PREDICTING MALARIA PARASITE RATES ACROSS AFRICA USING DEEP LEARNING WITH REMOTE SENSING DATA

Henrik Tornbjerg Carøe
201806624

Rasmus Klinkby Jørgensen
201808419

Advisors: Henrik Pedersen and Ira Assent

ABSTRACT

Malaria remains a significant public health concern, with millions of cases and half a million deaths reported annually, primarily in sub-Saharan Africa. Estimating malaria risk can be used in decision making about the distribution of medicine, cross-border policies, and city planning. This thesis uses RGB satellite images to predict malaria risk in contrast to most other researchers, which tend to use land cover. The advantages and disadvantages of the satellite and land cover data will be evaluated and discussed. As no benchmark dataset exists using RGB satellite images and parasite rates for the entirety of sub-Saharan Africa, we construct our own dataset. This is a challenging task. However, using the parasite rate data from the Malaria Atlas Project makes it possible.

Additionally, we also include other remote sensing data such as land cover, temperature, and precipitation. We therefore use different multi-modal data fusion techniques to utilize the data. Convolutional Neural Networks are employed to analyze the satellite images.

We experiment with various model architectures, techniques, and performance evaluation methods. Some of these architectures are a transfer learning model trained on remote sensing data and multi-task learning. Furthermore, we also convert the regression task formulation into a classification task formulation. Lastly, a comprehensive analysis is conducted using top-k and K-Nearest Neighbour analyses to inspect how the model performs and what it looks at.

The findings hint that the models' performance is limited by the quality and quantity of the data, indicating the need for higher-quality and more extensive datasets. Specifically, less noisy satellite images are needed together with more precise parasite rate data. This lack of data negatively affects the already few correlations in our dataset. Nevertheless, we conclude that RGB satellite images contain relevant information that can be utilized for making malaria risk predictions. Remarkably, it turns out that RGB satellite images can outperform the conventional land cover data in validation loss and root mean squared error. However, this improvement comes with the cost of a substantial increase in training time.

Contents

1 Introduction

In 2021, there were 247 million malaria cases worldwide, resulting in 619.000 deaths. Approximately 95% of these cases and deaths occurred in Africa. These numbers have been slowly increasing since 2015, when there were 230 million cases and 577.000 deaths [1].

One way of combating malaria is to estimate the risk of malaria in sub-Saharan Africa. The malaria risk estimation can be used for several things. It can assess the effectiveness of various control measures, such as insecticide-treated bed nets. Furthermore, risk estimation could also affect cross-border malaria control policies to restrict people from high-risk areas, thus limiting malaria spread [2]. This can be particularly crucial because infected individuals can transmit the malaria parasite to uninfected mosquitoes, which can further transmit the disease to others. Additionally, risk estimation can also be used to influence city planning, as it might help determine building materials and house positioning [3]. To estimate malaria risk, this thesis uses *deep learning* including *Convolutional Neural Networks* (CNNs).

When estimating or predicting malaria risk or more specifically, malaria parasite rates, it is important to understand the life of the mosquito, the main disease vector of malaria. Specifically, mosquitoes belonging to the Anopheles genus, the only species that transmits malaria [4], lay their eggs in water bodies where the mosquito larvae develop. Apart from when female mosquitoes consume blood for egg production, their regular diet consists of nectar and other plant juices, leading them to live in close proximity to vegetation [5]. As such, mosquitoes play a crucial role in the ecosystem for pollination in a wide range of plant species [6]. Considering this information, we include remote sensing data such as land cover and satellite imagery in estimating and predicting malaria parasite rate. These remote sensing data sources can assist in identifying water bodies and vegetation. Additionally, the time from an Anopheles egg being laid to being developed into an adult mosquito ranges from 6 to 16 days, highlighting the potential of mosquitoes to breed in temporary water puddles created by rainfall [7]. Therefore, precipitation data is relevant since it creates potential breeding sites. According to [8], these remote sensing data types have been utilized extensively in previous studies.

Another important aspect is that mosquito populations are constantly evolving. Therefore, researchers predict that rapid urban growth in Africa may adversely affect mosquito living conditions, potentially reducing their breeding sites and consequently lowering the malaria rate [9]. However, studies indicate that specific species such as Anopheles Stephensi flourish in urban settings [10]. Therefore, a decline in one species (e.g., due to urbanization) could result in the proliferation of another species. This further implies the need for city planning that may influence the prevalence of malaria.

The data mentioned above, including satellite imagery, land cover, precipitation, and other meteorological data, such as temperature, fall under the domain of remote sensing. Remote sensing is the technique of capturing data from a distant perspective of the Earth. Remote sensing can be categorized into two main groups: satellite data, which is obtained from satellites orbiting the Earth, and aerial data, acquired from aircraft or drones [11]. The resolution of the data and the distance from Earth can vary depending on which equipment and method is used. Before deep learning dominated the remote sensing field, traditional Neural Networks had been used but were slowly replaced by Support Vector Machines and Random Forests techniques [12]. However, in recent years the fast progression and development within the deep learning field have shifted the view back with great success [12]. Within the field of remote sensing, deep learning has been applied to change detection, scene classification, land cover detection, image segmentation, and other tasks [12].

The main contribution of this thesis is to utilize deep learning to predict malaria parasite rates using RGB satellite images collected between 2014 and 2020 across sub-Saharan Africa. This is possible by using malaria parasite rates from the Malaria Atlas Project (MAP) [13]. To utilize the different remote sensing data sources, this thesis uses multi-modal data fusion to combine the data and make predictions. Using multi-modal remote sensing data has been shown to improve predictions [8]. The Malaria Atlas Project uses multi-modal data in a Bayesian geostatistical model [14] to predict the parasite rate across the world and has an interactive visualization on its website [13]. To the best of our knowledge, this

thesis is the first to combine the commonly used remote sensing data types like land cover, temperature, and precipitation with RGB satellite images for malaria parasite rate prediction. As such CNNs will be used to analyse the images. CNNs have been used on satellite images before [15, 16, 17]. We suspect that RGB satellite images have not been used in malaria rate prediction because researchers see land cover data as more concise and still capture the essential information from the satellite images.

A problem within the remote sensing field is that there only exists benchmark datasets for high-resolution data, but none for both low- and medium-resolution satellite data [12]. This might restrict the progress within these topics. In line with this, to our knowledge, there does not exist a benchmark dataset for malaria parasite rate prediction. Due to this, we have developed a dataset using malaria parasite rate data from MAP [13] and satellite images that we scraped from the ArcGIS online web map [18].

The rest of the thesis is organized as follows: Section 2 provides the relevant related work. Section 3 describes and discusses the various data types. In section 4, the different models and methods used in the experiments are presented and explained. Section 5 showcases and discusses the results of the experiments. Section 6 presents an overall discussion of the experiments conducted. Finally, Section 7 provides a conclusion.

2 Related works

Keshavamurthy et al. (2022) [19] have surveyed the research on predicting the risk of infectious diseases such as COVID-19, influenza, and malaria. The surveyed papers' input data include remote sensing data and demographics. They observed that deep learning has increased exponentially in popularity in recent years. However, using CNNs to predict infectious diseases has only been used a few times. Only two of the 130 articles reviewed used CNNs to predict infectious diseases. A crucial aspect of the fight against malaria is understanding its spread and identifying high-risk regions. Malaria risk mapping is one way to do this.

Malaria risk mapping One of the aspects of the fight against malaria is knowing how it spreads and where it is most prevalent. A way to know this is to map malaria risk with statistical methods geographically. Odhiambo et al. (2020) [20] has systematically reviewed the literature on mapping malaria risk in sub-Saharan Africa. In 107 selected studies, they found that more than 50% of the studies used rainfall and temperature data. Furthermore, they find Bayesian geostatistical models to be the most used method (used by approximately 31%). It should be noted that none of their 107 selected studies used CNNs, which we have found to be a common theme when mapping the risk of malaria and other diseases. Odhiambo et al. [20] further find that different measures for malaria risk are used where malaria incidence (47%) and prevalence (35%) are the most common. Incidence denotes the proportion of new cases in a period, while prevalence denotes the proportion of all present cases at a specific time or period. Furthermore, 29% of papers used methods for spatial clustering of malaria. Half of the studies used model validation; most did this by making a training and validation dataset. Using variable selection techniques before mapping is also common (44%). One study that goes in great depth with variable selection is Weiss et al. (2015) [8], who are the people behind the Malaria Atlas Project (MAP).

MAP [13] has split Africa into grids of 5×5 km square cells and predicts the malaria parasite rates for each grid cell. This is done using remote sensing data and malaria surveys to which they have gained access from other research projects. Weiss et al. [8] have also reviewed which types of data other researchers have used as input features to their models. Going through 113 selected studies, Weiss et al. [8] found that temperature, rainfall, and land cover are the most used features, each used in over 50 other publications. Weiss et al. [8] thoroughly analyze the effectiveness of different variables' ability to predict malaria rates. They start with 33 initial variables and make new variations using different spatial and temporal summarizations, anomaly variables, and temporal lags to get 922 variables. Hereafter, these are transformed in 11 different ways, including square, square root, and log transforms, which results in 10142 variables. They then make all possible pairs, take the product between them, and end up with more than 50 million new covariates. Then, with

different processing techniques, they find the 20 best covariates. For example, one of the 20 best covariates is the product between the log of the mean annual precipitation and the square root of an anomaly variable of a vegetation index [1]. Each covariate has a coefficient telling whether it has a positive or negative correlation with the parasite rate. The example above has a positive coefficient, which means that when there is much rain and vegetation, this leads to a higher parasite rate. These 20 best pairs include 32 unique variable variants, consisting of 15 variables on temperature, 4 on vegetation, 3 on land cover classes, 3 on wetness, 2 on slope, 2 on precipitation, and 1 each for evapotranspiration, elevation, and brightness. The prominence of temperature variables among the best predictors suggests that temperature is highly relevant for predicting malaria. Looking at land cover, the three land cover classes were "Croplands", "Urban", and "Cropland/Natural vegetation mosaic". The "Croplands" class appears in one covariate pair with minimum nightly temperature and has a negative coefficient. The class "Urban" is in a pair with wetness, but "Urban" and wetness are transformed in a way that makes the coefficient hard to interpret. The class "Cropland/Natural vegetation mosaic" is in two pairs, but their coefficient is again hard to interpret. The paper [8] mentions that this lack of interpretability is a significant downside of their work.

In a later paper by Weiss et al. [14], they use the best variables found in their previous paper [8] and additionally use data about malaria control interventions such as bed nets, indoor residual spraying, and antimalarial drugs. They use a Bayesian space-time geostatistical model to predict the malaria rate among children in the age range of 2 to 10 years, and this is done with a 5 km resolution across sub-Saharan Africa annually from years 2000–17. One disadvantage of Weiss et al. [14] is that they write very little about their model's performance, as they focus more on the development and spread of the P. falciparum.

In a study by Krefis et al. (2011) [21], the impact of land cover on malaria incidence was investigated, which yielded more interpretable results than Weiss et al. [8]. The study area includes 12 villages in Ghana, and the researchers found that swampy areas and banana/plantain production were strong predictors of high malaria incidence. Specifically, a 10% increase in swampy areas within a 2 km radius of a village was associated with a 43% increase in malaria incidence and a 10% increase in banana/plantain production was linked to a 325% increase in malaria incidence. On the other hand, forested areas had a negative correlation with malaria incidence, with a 10% increase leading to a 47% decrease in malaria incidence. The study also argues that mosquitoes typically stay within a radius of 2 km of their breeding site. As such, Krefis et al. [21] use circles with radii of 0.5, 1, 1.5, and 2 km around their location of interest. Interestingly the different radii turned out to give similar results.

Mandike et al. (2016) [9] map malaria in Tanzania for children. Using a Random Forest algorithm, they generate their land cover images from satellite images. Land cover data are combined with data on wetlands, lakes, humidity, vegetation, and soil indicators and are used in a Boosted Regression Tree model to predict the malaria rate. As labels, they used 169 malaria surveys (similar to surveys used by MAP) from 2006 to 2014 for children aged 2 - 10. They find that proximity to rivers and vegetation has the most significant impact on parasite prevalence. Furthermore, they find that the parasite prevalence decreases within densely populated and urban areas.

Bui et al. (2018) [22] compare different machine learning models to predict malaria in Vietnam using remote sensing data, including weather, land cover, distance to road, and distance to a residential area. Their label data are much more precise than other research projects, as they have the specific coordinates of where a patient lives, both for the ones that have tested positive and negative. Many other research projects [8, 9, 21] have been conducting field surveys, for example, in hospitals, public schools, or libraries where that location has been registered instead of the patient's home address. Since they have data on whether each patient was infected, they can make a binary classification that predicts whether a patient is malaria positive or negative. They find that a type of decision tree called J48 with Random Subspace performs the best. Furthermore, they find that social

[1] The anomaly variable is the difference between monthly vegetation and the 13-year average vegetation index.

components, such as land use, distance to a residential area, and distance to a road, influence malaria risk the most.

Apart from mapping the malaria risk, models can also be built to forecast how the malaria risk will develop. This is known as an Early Warning System. Several attempts have been made to combat malaria epidemics using Early Warning Systems [23, 24, 2]. Such a system can help in determining which places the government should put extra focus on.

Land Cover As previously mentioned, land cover maps play a significant role in mapping malaria risk, and it is an active research area to enhance the accuracy of the land cover maps. According to the survey from Vali et al. [17] about land cover and land use classification with remote sensing data, they find that the topic has been growing steadily since 1970. Between 2015 and 2020, more than 2000 papers on land cover classification were published. It should be noted that according to Vali et al. [17], the use of deep learning for creating land cover maps emerged in 2015 and has increased in popularity ever since. Vali et al. [17] claim that deep learning models outperform traditional classifiers such as Support Vector Machines and Random Forest. However, deep learning models need a large quantity of training data before outperforming other classifiers. Furthermore, Vali et al. [17] also describe which methods can be used to combat the limited data problem. These methods include data augmentation, transfer learning, and Generative Adversarial Networks.

When working with remote sensing and land cover data, an important term to know is *spatial resolution*. The spatial resolution of a remote sensing image defines how large an area a single pixel corresponds to. Let us say we have a satellite image with 1000×1000 pixels, and the satellite image covers an area of 5×5 km, then the spatial resolution of the image is $5000m/1000pixels = 5m/pixel$. So assuming we have square images, let us call the number of pixels in a row of an image for p and the width of the physical area that the image covers for a, then the formula for spatial resolution is

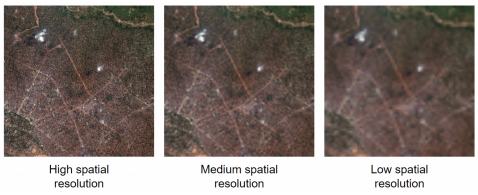


Figure 1: Different spatial resolutions.

$$\text{Spatial resolution} = \frac{a}{p}. \tag{1}$$

Counter-intuitively, when the spatial resolution value is small, e.g., $1m/pixel$, we say the image has a high spatial resolution. When the value is large, e.g., $300m/pixel$, the image has a low spatial resolution. This relation is likely tied to the standard definition of image resolution, i.e., the number of pixels in the image since images with a high spatial resolution often have a high image resolution. Figure 1 shows examples of images with different spatial resolutions.

A paper by Ma et al. [12] surveys the use of deep learning in the field of remote sensing. They look at around 200 papers from 2008 to 2018; more than half of them are from 2017 or 2018. Among the 200 papers, around 70 are about land cover classification, and around 90 use CNNs. Most of the papers use satellite images with a high spatial resolution, i.e., where each pixel corresponds to an area of 10×10 meters or less. However, one of the problems within the remote sensing field is the lack of benchmark datasets for images with a low spatial resolution, i.e., 30 meters/pixel or more. They claim this restricts the development within those areas.

Other projects The Zzapp/Omdena project [25] uses an alternative approach to combating malaria. They construct a map of the mosquitoes' breeding sites and manually spray them with insecticides. The map is used to register which water bodies have been sprayed. They use high spatial resolution satellite imagery to find the potential water bodies. However, they have a low amount of data problem, as high spatial resolution satellite is limited and can be expensive to acquire. Another issue is that some water bodies might not be visible

from the satellite images. This could be if the water body is covered by a roof like a water collection system or barrel.

There are other ways of combating malaria with statistics apart from using remote sensing data, where entirely different domains of data can be used. One of these ways is to automate parts of the malaria diagnosing process. Malaria can be diagnosed by looking at biopsies or blood samples of a patient. There exist benchmark datasets with blood samples for predicting whether malaria parasites are present in the blood samples. CNNs and other machine learning techniques have been used for this task [26].

3 Dataset

This section presents the various data types that we provide to our model. We utilize remote sensing data including satellite, land cover, temperature, and precipitation. Land cover data, temperature, and precipitation data have been previously explored in literature and shown to impact the parasite rate [8]. In addition, we will use parasite rates from surveys collected by the Malaria Atlas Project (MAP) [13] as our target values. MAP contains parasite rate data from different regions across most of Africa.

3.1 Malaria Atlas Project (MAP)

The Malaria Atlas Project has its roots at Oxford University and is a World Health Organization (WHO) Collaborating Centre [13]. It is one of WHO's main providers of data on malaria parasite rates in sub-Saharan Africa [1]. Much of the MAP's data is publicly accessible through the MalariaMap R package [27]. It should be noted that MAP is not responsible for going into the field and collecting the data. They make use of the data published for different research purposes. Due to this, the data come from different sources, giving rise to potential challenges. In MAP's papers [8, 14], they do not discuss or describe if they actively have done anything to ensure consistency nor if they have done any preprocessing or cleaning. If they have not done anything, this might lead to a noisy dataset, which can be challenging for the machine learning algorithm.

Some other challenges with the dataset are (1) that the longitude and latitude coordinates of some surveys (MAP does not describe which or how many) can be up to 5 kilometers from the actual site. (2) The temporal aspect has some uncertainty because the surveys span multiple months, and a person's initial infection could have happened months or even years before the survey date [8]. In order to combat the first challenge (1), the used remote sensing data should cover an area with at least a 5 km radius around the survey location. However, we chose to use a 5×5 km square with the survey location at the center, i.e. similar to a 2.5 km radius. We do this since it is what MAP uses, and it because it made it easier to get high spatial resolution satellite images.

In order to combat the second challenge (2), all our remote sensing data will be on an annual basis. Ideally, the dataset would have been more consistent and have better descriptions of the extent of their uncertainties.

MAP open-access dataset With the MAP R package [27], it is possible to access a malaria survey dataset openly. This open-access MAP dataset contains a total of 9524 surveys in Africa, ranging from 1984 to 2017, with most surveys from 2008. The surveys are from 47 out of the 54 African countries, with most surveys from Kenya which has 1610 surveys. Each survey/data entry has a longitude, latitude, number of participants, parasite rate, starting and ending month and year, age range of participants, method for diagnosing malaria, the malaria species examined in the survey, and a few other features. As the surveys have been performed for different research purposes, the number of participants can vary from 1 to more than 6000 for a particular survey. The average number of participants is 109.2 participants, the median is 54, and the most common size is 50, with 915 surveys. It would be preferable to have surveys of larger size, as this typically results in a more representative view (law of large numbers). Whether a size of 54 gives a representative view is hard to say and depends on the population size in the area.

Figure 2 plots the survey locations, survey sizes, and parasite rates across Africa for the Plasmodium falciparum parasite from the open access-dataset. These surveys are from different years so the parasite rate can vary a lot in local areas.

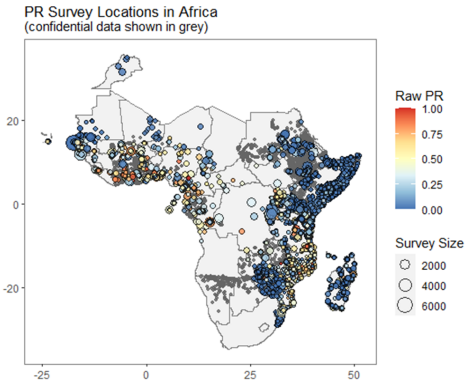


Figure 2: A plot showing the locations of surveys in the open access dataset in Africa, together with the size of the surveys and the parasite rate (PR).

MAP confidential dataset A confidential dataset can be requested from the Demographic and Health Surveys Program's website [2]. This confidential dataset combines the previously described open-access dataset with the new confidential data. This confidential dataset contains 24409 surveys in Africa from 1984 to 2018, with most from 2011 with 2616 surveys. The surveys are still from 47 countries, where most are from Tanzania with 3578 surveys. The average number of participants is 52.02, with a median of 19, and the most frequent size is 12, with 948 surveys of that size. This means that the average size has been more than halved, and the median is significantly lower, which is unpreferable as it might give a less representative view. However, we have chosen to use the confidential dataset over the open-access, as it contains more data and newer data. The newer data makes it possible to use newer satellite images (often clearer than older ones). Furthermore, having more data is almost always preferred if it gives a more representative view of the problem. By inspecting the MAP confidential dataset, we could observe that it contains more widespread data from different countries than the open-access dataset. This can lead to more noisy data as satellite images vary across Africa. This could result in the data becoming too diverse and without any correlations. On the other hand, it might give a better representation of malaria across Africa. Thus, our model might become better at generalizing, learning more general tendencies instead of local tendencies.

3.2 Pruned malaria data

We will be pruning upon the *Confidential* dataset. First, as only around 2% of the surveys are about another malaria species, namely the P. vivax, we will prune all surveys that are not about the P. falciparum species. The surveys about P. vivax generally contain a low participation count and parasite rate, so removing them simplifies the problem and we can ignore the imbalance between the two malaria variants. Secondly, we observe that some surveys have the same location and time span but cover different age ranges. In these cases, we have chosen to merge them into a single survey, as it simplifies the problem and results in a larger number of participants in the merged survey. Conversely, it could be interesting to do as [9] and only use data for kids aged between 2 and 10 years old, as they are more prone to become infected than adults [14]. Thirdly, we further prune any malaria surveys older than 2010, even though our satellite image only goes back to 2014. Meaning there will be a mismatch. However, as about 55% of our surveys are made between 2010 and 2014, we value more data higher than how accurate the match is between the survey date and the date of the satellite image. Finally, we prune surveys featuring less than 10 participants, as the survey size highly influences how representative the surveys are, directly reflecting on how representative the parasite rate is. After these prunings, the dataset has a size of 9826 data points. In the next section 3.3, we will be pruning the dataset further to make it possible to convert it into a classification problem with no class imbalance.

[2]Request access on: https://dhsprogram.com/

3.3 Regression as Classification

There are two main prediction problems for supervised learning: regression and classification [28]. They both have pros and cons and are used depending on the task that they try to solve. However, much more research has been put into the classification domain compared to the regression for neural networks [28]. Furthermore, a commonly seen practice is to convert the regression problem into a classification problem which usually can yield better results in terms of training time and fitting the data [28].

The problem of predicting the parasite rates is formulated as a regression problem, where the goal is to predict the numerical value of the parasite rate. However, the problem can be formulated as a classification problem by categorizing the parasite rate into e.g., high and low zones. The classification approach can still have the same use cases as the regression approach, e.g., be used as an early warning forecasting system to identify high and low risk areas. One challenge with the regression problem is that it needs to estimate the regression function, which makes the problem harder than classification, where it only needs to find the decision boundaries between classes.

Converting a regression problem to a classification problem is known as binning, discretization, bucketization, quantizing, or digitizing phenomenon [28]. In this thesis, we will be calling it binning. There are primarily two ways of doing it called *binning with equal-size* and *binning with equal-frequency* [29]. The equal-size is made by dividing the data into equally spaced bins. The equal-frequency is made by dividing the data into a set number of bins, such that each bin is balanced, i.e., contains the same amount of data points [29].

We want to perform an experiment to see how significant an impact the task formulation has on the remote sensing domain, specifically for satellite images. Thus, we will be converting our data into a classification problem.

(a) [Removed, Confidential], Parasite rates including 0.

(b) [Removed, Confidential],Parasite rates excluding 0.

Figure 3: Histograms showing the distribution of parasite rates, where parasite rates of 0 are respectively included and excluded. A bin size of 50 is used to make the plot more readable.

Figure 3 shows histograms of the parasite rates. There are 3360 points with a parasite rate of 0, which makes the first histogram (figure 3a) hard to read. So, we also show a second histogram (figure 3b), where they are excluded. Most data points have a low parasite rate and the count is steadily decreasing for higher parasite rates. So the class bins could not really be made according to natural boundaries in the data. However, roughly $1/3$ of our data has 0 in parasite rate, $1/2$ has between $[0 - 0.5]$, and $1/6$ between $[0.5 - 1]$. Using the equal-size bins will result in an imbalance as we have much more data closer to parasite rate 0 than 1. Thus we would have to accommodate for this imbalance when training and interpreting the results. So instead, we choose to use the equal-frequency binning method and make the intervals for the different classes as seen in table 1.

Classes	**Intervals**
None	[0]
Low	(0-0.2]
Medium	(0.2-0.5]
High	(0.5-1]

Table 1: Shows classes and intervals when converting to a classification problem.

The number of classes can influence the complexity of the problem. Separating it into numerous classes makes the problem harder, and separating it into a few classes makes it easier but less informative. Thus, a trade-off exists. In table 1, *High* is the class containing the lowest number of data points. Hence it was the dependent factor for the number of data points in a class. We chose the interval $(0.5 - 1]$ as this resulted in 1822 data points for each of the classes, which meant we would be able to use 7288 data points out of the 9826 points from our dataset described in section 3.2. The threshold between the *low* class and *medium* class was set such that they both would include at least 1822 elements which turned out to be a threshold at 0.2. Contrarily, Weiss et al. [14] define different levels of malaria endemicity. Weiss et al. [14] split it into four different categories. (1) *malaria-free* where the malaria rate is 0%, (2) *hypoendemic* where the malaria rate is between 0–10%, (3) *mesoendemic* from 10–50%,

and (4) *hyperendemic* or *holoendemic* where the malaria rate is above 50%. We choose to use a larger *low* class since if the interval was $(0.0 - 0.1]$, then the class would only have 1581 data points. Meaning that the *low* class instead would be the dependent factor resulting in 964 fewer data points in the final dataset.

For the classes *medium*, *low*, and *none*, we had to discard some data. This was done by inspecting the number of survey participants and selecting the ones with the most participants. Our converted dataset ended up with 7288 images while being balanced. Having an unbalanced dataset can result in less informative gradients, because most of the training time will be used optimizing for the classes with large amounts of data [30]. This effect is further amplified if using a small batch size. Therefore, these less informative gradients can result in slower training time. Interpreting the results also requires extra effort, as we have to consider the class imbalance.

3.4 Satellite data

The satellite data is from ArcGIS's online web map [18], where it is possible to view spatio-temporal satellite images for some specific dates back to 2014. We have developed a Python program to automatically scrape images by entering the coordinates and saving a screenshot of the location. The satellite images have a size of 5×5 km with the survey's location as the center. The satellite images are 1024×1024 pixels with roughly 5 meters per pixel spatial resolution.

We choose 5×5 km because the survey's locations can vary from the actual location as mentioned in section 3.1. Furthermore, we assume the people live some distance from the survey location. Another reason for using 5×5 km areas is that it is the same size that Weiss et al. [8] use. However, it should be noted that Weiss et al. [8] uses land cover images and not RGB satellite images. Lastly, as previously mentioned Krefis et al. [21] argue that mosquitoes generally stay within a 2 km radius of their breeding site.

Regarding the problem with the month of infection not being specified, we cannot simply take a satellite image from a specific month, even though it would be ideal. Furthermore, ArcGIS's web map has a lot of maps available, but they vary in capture date and only go back to 2014. When a location is entered, ArcGIS finds the nearest map based on the chosen date by going back in time. In most cases, ArcGIS finds an image from or close to the selected date, but sometimes it fails and might go back multiple months. We have chosen to use newer satellite images than the actual dates. For the data points from before 2014, we naturally use satellite images from 2014 since this is the closest we have. For the subsequent data points from after 2014, we map them to the closest image going forward in time, as shown in table 2. In order to make the scraping process more efficient, the dates are selected by inspecting different coordinates for different countries and seeing which specific map dates most often occur. Another strategy would be to make intervals starting one year before a map date and ending one year after, e.g., all data points with a survey date between 2016 and 2018 would be assigned a satellite image from 2017. Thus the data would be more centered around the map date. However, from a visual inspection of a sample of data points, it could be observed that local changes that happened in those years were not visible from the 5×5 km square. What was more noticeable was that newer satellite images are more sharp and clear. Hence, it is easier to see details such as houses and trees in those images. Therefore, by choosing the first strategy, we would use newer images for more data.

3.5 Land cover & land use data

Land cover and land use are closely related to a satellite image. Land use describes how people use the landscape, while land cover describes how much of the image is covered by water, vegetation, urban areas, farmlands, or other types. These classes vary a lot among different datasets.

Land cover and land use can be used for various purposes. This has led to much research within the topic, resulting in many datasets with very diverse data. The land cover data we choose to use is from Digital Earth Africa, and the data is freely available [31]. They have datasets with different resolutions, classes, and time spans [32]. We use the *ESA Climate*

Change Initiative (CCI) land cover dataset because of its timespan from 1992 to 2019. The CCI data has a spatial resolution of 300 meters, whereas other datasets have 100 or 10 meters. The CCI data has an accuracy of around 75% (based on an analysis of the data from 2015)[33], whereas some of the other datasets can have up to around 80% [32]. It is preferable to use higher accuracy and resolution. However, those from ESA with higher resolution only date back to 2015. Thus we will be trading off accuracy and instead use data with a longer time span. Since our satellite images each cover an area of 5×5 kilometers, our land cover images are going to be 17×17 pixels because $300m/pixels \cdot 17pixels = 5100m$. The CCI land cover dataset contains 38 classes, with some of them being different varieties of trees, urban areas, and water bodies (all the classes are listed in figure 4 and table 3). The data format is greyscale images, where each class has a different pixel value. The many classes may be useful as some of the classes might impact mosquitoes more (like banana trees were shown to do in [21]). One challenge is that having 38 classes, where many are subcategories of each other, can make the problem harder for the model to learn. Hence, an interesting experiment could be to try to merge some of the classes, e.g., some of the forest classes, to see if this impacts the performance of the model (we do this in section 4.12 and 5.9).

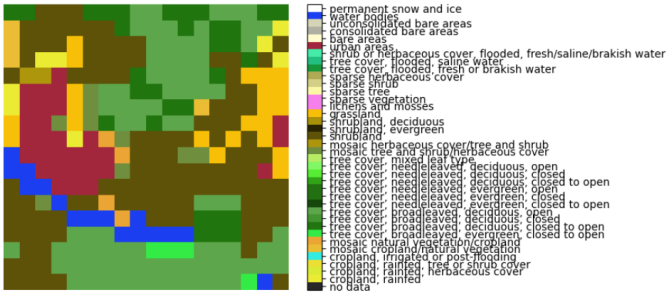


Figure 4: A picture of a colorcoded cci land cover example.

3.6 Precipitation data

Precipitation data are different types of water forming in the atmosphere and falling back to the earth. The most common is rain and snow. This thesis uses precipitation data collected from the U.S. National Oceanic and Atmospheric Administration which has data from the Global Precipitation Climatology Centre [3]. It contains monthly precipitation for the entire planet from January 1891 to December 2019 with a spatial resolution of 0.25×0.25 degrees, i.e., about 30×30 km per pixel. The data is quality-controlled and collected from 67200 weather stations worldwide, including Africa, where each station has data from at least the last 10 years. Thus, we can make crops accordingly to the survey locations. Since the spatial resolution of the precipitation data is 30×30, which is larger than our 5×5 km satellite images, we get a single value (i.e., an image with a single pixel) when we use 5×5 km areas for the precipitation data.

Intervals	Map dates
2010-2014	2014-11-12
2015-2017	2017-04-19
2018	2020-11-18

Table 2: Shows the ArcGIS image interval ranges and which map that has been used for them.

There are different ways of aggregating the precipitation data. In theory, the most precise and optimal solution would be to use data from the specific month(s) the surveys span over. However, as mentioned in the MAP section 3.1, one of the problems with the malaria data is that survey participants could have had their infection before the survey's time span. This means that even if we made the precipitation data match the survey months exactly, it might not match the date when a participant got infected. To use the exact survey months, we would need to know how often participants are infected months or years before the survey.

[3]The data can be accessed at `https://psl.noaa.gov/mddb2/makePlot.html?variableID=1627&fileID=532575`

However, Weiss et al. [8] do not specify this uncertainty. Due to this, we have instead gone with a more general solution. We take each survey location's average monthly precipitation for the entire year. By taking the yearly average, instead of only the months of a survey's time span, we might remove the noisy infection dates, which can improve our model's performance. However, an issue with this approach is that values can vary a lot during the seasons of a year, leading to high variance. Alternatively, we could use variance instead of average (Weiss et al. [8] experiment with different aggregation techniques) but have not done that in this thesis.

3.7 Temperature data

The temperature data is from NASA's Earth Data website [4]. The dataset contains land surface temperature with 1×1 km spatial resolution every 8'th days from 2000 to 2020. The data is captured by MODIS sensors on satellites. Specifically, we are using the *LST_Day_1km* layer from the MOD11A2 v006 dataset [34], which is the land surface temperature during the daytime. Like for the other data types, we take crops of the data for each survey's location. Unlike the precipitation data, the temperature data is captured every 8'th day. Again due to the same problem with the specific infection month not being specified in the malaria surveys, we take the average temperature across the year. Since the spatial resolution of the temperature data is 1 km, we get a 5×5 pixel image. In our data from 2018, the median temperature was 24 degrees Celsius, and the temperature ranged from -2.2 to 57.1 degrees Celcius. It could seem incredible with 57.1 degrees Celcius, but it should be noted, that it is surface temperature and not air temperature. So to put this into perspective, the highest ground surface temperature (i.e., measured with a hand-held infrared thermometer on the ground) ever recorded is 93.9 degrees Celcius [35].

4 Methods

This section presents the underlying theory of our experiments. In section 5, the experiments are further explained, and their results are shown and discussed.

4.1 Convolutional Neural Network

A Convolutional Neural Network (CNN) consists of two parts: an encoder and a decoder. The encoder is responsible for extracting meaningful features and capturing patterns in the input data. The encoder consists of a combination of convolutional layers and pooling layers (sometimes also other types such as Batch Normalization layers). These layers gradually reduce the image size while progressively extracting what the encoder finds to be the essential information (i.e., features). We can view the encoder's output as a compressed version of the input image. This output is then passed to the decoder. The decoder is responsible for learning the extracted features and mapping them to the correct class labels (for the classification domain) or numerical values (for the regression domain). The decoder is often a fully connected neural network.

Convolutional layers use convolutions, where a convolution involves applying filters to the input image, and the filters can have different sizes. A convolutional layer outputs a feature map (also called an activation map) that then serves as input for later layers. The number of filters in a convolutional layer can also vary. The filters have weights that are updated with backpropagation, like in a standard neural net.

During a convolution, each filter scans the input image using a sliding window. When a filter scans the image, element-wise multiplications are made between the filter weights and the corresponding pixels in the sliding window. The results of the element-wise multiplications are summed to produce a single value in the output feature map, so the width and height of the feature map will be smaller than in the input image. This means that a filter, e.g., of size 3×3 in a deeper layer will correspond to a larger area in the input image than a filter of the same size in an earlier layer. That area is called a receptive field.

[4] The data can be accessed at `https://www.earthdata.nasa.gov/`

By employing multiple filters, each responsible for detecting different features, CNNs can learn a hierarchy of features. In early layers, the network detects simple low-level features like edges and changes in color, and then in later layers, it detects more complex high-level features that define objects or image structures. High-level features could, e.g., be faces, wheels, or clouds. This is possible since deeper layers have a larger receptive field.

CNNs leverage parameter sharing, where it reuses the same filter weights across different parts of the image. By sharing weights, the network can detect the same feature regardless of its position. This is known as translation invariance. For example, if a CNN can recognize a particular object, like a cat, in one part of the image, it can also detect a cat if it is in another part of the image.

In summary, CNNs use convolutional layers to extract meaningful features from the input image through filter operations. These filters scan the image, capturing local patterns and gradually learning complex features.

4.2 Transfer learning

A large amount of data and computation time is needed to train a CNN. The idea in transfer learning is to reuse a pre-trained encoder and tune it for a new domain rather than starting from scratch. The decoder from the pre-trained model is often not used, as the domains and number of needed output neurons rarely are the same. Thus, a new decoder for the new domain is constructed instead. Some CNNs have already been trained on millions of images and are well-known for being good at generalizing. The encoders of these models have reusable properties as they learn general features which can be applied to other domains. As mentioned in section 4.1 above, the early convolutional layers have a small receptive field and detect small-scaled details like edges. Conversely, the later convolutional layers will have a larger receptive field and detect larger scaled structures such as car wheels, goldfish, or rivers. Updating some of the later layers might make sense if the new domain differs from the pre-trained model's domain. During the training of the new model, one often chooses to freeze the encoder, which means that its weights are not updated during training. Freezing layers prevents the model from fitting the encoder to the new domains. This can often be beneficial to prevent the encoder from overfitting to the new domain. If the encoder is unfrozen, it is called fine-tuning and is usually done after the decoder has been trained for some epochs. One can also choose only to unfreeze some of the later layers as their larger receptive field makes the layer more specific for a particular domain.

Furthermore, using a pre-trained encoder can also help reduce the risk of overfitting, as only the decoder needs to be trained, which only constitutes a small fraction of the total number of weights. One thing to note is that even though a model has been shown to generalize well on one task, it does not mean it is good at generalizing to another task. This is discussed in [17], where they mention the no-free-lunch theorem, referring to the fact that because a model performs exceptionally well for the ImageNet task, it is not guaranteed that it will also perform exceptionally well for all other tasks, such as for satellite images.

4.3 Baseline architecture

Our baseline is a regression model that predicts our target values, i.e., parasite rates, in the interval $[0, 1]$. As such, we will use the binary cross-entropy loss function, which works for values in the interval $[0, 1]$. Alternatively, Mean Squared Error could have been used as the loss function. For simplicity, our baseline model will only consider satellite images. In later experiments in section 5.7, the model is tested using different fusion methods to see the impact of the different data types. Figure 5 shows our baseline model's architecture.

Encoder We will use the Transfer Learning technique since we only have 7288 data points. We use the VGG16 model trained on the ImageNet dataset. Other models, such as ResNet and MobileNet, were also considered, but we chose to use VGG16 due to its simplicity. However, the millions of images from the ImageNet dataset that VGG16 has been trained on do not contain satellite images. The ImageNet dataset contains 1000 classes, many of them being different animals. As such, there is a big difference between an image of a frog and a

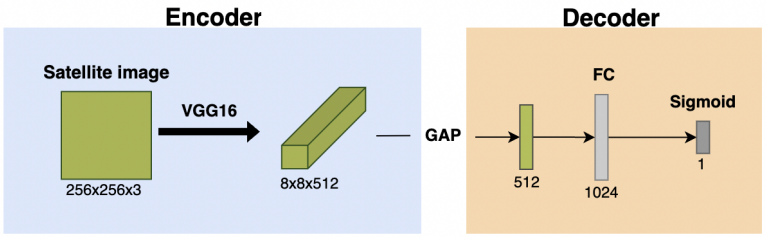


Figure 5: Shows the baseline model's architecture.

satellite image, even though both images can have green pixels. These two domains have little in common, which might result in our model having difficulty utilizing the pre-trained weights from VGG16. A way to combat this would be to unfreeze/fine-tune some of the later layers (i.e., with a large receptive field) of the encoder and thus retrain them for our domain. For all experiments, we will preprocess the satellite images the same way as when the VGG16 encoder was pre-trained on the ImageNet data. This preprocessing ensures that VGG16 gets inputs similar to its training images. Otherwise, VGG16 might not perform as well.

Decoder We do not use the decoder of VGG16 since it is trained to classify the 1000 categories in the ImageNet dataset. Thus, instead, we construct our own decoder. Firstly, since the encoder's output most often consists of volumes, the encoder's output need to be converted to a vector. There are different approaches to doing this. One is to reshape the volume to a 1-dimensional vector. This method is known as Flatten. One disadvantage of flattening is that the flattened vector quickly becomes enormous. When this large vector is given to the fully connected layer, there will be an enormous number of trainable parameters, which results in slower training times and higher memory consumption and makes the model more prone to overfitting. An alternative is Global Average Pooling (GAP), which averages the values for each channel in a feature map to get a single number and appends it to a vector. The result of GAP is a vector with a length equal to the number of channels in the feature map. Thus, the number of trainable parameters will not explode, and thus GAP does not have the disadvantages of Flatten. Due to this, we will in our experiments be using GAP. Then after GAP, we pass the vector to a Dense operation (i.e., a fully connected layer) with 1024 nodes (in section 5.4, we experiment with different fully connected layers). We do not have 1000 classes for our regression task, so our final layer only has a single output neuron. Finally, we will map the fully connected layer to a single numerical value between $[0, 1]$ using the Sigmoid activation function.

4.4 Root Mean Squared Error (RMSE)

We will use Root Mean Squared Error (RMSE) to measure the model's performance for the regression tasks. The performance should be measured by how far the predictions on the test dataset are from the ground truth values. Hence, Means Squared Error (MSE) can be used. MSE finds the average of the squared differences between predicted and ground truth values. To make the result more interpretable, we can take the square root of the result to transform the result back to the same scale as the original values.

Let N be the number of images in our dataset. Now for an image i, let the prediction be denoted $prediction_i$ and the ground truth value be denoted $ground\,truth_i$. Now we can calculate RMSE as shown in equation 2.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (prediction_i - ground\,truth_i)^2}{N}} \tag{2}$$

4.5 Downsampling

Image quality plays a crucial role in training convolutional neural networks (CNNs), as models can use more information from higher-quality images. Researchers, such as Thambawita et al. [36], have explored the impact of image size on CNN performance. They found that using smaller input images, and thus fewer input variables, reduces model overfitting as fewer parameters need to be optimized. However, Thambawita et al. also state that excessively reducing image size results in information loss. Therefore, there is a trade-off between image size and overfitting. Furthermore, Thambawita et al. mention that a trade-off arises when optimizing a convolutional neural network using a Graphics Processing Unit (GPU), as an increase in the input image size results in a higher memory requirement. This increase in memory consumption means the batch size needs to be lowered, which can negatively impact the model's performance. This trade-off must also be taken into account for our model. We have large input images with dimensions $1024 \times 1024 \times 3$, so a significant amount of GPU memory is needed. This leads us to slower training times, as our batch size needs to be adjusted accordingly.

4.6 Dropout

Dropout is a well-known regularization technique to combat overfitting. The idea is to randomly turn off some neurons in the fully connected layers during training. After each batch, new neurons are chosen to be turned off. Hence, they are turned on and off multiple times during an epoch. The neurons are turned off by setting their output to zero during the forward pass. It is a computationally cheap regularization technique compared to, for example, data augmentations. The intuition behind dropout is that we do not fully believe that the encoder or a fully connected layer is entirely correct. As such, we introduce some noise by using dropout. This noise makes sure the model can not have a single neuron that memorizes a pattern since there is a chance that this neuron is turned off. So the model needs multiple neurons that detect the same thing. The probability of turning off a neuron is a hyperparameter commonly set to 0.5. We will be testing different values to see how they impact model performance. However, not for values higher than 0.5 because then the model cannot train the neurons correctly [37]. Furthermore, there are various places the dropout layer can be placed. One is before the fully connected layers (i.e., right after the encoder), and another is after a fully connected layer. The only difference between adding dropout before and after the fully connected layers is where in the pipeline we add noise.

4.7 Fully connected layers

The decoder varies a lot between models. Both in terms of how many fully connected layers are used and the number of neurons in them. AlexNet and VGG16 architectures have two dense layers, each with 4096 neurons, before their output layer with 1000 neurons. On the other hand, ResNet only has an output layer with 1000 neurons. Lastly, MobileNet has a single layer with 1024 neurons before the output layer. Having a complex decoder results in more trainable parameters, which can give the model more complexity. This may result in slower training time and make the model more prone to overfitting. However, having a too simple decoder may result in underfitting as the model cannot capture all the patterns in the data. As such, we need to decide which configuration we want to use. In section 5.4, we experiment with different fully connected layers both in terms of the number of layers and neurons.

4.8 Data augmentations

Data augmentation is another regularization technique. It works by modifying/augmenting the input image. There are many ways to augment the input images, categorized as position augmentations, color augmentations, and other augmentations. Other augmentations include adding noise or using occlusion to cover some parts of the image. Position augmentations include making random crops, rotations, flips, and resizing images. Color augmentations include changing brightness, contrast, saturation, changing the color hue (e.g., changing green to blue), and changing RGB to greyscale.

Our satellite images are captured by different instruments and are captured at different times of the year. Hence, they differ in color, saturation, contrast, and brightness, as shown in figure 6. Thus, by applying data augmentations, the hope is that it makes the model pay less attention to, for instance, color differences. An image with misleading color differences is figure 6i, where the left side is orange while the right is green. This color change is possible since two satellite images have been merged together. The orange part might have been taken during the evening, while the green part might have been taken during noon. However, the model should not consider whether an image was taken during noon or evening. So augmenting the hue could help prevent that. On the other hand, augmenting the colors could have negative effects if the color represents useful information in some images. For example, the specific color of some vegetation could indicate a good breeding habitat for mosquitoes. For instance, according to [21], banana trees have a positive correlation with the parasite rate.

Augmentations can effectively increase the dataset size and thus reduce overfitting, which is mainly what we try to achieve with positional augmentations. Regarding flipping the images, it is possible to do it vertically and horizontally. However, some positional augmentations might not make sense for some images. For instance, vertically flipping an image of a building on its head is unrealistic. Thus, vertical flips might not improve the model's performance to generalize to unseen data, as it will have learned something which is impossible in reality. The same is also true for rotating the images. In contrast, these positional augmentations make more sense for satellite images.

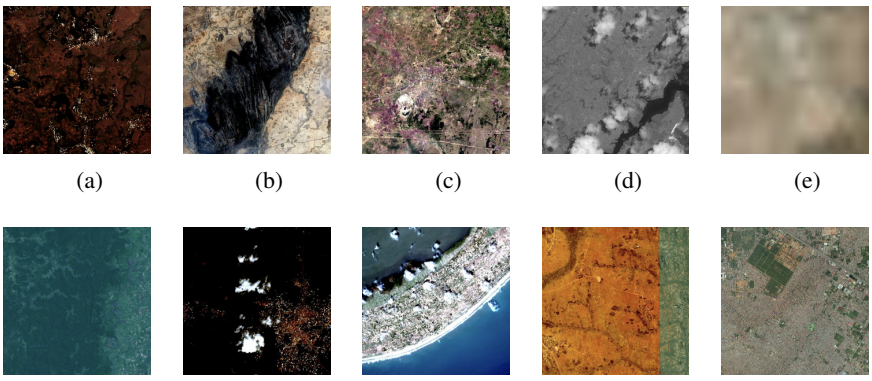


Figure 6: Examples of our raw satellite images with no preprocessing. Here (a) is dark with red colors, (b) has contrasting colors with a dark mountain, (c) has high saturation with pink spots, (d) is in greyscale, (e) is very blurry, (f) has a blue tint, (g) is mostly black, (h) has high brightness, (i) has an orange tint in one side, and finally (j) is a city with standard colors.

4.9 Transfer learning with a model pretrained on remote sensing data

It has been shown to improve remote sensing models if they have been trained from scratch on remote sensing data instead of using transfer learning from models pretrained on the ImageNet dataset [15, 38]. Wang et al. [15] has trained a ResNet50 architecture from scratch on a dataset called MillionAID, where the spatial resolution of the images varies between 0.5 meters and 153 meters. Our 256×256 images' spatial resolution is roughly 20 meters/pixel. Since this is between 0.5 meters and 153 meters, there is potential for Wang et al.'s model to generalize to our domain. Wang et al. trained models for different tasks. One of these tasks was scene recognition, which could, e.g., be to classify a satellite image as a forest, basketball court, parking lot, etc. Specifically, their model could predict 51 classes, including agriculture-, commercial-, industrial-, public service-, residential-, transportation-, and unutilized land, and finally, water areas. On this scene recognition task, their new ResNet50 model got an accuracy of around 97%, whereas the standard ResNet50 pretrained on ImageNet got around 95%.

4.10 Multi-modal data

Weiss et al. [8] show that providing their model with multiple types of data, also known as a multi-modal dataset, improves its performance. They further show that many different authors also use more than one type of data in their model.

4.10.1 Data fusion: early, feature, and late fusion

One of the challenges when having multiple types of input data is how the data should be processed. The three main methods for handling this are Early, Feature and Late Fusion [17]. They indicate where in the pipeline we fuse the inputs. In Early Fusion, the inputs are fused in the beginning before they are given as input to the model. This can result in having an input tensor with multiple channels, which is a disadvantage since most transfer learning models then can not be used. Most transfer learning models are designed to accept images with 3 channels (Red, Green, and Blue). The advantage is that the inputs are fused early in the process, so only one model has to be trained instead of the multiple models that are needed in Late Fusion. In Late Fusion, all the inputs are run independently, each through their own encoder and decoder, and fused together in the end at a final decision layer. This will result in higher computational costs than Early Fusion but opens up the possibility of using transfer learning. Feature fusion is in between and takes feature vectors from different encoders (e.g., convolutional streams) and concatenates the vectors, and then uses a common decoder. Below is a more detailed description of the fusion methods and how they can be used in our domain.

4.10.2 Early fusion

Others have used early fusion (also called data level fusion) for different tasks [39], for example, for action recognition where they use RGB images, depth images, and skeletal data to classify actions (e.g., jumping). Early fusion can be done in several ways. One way is to combine the data into a multi-dimensional tensor, and another is to put the data next to each other, creating a larger RGB image.

Early fusion with a multi-dimensional tensor The first way of fusing the data is shown in figure 7, where we put the data types on top of each other. We upsample/stretch (with nearest neighbour interpolation) the land cover, precipitation, and temperature data to have the same width and height as the satellite images. We do it this way to ensure that the geographical locations are overlaid correctly on top of each other.

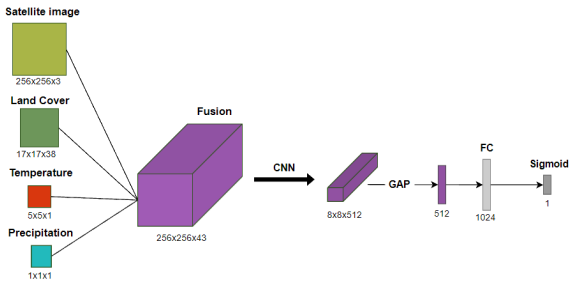


Figure 7: Shows the model architecture when using all the remote sensing data types with early fusion with a multi-dimensional tensor.

Early fusion with RGB images The other way we make early fusion is by putting the images next to each other, where the land cover data is now a greyscale image with pixel values representing the different land cover classes (i.e., the same format as when the land cover data was downloaded [32]). The land cover, precipitation, and temperature are all greyscale and are then made into 3 channels by copying the single channel into the two new channels. They are also scaled up to be size 256×256 with nearest neighbour interpolation. Figure 8 shows an example of an input image. An advantage of this method is that we can use transfer learning (e.g., VGG16) since these new input images have 3 channels.

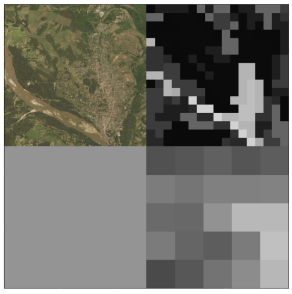


Figure 8: An example of an input image of early fusion where the images are next to each other. The top left is the satellite image, top right is land cover, bottom left is precipitation, and bottom right is temperature.

4.10.3 Feature fusion

Feature fusion (also called representation or intermediate fusion) has, like early fusion, been used for action recognition [39]. In our feature fusion model, we have a VGG16 encoder for satellite images. The output of that encoder is made into a vector with GAP, and the other data types are just made into vectors with GAP. Then all the vectors are fused by concatenating them before the fully connected part of the model, as shown in figure 9. A more involved feature selection could be made on the feature vectors (as in [39]), but we just concatenate them for simplicity.

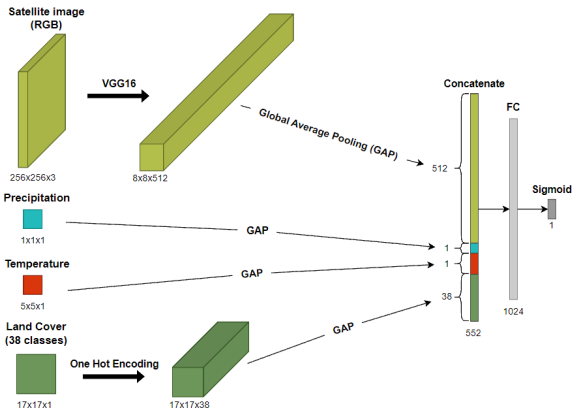


Figure 9: Shows the model architecture when using all the remote sensing data types with feature fusion.

4.10.4 Late fusion

Late fusion (also called decision level fusion [17]) is among others used by [40] for action recognition. They use a two-stream CNN, i.e., an ensemble model where the first CNN takes an image and the other a video. They use late fusion by averaging the Softmax scores of their two CNNs. Inspired by this, we make a four-stream ensemble model with a similar late fusion strategy on the satellite, land cover, precipitation, and temperature data. The satellite data is still going through VGG16, then Global Average Pooling, and then a 1024-node fully connected layer with ReLU followed by a 1-node Softmax layer. The other data types also each go through an identical architecture except for the transfer learning, and then we take the average value of the four streams and use this as our predicted regression value. This architecture can be seen in figure 10.

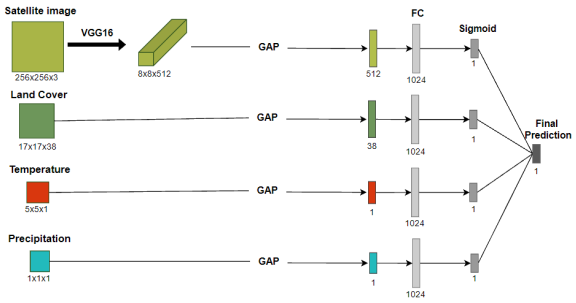


Figure 10: Shows the model architecture when using all the remote sensing data types with late fusion.

4.11 Land cover model with more data

Adding more data to a model can often be beneficial. One reason is that a larger dataset will likely contain more diverse examples, including more edge cases. Another reason is that more data makes it harder for the model to memorize specific data points, and thus it reduces overfitting. However, it is not guaranteed that adding more data will help, e.g., if the quality of the new data is lower or less relevant in some way. We have land cover data from 1992 onward, and malaria data dating back to 1984. So a model using only land cover data will have more data points than our dataset with data only going back to 2010. However, it is possible that the further back in time we go the less relevant the data becomes. There could be several reasons for this. It could be because the instruments capturing the data were less accurate in the past, or some trends in the older data could be less relevant for predicting malaria rates in the present day.

4.12 Combining land cover classes

Our land cover data has 38 classes, but some of the classes are similar to each other. By merging classes, it can remove redundancy which can reduce the training time, simplify the model, and reduce the chance of overfitting. Other papers often use less than 20 land cover classes by merging some of the classes. Krefis et al. [21] use 9 land cover classes, Mandike et al.[9] started with 20 classes and merged them into 13 classes, and Weiss et al. [8] use 17 land cover classes. We have merged our 38 land cover classes into 13 new classes as shown in table 3.

4.13 Regressions as classification

As described in section 3.3, the regression problem is often harder than classification since the model has to estimate the regression function compared to determining a set amount of decision boundaries. Hence, by converting the problem into a classification problem, the problem can, according to [28], improve the model's performance in terms of training time and accuracy. When transforming the problem into classification, we also have to modify the model. First, we change the model to use SoftMax instead of Sigmoid. Second, we must change the loss function to categorical cross-entropy.

4.14 Multi-task learning

In multi-task learning, the model is constructed in such a way that it is training on two or more different tasks simultaneously. The tasks could be regression and classification. According to [41], multi-task learning is under-explored for mosquito control despite its applications in related topics. Multi-task learning can be helpful as it restricts the model from getting superior for only one of the tasks. The model will have to perform well for both of them. It is possible to adjust how much each of the tasks should weigh. Weighting them is possible as the loss function consists of two terms, one for each task. We will use the categorical cross-entropy loss function for classification and the binary cross-entropy loss function for regression. It should be noted that the encoder is identical for both tasks,

New class	Old classes
No data	• No data
Cropland	• Cropland, rainfed • Cropland, rainfed, herbaceous cover • Cropland, rainfed, tree or shrub cover • Cropland, irrigated or post-flooding
Mixed crops and natural vegetation	• Mosaic cropland/natural vegetation • Mosaic natural vegetation/cropland
Tree cover, broadleaved	• Tree cover, broadleaved, evergreen, closed to open • Tree cover, broadleaved, deciduous, closed to open • Tree cover, broadleaved, deciduous, closed • Tree cover, broadleaved, deciduous, open
Tree cover, needleleaved	• Tree cover, needleleaved, evergreen, closed to open • Tree cover, needleleaved, evergreen, closed • Tree cover, needleleaved, evergreen, open • Tree cover, needleleaved, deciduous, closed to open • Tree cover, needleleaved, deciduous, closed • Tree cover, needleleaved, deciduous, open
Mixed trees and mixed trees and shrubs	• Tree cover, mixed leaf type • Mosaic tree and shrub/herbaceous cover • Mosaic herbaceous cover/tree and shrub
Shrubs, grass, moss and lichens	• Shrubland • Shrubland, evergreen • Shrubland, deciduous • Grassland • Lichens and mosses
Sparse vegetation	• Sparse vegetation • Sparse tree • Sparse shrub • Sparse herbaceous cover
Flooded vegetation	• Tree cover, flooded, fresh or brakish water • Tree cover, flooded, saline water • Shrub or herbaceous cover, flooded, fresh/saline/brakish water
Urban	• Urban areas
Bare area	• Bare areas • Consolidated bare areas • Unconsolidated bare areas
Water	• Water bodies
Permanent snow/ice	• Permanent snow and ice

Table 3: Shows the 38 original landcover classes merged into 13 new classes.

and the features the model gives as input to the two tasks also is identical. Hence, the fully connected layers should be able to use the features for two different tasks. As such, the only difference between the baseline model and the multi-task learning model is that for multi-task learning, the model has two paths after the fully connected layer: one for the regression task using Sigmoid to get a numerical value and another for the classification task using SoftMax.

Being able to weigh the loss function individually means we can bias the model to consider one of the tasks more. Therefore, we will be testing the following three cases. (1) Even weights, (2) High regression and low classification, and (3) Low regression and high classification. For all three cases, we can consider four possible outcomes. Namely, the model performs (A) better for both tasks, (B) worse in both tasks, (C) better at only one of the tasks, or (D) neither performs better or worse.

If the model's features are useful for both tasks, event (A) might occur. Otherwise, if the features are not useful for both tasks, we might see events (B) or (C) occurring. Lastly, two things can make the event (D) occur. First, the fully connected layers find the same

tendencies as in the baseline. Second, the fully connected layer performs better at some features but worse at others. Meaning they cancel out with each other.

4.15 Ordinal classification

Ordinal classification, also known as ordinal regression, refers to the classification task where the classes contain an ordering, and the model takes this into account. Our classes have an ordering since the class containing the points with a parasite rate of 0 is closest to the class where they are between 0 and 0.2, the second closest to the class with parasite rates between 0.2 and 0.5, and the furthest away from the last class. Another example of ordinal classes is the age ranges of people, e.g., the classes could be child, teenager, young adult, adult, and elderly. A paper [42] introduces a framework called CORAL for ordinal classification and uses it to predict a person's age based on a picture of the person's face. An idea that ordinal classifiers often use is to make K-1 binary classifiers, where K is the number of classes, and use these instead of a multi-class classifier. CORAL does this by doing a variation of multi-task learning, where there is a common encoder (they use ResNet-34 in their age classifier), which then splits into K-1 sub-tasks which are converted to a single label. Furthermore, CORAL enforces rank consistency, which means that if the model is most confident that a data point is in class 1 (e.g., age between 20-30), then the model should be second-most confident that the data point belongs in class 2 (e.g., age between 30-40) and so on. As CORAL improves the model's accuracy for their domain, it is interesting to see if this also affects our domain.

4.16 Saliency maps and Gradient-weighted Class Activation Mapping (Grad-CAM)

Saliency refers to the quality of being noticeable and important. Thus, saliency maps tell which pixels in an image are most important. Saliency maps can be made with respect to the human visual system. Bright colors catch the attention of the human eye, so a saliency map should have higher values (i.e., higher importance) in those areas of the image [43]. Saliency maps can also be made for computer vision models where the saliency map shows which input pixels influence the prediction most. Saliency maps have several use cases, such as object localization[44], model interpretability, and network debugging. Saliency maps can both be used for classification and regression. For regression, the saliency maps can show whether pixels in the input image are increasing or decreasing the predicted value. For classification, saliency maps show the regions that contribute the most to activating a given output neuron.

Simonyan et al. [44] make a saliency map for a CNN. With a single backpropagation pass through the network, they calculate the gradients for an image. CNNs are complex and generally contain non-linear operations, e.g., the ReLU activation function, so it is hard to describe the exact function for predicting the class of an input image. So Simonyan et al. approximate this function (they call it the class score function) by using Taylor expansions. They can make saliency maps visualising neurons in a fully connected layer of their network.

Grad-CAM stands for Gradient-weighted Class Activation Mapping. As with Simonyan et al.'s saliency map, Grad-CAM calculates the gradients for an input image with a backpropagation pass through the network. The gradients are then used to weigh the output values of a specific convolutional layer. This results in a weighted feature map with many channels that are added together element-wise to get a 2D map. A ReLU operation is used on the map to get the features that increase the chance of predicting the target class of the input image. Finally, Grad-CAM's saliency map is resized to be the size of the input image. Grad-CAM++ builds upon Grad-CAM and makes more accurate saliency maps. Grad-CAM++ uses a more sophisticated backpropagation to achieve its results [45].

4.17 Model inspection using top-k analysis

To inspect which images the network performs the best and worst on, we can make a top-k analysis and find the k best and worst images. This can be done using a trained model to predict on some images and store the errors for each image. Then we can find the top-k best and worst images by sorting the errors. This data can be used for several things. (1) It can

visually showcase if there are any similarities in the best or worst images. Thus, it can give an idea of if there are any tendencies in the images. (2) It can be used to compare the data from the top-k best and worst to the overall distribution of the data and give an idea of what data the model performs better on. This information can furthermore be used to investigate why the model performs better on some data than others. Specifically, we will be looking into the distributions of years and countries as these might have an impact. Looking into the years is interesting as newer images might be clearer and contain fewer merged images. Countries are interesting as MAP uses other researchers' surveys resulting in an unbalanced distribution.

There are several ways to calculate the error. Let Y be the ground truth and let $\hat{Y}$ be the prediction.

Simple approach The simple way to calculate error is to use the absolute difference between the ground truth value and the prediction value:

$$error = |Y - \hat{Y}|$$

However, one issue with this way is that the values near the ends (i.e., 0 and 1) can have a much larger error than those in-between. For example, a value such as 0.5 can maximally have an error of 0.5 as $|0.5 - 1| = 0.5$ and $|0.5 - 0| = 0.5$. Whereas a value such as 0 can maximally have an error of 1.

Normalized approach To solve this conflict, we can normalize the values according to the maximal error they could have had.

$$error = \frac{|Y - \hat{Y}|}{\max(|Y - 0|, |Y - 1|)}$$

Thus, by normalizing the values, the values near the boundaries (i.e., 0 and 1) will not have a greater chance of having the largest error.

The normalized approach is used to implement the top-k in section 5.13, as it does not have the bias.

4.18 K-Nearest Neighbour feature vector analysis

The fully connected layers in our network can be inspected using a K-Nearest Neighbour approach. It works by first modifying the model to output the feature vectors from the fully connected layer by removing the Sigmoid or SoftMax layer (if having multiple fully connected layers, more layers can be removed if one wants to inspect deeper layers). Then we predict on all our images and store the feature vectors for each image. We can now take a test image and use the model to get the feature vector. Then we use K-Nearest Neighbours to find the nearest neighbours for the given test image among the stored feature vectors. Lastly, we can plot the test image and the images that the nearest neighbours model finds and visually inspect them for similarities and dissimilarities.

Inspecting these images can give a good idea of if the models look for the correct thing. For example, assume we give the model a picture of a winding river. Then we hope the model has learned that winding rivers can have many shapes and not simply give images with identical shaped windings, size, and placement of the river. More formally, we do not want the model to learn low-level features such as color and pixel locations. Instead, we want the model to learn the high-level feature, such as the conceptual understanding of a river.

However, this approach is not perfect, as images often vary a lot, even though they capture similar things. This results getting different feature vectors. Thus, the testing images can influence how well the results of the K-Nearest Neighbour are. As such, it should be advised to make multiple tests with different images to ensure that the model performs as expected.

5 Experiments

In this section, we will present and discuss the results of the different experiments. We mainly use Python with the high-level API of TensorFlow called Keras for the implementation [5]. In the first experiment, we will downsample the satellite images. The result will be used to define the baseline, which will be used for the remaining experiments. Although, for succeeding experiments, we will not be using the best results from previous experiments; instead, we will always go back to the baseline.

5.1 Downsampling

In this experiment, we use our baseline architecture (see figure 5) with the satellite images with the following resolutions: $1024, 512, 256, 128, 64$.

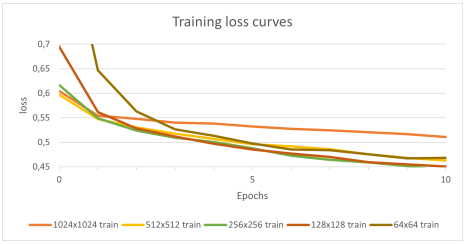
We expect that most of our essential information is not in the fine details but instead consists of larger structures and contrasts in the images (e.g., cities and water bodies). However, the details might make the prominent features stand out. Thus, when downsampling the images, these details might be lost.

Sizes	RMSE	Time (seconds)
1024	0.227	420
512	0.248	215
256	0.259	56
128	0.287	24
64	0.301	17

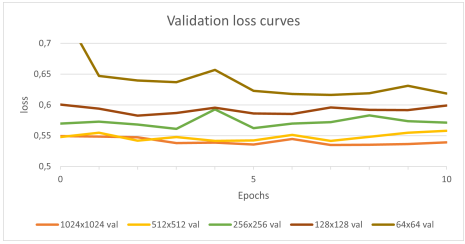
Table 4: Shows the RMSE values and the average training time per epoch for the downsample experiments on the test dataset.

Thus, we expect a decrease in root mean squared error (RMSE) as downsampling the images diminishes the fine details, which might make the more prominent features stand out. Meanwhile, we also expect the training time to become faster as size decreases, as the batch size can be increased.

The training and validation loss curves can be seen in figure 11a, 11b while the RMSE can be seen in figure 4.



(a) Shows the training loss for the different models as a function of the epochs.



(b) Shows the validation loss for the different models as a function of the epochs.

Figure 11: Shows the training and validation loss curves for the baseline architecture using different satellite image resolutions.

The training losses (figure 11a) show that all except the 1024×1024 model converge to the same training loss. This is unexpected, as we expected that it would be easiest for the model to memorize the 64×64 images since these are smaller and less complex.

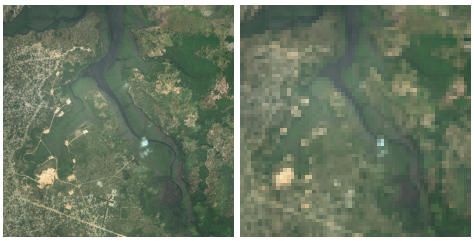
Figure 11b shows that validation loss increases as the image resolution decreases. This might indicate that some important information has been lost, making the model less confident about its predictions. The same can also be seen when looking at the RMSE (table 4). Thus, lowering the image size decreases the performance but greatly reduces training time. So as expected, the model gets more uncertain with downsampled images and performs worse.

Figure 12 shows that it visually is easier to find cities and water bodies on images of size 512×512 compared to 64×64.

Regarding speedup, the average training times of an epoch for the different resolutions are shown in table 4. We can see a significant difference from the 1024×1024 to the

[5] Our code can be found at GitHub here: `https://github.com/RasmusKlinkJoerg/geospatialMalariaPrediction`.

others. This makes sense as we reduce the number of pixels with a factor 4 between each downsampling. Thus, for 1024 to 512, the number of pixels is reduced by a factor 4, while for 1024 to 64, it is reduced by a factor 256. A central part of the speedup comes as we can increase the batch size when using smaller images. Furthermore, it is faster to load smaller images into memory and preprocess them.



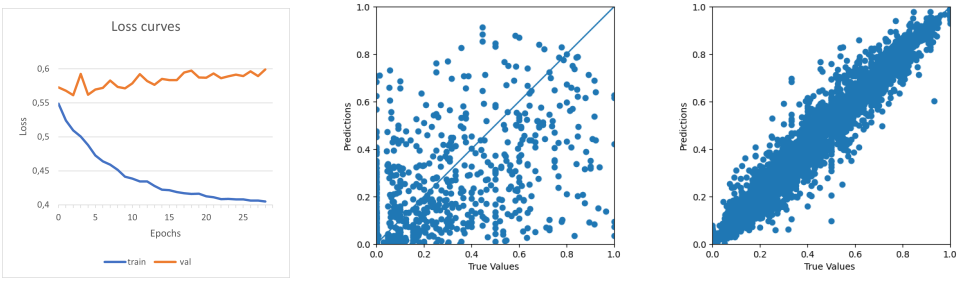
(a) 512×512 (b) 64×64

Figure 12: Shows an example of the quality of the images when changing the resolution.

Based on the experiments above, we have decided to use the 256×256 images for the remaining experiments. Vincenzi et al. [16] also experimented with different resolutions of satellite images in a remote sensing task and found that using images with 224×224 resolution was a good tradeoff between spatial resolution and overall memory footprint. In Vincenzi et al.'s paper, a 224×224 resolution resulted in a spatial resolution of 20 meters/pixel (i.e., each pixel corresponds to an area of 20×20 meters). Our satellite images with size 1024×1024 pixels have a spatial resolution of roughly 4.9 meters/pixel, as mentioned in section 3.4, which means that our 256×256 images are roughly 20 meters/pixel. So Vincenzi et al. reached the same conclusion as us. The training and validation scores for the different resolutions respectively differ with less than 0.1 binary cross-entropy loss, and the same is true for the RMSE scores. This indicates that changing the resolution does not make the biggest difference quality-wise. By picking the 256×256 resolution, the validation loss increases by around 6% compared to the 1024×1024 resolution, and the RMSE increases by 14%. However, we will get a 6 times speedup, making our experiments take roughly 1 hour to perform on our setup instead of 6 hours.

5.2 Baseline

The above-described model with 256×256 resolution will be our baseline model, so we will compare the following experiments' performance to this. We expect that the model will be able to fit our data but overfit, as we do not have anything apart from batch shuffling to combat overfitting. We also expect that the model will have difficulty utilizing the pretrained weights from VGG16, as our domain differs significantly from the natural images in the ImageNet dataset. The results of the experiment can be seen in figure 13.



(a) Loss curves (b) Test dataset (c) Train dataset

Figure 13: Shows the results from the baseline experiment. In figure 13a, the training loss and validation loss curves are plotted as a function of epochs. In figures 13b and 13c, the diagonal line represents the ideal prediction where each data point hits its true value. In figure 13b, the predictions on the test dataset are shown. Figure 13c shows the predictions on the training dataset.

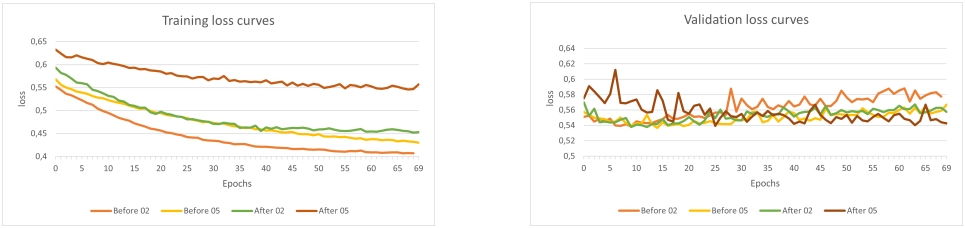
In the loss curves (figure 13a), we can see that the training loss keeps decreasing but the validation plateaus. This is usually a sign of overfitting since the curves ideally should follow each other. This is also what we can see on the prediction for the training dataset in figure 13c where the points fit the line much better than on the testing dataset 13b. From this, we can see that the model can overfit the data and has difficulty generalizing to unseen data.

5.3 Dropout

As described in section 4.6, we will use different dropout values on different locations to see dropout's impact. We have decided to use the dropout values 0.2 and 0.5. These will be used both before the fully connected layer and after the fully connected layer in the baseline model. The dropout value 0.5 is used as a rule of thumb, and 0.2 is arbitrarily chosen to have a value smaller than the rule of thumb. As mentioned in section 4.6, dropout values greater than 0.5 should not be used. Additionally, if the dropout value is too low, the effect of dropout disappears. We expect positioning the dropout before and after the fully connected layer will positively impact the model. Additionally, we expect the 0.5 dropout value to be more restrictive than the 0.2 value and regularize the model more.

The results can be seen in figure 14. The training loss curves (figure 14a) show that both dropout values after the fully connected layer result in the largest training losses. So it seems that placing the dropout after the fully connected layer is better at preventing overfitting. Additionally, we can observe that the experiments with dropout before the fully connected layer have more similar loss curves. This indicates that a dropout layer before the fully connected layer does not affect the model much. Almost the same can be seen for the validation loss curves (figure 14b). At 70 epochs, the difference in validation loss between the models with the highest and lowest loss is a mere 0.04. This indicates that the choices of dropout values and positioning do not impact the model much. However, the experiment with dropout after the fully connected layer and dropout value 0.5 is the only experiment where the training loss and validation loss closely follow each other. This is what we expect to see, as it indicates that the model does not overfit. Unexpectedly, adding dropout did not improve the validation loss much compared to the baseline. However, utilizing dropout restricted the model such that the training and validation losses followed each other much closer.

To summarize, having a too-small dropout value or the dropout layer placed directly after the encoder does not seem to impact the model's performance much. However, using the dropout value of 0.5 and positioning the dropout layer after the fully connected layer has the strongest effect on preventing the model from overfitting.



(a) Shows the training loss curves.

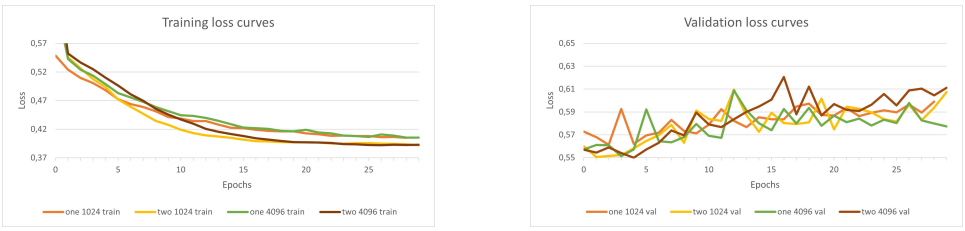
(b) Shows validation loss curves.

Figure 14: The training and validation loss curves for the dropout experiment. Here "Before 02" means that dropout with a value of 0.2 was introduced before the fully connected layer.

5.4 Fully connected layers

In this experiment, we use different decoders where we vary the number of fully connected layers and their width. The baseline model uses a single fully connected layer with 1024 nodes. Then we try with two layers with 1024 nodes and one and two layers with 4096 nodes. We expected that the decoder with one layer with 1024 nodes would be sufficient since our output is a single node instead of 1000 nodes, as in the ImageNet dataset. Therefore, we expect the other larger decoders would cause more overfitting. Each model is trained for 30 epochs. The results are shown in figure 15.

The training loss curves follow the same trend, but the models with two layers plateau at around 0.39, whereas the models with one plateau at about 0.40. The validation loss curves are similar to each other. They start around 0.56 and are overfitting, and finish at approximately 0.60. So the more complex decoders overfit slightly more, but overall there was no significant difference.

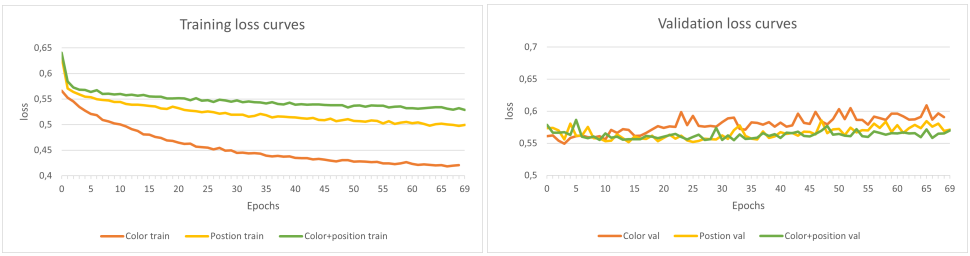


(a) (b)

Figure 15: Training (a) and validation (b) loss curves for models with different fully connected layers. Here "one 1024" means that a single fully connected layer with 1024 nodes was used.

5.5 Data augmentations

Our hypothesis with adding data augmentations is that it will help our model to better generalize on unseen data. We will perform three experiments. (1) We will add color augmentations where we change the hue, saturation, contrast, and brightness. Doing this forces the model to find more general features, meaning we are limiting its ability to learn from the color of the images. (2) We will add position augmentations, including flipping both vertically and horizontally, rotation, and zoom. (3) We will combine both the color and positional augmentations.



(a) Training loss curves (b) Validation loss curves

Figure 16: Figure 16a shows the training loss curves while figure 16b shows the validation loss curves.

The results can be seen in figure 16. It can be observed that the data augmentations result in very similar validation loss values. Furthermore, the validation loss is almost constant and does not improve. This might indicate that we already have much noise in the dataset, indicating that the baseline model already does not consider color, brightness, etc. Looking at the training losses (figure 16a), we can see that the positional and combined data augmentation slowed the overfitting, as expected. It also makes sense that combining both position and color will become more diverse and thus reduce overfitting even more.

5.6 Transfer learning with a model pretrained on remote sensing data

In this experiment, we take Wang et al. [15] ResNet50 model pretrained on remote sensing data and use it in our model instead of the VGG16 model we originally used for transfer learning. We expect that this change will improve our model. However, it might be hard to compare them since we also change the architecture from VGG16 to ResNet50. Thus we make a version of our baseline that uses the standard ResNet50 pretrained on ImageNet instead of VGG16.

Our baseline with ResNet50 after training for 70 epochs got an RMSE of 0.2717, and the model pretrained on the remote sensing task got an RMSE of 0.2705. So there is no significant difference. The outcome could be explained by the fact that Wang et al.'s [15] models were trained for another task, i.e., scene recognition. Furthermore, their model only resulted in an increase in accuracy from 95% to 97%.

5.7 Multi-modal data

5.7.1 Data impact

In this experiment, we wanted to see if it was possible to improve the model by adding additional data types. We run a separate model for each data type to evaluate the impact of the different types of data. Then we run models with the following combinations of data: satellite and land cover data, satellite and precipitation, satellite and temperature, and finally, all four together. As usual, the satellite images are preprocessed with VGG16's preprocessing function. The precipitation and temperature data are normalized using the overall minimum and maximum values in their respective datasets. The following formula is used, $x' = (x - X_{min})/(X_{max} - X_{min})$, where x is the original data point, x' the new data point, and X_{min} and X_{max} are the overall minimum and maximum. The land cover data is transformed to one-hot encoded tensors, so a land cover image gets the shape $(17 \times 17 \times 38)$ where each channel represents a land cover class.

Each data type separately We already have the baseline model using the satellite data separately. For the other data types, we use an architecture similar to figure 9 and 10, just where we only use one data type. So, e.g., the model using only temperature data consists of a GAP operation followed by a fully connected layer with 1024 nodes and then an output layer. The results are that land cover got a validation loss of 0.5447 after 70 epochs and an RMSE of 0.2530, so slightly better than the baseline. Precipitation and temperature both got a validation loss of about 0.59. It makes sense that the performance with land cover is close to satellite since it is a high-level representation of satellite images. It also makes sense that precipitation and temperature perform worse since it is hard to see a clear correlation between them and the malaria parasite rate, as seen in figure 17. In the temperature heatmap (figure 17a), the maximum value in the 5×5 image is used, and not the average since some images had missing values, which were set to -273.15 degrees Celsius, and those would have skewed the heatmap. It is worth noting that these models are significantly faster to train than the baseline. For example, the model using only land cover takes about 3 seconds per epoch compared to 56 seconds for the baseline.

(a) [Removed, Confidential], A heatmap showing the maximum temperature in each 5×5 image plotted against parasite rate.

(b) [Removed, Confidential],A heatmap showing precipitation plotted against parasite rate.

Figure 17: Heatmaps showing precipitation and temperature respectively plotted against parasite rate.

Combining data types with feature fusion This experiment evaluates the impact of combining the different data types, specifically models combining satellite and land cover data, satellite and precipitation, satellite and temperature, and all four together. We used feature fusion to combine the data with the architecture for all four data types shown in figure 9. We used the same architecture for the other combinations, just where only the respective data types are included. We expected that adding the new data types would improve our model since the added data types often are used for predicting malaria parasite rates. The results were that it did not significantly change the model performance in any of the experiments. The loss curves were practically identical to those in the baseline. We suspect this is because the fusion strategy we are using is weighing the satellite data higher since it takes up most of the concatenated feature vector. So this brings us to try different types of data fusion.

5.7.2 Early fusion

In this experiment, we perform the two ways of early fusion described in section 4.10.1.

Early fusion with a multi-dimensional tensor First we use the architecture shown in figure 7. The CNN encoder used in this model is VGG16 but with random initial weights since we can not use transfer learning because our input has 43 channels. We expected this model to perform worse than the baseline since we cannot use transfer learning. This

expectation was accurate, and the model was not better than the baseline. Its training and validation loss plateaued at 0.6 after the first epoch. So, it seems that we do not have enough data to train a VGG16 model from scratch. Additionally, the VGG16 architecture may not work well for large tensors. Moreover, the training time for the model was long. Since each input image consisted of a $256 \times 256 \times 43$ tensor, the training time was around 350 seconds per epoch compared to 56 seconds for the baseline.

Early fusion with RGB images The model architecture used in this experiment is the same as in the baseline model described in section 4.3, just with fused RGB images as shown in figure 8.

Before we fuse the data types into a single RGB image, we preprocess the precipitation data by scaling it to the range [0, 255] using the minimum and maximum across all precipitation values in the dataset. Then we cast the data to integers, so it is similar to the ImageNet data that VGG16 was trained on, even though we lose some information when going from reals to integers. We preprocess the temperature data in the same way, except that we ignore the actual minimum value. The minimum value is a placeholder value for missing data (-273.15 degrees Celcius), so instead, we use the second smallest observed value (13.25268 degrees Celcius [6]). By not using the minimum value, we are not skewing the normalized data, and it gets a higher variance which is preferable. Then we preprocess the concatenated images with the preprocessing function of VGG16. We expected this model to be better than the baseline and the feature fusion model. Here the data types are weighted equally by having the same number of pixels in the input image, as opposed to in the feature fusion model where most of the concatenated vector was from satellite data. The result of this model was similar to the baseline, with a validation loss of around 0.56. This indicates that fusing these data types in this way does not help the model.

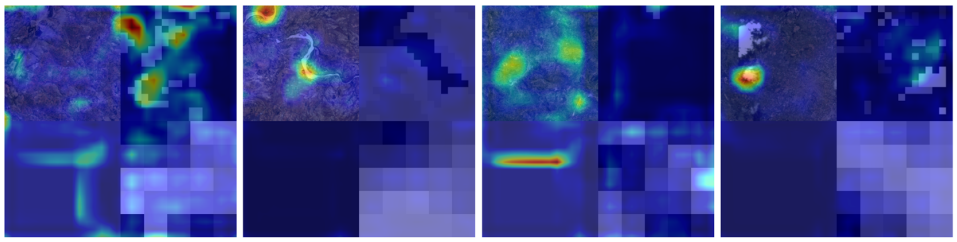


Figure 18: Grad-CAM++ run on fused images. In each fused image, the top left is the satellite image, top right is land cover, bottom left is precipitation, and bottom right is temperature.

To understand which data type the model uses most, we run Grad-CAM++ on some examples, as seen in figure 18. It varies which type of data has the most significant influence according to Grad-CAM++, and often multiple data types have a strong influence. It is hard to tell if this is the behaviour we want to see since it could vary from picture to picture what data is most important.

5.7.3 Late fusion

Here we experiment with late fusion. The used architecture can be seen in figure 10. The final prediction is made by averaging the outputs of the Sigmoid layers. We expected this model to be better than the baseline since it uses new relevant data types, and each data type has the same influence on the output. After training this model for 70 epochs, the validation loss, after steadily decreasing, had plateaued at 0.55, and the training loss was at 0.54 and slowly decreasing. So this experiment was not significantly better than the baseline. This could be because the data types still need more information to predict the target value consistently. Furthermore, an improvement to this model could be to swap out the averaging operation. When the loss is propagated back through the average operation, it is split evenly through the input nodes, even though their activations in the forward pass were not the same.

[6] Note that the lowest temperature here is larger than the -2.2 mentioned in section 3.7 because that was across all of Africa whereas this is only for our survey locations.

5.8 Land cover model with more data

In this experiment, we make a model identical to the model that only uses land cover data from section 5.7.1 but now uses data from 1992 instead of 2010 and onward. We still prune out surveys with examined participants below 10, resulting in a dataset of 17611 data points. The distribution of parasite rates is the following, 30% of surveys have a parasite rate of 0, 30% have between 0 and 0.2, 22% have between 0.2 and 0.5, and 18% have above 0.5. We expect the new model containing more data to perform better, as more data usually is better, as described in section 4.11. The earlier land cover model that only used land cover data from 2010 and onward had, after 70 epochs, a binary cross-entropy loss of 0.5447 and an RMSE of 0.2530, while the new model has a binary cross-entropy loss of 0.49 and an RMSE of 0.2197. So the new model improved by decreasing loss by 10%, and the RMSE decreased by 13%. Meaning in this case, it turned out that including the older data was still relevant for the model and improved its performance, as expected.

5.9 Combining land cover classes

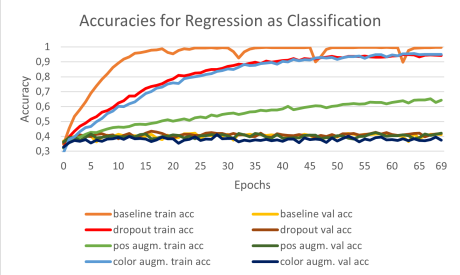
In this experiment, we use land cover data from 2010 and onward, where the land cover classes have been combined into 13 classes instead of 38 (the combinations are described in section 4.12). The same model architecture from section 5.7.1 that only uses land cover data is used. As described in section 4.12, merging classes often can have many advantages and is used by many other papers. As such, we expect that it also improves the performance of our data.

The results gave a validation loss of 0.551 and an RMSE of 0.2378. Thus, it resulted in a slightly worse validation loss but a slightly better RMSE than using the 38 land cover classes in section 5.7.1. We expected to see a larger difference and that merging would have performed better for both the validation loss and RMSE. However, as the change is small, it might be due to random aspects such as weight initialization that we see a change and that the merge itself did not do much.

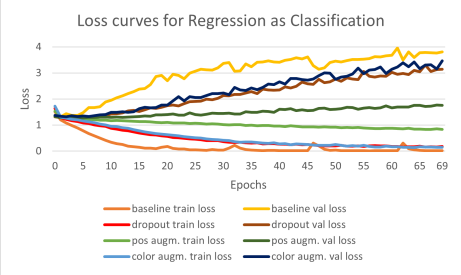
5.10 Regression as classification

In this experiment, we will test the impact of the task formulation. As described in section 4.13, the regression task might be harder than the classification task. As such, we will convert our model into solving a classification problem as described in 4.13.

We expect the classification problem to converge quicker than the regression, as finding decision boundaries is easier than estimating the regression function. We further expect that due to the way, we converted the data into a classification task, it will struggle more near the boundaries of the class divisions as they were not natural boundaries in the distribution of the target values. We will perform the following experiments for the classification task: the baseline, dropout, and data augmentation.



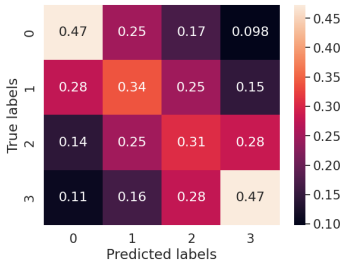
(a) Training and validation accuracies for the different experiments.



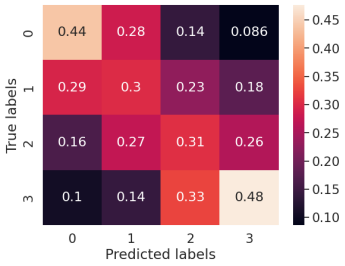
(b) Training and validation loss curves for the different experiments.

Figure 19: Shows the accuracies (figure 19a) and loss curves (figure 19b) from the baseline, dropout, and data augmentation experiments performed for regression as classification.

The results of the experiment can be seen in figure 19. The results show that in almost all of the experiments, the validation accuracies consistently converge towards 0.4. For the test set, the baseline model got an accuracy of 0.378. As we have 4 perfectly balanced classes, random guessing would give an accuracy of 0.25. Thus, the model performs better than random guessing, meaning it learns some tendencies. To inspect which classes were easier and harder, a confusion matrix was constructed. As the confusion matrices almost showed the same results for all the experiments, we have only shown the best (i.e., the baseline) and worst (i.e., the color data augmentation). They can be found in figure 20.



(a) Shows the confusion matrix for the baseline regression as classification experiment.



(b) Shows the confusion matrix for the color data augmentation experiment.

Figure 20: Confusion matrices for baseline regression as classification and color data augmentations experiments.

The confusion matrices in figure 20 show that it is easiest for the model to classify the classes in the ends (i.e., classes 0 and 3). This makes sense as classes 0 and 3 only need a single decision boundary between them and the other classes, whereas the middle classes (i.e., 1 and 2) each need two decision boundaries. The confusion matrices further show that the correct class is picked the most, except in the color data augmentation for class 2. This is as we have hoped, as it means that the model mostly picks the correct one. However, it is a very small majority for the middle classes with only around 30% accuracy. Thus, we can see that the model has a hard time separating classes close to each other. This indicate that our data lacks enough distinguishable information to separate these middle classes.

Another interesting observation is that the validation accuracy remains constant at 0.4 in the experiments while the validation loss steadily increases. This is peculiar, as we would have expected some degree of correlation between the accuracy and loss. For example, if the loss increases, the accuracy should decrease. However, accuracy solely measures the correct predictions, whereas cross-entropy measures the certainty of predictions. Therefore, it is possible that after a training step, the model still makes a correct prediction but has become more uncertain about it. Additionally, it has the same effect when the model becomes more certain about a wrong prediction. This means that it will penalize the model by increasing the loss. However, the accuracy stays the same.

5.11 Multi-task learning

In this experiment, we try to improve the performance by combining the regression and classification tasks. The model must be modified to allow multiple tasks and loss functions. The experiment's modifications and theory can be found in greater detail in section 4.14.

We will weigh the two tasks accordingly to the columns *Regression* and *Classification* in table 5. We expect that we will see an increase in accuracy for classification and a decrease in RMSE for regression, as according to [41], multi-task learning has shown great results for similar topics. As such, we expect event (A) (from section 4.14), where the features are useful for both tasks.

The RMSE of the different experiments can be seen in table 5. The loss curves and confusion matrices performed similarly to the ones previously presented (figure 13a, figure 19 and figure 20a), as such they have been omitted in this section. However, even though the accuracies and loss values were similar, we can in table 5 see that all the experiments' RMSE values are higher than the baseline model's RMSE of 0.259 (section 5.2). This

Name	Regression	Classification	RMSE
Even weights	1.0	1.0	0.268
High regression, low classification	1.0	0.5	0.267
Low regression, high classification	0.5	1.0	0.279

Table 5: Shows the weights and RMSE values for the different multi-task learning experiments.

indicates that instead of the expected event (A) occurring, event (C) is occurring. Meaning the features learned by the encoder might not be useful for both tasks. This can also be seen when we lower the weight for the regression task. Then the model performs worse than when the weights are even. Otherwise, if we increase the regression task's weight, the model performs slightly better than when using even weights. As such, we can see that however we weight the tasks, the RMSE cannot be improved using multi-task learning.

5.12 Ordinal classification

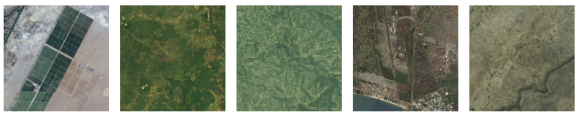
This experiment will be conducted using CORAL. The experiment and CORAL are described in more detail in section 4.15. CORAL is a library that can be used together with TensorFlow and Keras [46] to make a new model that considers the ordering of our classes. Due to CORAL improved the accuracy in their paper's [42] domain, we expect that by considering the ordering for our domain, we will also be able to see an improvement in the accuracy. Intuitively this makes sense, as giving the model an understanding of how the classes are related should make it more certain about if it is closer to 0, 1, or somewhat in the middle.

The results were the following: for ordinal classification, it got an accuracy of 37.4%, whereas the baseline got an accuracy of 37.8%. Both of the accuracies are computed on the test set.

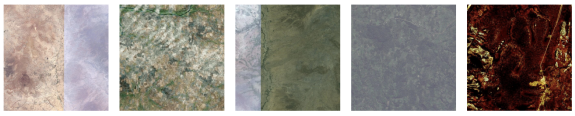
Unexpectedly, this means that ordinal classification slightly worsens the model's performance. However, this difference is very small. Thus, it may be due to random aspects such as weight initialization and random batches. As such, this indicates that even though an ordering is introduced, it is not guaranteed to make better predictions.

5.13 Model inspection using top-k analysis

In the previous experiments, we see that our models struggle to fit the validation data. To better understand what the baseline model (shown in figure 5) reacts to, we will perform a top-k analysis and find the k best and worst images. We will perform the experiment as described in section 4.17, using the normalized approach for calculating the errors. Specifically, we will visually inspect the top-100 best and worst images and compare them to look for similarities and differences. As it would require too much space to have all the top-100 best and worst, a smaller plot of the top-5 best and worst images is shown in figure 21.



(a) The top-5 best images



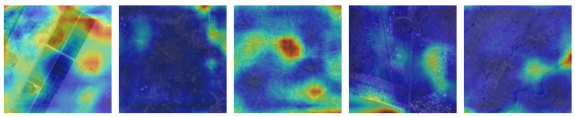
(b) The top-5 worst images

Figure 21: Shows the top-5 best and worst images for the baseline model.

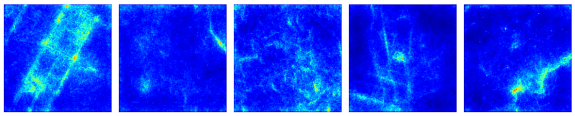
We observed the following trends when inspecting the top-100 best and worst. (1) The colors differ between the best and worst images. The best images seem to have more vibrant colors, contrasts, and sharper images. (2) The best images more often have human-constructed structures and large water bodies, while the worst images in many cases, look like a rural landscape without much civilization. (3) A large part of the worst images consists of merged images, specifically 15 out of the 100 images. Three of these can be seen in figure 21b. These merged images might give complications as they introduce unnatural contours and contrasts. To verify this, we will apply two different saliency map methods. We will use Simonyan et al.'s method [44] and Grad-CAM++ as described in section 4.16. For the remainder of this thesis, we will use Saliency Map to denote Simonyan et al.'s method and Grad-CAM++ to denote the Grad-CAM++ method.

Saliency Maps and Grad-CAM++ In figure 22, the Saliency Map and Grad-CAM++ are shown for the top-5 best images, while figure 23 shows the top-5 worst images. In the Grad-CAM++ images (figures 22a, 23a), we can see that the baseline model reacts to villages/cities and roads. This seems plausible as it shows human activity, which might impact the model's decision. For the Saliency Maps (figures 22b, 23b), the model also reacts to villages and roads, but it also shows that clouds and noisy merges influence the model. It is especially seen in the first and third images in figure 23b, where the merged areas are where the model reacts the most. When zooming in, we can see small villages where the two images merge. However, these details are too small to be captured in the satellite images. Thus, either this is a coincidence, and the model performs as it should, or it might indicate that the noisy merges and clouds influence the model's decision, which is unwanted. Additionally, when comparing the Saliency Maps from the best and worst images, we can see that the points are more scattered in the worst images. At the same time, it is more centered around the cities or water bodies for the best images. This makes it further plausible that human-made structures positively influence the model.

Furthermore, we can see in the last Grad-CAM++ image 23a of the top-5 worst images that the model reacts strongly only in a small area in the upper left corner. Inspecting the location on a newer image without color defects shows that the place the model reacted to is a tiny part of a river. As such, the model is not wrong to react there. However, it would have made sense if the model also reacted to the straight road as the Saliency Map does.



(a) Grad-CAM++ heatmaps for the top-5 best images.

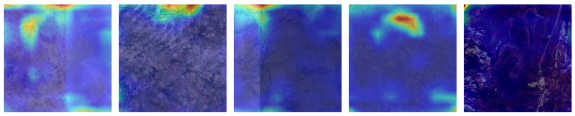


(b) Saliency Maps for the top-5 best images

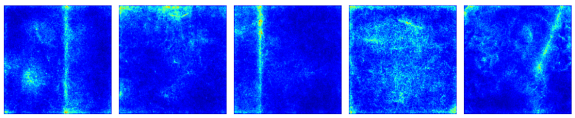
Figure 22: Shows the Saliency and Grad-CAM++ maps for the top-5 best images.

To better understand the top-100 analysis, a statistical analysis will be performed. We will inspect the distributions for the images in the best and worst and compare them to the overall distribution.

Distribution of the capture year of the satellite images We know from the image scraping process that many merged satellite images are from the early years, with less data. Thus ArcGIS has to stitch multiple images together, potentially from different years and captured using different methods. This experiment can also help to tell if the merged images influence the model's decisions.



(a) Grad-CAM++ heatmaps for the top-5 worst images.



(b) Saliency Maps for the top-5 worst images

Figure 23: Shows the Saliency and Grad-CAM++ maps for the top-5 worst images.

We expect that we will have more images in the top-100 worst images from the early years than the later. Likewise, we expect more satellite images from the later years in the top-100 best, as these contain clearer images and fewer merges.

(a) [Removed, Confidential], Shows the distribution of the years for the top-100 best.

(b) [Removed, Confidential],Shows the distribution of the years for the top-100 worst.

(c) [Removed, Confidential], Shows the distribution of the years for the entire dataset.

Figure 24: [Removed, Confidential], Shows the different distributions of the years the satellite images are from.

The distributions of the years can be seen in figure 24. Both the best (figure 24a) and worst (figure 24b) closely follow the global distribution from the entire dataset (24c). However, we can see that the number of images in 2014 is slightly higher for the top-100 worst. As previously mentioned, this might be due to the merged images. Unexpectedly, the difference between the top-100 best and worst is relatively small, which indicates that it is not as big of a challenge as we expected. Another thing we can observe is that none of the newest satellite images occur in the top-100 best, but a few do in the top-100 worst. This is unexpected as we had guessed that they would be more occurring in the top-100 best due to their improved image quality. Visually inspecting these images, we can observe that they do not feature any larger cities. Furthermore, only one of the top-100 best images has a coastline where the model reacts. By finding the exact locations on ArcGIS, we could zoom in and see that, for all cases, it is a rural area where the population density is much smaller and not as clustered as in many of the other images. Hence, the houses and paths become too small for the image to capture when zooming out. Indicating that it is not ideal to zoom this much out.

To summarize, it does seem like clouds and the noise in the merged images influences the results a little, meaning it could make sense to remove these noisy images from the dataset and retrain the model. However, as we only expected that this would change the result slightly and not create some correlations in the dataset, which were not there before, we decided not to perform an experiment where we removed the data because it would take much time to prune the images manually.

Distribution of the countries As the surveys used to make our labels are made for different research purposes, the selected countries vary a lot. As such, we have an imbalance in the amount of data for each country, which might result in the model being more biased toward some countries. To verify this, we compare the global country distribution for the entire dataset to the top-100 best and worst images. Our hypothesis is that these biases might result in the model performing better for some countries with more data, as they are more likely to have a representative view across multiple years. The distributions can be seen in figure 25.

From figure 25c, we can observe that Tanzania has most data. Furthermore, we can observe that the model performs well for Tanzania, as it has a high amount of images in the top-100 best images (figure 25a) and only a few in the top-100 worst images (figure 25b). This is what we expect to see, as having more data can give a more representative view. However,

(a) [Removed, Confidential], Shows the distribution of the countries for the top-100 best.

(b) [Removed, Confidential], Shows the distribution of the countries for the top-100 worst.

(c) [Removed, Confidential], Shows the distribution of the countries for the entire dataset.

Figure 25: Shows the different distributions of the years.

this might not always be the case, as the images are not guaranteed to provide a more representative view. For example, let us look at Burkina Faso and Mozambique. They are the countries with the second and third most images in the dataset but perform the opposite of Tanzania. The top-100 best images have a low amount of images from Burkina Faso and Mozambique, and in the top-100 worst images, they are the two countries with the most images, constituting about 30% of the images. This indicates that there might not be a correlation between the number of images from a country and the model's performance for that country.

We can visually inspect the worst image from Burkina Faso and Mozambique to try to find some tendencies that might be the reason for the poor performance. Many of these worst images contain merges similar to those in figure 6. As such, it seems to be a tendency for merged images to make the model perform worse. Additionally, as Burkina Faso and Mozambique constitute around 30% of the top-100 worst images, the merged images might be one reason for the poor model performance.

New model without data from Burkina Faso and Mozambique In the sub-experiment, we exclude the images from Burkina Faso and Mozambique to see their impact on the model. Burkina Faso and Mozambique have the second and third most data constituting around 15% of our dataset (approximately 1100 images). Thus, excluding them, three scenarios might happen. (1) The model can perform better if it turns out that these images cannot contribute to the model. (2) The model can perform worse as we reduce our already small dataset size. (3) We might not see any changes if Burkina Faso and Mozambique do not impact the model.

We expect that excluding the images will improve the performance as they constitute 30% of bad images in the top-100 worst, even though we reduce our dataset by 15%.

The new model performed marginally better, but unexpectedly only with a small reduction in RMSE of 0.0085. The reason might be linked to the other experiments where we neither saw any improvements. Hence, all our experiments point towards a data problem. More specifically, our dataset is too noisy, or there is too little correlation between our satellite data and our parasite rates. This could explain it as even though we remove the images which perform badly, it does not create some new correlations in the remaining data, resulting in the model not performing better. Furthermore, this indicates that it is more than a local problem for some specific countries.

New model with data only from Tanzania We can try to perform the opposite experiment, where we only include a country with good data (i.e., Tanzania). If we have some correlations in the data, it should be for the images that the baseline model actually seems to be able to learn something from and performs well at. As such, we expect that we will see an increase in model performance when only including the images from Tanzania. There are 775 data points from Tanzania.

This new Tanzania model resulted in a 55% reduction in RMSE from 0.259 to 0.1165. At first glance, this indicates that our data for Tanzania has some correlation, which the model can learn. Hence, the model performed better, as we had expected. However, we must keep in mind that the problem has changed and most likely become easier since we are now only looking at a single country.

To inspect its performance, we first use Grad-CAM++ for the previously shown top-5 best and worst (figures 21a,21b). The idea is to look for differences now that our new model performs much better. Unfortunately, only some minor differences were visible. In most cases, the baseline and Tanzania models reacted at the same location with almost the same weights. This is unexpected as our new model performs much better, meaning it might have

learned to look for something else than the baseline model. However, this result indicates that the Tanzania model has not learned anything new from our baseline model.

One thing we can notice when plotting the predictions as a function of the ground truth (see figure 26) is that there is a bias in the parasite rate distribution. This might also be why the Tanzania model performs better, as the values are more clustered. However, we can still see that it struggles to fit the data completely. From figure 26, we can observe that nearly all the data has a parasite rate below 0.5. Furthermore, by inspecting the data, we find that approximately 60% of the training and test data in Tanzania has a parasite rate of 0. This is not easy to see from the image, but it indicates that we have a large imbalance in the parasite rate. Meaning the imbalance might be the reason for the model to perform better for Tanzania. To better understanding the imbalance, we make a statistical analysis of the parasite rates.

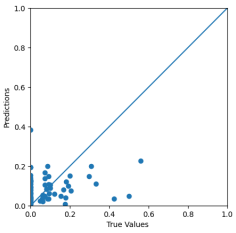


Figure 26: Shows the predicted values as a function of the ground truth values for the Tanzania model.

Distribution of parasite rates In this sub-experiment, we will compare the top-100 best and worst images and inspect their distributions. Due to the imbalance, we expect more data to have a parasite rate in the interval $[0.5, 1]$ in the top-100 worst images than in the top-100 best images. The results can be seen in figure 27.

Firstly, we can notice that for the best images (figure 27a), almost all of our images have a parasite rate of 0. It clearly states that our baseline model has a bias towards 0. Whereas for the worst images (figure 27b), the distribution has a higher variance. We can also observe that the baseline model struggles with parasite rates in the interval $[0.4, 1]$. This might be because of the high imbalance. We have 1822 images directly for the parasite rate 0 but only 51 from the parasite rate 1. This imbalance occurs since we choose the same amount of images for parasite 0 as for the parasite rates in the interval $[0.5, 1]$. This was made to have a balanced dataset for the classification task. Unfortunately, this has resulted in an imbalance for the regression task, which means that 25% of our dataset has a parasite rate of 0. This will be discussed more in the discussion (section 6).

This imbalance can also explain why our baseline model performs so well for Tanzania. Looking at figure 26, we can see that almost all of Tanzania's parasite rates are near 0. With more than 60% of the top-100 best images (figure 27a) having parasite 0, it indicates that our baseline model seems to perform well for values close to parasite 0.

An interesting follow-up experiment is to eliminate the imbalance. This can be done by simplifying the problem further to a binary classification problem. Thus, the problem is whether an image has a parasite rate of 0 or above. By doing this, we expect the model to fit the data better if a correlation exists between those areas with and without malaria. However, simplifying the problem does make the problem less informative and thereby also decreases its usefulness. Due to time constraints, we have not performed this experiment.

(a) [Removed, Confidential], Shows the parasite rate distribution for the top-100 best images.

(b) [Removed, Confidential], Shows the parasite rate distribution for the top-100 worst images.

Figure 27: Shows the parasite rate distribution for the top-100 best and worst images. To make the results more readable, a bin size of 20 is used.

5.14 K-Nearest Neighbour feature vector analysis

This experiment inspects the fully connected layers in our baseline model (shown in figure 5). To perform the experiment, we will do as described in section 4.18, where we find the K-Nearest Neighbours for some selected test images and visually inspect and discuss the results. The K-Nearest Neighbours uses the training data while the test images are taken from the testing data. This is done as we have more data in the training dataset, meaning the results are more representative. As we have not trained the model to classify the different

categories described below (and shown in figure 28), it should not be a problem to use the training set images.

The images used for testing can be seen in figure 28. Notice that we also tested with other images of similar types but did not include them due to space consumption. We will be providing images containing merged images (figure 28a), clouds (figure 28b), water bodies (figure 28c), urban area (figure 28d), and rural area (figure 28e). By rural, we mean images with few to no buildings. Often these buildings are too small to be seen on the satellite images.

(a) Merged (b) Clouds (c) Water bodies (d) Urban (e) Rural

Figure 28: Shows the different input images for the different categories.

We expect all the categories to have their own characteristic and thus will be distinguishable. We also expect that for some of the images, there might be an overlap between the different categories, as some of the characteristics can be correlated. For example, if we insert an image containing a water body in the K-Nearest Neighbours algorithm, there might also be buildings in the nearest images, as cities often are located close to water bodies. Lastly, we expect that the model will have a harder time for rural areas as their features might not be as dominant, in terms of large contrasts, as for the other categories. However, we expect the model to be able to differentiate between rural and urban.

For the experiments, we have inspected the 50-Nearest Neighbours, but for simplicity and space consumption, we will only be showing the 5-Nearest Neighbours for each of the categories. These can be seen in figure 29.

We will now be commenting on the results for the categories one by one.

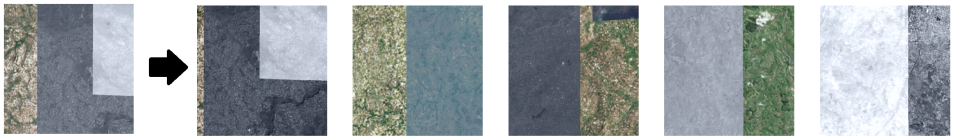
Merged In all the neighbouring images, we can see they all consist of merged images. However, we can see that the merges can have many different shapes, locations, and colors. This is what we want to see, as it means that the model's learned features are independent of the low-level features.

Clouds For almost all the neighbouring images, we can see that there also are clouds in different locations and with different shapes and sizes. Only in a few cases of the 50-Nearest Neighbours were there some images without any clouds, where they were with urban areas instead. But for these cases, we could see some brown spots, as we also see in the first neighbour's image in figure 28b.

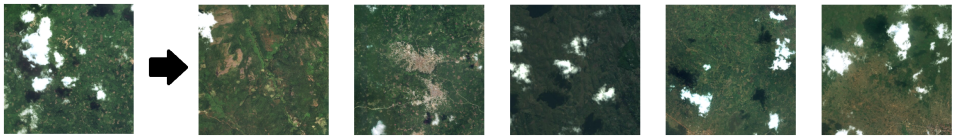
Water bodies In the nearest neighbouring images, we see winding rivers with different colors, shapes, sizes, etc. This is what we hope to see, as it indicates that the model not only has learned particular meanders but the general idea of water bodies and how they should look compared to their surroundings. For some of the images in the 50-Nearest Neighbours, it was possible to see larger lakes or seawater. This might be because rivers are often close to larger water bodies in our dataset.

Urban Most of the nearest neighbours contain larger cities with buildings covering most of the image. However, in some of the images, it is possible to see some rural landscapes. For example, the one shown at the end in figure 29 looks like a rural agricultural area. A few other images in the 50-Nearest Neighbours also looked rural. As such, the model is in most cases able to differentiate between urban and rural as we expected.

Rural Many of these images are a combination of rural and urban. We expected to find images without any larger cities. An explanation could be that, from our visual inspection,

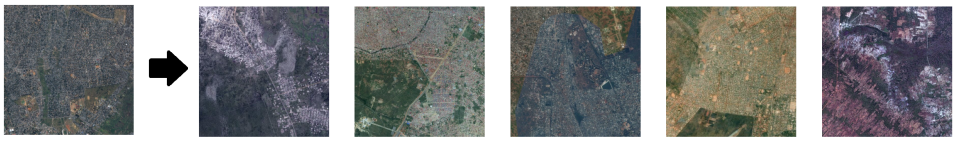


(a) Merged

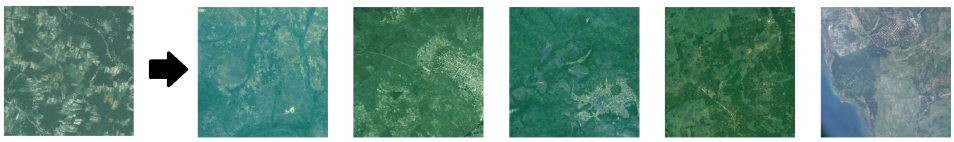


(b) Cloud

(c) Water bodies



(d) Urban



(e) Rural

Figure 29: In each row, the image left of the arrow is the test input image, while the images to the right are the 5-Nearest Neighbours in the training set. In each row, the images for a different category can be seen.

the majority of the satellite images seem to contain some urban areas. One thing to notice is many of the images have green nuances. This might indicate that color is a strong indication of when it is rural. As previously mentioned, we tried other rural images with both green and brown nuances as input to the K-Nearest Neighbour algorithm. Those we tested all seem to be independent of the green nuance and do not show any larger cities. One of the images with a mix of green and brown can be seen in figure 30.

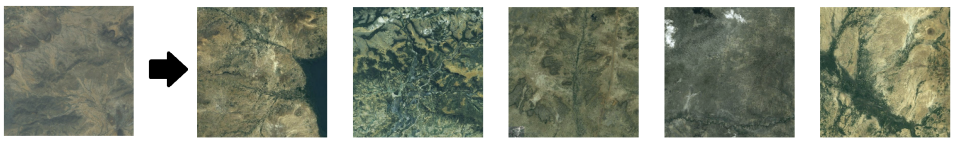


Figure 30: Shows a better example for rural areas.

To summarize, the model performs as expected for all categories except for the rural areas. This indicates that the model has learned certain general features and is capable of detecting factors beyond meanders in the winding rivers, pixels, and colors. However, distinguishing between rural and urban areas seems to be a challenge for the model, but not the converse. This could be due to the nature of our data, with a majority being around larger cities.

One can argue that this analysis is not that in-depth, thus we cannot make any conclusions based on it. This is because the method is very dependent on the input image. This also happened for the rural category, where it in the figure 28e seemed to be depending on the color, but for the other rural input images, it did not. Apart from this disadvantage, the method does give a good visual intuition of what the features in our fully connected layers are capable of.

6 Discussion

Model performance A general tendency for all of our models was that they struggled to achieve good results. The highest performing models were those respectively using 1024×1024 satellite images, land cover data back to 1992, and only using data from Tanzania. The model with 1024×1024 satellite images has a validation loss of 0.54 and an RMSE of 0.227, the land cover model has a validation loss of 0.49 and an RMSE of 0.220, and finally, the Tanzania model has a validation loss of 0.31 and an RMSE of 0.117. However, as mentioned in section 5.13 and shown in figure 26, the Tanzania model's predictions still do not fit the true values particularly well, and the improvement most likely was due to the data imbalance. As such, the RMSE can be misleading.

Our classification models got an accuracy of around 40%. Random guessing is 25%, and since we have balanced classes if a model always predicts one class, the accuracy would also be 25%. So our accuracy of 40% indicates that it is possible to learn some tendencies from our data; however, we expected it to be more accurate.

Most of our experiments did not result in any improvements. All of the following techniques resulted in a model that performed similar to or worse than the baseline model: dropout, different fully connected layers, data augmentations, different transfer learning models, using more data types with different fusion strategies, multi-task learning, and removing data from Burkina Faso and Mozambique. For the classification task, the same happened, where dropout, data augmentations, ordinal classification, and multi-task learning did not improve the model.

We investigated the problem by conducting the top-k and K-Nearest Neighbour analyses, but we did not see any clear reasons why the model struggled to fit the data. As such, we suspect the biggest reason is the lack of correlations in our dataset. As no benchmark dataset exists for our domain, we had to construct a dataset ourselves. In this process, we had to use what was possible to get access to, and as we wanted a large scale (i.e., the entire continent of Africa), the malaria parasite rate data was very limited. However, we managed to construct the dataset.

Data quality We suspect the quality of our data has a high influence on the poor performance of our models. Our RGB satellite images vary in quality; some consist of merged images, others have clouds, are greyscale, taken at night, are blurry, and some are zoomed out, covering an area much larger than 5×5 km. The land cover data could also be better, with an accuracy of approximately 75%. In the malaria data, the survey date is not necessarily corresponding to each participant's infection date, and the geographical location of the survey may be off by 5 km. Furthermore, our dataset size is small compared to other computer vision datasets, e.g., ImageNet with millions of images. As such, our data sources have some problems.

Old data and the evolution of malaria risk factors Higher quality and quantity of malaria data would likely improve our model's performance. Additionally, if the model should be able to inform us of the current situation on malaria risk factors, then more recent data would be preferred. Our malaria data dates back to 1984, and the older data might be less relevant for understanding malaria in the present. The factors influencing malaria risk are evolving. For example, the biological factors are evolving as mosquitoes develop resistance to insecticides [47]. Socioeconomic factors, like improved housing, are also evolving [48]. Furthermore, new inventions like vaccines [49] and genetically modified mosquitoes [50] are made to fight malaria. So in this changing world, new interventions to acquire higher-quality malaria data are crucial for improving malaria risk prediction.

More data types Adding more data types could improve our model. According to Weiss et al. [8], surface moisture, vector breeding sites, vegetation indices, elevation, and humidity are the next most used data types, after temperature, precipitation, and land cover. Other data types that could be interesting to use are malaria control interventions such as insecticide-treated bed nets, insect sprays, and antimalarial drugs. These interventions should all correlate with how much malaria is in an area, as the need for interventions increases as malaria's prevalence increases.

We decided not to use the year as an input for our model. Our model should not learn the yearly development of malaria but determine which geographical, meteorological, and possibly socioeconomic factors influence malaria. However, it would have been better to include the year in the model and then use some detrending approach to make the model independent of the year afterward. We did an experiment using the year as input (not included in the experiments section), showing a slight improvement in the model's performance. However, detrending the year is left as future work.

Use of RGB satellite images Whether RGB satellite images show any promise for predicting malaria parasite rates, they show the most promise with high-resolution images. We have mentioned earlier that most researchers are not using RGB satellite images but instead use land cover data. The model with land cover data from 2010 onward performed slightly better than the baseline (using 256×256 RGB satellite images). However, the model using 1024×1024 RGB satellite images performed better than the land cover model, with a 1% difference in validation loss and a 10% difference in RMSE. One advantage of the land cover data is that it is more compact than the satellite images. This greatly impacts the training time, where an epoch in the land cover model takes around 3 seconds, whereas it takes 420 seconds for the 1024×1024 satellite image model. Thus, the training time may be one of the reasons why RGB satellite images are not as widely used to predict malaria parasite rates.

Use of CNNs As mentioned in our related works section CNNs have not been used much in mapping malaria risk [20] nor for risk mapping of other diseases [19]. We suspect that this is because other authors see land cover images as the essence of satellite images. Land cover images often have a much lower resolution than satellite images; thus, using a CNN model might not be necessary. Instead, the land cover data can, for example, be transformed into a vector with entries representing percentages of land cover classes in an area, and this vector could then be used in a simple and more traditional machine learning model. Furthermore, in our experience, it is also easier to find good, freely accessible land cover datasets than satellite images. A potential downside with using land cover data instead of satellite data is that the machine learning algorithm used for generating the land cover might be flawed and not be 100% accurate. Hence, if we use flawed land cover data to train our model, our model might inherit these flaws.

Skewed target data We inspected how the model performed in the top-k and K-Nearest Neighbours analysis. Here we also were reminded that we have a data imbalance in the regression task, i.e., the distribution of malaria parasite rates has a positive skew, with most of our target values being 0. Regression models work worse for skewed data [51], so future work could be to fix the skewness by transforming the data.

Running experiments and early stopping We ran most of our experiments for 70 epochs. However, in most cases, the model had its best validation performance much earlier. Thus, the model ended up overfitting. A way of fixing this is to use early stopping. Early stopping could also give a more accurate view of the validation loss, as it often worsens after many epochs. Therefore, it could be better to save with the best validation performance and calculate the RMSE from this instead of the overfitted model. We did not use early stopping in the experiments since we wanted to see how the models developed over time. For example, we did this in the dropout experiment to determine which model overfits the most.

Other model architectures Apart from more data, other model architectures could also result in higher performance. Architectures we have considered are Vision Transformers,

Bayesian geostatistical models, and using more traditional machine learning techniques such as K-nearest Neighbor Regression.

7 Conclusion

This thesis investigates the application of machine learning and remote sensing data for malaria risk prediction. We analyze various model architectures and techniques, such as dropout, data augmentations, transfer learning, multi-modal fusion strategies, ordinal classification, and multi-task learning. We evaluate their performance using the following explainability techniques: Saliency Maps with Simonyan et al.'s method [44], Grad-CAM++, top-k analysis, and K-nearest Neighbour feature vector analysis. Furthermore, this thesis concludes that high-resolution RGB satellite images can outperform the more widely used land cover data. However, using high-resolution RGB satellite images comes with the cost of much slower training times.

Noise in our satellite images, land cover data, ground truth malaria data, and maybe temperature and precipitation data has made it challenging to train robust models. Thus we believe a higher quality and quantity of data is needed to improve model performance.

In conclusion, this thesis contributes to the growing body of research in the field of disease surveillance and highlights the potential of machine learning in improving malaria risk prediction. Continued research and advancements in data quality, model architectures, and the incorporation of additional relevant data types are crucial for improving the performance and reliability of malaria risk prediction models. Hence, this thesis helps pave the way for future advancements in leveraging remote sensing data for public health applications and ultimately supports global efforts in malaria control and eradication.

References

[1] World Health Organization. *World malaria report 2022*, page x and xxi (12 and 23 in the pdf). World Health Organization, Genève, Switzerland, December 2022.

[2] Takayoshi Ikeda, Swadhin K. Behera, Yushi Morioka, Noboru Minakawa, Masahiro Hashizume, Ataru Tsuzuki, Rajendra Maharaj, and Philip Kruger. Seasonally lagged effects of climatic factors on malaria incidence in south africa. *Scientific Reports*, 7(1), May 2017.

[3] Upcoming paper: Risk-assessment of vector-borne diseases based on deep learning and remote sensing. `https://datascience.novonordiskfonden.dk/projects/risk-assessment-of-vector-borne-diseases-based-on-deep-learning-and-remote-sensing/`. Accessed: 2023-03-20.

[4] About malaria - cdc. `https://www.cdc.gov/malaria/about/faqs.html#:~:text=Usually%2C%20people%20get%20malaria%20by,taken%20from%20an%20infected%20person.` Accessed: 2023-03-20.

[5] Mosquitoes, wikipedia. `https://en.wikipedia.org/wiki/Mosquito`. Accessed: 2023-05-19.

[6] A world without mosquitoes? it's not as great an idea as it may seem, reconnectwithnature. `https://www.reconnectwithnature.org/news-events/the-buzz/world-without-mosquitoes-not-as-easy-as-it-seems/`. Accessed: 2023-05-29.

[7] Lifecycle of anopheles mosquitoes. `https://www.cdc.gov/mosquitoes/about/life-cycles/anopheles.html`. Accessed: 2023-03-20.

[8] Daniel J Weiss, Bonnie Mappin, Ursula Dalrymple, Samir Bhatt, Ewan Cameron, Simon I Hay, and Peter W Gething. Re-examining environmental correlates of plasmodium falciparum malaria endemicity: a data-intensive variable selection approach. *Malaria Journal*, 14(1), feb 2015.

[9] C.W. Kabaria, F. Molteni, and R. et al. Mandike. Mapping intra-urban malaria risk using high resolution satellite imagery: a case study of dar es salaam. In *Int J Health Geogr.*, 2016.

[10] M. E. Sinka, S. Pironon, N. C. Massey, J. Longbottom, J. Hemingway, C. L. Moyes, and K. J. Willis. A new malaria vector in africa: Predicting the expansion range of anopheles stephensi and identifying the urban populations at risk. *Proceedings of the National Academy of Sciences*, 117(40):24900–24908, 2020.

[11] John E. Ball, Derek T. Anderson, and Chee Seng Chan. Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community. *Journal of Applied Remote Sensing*, 11(04):1, sep 2017.

[12] Lei Ma, Yu Liu, Xueliang Zhang, Yuanxin Ye, Gaofei Yin, and Brian Alan Johnson. Deep learning in remote sensing applications: A meta-analysis and review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 152:166–177, 2019.

[13] Malaria atlas project. `https://data.malariaatlas.org/`. Accessed: 2023-01-30.

[14] Daniel J Weiss, Tim C D Lucas, Michele Nguyen, Anita K Nandi, Donal Bisanzio, Katherine E Battle, Ewan Cameron, Katherine A Twohig, Daniel A Pfeffer, Jennifer A Rozier, Harry S Gibson, Puja C Rao, Daniel Casey, Amelia Bertozzi-Villa, Emma L Collins, Ursula Dalrymple, Naomi Gray, Joseph R Harris, Rosalind E Howes, Sun Yun Kang, Suzanne H Keddie, Daniel May, Susan Rumisha, Michael P Thorn, Ryan Barber, Nancy Fullman, Chantal K Huynh, Xie Kulikoff, Michael J Kutz, Alan D Lopez, Ali H Mokdad, Mohsen Naghavi, Grant Nguyen, Katya Anne Shackelford, Theo Vos, Haidong Wang, David L Smith, Stephen S Lim, Christopher J L Murray, Samir Bhatt, Simon I Hay, and Peter W Gething. Mapping the global prevalence, incidence, and mortality of plasmodium falciparum, 2000–17: a spatial and temporal modelling study. *The Lancet*, 394(10195):322–331, July 2019.

[15] Di Wang, Jing Zhang, Bo Du, Guisong Xia, and Dacheng Tao. An empirical study of remote sensing pretraining. *IEEE Transactions on Geoscience and Remote Sensing*, PP:1–1, 2022.

[16] Stefano Vincenzi, Angelo Porrello, Pietro Buzzega, Annamaria Conte, Carla Ippoliti, Luca Candeloro, Alessio Di Lorenzo, Andrea Capobianco Dondona, and Simone Calderara. Spotting insects from satellites: modeling the presence of culicoides imicola through deep cnns, 2019.

[17] Ava Vali, Sara Comai, and Matteo Matteucci. Deep learning for land use and land cover classification based on hyperspectral and multispectral earth observation data: A review. *Remote Sensing*, 12(15), 2020.

[18] Arcgis online web map. `https://livingatlas.arcgis.com/wayback/`. Accessed: 2023-02-09.

[19] Ravikiran Keshavamurthy, Samuel Dixon, Karl T. Pazdernik, and Lauren E. Charles. Predicting infectious disease for biopreparedness and response: A systematic review of machine learning and deep learning approaches. *One Health*, 15:100439, 2022.

[20] Julius Nyerere Odhiambo, Chester Kalinda, Peter M Macharia, Robert W Snow, and Benn Sartorius. Spatial and spatio-temporal methods for mapping malaria risk: a systematic review. *BMJ Global Health*, 5(10):e002919, October 2020.

[21] Anne Caroline Krefis, Norbert Georg Schwarz, Bernard Nkrumah, Samuel Acquah, Wibke Loag, Jens Oldeland, Nimako Sarpong, Yaw Adu-Sarkodie, Ulrich Ranft, and Jürgen May. Spatial analysis of land cover determinants of malaria incidence in the ashanti region, ghana. *PLoS ONE*, 6(3):e17905, March 2011.

[22] Quang-Thanh Bui, Quoc-Huy Nguyen, Van Manh Pham, Minh Hai Pham, and Anh Tuan Tran. Understanding spatial variations of malaria in vietnam using remotely sensed data integrated into gis and machine learning classifiers. *Geocarto International*, 34(12):1300–1314, 2019.

[23] Michael C. Wimberly, Dawn M. Nekorchuk, and Ramcharan R. Kankanala. Cloud-based applications for accessing satellite earth observations to support malaria early warning. *Scientific Data*, 9(1), May 2022.

[24] David Harvey, Wessel Valkenburg, and Amara Amara. Predicting malaria epidemics in burkina faso with machine learning. *PLOS ONE*, 16(6):e0253302, June 2021.

[25] Ai for malaria prevention: Identifying water bodies through satellite imagery. https://omdena.com/blog/artificial-intelligence-malaria/. Accessed: 2023-03-10.

[26] Imen Jdey, Ghazala Hcini, and Hela Ltifi. Deep learning and machine learning for malaria detection: overview, challenges and future directions, 2022.

[27] Malaria atlas project r package. https://cran.r-project.org/web/packages/malariaAtlas/index.html. Accessed: 2023-02-20.

[28] Lawrence Stewart, Francis Bach, Quentin Berthet, and Jean-Philippe Vert. Regression as classification: Influence of task formulation on neural network features, 2023.

[29] Raied Salman and Vojislav Kecman. Regression as classification. In *2012 Proceedings of IEEE Southeastcon*, pages 1–6, 2012.

[30] Imbalanced data. https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data. Accessed: 2023-03-27.

[31] Land cover data. https://explorer.digitalearth.africa/products.

[32] Deafrica landcover explanations. https://github.com/digitalearthafrica/deafrica-sandbox-notebooks/blob/main/Datasets/Landcover_Classification.ipynb. Accessed: 2023-05-01.

[33] Esa climate change initiative land cover at 300m spatial resolution. https://docs.digitalearthafrica.org/en/latest/data_specs/CCI_Landcover_specs.html#ESA-Climate-Change-Initiative-Land-Cover-at-300m-spatial-resolution.

[34] Temperature data description. https://lpdaac.usgs.gov/products/mod11a2v006/. Accessed: 2023-02-09.

[35] List of weather records, wikipedia. https://en.wikipedia.org/wiki/List_of_weather_records. Accessed: 2023-05-22.

[36] Vajira Thambawita, Inga Strümke, Steven A. Hicks, Pål Halvorsen, Sravanthi Parasa, and Michael A. Riegler. Impact of image resolution on deep learning performance in endoscopy image classification: An experimental study using a large dataset of endoscopic images. *Diagnostics*, 11(12), 2021.

[37] Understanding dropout with the simplified math behind it. https://towardsdatascience.com/simplified-math-behind-dropout-in-deep-learning-6d50f3f47275.

[38] Gencer Sumbul, Arne de Wall, Tristan Kreuziger, Filipe Marcelino, Hugo Costa, Pedro Benevides, Mario Caetano, Begüm Demir, and Volker Markl. Bigearthnet-mm: A large scale multi-modal multi-label benchmark archive for remote sensing image classification and retrieval. *CoRR*, 2021.

[39] Said Yacine Boulahia, Abdenour Amamra, Mohamed Ridha Madi, and Said Daikh. Early, intermediate and late fusion strategies for robust deep learning-based multimodal action recognition. *Machine Vision and Applications*, 32(6), September 2021.

[40] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos, 2014.

[41] Ananya Joshi and Clayton Miller. Review of machine learning techniques for mosquito control in urban environments. *Ecological Informatics*, 61:101241, 2021.

[42] Wenzhi Cao, Vahid Mirjalili, and Sebastian Raschka. Rank consistent ordinal regression for neural networks with application to age estimation. *Pattern Recognition Letters*, 140:325–331, 2020.

[43] Saliency map. https://en.wikipedia.org/wiki/Saliency_map. Accessed: 2023-05-31.

[44] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps, 2014.

[45] Review — grad-cam++: Improved visual explanations for deep convolutional networks (weakly supervised object localization). https://medium.com/swlh/review-grad-cam-improved-visual-explanations-for-deep-convolutional-networks-weakly-760264e66bc6. Accessed: 2023-05-28.

[46] Ordinal regression in tensorflow keras, coral. https://github.com/ck37/coral-ordinal. Accessed: 2023-05-09.

[47] Malaria threat map, threats to malaria control. https://apps.who.int/malaria/maps/threats/. Accessed: 2023-06-01.

[48] Lucy S. Tusting, Donal Bisanzio, Graham Alabaster, Ewan Cameron, Richard Cibulskis, Michael Davies, Seth Flaxman, Harry S. Gibson, Jakob Knudsen, Charles Mbogo, Fredros O. Okumu, Lorenz von Seidlein, Daniel J. Weiss, Steve W. Lindsay, Peter W. Gething, and Samir Bhatt. Mapping changes in housing in sub-saharan africa from 2000 to 2015. *Nature*, 568(7752):391–394, March 2019.

[49] Malaria: The malaria vaccine implementation programme (mvip). https://www.who.int/news-room/questions-and-answers/item/malaria-vaccine-implementation-programme. Accessed: 2023-06-01.

[50] Andrew Hammond, Paola Pollegioni, Tania Persampieri, Ace North, Roxana Minuz, Alessandro Trusso, Alessandro Bucci, Kyros Kyrou, Ioanna Morianou, Alekos Simoni, Tony Nolan, Ruth Müller, and Andrea Crisanti. Gene-drive suppression of mosquito populations in large cages as a bridge between lab and field. *Nature Communications*, 12(1), July 2021.

[51] How to deal with skewed dataset in machine learning? https://becominghuman.ai/how-to-deal-with-skewed-dataset-in-machine-learning-afd2928011cc.