, Jonathan Khandker, S. R. (2009). *Handbook on Poverty and Inequality*. The World Bank.

He, K., Gkioxari, G., Dollar, P., and Girshick, R. (2017). Mask R–CNN. *Proceedings of the IEEE International Conference on Computer Vision*, pages 2980–2988.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proceedings of the IEEE International Conference on Computer Vision*, 2015 International Conference on Computer Vision, ICCV 2015:1026–1034.

Head, A., Manguin, M., Tran, N., and Blumenstock, J. E. (2017). Can Human Development be Measured with Satellite Imagery? *Proceedings of the Ninth International Conference on Information and Communication Technologies and Development - ICTD '17*, pages 1–11.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge Regression: Biased Estimation for Problems Nonorthogonal. *Technometrics*, 12(1):55–67.

Hron, K., F. P. and Thompson, K. (2012). Linear regression with compositional explanatory variables. *Journal of Applied Statistics*.

Instituto Nacional de Estadstica (2017). Poverty and its measurement.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning with Applications in R*. Springer New York Heidelberg Dordrecht London.

Jean, N., Burke, M., Xie, M., Davis, W. M., Lobell, D. B., and Ermon, S. (2016). Machine Learning To Predict Poverty. *Science*, 353(6301):790–794.

Kelley, K., Clark, B., Brown, V., and Sitzia, J. (2003). Good practice in the conduct and reporting of survey research. *International Journal for Quality in Health Care*, 15(3):261–266.

Khorram, S., Nelson, S. A. C., Koch, F. H., and Wiele, C. F. V. D. (2012). Data Acquisition. In *Remote Sensing*, pages 17–37. Springer-Verlag New York, 1 edition.

Kim, J., Lee, J. K., and Lee, K. M. (2016). Accurate image super-resolution using very deep convolutional networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 1646–1654.

Kleinberg, R., Li, Y., and Yuan, Y. (2018). An Alternative View: When Does SGD Escape Local Minima? *CoRR*, abs/1802.06175:1–16.

Kong, C. and Lucey, S. (2017). Take it in your stride: Do we need striding in CNNs? *CoRR*, abs/1712.02502.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems*, pages 1–9.

Kshirsagar, V., Wieczorek, J., Ramanathan, S., and Wells, R. (2017). Household poverty classification in data-scarce environments: a machine learning approach. (Nips).

Lancaster, H. O. (1965). The helmert matrices. *The American Mathematical Monthly*.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., and Jackel, L. D. (1990). Handwritten Digit Recognition with a Back-Propagation Network. *Advances in Neural Information Processing Systems*, pages 396–404.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2323.

Lecun, Y., Kavukcuoglu, K., and Farabet, C. (2010). Convolutional Networks and Applications in Vision. *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 253–256.

Lee, C. Y., Gallagher, P., and Tu, Z. (2018). Generalizing Pooling Functions in CNNs: Mixed, Gated, and Tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):863–875.

Lewbel, A. and Pendakur, K. (2006). *EQUIVALENCE SCALES Entry for The New Palgrave Dictionary of Economics , 2nd edition.* Palgrave Macmillan.

Liew, S. S., Khali-Hani, M., Ahmad Radzi, S., and Bakhteri, R. (2016). Gender classification: a convolutional neural network approach. *Turkish Journal of Electrical Engineering & Computer Sciences*, 24:1248–1264.

Lok-Dessalien, R. (2000). Review of poverty concepts and indicators. *UNDP Social Development and Poverty Elimination Division.*

Michigan State University Map Library (2016). South africa 1:50,000.

Mohanty, S. P. (2018). Crowdai mapping challenge 2018 : Baseline with mask rcnn. `https://github.com/crowdai/crowdai-mapping-challenge-mask-rcnn`.

Narayan, D., Patel, R., Schafft, K., Rademacher, A., and Koch-Schulte, S. (1999). Can anyone hear us? *World Bank*, pages 26–64.

National Planning Commission (2011). *Our future - make it work.*

Petty, M. D. (2012). Calculating and using confidence intervals for model validation.

Poobalan, G., Nilen, K., Nicolette, P., Andrew, R., Greg, T., and Zyl, N. V. (2007). Poverty and inequality in South Africa and th world. *South African Actuarial Journal*, 7.

Queiroz, C. and Gautam, S. (1992). Road infrastructure and economic development : some diagnostic indicators. *Policy Research working papers ; no. WPS 921. Transport. Washington, DC: World Bank.*

Raul, R. (1996). *Neural Networks - A Systematic Introduction.* Springer.

Reisert, M. and Burkhardt, H. (2007). Learning equivariant functions with matrix valued kernels. *Journal of Machine Learning Research*, 8(June):385–408.

Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., and Savarese, I. R. S. (2019). Generalized intersection over union: A metric and a loss for bounding box regression.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747.

Sen, A. (1976). Poverty: An Ordinal Aproach to Measurement. *Econometrica*, 44(44):219–231.

Shorrocks, A. F. (1995). Revisiting the Sen Poverty Index. *Econometrica*, 63(5):1225.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.*

Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Inf. Process. Manage.*, 45(4):427–437.

Southern Africa Labour and Development Research Unit (2016). National Income Dynamics Study 2012, Wave 3 [dataset]. Version 2.1. Cape Town: Southern Africa Labour and Development Research Unit [producer]. Cape Town: DataFirst [distributor], 2016.

Statistics South Africa (2017). Poverty Trends in South Africa: An examination of absolute poverty between 2006 and 2015. Technical report, Statistics South Africa, Pretoria.

Szegedy, C., Ioffe, S., and Vanhoucke, V. (2016). Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261.

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 4278–4284.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014). Going deeper with convolutions. *CoRR*, abs/1409.4842.

Thon, D. (1979). On measuring poverty. *Review of Income and Wealth*, 25(4):429–439.

Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society*, 58(1):267–288.

Tsagris, M.and Stewart, C. (2018). A dirichlet regression model for compositional data with zeros. *Lobachevskii Journal of Mathematics*.

Tsagris, M. (2015). Regression analysis with compositional data containing zero values. *Chilean Journal of Statistics*.

Tsagris, M., Preston, S., and Wood, A. (2011). A data-based power transformation for compositional data. *In Proceedings of the 4th Compositional Data Analysis Workshop*.

Tzutalin (2015). labelimg.

van Houwelingen, H. C. and Sauerbrei, W. (2013). Cross-Validation, Shrinkage and Variable Selection in Linear Regression Revisited. *Open Journal of Statistics*, 03(02):79–102.

Wang, H., M. J. and Tenenhaus, M. (2010). Regression modelling analysis on compositional data. *Springer Berlin Heidelberg*.

Wang, H., S. L. W. J. and Guan, R. (2013). Multiple linear regression modeling for compositional data. *Neurocomputing*.

Wei, Q. and Dunbrack, R. L. (2013). The role of balanced training and testing data sets for binary classifiers in bioinformatics. *PLoS ONE*.

Woolard, I., , and Murray, M. P. i. S. A. I. W. (1999). *Measuring Poverty in South Africa*. Number 99.

Wooldridge, J. (2014). *Introduction to Econometrics*. Cengage.

Xi Chena and William D. Nordhausa (2011). Using luminosity data as a proxy for economic statistics. *Proceedings of the National Academy of Sciences*, 108(21):8589–8594.

Xie, M., Jean, N., Burke, M., Lobell, D., and Ermon, S. (2016). Transfer Learning from Deep Features for Remote Sensing and Poverty Mapping. *CoRR*, abs/1510.0.

Zhifei Zhang (2016). Derivation of Backpropagation in Convolutional Neural Network. Technical report, University of Tennessee, Knoxvill, TN.

Appendix A

South Africa's National Poverty Lines

Table A.1: Poverty Line in Rands (Statistics South Africa, 2017).

Year	Food Poverty Line (FPL)	Lower-Bound Poverty Line (LBPL)	Upper-Bound Poverty Line (UBPL)
2006	219	370	575
2007	237	396	613
2008	274	447	682
2009	318	456	709
2010	320	466	733
2011	335	501	779
2012	366	541	834
2013	386	572	883
2014	417	613	942
2015	441	647	992
2016	498	714	1077
2017	531	758	1138

Adpated from (Statistics South Africa, 2017).

Appendix B

Map Sheet of Kwa-Zulu Natal

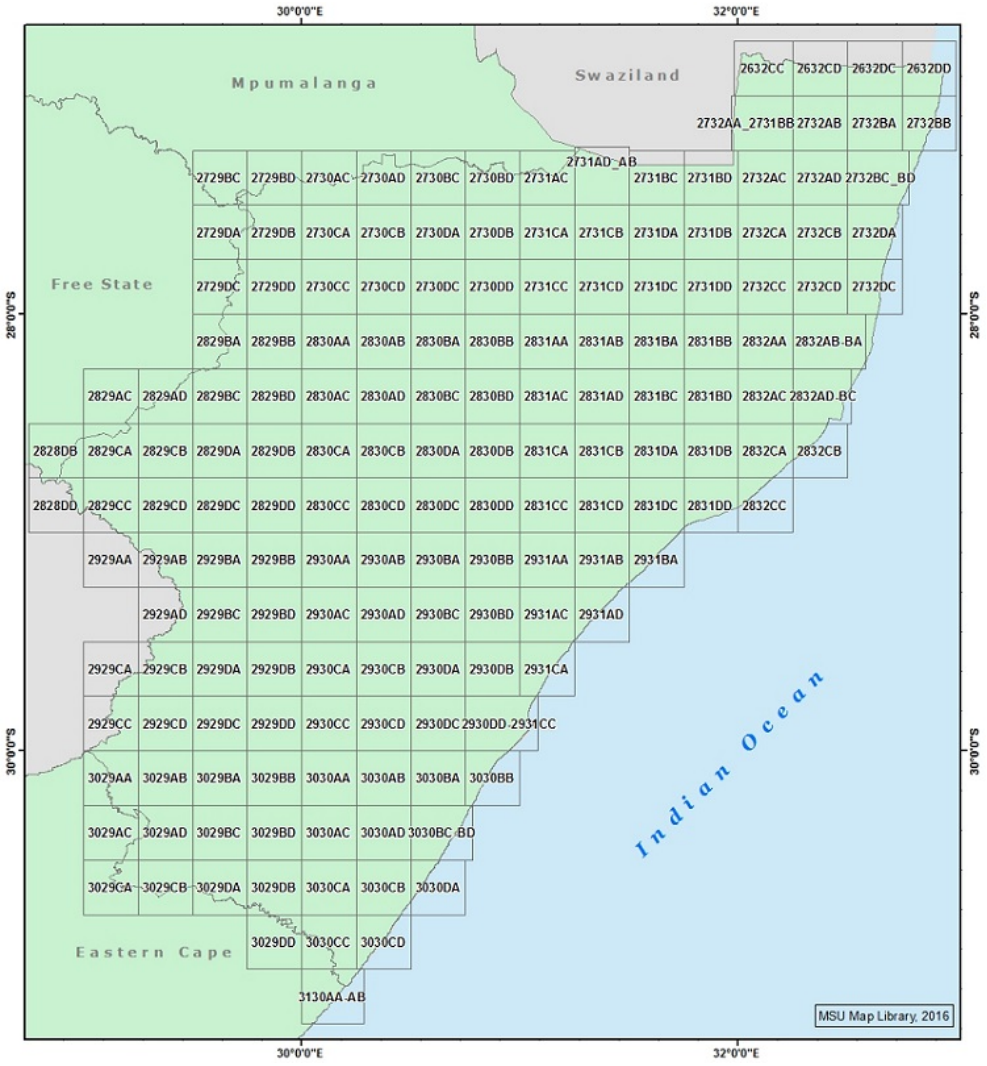


Figure B.1: Map sheet of Kwa-Zulu Natal (KZN), South Africa with overlaid topological codes (Michigan State University Map Library, 2016)

Appendix C

Geo-type Classification Model Architecture

C.1 Base-line Convolutional Neural Network Model Architecture

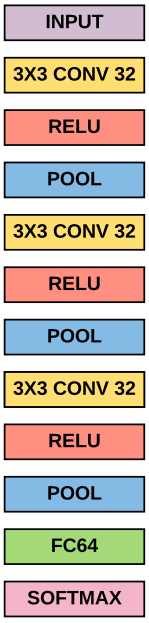


Figure C.1: Basic Convolutional Neural Network (CNN) is a 3-Layered CNN model with there convolutional layers each followed by a pooling layer. Adapted from (Goodfellow et al., 2016).

C.2 VGG19 Model Architecture

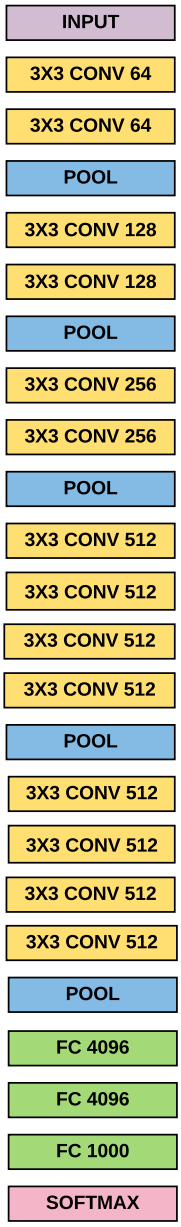


Figure C.2: Model Architecture of VGG19, adapted from (Simonyan and Zisserman, 2015)

C.3 InceptionV4 Model Architecture

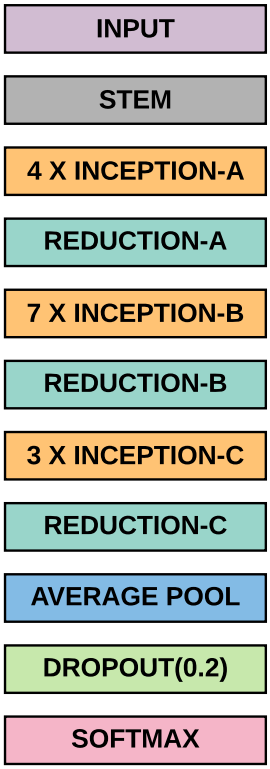
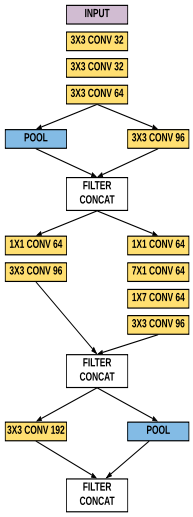
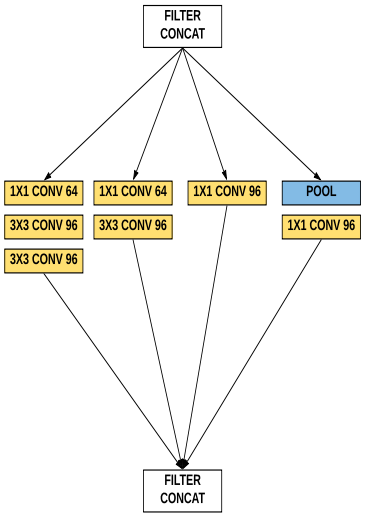


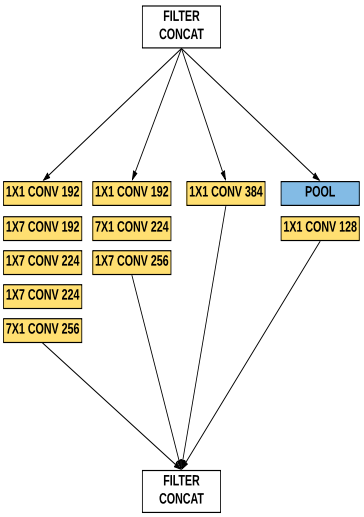
Figure C.3: InceptionV4 model architecture, adapted from (Szegedy et al., 2017).



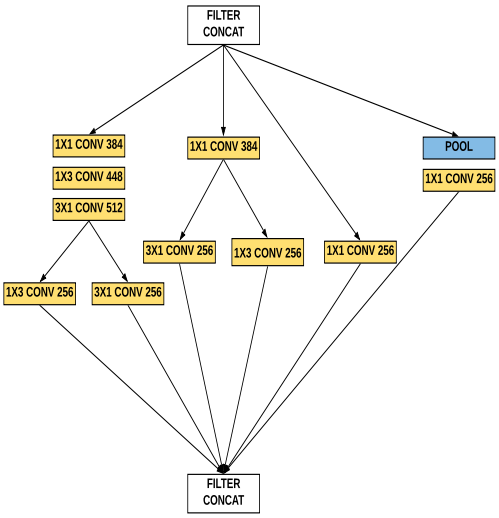
(a) Stem Module



(b) Inception-A Module



(c) Inception-B Module



(d) Inception-C Module

Figure C.4: Inception Modules, adapted from (Szegedy et al., 2017).

Appendix D

Multiple Linear Regression Assumptions

D.1 MLR fitted using all the independent variables

Table D.1: Multiple Linear Regression Model Summary

Term	**Estimate**	**Std. Error**	**t-value**	$\boldsymbol{Pr(>\lvert t\rvert)}^*$
Intercept	0.3946	0.012	31.798	**0.000**
Brick_house_Rural	0.0485	0.012	4.164	**0.000**
Traditional_house_Rural	0.0088	0.014	0.618	0.540
Informal_settlement_Rural	-0.0387	0.014	-2.768	**0.008**
Brick_house_Urban	-0.0304	0.013	-2.256	**0.029**
Traditional_house_Urban	0.0027	0.017	0.160	0.874
Informal_settlement_Urban	-0.0422	0.023	-1.852	0.071
Brick_house_Farms	-0.0344	0.016	-2.163	**0.036**
Traditional_house_Farms	0.0187	0.014	1.329	0.191
Informal_settlement_Farms	-0.0265	0.013	-2.067	**0.045**
	F-Statistic	13.08	*p-value*	2.63e-09

* Bold $Pr(>|t|)$ values indicates significant independent variables at 0.05 significance level.

D.2 MLR1 and MLR4: Model is linear in parameters and has homoskedastic errors

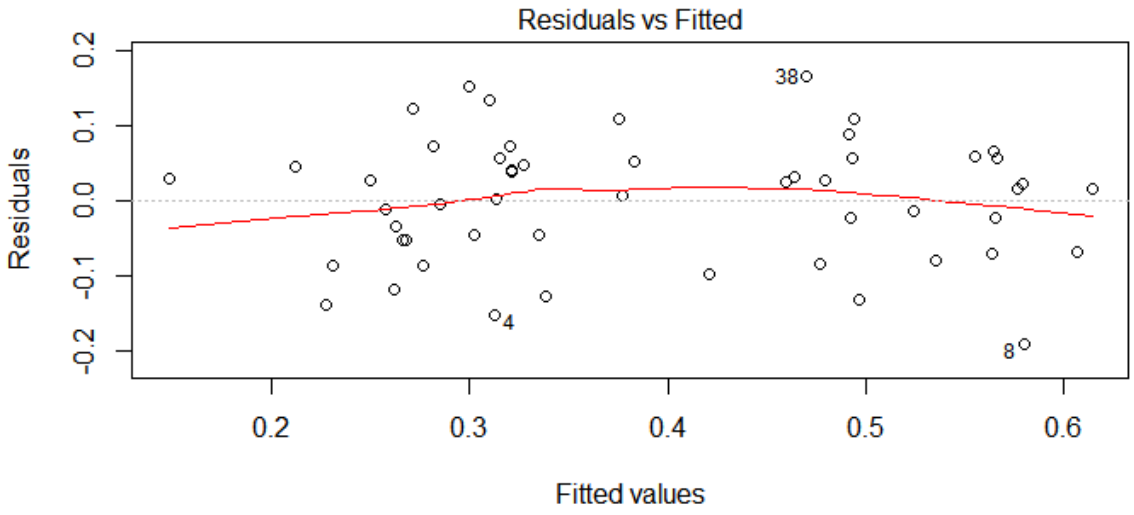


Figure D.1: Residuals against fitted values plot, which is used to assess that the model is linear in parameters and the errors are homoskedastic.

From Figure D.1, it appears that the observations are random and the line appears slightly flat with no trend. Furthermore, the residuals are spread equally around the line $y = 0$, which indicates that the model is linear in parameters and the error has the same variance given any independent variable. Therefore, the assumption that the model is linear in parameters and that the errors are homoskedastic is met.

D.3 MLR2: Normally distributed errors

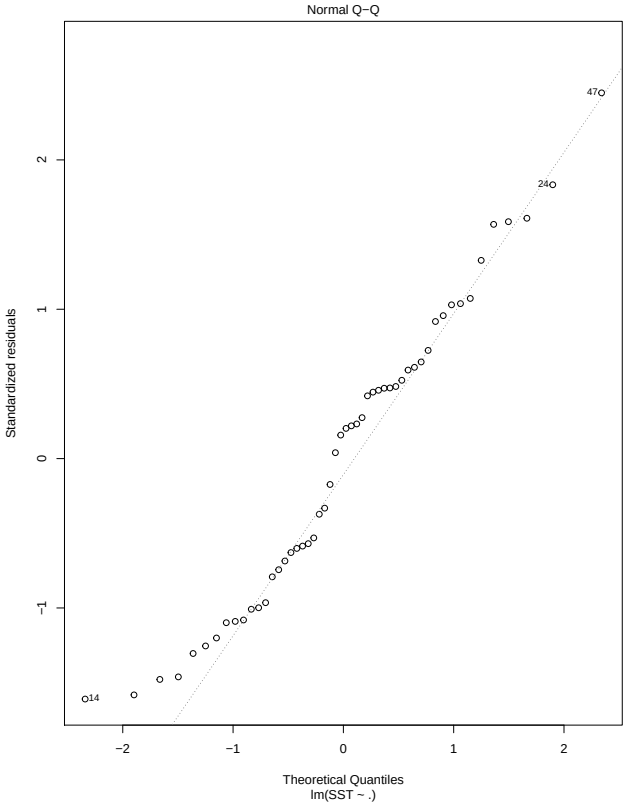


Figure D.2: Q-Q plot of the errors, which is used to assess whether the errors are normally distributed

It is observed in Figure D.2 that errors are normally distributes, because the observation follow the dotted straight line of the Q-Q plot, which is a plot used to assess whether the error are normally distributed. Therefore, the assumption of normally distributed errors is met. Shapiro-Wilk normality test is implemented at a significance level(α) of 0.05, to further test for normality.

Null Hypothesis: The errors are normally distributed.

Output: Test Statistic : W = 0.9847, p-value = 0.8890

Therefore, fail to reject the null hypothesis because the p-value = $0.8890 > 0.05 = \alpha$ with a test statistic of 0.9847, which suggests that the there is not enough evidence to conclude that the errors are not normally distributed. Hence, taking into account

the q-q plot in Figure D.2 and the Shapiro-Wilk test, the errors can be considered as being normally distributed.

D.4 MLR4: No perfect multicollinearity

Table D.2: Variance Inflation Factor(VIF) of the independent variables.

Independent Variables	VIF
Brick_house_Rural	2.342724
Traditional_house_Rural	1.479738
Informal_settlement_Rural	1.259692
Brick_house_Urban	6.975168
Traditional_house_Urban	2.129294
Informal_settlement_Urban	5.345794
Brick_house_Farms	1.664058
Traditional_house_Farms	1.262957
Informal_settlement_Farms	1.428227

There is no perfect multicollinearity among the independent variables because it is observed from Table D.2 that all the VIF values are less than 10, which is a rule of thumb VIF threshold value used to assess the presence of multicollinearity (James et al., 2013). Therefore, the assumption that there is no perfect multicollinearity is met.