Better Routing in Developing Regions

Stienen, Valentijn; den Hertog, Dick; Wagenaar, J.C.; Zegher, J.F.

Publication date:
2023

Document Version
Early version, also known as pre-print

No. 2023-025

BETTER ROUTING IN DEVELOPING REGIONS: WEATHER AND SATELLITE-INFORMED ROAD SPEED PREDICTION

By

V.F. Stienen, D. den Hertog,
J.C. Wagenaar, J.F. de Zegher

18 September 2023

ISSN 0924-7815
ISSN 2213-9532

Better routing in developing regions: weather and satellite-informed road speed prediction

V.F. Stienen[1], D. den Hertog[2], J.C. Wagenaar[1], J.F. de Zegher[3]

[1] Tilburg University, Department of Econometrics and Operations Research, Zero Hunger Lab, Tilburg, The Netherlands.

[2] University of Amsterdam, Amsterdam Business School, Amsterdam, The Netherlands.

[3] Massachusetts Institute of Technology, Sloan School of Management, Operations Management, Cambridge MA, USA.

September, 2023

Abstract Inaccurate digital road networks significantly complicate the use of analytics in developing, data scarce, environments. For routing purposes, the most important characteristic of a digital road network is the information about travel times/speeds of roads. In developing regions, these are often unknown, and heavily dependent on the weather (*e.g.*, rainfall). This may, for instance, cause vehicles to experience longer travel times than expected. Current methods to predict the travel speeds are designed for the short upcoming period (minutes or hours). They make use of data about the position of the vehicle, the average speed on a given road (section), or patterns of traffic flow in certain periods, which are typically not available in more developing regions. This paper presents a novel deep learning method that predicts the travel speeds for all roads in a data scarce environment using GPS trajectory data and open-source satellite imagery. The method is capable of predicting speeds for previously unobserved roads and incorporates specific circumstances, which are characterized by the time of the day and the rainfall during the last hour. In collaboration with the organization PemPem, we perform a case study in which we show that our proposed procedure predicts the average travel speed of roads in the area (that may not exist in the GPS trajectory data) with an average RMSE of 8.5 km/h.

Keywords Traffic speed · Road attribute prediction · (Convolutional) neural network · Satellite imagery · Weather information

1 Introduction

An accurate representation of a road network is essential when performing network analyses. As discussed in Stienen et al. (2022), an accurate spatial representation of the road network geometry is, among other things, important to optimize the trade of micro-enterprises (SDG 8) and determining the placement of hospitals in a region (SDG 3). However, optimizing routing decisions in these situations necessitates knowledge of the speed one is expected to drive on a given road, which we refer to as the *travel speed*. This information can significantly impact the optimal route, as certain roads may take more or less time to traverse compared to their length (*e.g.,* highways or rural roads). Moreover, in less developed regions, the travel speed may be highly influenced by external factors such as the time of day or weather conditions (*e.g.,* rainfall). While we often rely on

information from sources like OpenStreetMap (2023) (OSM) or Google Maps, this data may not always be available, especially not in developing regions.

We collaborate with PemPem (a marketplace for micro-enterprises in upstream commodity markets), which has to handle inaccurate digital road networks in their operations (in remote areas in developing countries). In this paper, we enhance road networks with information about the expected speed a vehicle can drive on each road. Usually, one may get an indication by looking at the speed limit of a road, but, often, in developing regions, this information is not readily available. For instance, in our region of interest (where PemPem operates), we observe that, in OSM, information about the speed limit is available for less than 5% of the roads (which cover less than 4% of the total amount of kilometers) in our road network (without knowing whether this speed limit is actually reliable). Moreover, especially in developing regions, the travel speed may often be affected by external factors, such as rainfall, or whether it is dark outside when driving. Recently, PemPem equipped several trucks with GPS trackers, in order to gain additional information about the road network. The GPS trackers generate continuous streams of speed observations as the vehicles drive around in the designated areas.

In this paper, we aim to predict travel speeds for roads in a road network using a limited dataset of speed observations, obtained via GPS trackers, from vehicles driving around in the area. The data may not cover all roads in the network, and our goal is to estimate the travel speeds (of all roads) based on known characteristics, which may be shared with other roads. To enhance our predictions, we leverage satellite imagery, weather information, and data on whether it is dark outside. Specifically, we propose a novel deep learning approach that utilizes a data-augmented set of speed records along with weather and satellite data to estimate the expected time a vehicle will drive on a specific road under particular conditions. We apply this method to a case study in Indonesia, in collaboration with PemPem.

Literature survey

In recent years, many papers have been published about *travel speed prediction*. Similar papers have also tried to predict the travel time of certain road (sections), which often boils down to using similar techniques. Moreover, sometimes travel speed is also considered as an attribute, a specific characteristic, of a road. Several papers have been published about *predicting* such *road characteristics/attributes*, most of which focus on the prediction of the road quality. In our research, we combine these two streams of literature. Therefore, we first, briefly, discuss the existing literature on both these topics.

Travel speed (or travel time) prediction is often referred to as the usage of a learnable function that takes as input the historical traffic data from several previous time-steps in order to predict the traffic in the future. From the perspective of temporal analysis, a strong correlation usually exists among traffic time series, where previous traffic conditions likely have a large impact on future traffic. Historical data often consists of information about the spatial location of vehicles annotated with temporal information (time-stamps), and often comes from sensors or from trajectory data. In this paper, we only discuss those that use trajectory data to predict the travel speed/time, which is the most relevant in our setting. For an extensive review on speed prediction methods, we refer to the recently published article written by Zhou et al. (2022).

Early methods use statistical models, predicting the mean and variance of travel times based on known explanatory variables (Jenelius and Koutsopoulos (2013)), or test whether a normal distribution is recognized in travel times of trips (Jiang and Li (2013)). In the years thereafter, researchers started to use more sophisticated, but more complicated methods to predict travel times, such as compressive sensing (Liu et al. (2016)), or Support Vector Machines (SVMs) (Yao et al. (2017)).

More recently, as noted in the literature review written by Tedjopurnomo et al. (2020), deep learning methods gained popularity in the field of travel speed prediction. The ability to model complex and deep structures in order to achieve large prediction power resulted in deep learning being preferred over, for instance, model-based methods. One of the earliest attempts to use deep learning methods to predict the travel speed are employing Convolutional Neural Networks (CNNs) (Zhang et al. (2016), Yu et al. (2017), Ma et al. (2017)). Speeds are converted into a series of static images, where, for instance, the axes of the image represent time and space. Instead of using convolutional neural networks, people also use different, (or combinations of) deep learning models to predict the speed, such as Recurrent Neural Networks (RNNs) (Lv et al. (2018)), or Graph Neural Networks (GNNs) (Xie et al. (2019)). Next to these studies that focus on the spatial and temporal data to predict the future speed of a vehicle, there are also several studies that use additional features that may influence the speed. For instance, Jia et al. (2017) try to include the effect of rainfall on speed prediction. Furthermore, Abdollahi et al. (2020) augment their available dataset with external weather data to enhance predictions. Additionally, Huang et al. (2022) propose utilizing points of interest (POIs), such as restaurants, to improve travel time predictions.

The main problem with previous methods is that they are designed for the short upcoming period (minutes or hours). They make use of data about the current position of the vehicle, the average speed on a given road (section), or patterns of traffic flow in specified periods. However, in this research we seek to predict the speed that a vehicle will drive at a specific moment in time, which is not minutes or hours away. The prediction should be available before someone starts driving a road, in the *route planning phase*. This means that any features about the current state of a vehicle, such as its current position or acceleration, are not applicable. Moreover, when dealing with limited observations, it is essential that the model is able to accurately predict travel speeds of roads for which no speed data are observed. For instance, by utilizing information from roads with similar characteristics. This is also not possible when using data about the current position of a vehicle.

The stream of literature on road attribute prediction has become more popular in recent years. Popular predictions are about the road quality (Cadamuro et al. (2018), Brewer et al. (2021)) or road features that exist in OSM, such as the number of lanes on a road, or (a subset of) the road types[1] (He et al. (2020), Iddianozie and Mcardle (2021)). Where information about OSM features is often readily available, information about the road quality needs to be gathered from vehicles equipped with, for instance, vibration detectors. Most of the times, high resolution satellite imagery is the main source of information for making the prediction (Cadamuro et al. (2018), He et al. (2020), Brewer et al. (2021)).

We do note that high-resolution satellite imagery is often not (publicly) available. Moreover, when predicting road features in OSM, we may have to deal with noise in the labels. The labels may not all be based on accurate information. For a part in Indonesia, we see many features having either inaccurate or wrong labels. To minimize the possible effect of noisy labels, one could use (ideally, high-resolution) satellite imagery to predict the travel speed based on the *observed* speed registrations.

Contributions

The first contribution of this work is that we propose a novel deep learning method that, based on data scarce GPS trajectory information, predicts the average travel speed for *every road* in a road network (also the roads that are not driven by the vehicles sending trajectory data) under specific circumstances, such as driving in darkness or with heavy rainfall in the last few hours. This is done by exploiting i) open-source satellite imagery of roads (*e.g.*, that may provide information about the color

[1]The (OSM) road type indicates the importance of the road within the road network as a whole. For more information, see `https://wiki.openstreetmap.org/wiki/Key:highway`

of the road, the width of the road, any neighbouring buildings, etc.), ii) the time of the day (*e.g.*, whether it is dark outside), and iii) the weather (the amount of rainfall in the last hour).

Secondly, we perform a case study in cooperation with PemPem, in which we show that we can predict the travel speed of any given road in the area of interest, under specific circumstances, with an average Root Mean Squared Error of 8.5 km/h.

Finally, we do note that all code is publicly available at `https://github.com/valentijnstienen/PredictVelocity`.

2 Data preprocessing and feature augmentation

In this section, we discuss how we obtain and preprocess the data used for our travel speed prediction model. We start with creating a dataset that contains speed observations for specific roads at specific time instants (Section 2.1). Then, we discuss how we augment this dataset with additional information (satellite imagery (Section 2.2) and weather information (Section 2.3)), that may explain variations in travel speed.

2.1 Speed observations (from trajectory data)

We want to create a model that predicts the speed one is expected to drive on a given road at a specific moment in time. To be able to develop an empirically grounded model, we require actual observations of travel speeds that have been driven on various roads. For this, we use GPS trajectory data that is received from vehicles driving around in the area of interest. These vehicles continuously generate streams of their location (latitude/longitude) while driving, with time intervals of 10, 30, 60, or more seconds. Our goal is to accurately match these GPS streams onto the corresponding roads in the road network. This process, commonly referred to as trajectory matching, involves identifying and linking parts of GPS trajectories to specific roads in an existing graph. In this research, we make use of the algorithm developed by Stienen et al. (2022). They propose a method to extend a digital road network by using GPS trajectory data. They propose a projection-based incremental insertion method that incrementally adds new information to existing road networks. In this algorithm, GPS points are sequentially checked whether they may have been received while a vehicle was driving on a (known) road, a road that is already present in the current digital road network. When this is the case, the point is *absorbed* in the current digital road network. All the information of the absorbed GPS points is then saved in attributes of the corresponding edge. In this way, in the end, we have for a set of roads, a list of combinations of dates and travel speeds. This dataset can then be unpacked to have an observation (row) for a specific road on a specific day at a specific time. This dataset looks as in Table 1.

Edge	Features (see Stienen et al. (2022))	Date / Time	Speed
$(a, b, 0)$	$\cdots$	2020-08-30 23:55:26	28.0
$(b, a, 0)$	$\cdots$	2020-07-14 07:10:42	43.0
$(b, a, 0)$	$\cdots$	2020-07-14 07:11:12	46.0
$(b, a, 0)$	$\cdots$	2020-10-04 09:56:54	38.0
$(b, a, 0)$	$\cdots$	2020-10-04 14:50:12	38.0

Table 1: Dataset obtained when running the adjusted version of the algorithm developed by Stienen et al. (2022).

Note that an edge here consists of a starting node, an end node and a key (that distinguishes edges with the same start and end

node). Additionally, some roads might provide supplementary features, like road type data from OSM. However, it is worth noting that not all roads will have these supplementary features available.

2.2 Satellite images

The first feature that is added to the available data is a satellite image of the road. Based on this image, one may recognize specific characteristics that may influence the speed one can drive on this road (*e.g.*, road quality, as shown by Brewer et al. (2021)). To obtain such an image, we use Google Earth Engine (Gorelick et al. (2017)), in particular, open source data from the Sentinel-2A satellite (ESA (2022)), which has a resolution of 10m per pixel.

To get a representative image of a road, we propose to compute an *average image* of several images of the road. We do this to avoid taking an image of the road that is not representative due to for instance occlusion (see He et al. (2020)). First, we extract a maximum of five, evenly spread, 160m $\times$ 160m images from the road (less images if the road is not that long). Secondly, these images are then aligned, they are rotated according to the heading of the road at the corresponding points, such that all images have roads pointing in the same direction. Note that after rotating images, the 160 $\times$ 160 image may contain unknown pixels due to rotating the image to a certain degree. Therefore, we crop the image such that only the *middle square* (100m $\times$ 100m) is retained, and the boundaries that may change due to rotating images are removed. The resolution of the Sentinel-2A images is 10m per pixel, which means that this leads to a colored 10 $\times$ 10 pixel image. Now, we can find an average image by computing average pixel values of this square (average RGB colors). The whole procedure is visualized in Figure 1.

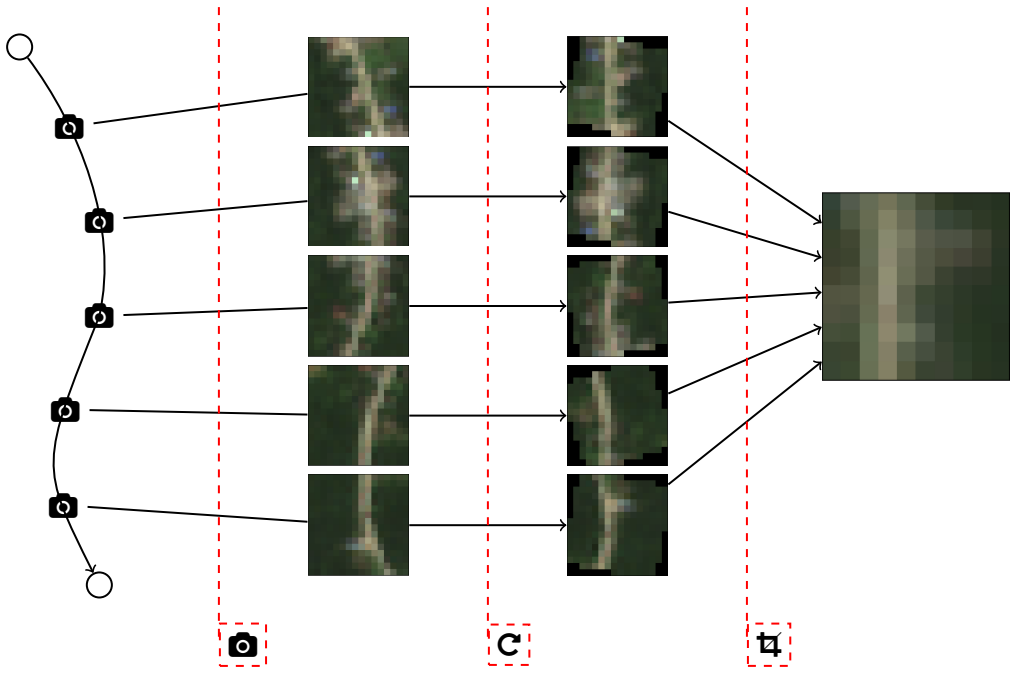


Figure 1: Illustration of obtaining a picture of a representative point of road. Left: different images of the road are extracted from Sentinel-2A data (📷). Middle: These images are rotated according the heading of the road (↻). Right: The aligned images are combined into one picture. Only the middle part is used, getting rid of missing colors due to rotations of images (⧉).

For each of the five images, we need images without any cloud obstructions, taken during the relevant time frame (when speed observations were made). These images are extracted from larger *satellite photos* taken from the Sentinel-2A satellite. For a given satellite photo (taken on a specific day), we have information available about i) the percentage of pixels that is estimated to be covered by a cloud (referred to as the `CLOUDY_PIXEL_PERCENTAGE`), and ii) for each pixel within this photo the probability that this pixel is covered by a cloud (referred to as the `MSK_CLDPR`).

Since our focus is solely on the section of the photo encompassing the road, we compute the relevant cloud probability for this part using the `MSK_CLDPR` information. Note that we typically have many photos available as the Sentinel-2A has maximum revisit time of ten days. Therefore, to speed up the process, we only compute this probability for satellite photos for which the `CLOUDY_PIXEL_PERCENTAGE` (for the entire photo) is less than or equal to 50%. Then, we choose the image with the lowest cloud probability (for our relevant region) as the most suitable representation of the road.

In addition to the road image, we augment our available data with additional information obtained from the satellite image that has the potential to impact travel speed on the road: measurements of the mean, maximum, and minimum values of the Short-Wave Infrared (SWIR) band. Measurements of the SWIR band are relevant in our situation, as they are sensitive to various surface materials and their properties. For instance, the SWIR band can provide insights into the composition and condition of the road surface, including features such as asphalt, concrete, and different levels of degradation or deterioration. Higher SWIR values are often associated with smoother road surfaces, while lower values often indicate worse road sections. Therefore, by integrating the SWIR measurements, derived from the satellite image, we are able to capture the influence of road surface characteristics on travel speed. In the remainder of this research, we refer to these variables as *SWIR info*.

2.3 Weather

The second feature that is added to the available data is information about the weather at the time of receiving a GPS point. Note that especially information about (current) rainfall may have a significant impact on the speed one may drive at a given moment in time (*e.g.*, see Jia et al. (2017)).

The weather information we use is obtained using APIs of OpenWeather (2022). The features obtained include, among other things, information about current wind speed/gust, and information about the current rainfall (of the last hour). In this research, we only exploit the information about the amount of rainfall in the last hour.

2.4 Final dataset

After augmenting the data obtained from the GPS trajectories with the satellite images, the SWIR info, and with the weather information, we obtain our (final) dataset that looks as in Table 2.

Here, the location of an edge is assumed to be the *centroid* (the center point) of that edge. Note that not all information may be available for each road. For instance, the road type is unknown for many roads (*e.g.*, newly added roads), but also, we did not observe a travel speed on every road under every possible weather condition. Using this dataset, we thus want to predict the travel speed of a given road (edge), at a given *moment in time*. In the next section, we will discuss the concept *moment in time* in more detail. This concept encompasses several factors, such as information about the road itself (satellite image), the time of day, or the amount of rainfall in the last hour,

Edge	Location	Road type	Hour	Rain (1h)	Image (100m × 100m)	SWIR info	Speed
$(a, b, 0)$	(102.41, -0.55)	trunk	11	0.8 mm	[[[122, 114, 91], [139, ⋯	⋯	28.0
$(a, b, 1)$	(102.41, -0.55)	trunk	3	1.8 mm	[[[122, 114, 91], [139, ⋯	⋯	43.0
$(b, a, 0)$	(102.41, -0.55)	trunk	5	1.6 mm	[[[122, 114, 91], [139, ⋯	⋯	46.0
$(b, a, 0)$	(102.41, -0.55)	trunk	7	0 mm	[[[122, 114, 91], [139, ⋯	⋯	38.0
$(b, a, 0)$	(102.41, -0.55)	trunk	14	2.3 mm	[[[122, 114, 91], [139, ⋯	⋯	38.0

Table 2: Final dataset including all information that can be used.

3 Mathematical approach

In this section, we describe our mathematical approach and our proposed algorithm. We start with a discussion of the assumptions we make (Section 3.1). Then, in Section 3.2, we describe the deep learning method that is used to predict the travel speed on a road at a specific moment in time.

3.1 Assumptions

The first two assumptions we have are regarding the predictions for given roads:

A.1 A road is defined as the segment that lies between two intersections. Moreover, the speed one can drive on a particular road is the same over that whole road.

A.2 The characteristics that determine the travel speed of unobserved roads are similar to those that are observed in the dataset.

Note that A.2 indicates that a model is able to predict the travel speed for roads that are not driven by the vehicles that created the (trajectory) data. This assumption also introduces our goal to only use universally available road information, such as the location or satellite data of the road. We, on purpose, do not utilize information that is not consistently available for all roads (*e.g.,* road type data). In this way, we ensure a robust and widely applicable approach, allowing the model to make predictions for an entire road network.

Next, we make assumptions about the *circumstances* we use when predicting the speed on a given road. As an example, a circumstance could be: *Driving on Road X, at 11pm, where 3mm of rainfall has been experienced over the last hour*. Obviously, this is very specific (*e.g.*, 11pm and 3mm of rainfall). If we use such a detailed classification for these features, we do not have many observations *per road per circumstance* (we do have, however, a lot of circumstances...). This reduces the reliability of, for instance, the average speed that is driven on a road during specific circumstances.

To increase the number of observations per circumstance, we propose to cluster circumstances. As an example, whether a road experienced 3mm or 3.1mm of rainfall in the last hour will most likely not affect the average travel speed of this road. Also, whether it is 2pm or 4pm, this will not affect the travel speed much. Therefore, we make the following assumptions in order to increase the amount of observations per road per circumstance:

A.3 As temporal features we use whether it is dark outside or not. In other words, it does not matter if you drive on Monday or Saturday, and it does not matter whether you drive in the morning or in the afternoon, as long as there is enough daylight. Note that typically in the developing regions we consider, traffic jams are not very common during, for instance, rush hours on weekdays.

A.4 As weather features, we will use the amount of rain that fell in the last hour. We distinguish three categories of rainfall: no/light, moderate and heavy rainfall, which we will refer to as rainfall category 0, 1, or 2, respectively. Since our case study is done in collaboration with PemPem, in an area in Indonesia, the corresponding cumulative amount of rainfall (*i.e.*, the rainfall classification) is based on information from BMKG (2022), an Indonesian non-departmental government agency for meteorology, climatology, and geophysics. A summary of the rainfall classification is shown in Table 3.

	Cat.	mm/1h
No/Light rain	0	< 0.8
Moderate rain	1	$0.8 - 2.1$
Heavy rain	2	> 2.1

Table 3: Rainfall intensity classification, based on information of BMKG (2022).

Using Assumption A.3 and A.4, we get more observations that we can use for training our model, and therefore travel speeds can be predicted more reliably. In our case, there are $28,784$ roads in our region of interest. For around 28% ($8,079$ roads) of the roads a vehicle has driven at least once (in our dataset), which indicates that there are many roads that are not even driven once for which we also want to know the speed one can drive under specific circumstances. Each road has 3×2 possible circumstances, which means that there are $28,784 \times 3 \times 2 = 172,704$ travel speeds that we want to predict for these roads. Based on the data, we have at least one observation for $17,493/172,704 = 10\%$ of these travel speeds. However, to make sure that we are not including information from outliers, we choose to only use the information that is based on at least 5 speed observations. So, if there is a single observation for a road in one specific circumstance, we do not include this information in our analyses. Consequently, merely 4% ($6,961$ out of $172,704$) of the travel speeds intended for prediction have a minimum of 5 observations. In other words, there are many edges that are not driven under specific circumstances. Note that having a higher number of rain intensity classes, or temporal features, lowers this percentage significantly.

Regarding the observations, it is worth mentioning that current driving behavior (personal, or moment-specific) may cause differences in the observations of speed during certain circumstances. There may occur different speed observations during similar circumstances. In this research, we therefore focus on predicting the *average* speed one can drive on a road under specific circumstances. To calculate the average speed for a specific road based on the observed speed registrations, a few preprocessing steps are therefore necessary. It is important to note that for each road-circumstance combination we may have multiple speed registrations available, which could have been obtained from different GPS trajectories. An example is given in Figure 2. In this figure, we observe nine speed registrations. However, these speed registrations are obtained over two trips, with 30 seconds in-between two consecutive GPS points. Since the second trip (right) has a higher average speed, there are less GPS points received (the vehicle traversed this road in less time). The average speed one can drive on this road, under the given circumstances, will therefore be the average of the average speeds of the two different trips. In this way, both trips are weighed equally.

In short, taking all assumptions into account, we use the following columns of the augmented dataset we created in Section 2: the image, the location (lat/lon), the amount of rainfall in the last hour (`Rainfall 1h (cat)`), whether it is dark outside or not (`Dark`), and the SWIR information. These columns are tabulated in Table 4. In this table, there is one row for a specific road-circumstance combination. The speed column now represents the average speed one can drive on this road under the

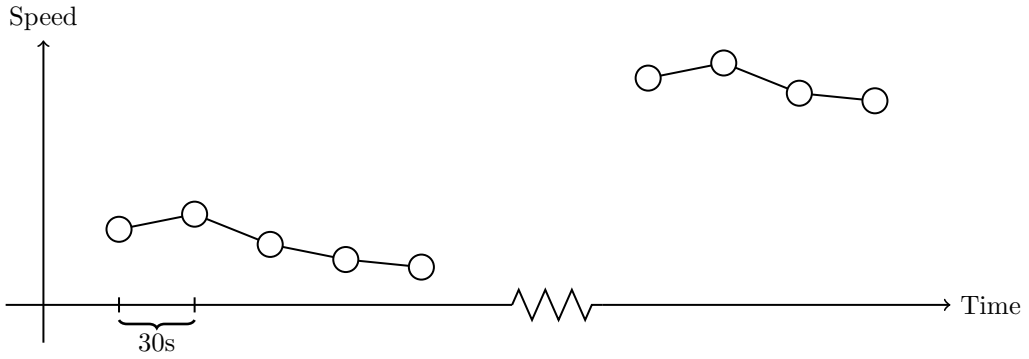


Figure 2: Illustration of the average speed computation on a road. In this figure, we observe two GPS trajectory trips, where the first trip has 5 GPS points (received when driving with low speed) and the right trajectory has only 4 GPS points (received when driving with higher speed).

corresponding circumstances (travel speed).

Image (100m × 100m)	Location	Rainfall 1h (cat)	Dark	SWIR info	Speed (km/h)
[[[122, 114, 91], [139, 131, ...	(102.41 -0.54)	1	Dark	···	33
[[[122, 114, 91], [139, 131, ...	(102.30 -0.55)	0	Light	···	42
[[[112, 112, 95], [139, 131, ...	(101.89 -0.56)	1	Light	···	46

Table 4: Final dataset used (after incorporating all assumptions).

3.2 Deep learning structure

Next, we discuss the model that we will use to predict the travel speed. We exploit the use of a deep learning structure. There are two types of data that we use: tabular data (location, the amount of rainfall, whether or not it is dark, and the SWIR info), and image data (satellite image of the road). We first extract the features of the image (*e.g.*, color, width, surrounding buildings, etc.) using a convolutional neural network (CNN). Then, these features are concatenated with the tabular features. After this concatenation, we add hidden layers before obtaining the final output layer (which is just one neuron), representing the predicted travel speed. This last part of the network is referred to as the Merged Part (MP). A visualization of the complete model is shown in Figure 3.

Next, we discuss the CNN and Merged Part (MP) of the structure in more detail.

CNN part

For the CNN, we make use of CNN architectures that have shown to perform well in the literature. First of all, we note that the images we use are $10 \times 10 \times 3$, where the 3 stands for the different RGB channels of the image. Since we exploit such small, low resolution, images, large and very deep architectures are less appropriate. We base our structure on two well known architectures that have shown their exceptional performance in the field early on: AlexNet (Krizhevsky et al. (2012)), the winner of the ImageNet image classification competition in 2010, and VGGNet (Simonyan and Zisserman (2014)), which also achieved remarkable results on the ImageNet dataset. Examining Alexnet and VGGNet, we observe that these networks take

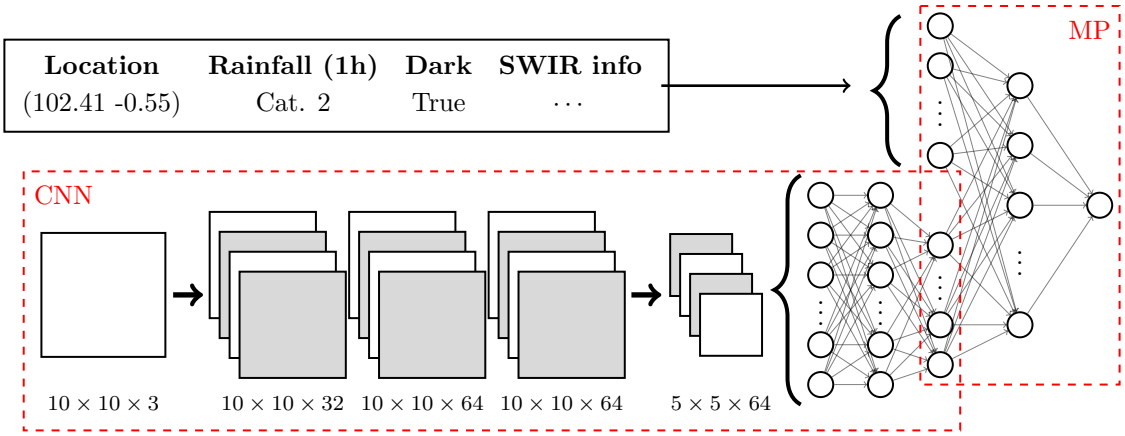


Figure 3: Deep learning framework. Images are first processed through a convolutional neural network to extract key features in an image (CNN part). These features are then combined (concatenated) with the features that are represented in the tabular data (rainfall, location, darkness, SWIR info) and processed through a fully connected neural network (Merged Part, MP).

$224 \times 224 \times 3$ images as input. Therefore, we only look at the tail structures of these architectures; when the data has similar dimensions as our input image ($10 \times 10 \times 3$). For both networks, we see that, at the end, they use three convolutional layers (kernel size 3×3) followed by a single max pooling layer, that halfs the size of the data, and two fully connected (FC) layers. We use a similar structure for the CNN part of our model. Next, we discuss the parameters in our proposed CNN part in more detail.

First, we look at the convolutional layers that convolve the images. For both Alexnet and VGGNet, we note that, many filters are used in the tail of the network. In their situation, this is the tail of the network (they start with larger images), where for us it is the whole network. Using such a large number of filters increases the amount of parameters that needs to be learned significantly. Therefore, these known structures may contain too much filters/neurons to obtain good estimates of the speed. We decide to start with 32 filters in the first convolutional layer and increase this amount to 64 in the second and third convolutional layer.

Secondly, we discuss the amount of neurons in the two hidden layers in the FC part of the CNN. Both Alexnet and VGGNet end up with a (flattened) layer of a certain amount of neurons. Then, VGGNet use $\frac{1}{5}$ of the amount, and Alexnet use $\frac{1}{2}$ of the amount of neurons. In this research we consider these two possibilities and one possibility in-between as possible amount of neurons in the hidden layers, *i.e.*, fractions of $\frac{1}{5}$, $\frac{7}{20}$ and $\frac{1}{2}$. After the two hidden layers, we end the CNN part with an output layer. In a way, we see the number of neurons in this output layer as the amount of information that is retrieved from the satellite image (regarding travel speed prediction). Examples could be, the color of roads, the shape of roads, or nearby buildings. We use an output layer of a similar amount of neurons as the amount of input neurons that correspond to the tabular data. We do this to ensure that, at the start of training, equal importance is given to both input sources of the merged part.

Finally, the goal of our paper is not to classify images, which is the goal of Alexnet and VGGNet. Instead, we want to extract features that can be used in another part of our model. Therefore, we do not use a Softmax activation layer at the end, but a ReLU (rectified linear unit) activation function to activate neurons. Note that by using ReLU, we increase the non-linearity of the network without affecting the receptive fields of convolution layers. The structure for our CNN is tabulated in Table 5.

Layer		# Filters	Size	Kernel size	Stride	Activation
Input	Image	-	$10 \times 10 \times 3$			
1	Convolution	32	$10 \times 10 \times 32$	3×3	1	ReLU
2	Convolution	64	$10 \times 10 \times 64$	3×3	1	ReLU
3	Convolution	64	$10 \times 10 \times 64$	3×3	1	ReLU
4	Max Pooling	64	$5 \times 5 \times 64$	3×3	2	ReLU
5	Flatten	-	$1,600$	-	-	ReLU
6	FC	-	$1,600 \cdot \{\frac{1}{5}, \frac{7}{20}, \frac{1}{2}\}$	-	-	ReLU
7	FC	-	$1,600 \cdot \{\frac{1}{5}, \frac{7}{20}, \frac{1}{2}\}$	-	-	ReLU
Output	FC	-	8	-	-	ReLU

Table 5: Tabular form of the CNN structure that is used to process the $10 \times 10 \times 3$ satellite images of the roads.

Merged part (MP)

Next to this CNN part where we examine the satellite image of the road, we also have the tabular data we want to include (location (lat/lon), rainfall, darkness, SWIR info). We want to combine this information with the information that resulted from the CNN. Therefore, we concatenate the output layer of the CNN with eight new neurons that represent this tabular information (note that the location will use two neurons, one for latitude and one for longitude). Now, we have 16 entries that we can use in a regular feed-forward neural network to predict the average travel speed (one output neuron). We use one hidden layer in-between to capture any nonlinearities in the prediction model. In this layer, we start with an input layer that consists of 16 neurons. There is an output layer of a single neuron, which represents the travel speed of the road under specified circumstances. Therefore, in the hidden layer we need to find a way to convert the input layer information to a prediction for the speed. For this, we examine different structures, from a very tight to a very wide structure. Specifically, we examine sizes (amount of neurons) that are powers of 2: from 2^i for $i \in \{6, 8, 10, 12\}$.

As in the CNN part, in all the layers in this Merged Part, we use ReLU activation functions to activate neurons.

Remaining settings

We conclude this section by discussing all (additional) choices regarding the architecture of the proposed deep learning structure. First, in order to prevent overfitting, and increase the chances of learning general features, we use dropout (Srivastava et al. (2014)) in all the layers in the network. Dropout ignores a portion of the neurons during each training batch update. Each training batch will use a different combination of neurons that they can use. In this way, neurons are more likely to work together to generalize to unseen data. During prediction, all units are used. This becomes essentially an ensemble of all the different combinations of units that were used. Note that one may also consider reducing the amount of filters/neurons in the layers. However, this approach will only learn what those fewer units can be optimized for, which would deteriorate the performance of the prediction model. Considering our deep learning structure, there are two types of layers for which we can apply a dropout layer; convolutional and fully connected layers. Convolutional layers typically have a lower drop-out rate, as there are less parameters to tune (and therefore less risk for overfitting). In our case, we examine the values 0 and 0.25. For the fully connected hidden layers, we typically have more neurons/parameters to fit, which makes this more prone to overfitting. The highest number of neurons in a hidden layer is in the CNN part. Therefore, we also use a higher dropout rate in this part;

0.5 or 0.75. In the hidden layer of the merged part of the network, we have a large range of possible values (very wide to very tight). Therefore, we examine dropout rates of 0.25, 0.5 and 0.75. Finally, note that we will not apply dropout to the input layer of the merged part. All these eight neurons should play a role in determining the traffic speed. Missing one (or more) neurons could deteriorate the performance significantly.

The second choice we make is to use the Adam algorithm (Kingma and Ba (2014)) to optimize the weights and biases of the deep learning structure. For this algorithm, we use the default settings recommended by Kingma and Ba (2014), with the exception of the learning rate. Choosing the learning rate is crucial, as setting it too small may lead to very slow convergence, while setting it too large may cause the algorithm to overshoot the optimal value, which may prevent the algorithm from converging. We consider three different values: the default value of 0.001, along with alternative values of 0.002 and 0.003.

Finally, there are two parameters that define how long training takes: the amount of *epochs* and the *batch size*. The amount of epochs is the amount of times the observations in the (training) dataset are used to improve the prediction model. The batch size refers to the amount of times the training dataset is split in sub-datasets, which are subsequently all used to improve the model. So, the more batches, the more opportunities the model has to improve, but also the less accurate possible improvements could be (*e.g.*, tuning the model based on a single observation). Regarding the amount of epochs, we overestimate this number at 50 and let the model stop early when the performance is not improved over the last 5 epochs. In this way, we also prevent the model from over-fitting the training data. The best model is then the latest model that improved the performance on the validation dataset.

Regarding the batch size, we want a batch size that is representable for the whole dataset, so not too large and not too small. Moreover, we would like to use batch sizes that *exhaust* the training dataset (we want to use *all* training samples in one epoch). This means that we can approximate the batch sizes with numbers that are equal to 2^x for $x \in \{0, 1, 2, ..., 14\}$. Out of this set, we choose to examine values for x between 7 and 10.

Note that for several parameters we proposed *possible* values, instead of a single number. These parameters are called the *hyperparameters* that still need to be set before solving the model. We will perform a *grid search* on these combinations of parameters. In other words, we test combinations and choose the one that performs best on the validation dataset. A summary of the hyperparameters that are examined, along with their possible values is shown in Table 6.

These possibilities result in 1,728 different settings that we examine for a given set of features (*e.g.*, using the rainfall in the last hour).

4 Numerical results and conclusions

This section presents a case study conducted in collaboration with PemPem. We start by describing the data and any preprocessing steps performed (Section 4.1). In Section 4.2, we discuss the proposed (best) model in more detail. Then, in Section 4.3, we evaluate the proposed model on unknown test data to examine its performance. In Section 4.4, we analyze predictions that are made with the proposed model. Finally, in Section 4.5, we perform several ablation analyses, investigating the usefulness of various information sources in this case study.

	Possibilities
CNN Part (CNN)	
Dropout % per convolutional layer	[0, 0.25]
# neurons hidden layers	$x \cdot 1,600$ for $x \in \{\frac{1}{5}, \frac{7}{20}, \frac{1}{2}\}$ $= [320, 560, 800]$
Dropout % hidden layers	[0.5, 0.75]
Merged Part (MP)	
# neurons hidden layer	2^x for $x \in \{6, 8, 10, 12\}$
Dropout % hidden layer	[0.25, 0.5, 0.75]
Learning rate	[0.001, 0.002, 0.003]
Batch size	2^x for $x \in \{7, 8, ..., 10\}$

Table 6: Hyperparameters of the proposed structure.

4.1 Data and preparation

The goal of PemPem is to make Enterprise Resource Planning (ERP) systems available to help smallholder farmers transition out of an informal cash-based economy, into a digital, cashless and technology-based economy (PemPem (2023)). This case study involves enhancing the travel speed information in existing road networks, which can be used for routing decisions. In this study, we focus on a region in Sumatra (Indonesia) where PemPem is currently active. In this region, PemPem has access to the information in OSM, but this information turns out to be incomplete. Recall that the information about the speed limit is available for less than 5% of the roads (which cover less than 4% of the total amount of kilometers) in the relevant road network. Recently, they equipped trucks with GPS trackers, generating streams of GPS traces of where a truck has been driving, at what time, and with what speed. The frequency rate of the trackers varies, most of the time, between 10, 30 and 60 seconds. We want to extend the current road network information (the information in OSM) with the information in the GPS trajectories. In Stienen et al. (2022), the geometries of the network are extended. In this paper, we include information about the travel speed of roads under different circumstances.

The GPS trajectory information that is available is received between January, 2020, and June, 2021. As discussed in Section 2, we process these GPS trajectories through the algorithm developed by Stienen et al. (2022). After preprocessing the output of the algorithm, we end up with 184,160 speed registrations, which cover 17,493 different roads in the area. This data is subsequently augmented with information about the weather and satellite imagery (see Section 2). For the weather, we use information from OpenWeather (OpenWeather (2022)). We specify the center point of our region of interest and retrieve the hourly weather that includes the rainfall in the last hour. This information is then combined with the speed registration data. For the satellite imagery, we utilize Sentinel-2A data obtained through Google Earth Engine (Gorelick et al. (2017)). More specifically, we use pictures of the road that are taken between January, 2020, and June, 2021.

To recommend a model based on various performance evaluations, we first divide the augmented dataset in a training, validation, and test (data)sets. The training set is utilized to train the model (the deep learning network), this information is

made available for optimization. The validation set is used for comparing hyperparameter configurations. Once a model is created using the training data, its performance is evaluated on the validation set. Finally, the model that performs best on the validation set is selected, and its performance is assessed on the test set. So, to evaluate the performance of the proposed procedure, we exclusively rely on the test set, which is never utilized before the prediction phase.

The aim is to develop a model that also performs well on travel speed prediction tasks on roads that did not occur in the GPS trajectory dataset. We refer to these roads as *unknown* roads. As an example, if we know the travel speed of a given road during heavy rainfall, we may be able to predict the (unknown) travel speed of this same road during moderate rainfall in a better way than predicting the travel speed on a road that does not exist in the GPS trajectory data (unknown road). Therefore, to create the test set, we take 10% *of the roads* in the data set that we consider as unknown roads. Note that for a single road, multiple circumstances may have been observed. This means that this part of the test set contains more than 10% of all data points. Additionally, of the remaining data points, we take a random 10% of the observations that is used to determine the performance of the model on known roads, but under different (unknown) circumstances. The remaining ($\pm$80%) data points are used for the training and validation dataset. We choose to use 80% of this remaining data as our training data and 20% of as our validation data. In this case, we do *not* ensure that the validation set contains a large variety of different roads, as these are not used for performance evaluation (they are only used to differentiate between hyperparameter configurations). To ensure fair performance evaluation of our model, we intentionally select only unknown roads for the validation set. This approach prevents our model from memorizing features of known roads and focuses on predicting travel speeds for unknown roads (which is a more difficult task).

To evaluate the results, we use two metrics: the Root Mean Squared Error (RMSE) and the Mean Absolute Error (MAE). The RMSE is computed by taking the square root of the mean of the squared differences between observed values (σ_i) and corresponding predicted values ($\hat{\sigma}_i$) for each prediction $i = 1, ..., n$. MAE, on the other hand, is calculated by taking the mean of the absolute differences between observed and predicted values. The mathematical expressions of these metrics are given below:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\sigma_i - \hat{\sigma}_i)^2}, \qquad MAE = \frac{1}{n}\sum_{i=1}^{n}|\sigma_i - \hat{\sigma}_i|.$$

Both metrics quantify the average difference between predicted and observed values. The RMSE, however, incorporates a higher penalty for outliers due to squaring the differences. On the contrary, the MAE treats all errors equally, without emphasizing outliers.

During the optimization of the weights and biases of our deep learning structure, we utilize the RMSE as our performance metric. Consequently, we adjust our early stopping criteria, such that early stopping is only applied if our current solution is performing better than the solution that always predicts the mean of all validation samples. Let y_i represent the individual samples in the validation set. Predicting the mean value, denoted as μ, yields the following RMSE:

$$RMSE(\mu) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \mu)^2}, \text{ where } \mu = \frac{1}{n}\sum_{i=1}^{n} y_i.$$

Note that this RMSE value is equivalent to the standard deviation of the values y_i for $i = 1, ..., n$ in the validation set. Therefore, early stopping is only triggered if the validation RMSE is smaller than the standard deviation of the y_i values.

4.2 Obtaining the *best* model configuration

Next, we discuss the results of the grid search process. As mentioned above there exist several (hyper)parameters that can be fine-tuned to optimize performance. To identify the *best* configuration, we perform a grid search in which we use a validation set to compare configurations. Moreover, we use three different seeds, where each seed corresponds to a different training/validation/test set. Based on the average performance over the different seeds, we compare the model configurations. We use an RTX 4090 GPU to conduct our experiments. The best model configurations are shown in Table 7.

	Possibilities	**Choice** (#1 (#2))
CNN Part (CNN)		
Dropout % per convolutional layer	[0, 0.25]	0.25 (0.25)
# neurons hidden layers	$[320, 560, 800]$	320 (560)
Dropout % hidden layers	[0.5, 0.75]	0.5 (0.5)
Merged Part (MP)		
# neurons hidden layer	2^x for $x \in \{6, 8, 10, 12\}$	2^{12} (2^{12})
Dropout % hidden layer	[0.25, 0.5, 0.75]	0.5 (0.25)
Learning rate	[0.001, 0.002, 0.003]	0.001 (0.003)
Batch size	2^x for $x \in \{7, ..., 10\}$	2^7 (2^8)
Average MSE (validation sets)		**72.8** (73.9)

Table 7: Hyperparameters of the proposed structure with the *optimal* choice of each of the hyperparameters in the columns on the right. Between parentheses, we show the second best performing model configuration.

We first observe that the second best model configuration (between parentheses) is very similar to the best model configuration. This, in turn, also holds for the third best, fourth best, etc. models. The model configuration is therefore considered to be robust in its parameter choices. In the rest of this section, we show results for the model that resulted in the lowest average MSE. Similar results are obtained for the other models, which can be run using the code on Github.

So, the Choice (#1) model is the model architecture that we propose to use for predicting the travel speed on different roads under different circumstances. In the rest of this section, we perform additional analyses using this model. For the sake of reproducibility and to facilitate the creation of visualizations, we rerun the model on a local CPU. Subsequently, all analyses are carried out using this model. It is important to note that slight variations in results might occur between the GPUs and the CPU due to differences in hardware and sources of randomness. As a result, a model could be *optimal* on the GPU but not on the CPU and vice versa. However, the differences are expected to be small.

We start with examining the learning curves of the proposed model, which are shown in Figure 4. Note that these curves are represented as bands, as we have three different validation sets (three different seeds) that all have corresponding learning curves. The bands are the convex hulls of the three different curves. The corresponding final training and validation losses (MSE's) are shown in Table 8. For clarity, we also show the Root Mean Squared Error (RMSE).

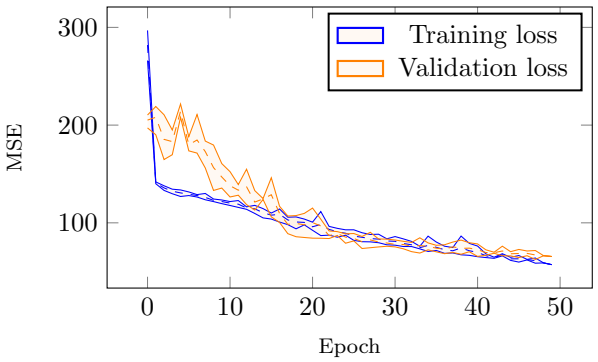


Figure 4: Learning bands for the best model using three different seeds.

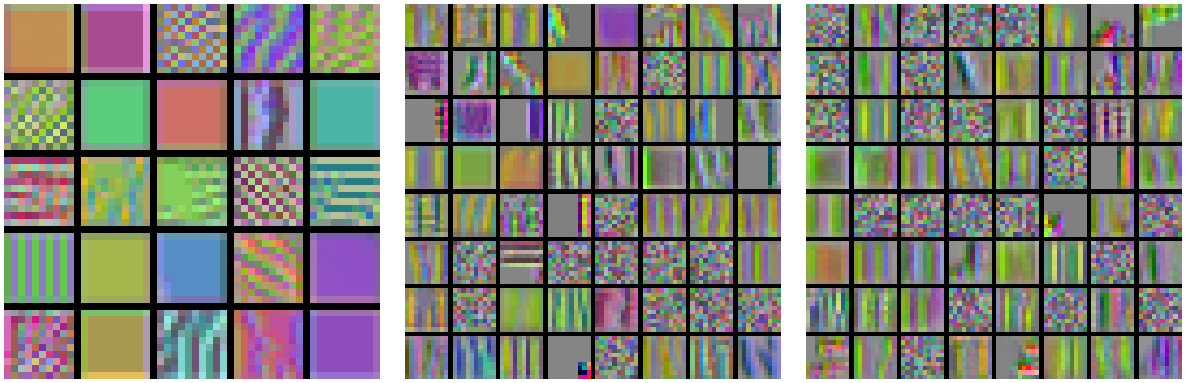
	Training loss		**Validation loss**	
	MSE	RMSE	MSE	RMSE
Seed 0	80.3	9.0	77.0	8.8
Seed 1	58.7	7.7	62.5	7.9
Seed 2	64.8	8.1	69.9	8.4
Average	**68.0**	**8.2**	**69.8**	**8.4**

Table 8: Final training and validation losses for the best model using the three different seeds.

Based on Figure 4 and Table 8, we conclude that the model does not overfit the data. The validation loss is approximately similar to the training loss, and we do not see a divergence between the two learning curves.

To get a better insight in the proposed (best) model, we analyze the model a little further. We choose to do the analyses using the model obtained with Seed 1, as this model resulted in the lowest validation loss.

We start with visualizing the optimized weights of the filters of the CNN part of our network. Note that we have three convolutional layers, which means that we have three sets of optimized filters with sizes 32, 64 and 64 respectively. For each filter, we create images that maximize the activation of this filter. This image could therefore be seen as a visualization of the pattern to which that particular filter responds to. In Figure 5, we show for each convolutional layer (a part of) the images that maximize the activation of the filters.



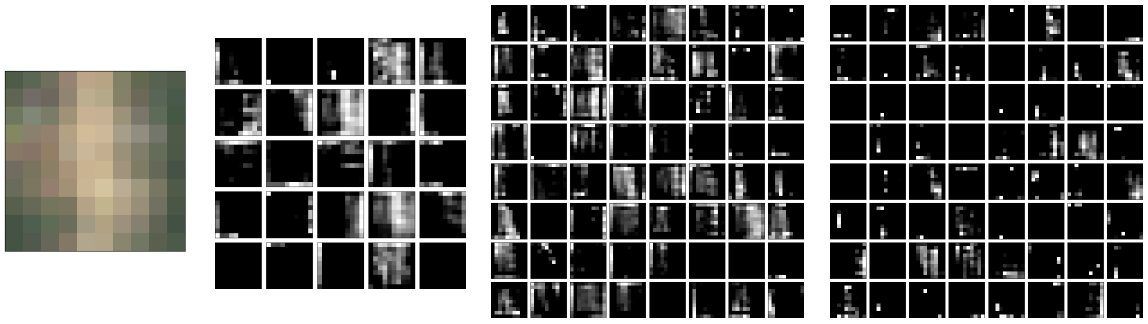
(a) Layer 1: 25 out of 32 filters. **(b)** Layer 2: 64 out of 64 filters. **(c)** Layer 3: 64 out of 64 filters.

Figure 5: Images of the filters.

We observe that in the first layer, the network seems to extract colors and some textures from the image. In the second and third layer, the network searches for more complex patterns in the images. Also, we see that there are still some noise filters (squares with seemingly random color dots) present in the second and third layer. This might indicate that we could reduce

the amount of filters used.

Secondly, we look at images of roads that are convolved through the CNN part of our network. We look at how the image is constructed out of the optimized filters. We expect that in the first layer of filters, we can recognize the most elements of a road. The model extracts the features into more general concepts when getting deeper into the network. Therefore, the images may become more and more abstract. For a specific road, we show a selection of convolved images in Figure 6.



(a) Layer 1: 25 out of 32 filters

(b) Layer 2: 64 out of 64 filters

(c) Layer 3: 64 out of 64 filters

Figure 6: Convolved figures for an example road.

We indeed observe that in the first figure some bright parts of the image of the road can be recognized. Note that for this we only look at the filters that contain many bright part. The other filters are apparently not active for this specific image of a road. In the second and third layer, we may recognize some parts of edges, which seem to represent more general concepts. We also see pictures in which we may not immediately recognize any features, which are therefore also more likely to represent more general and abstract concepts.

4.3 Evaluating the proposed model on test data

In this section, we examine the performance of the proposed model on the test data. For this, we use the two types of test data we mentioned before. First, the test data that consists of roads that are, in different circumstances, present in the training set (known roads), and secondly, the test data that consists of roads that are not seen during training or validation (unknown roads). The results of the analyses are tabulated in Table 9.

We observe that the performance of the model on known roads is slightly better than the performance on the unknown roads in the test data. This makes sense, as if we know how one drives on a specific road under specific circumstances, we may be able to better guess the speed one may drive on this same road under different circumstances. However, the performance gap between known and unknown roads seems relatively small. This suggests that the model did not overfit the training and validation data, and is able to predict unknown roads efficiently compared to known roads.

Next, we look at where the model makes wrong predictions (without distinguishing between predicting known and unknown roads). We look at the distribution of speeds that are predicted and observed. Again, we specifically do this for the model with seed 1. The result is shown in Figure 7.

	Known roads			Unknown roads		
Test data	MSE	RMSE	MAE	MSE	RMSE	MAE
Seed 0	84.6	9.2	7.4	83.9	9.2	7.2
Seed 1	56.1	7.5	5.8	66.0	8.1	6.3
Seed 2	63.8	8.0	6.4	75.8	8.7	7.1
Average	**68.2**	**8.2**	**6.5**	**75.3**	**8.7**	**6.9**

Table 9: Average travel speed prediction results (based on 3 different test sets, 3 seeds) on the test dataset, containing known and unknown roads.

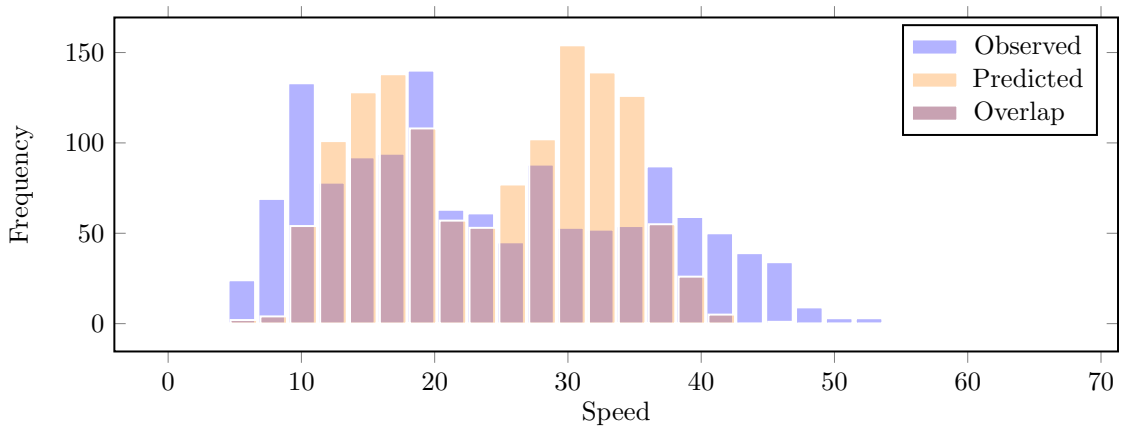


Figure 7: Frequency plot of the observed (blue) and the predicted (orange) travel speeds.

We observe that slightly less predictions are made about speeds in the right tail of the speed distribution. The predictions seem more conservative and more towards an *average* travel speed, instead of outlying predictions, which is, in practice, a good solution.

4.4 Evaluating predictions made with the proposed model

Next, we use the proposed model to make a prediction for the roads in the relevant road network. In this section, we discuss these *predictions* made with the proposed model. One of the key features of our approach is that we can predict the speed one can drive on every road in the area, including those that are not in the dataset used. All information, the location, the satellite image, the rainfall in the last hour, and whether it is dark, is available. In this section we analyze the predictions for every road in the area in which PemPem operates (the region within Sumatra, Indonesia) and discuss the results.

First, we examine the variation in predicted speed between circumstances for different roads. This gives us insight in whether there is a large difference to be expected when driving in different circumstances. Moreover, it gives us the opportunity to assess the necessity and impact of including all the information currently utilized in our predictions. Note that, due to the limited amount of data, we are not able to analyze the variation between different circumstances for the *observed* speed. Insufficient

speed observations are available for most of the road-circumstance combinations. In Figure 8, we plot the maximum percentage differences in predicted speed for every road under all different circumstances. Specifically, we have focused on displaying only the cases where the difference exceeds 0%.

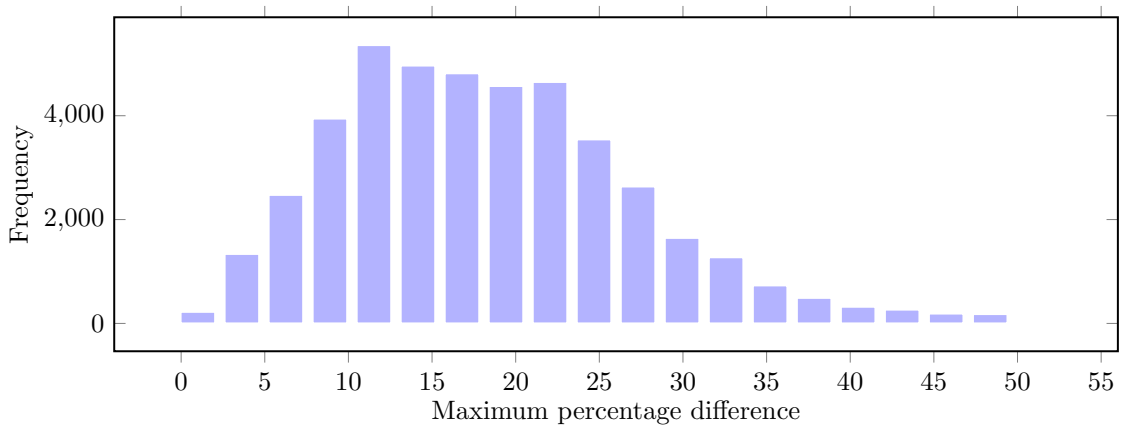


Figure 8: Frequency plot of the maximum percentage differences between different driving circumstances.

We observe that most of the maximum percentage differences are between the 10 and 35%. There are some roads for which the maximum percentage difference increases to between 40 and 50%. These are often roads with already a smaller travel speed. It is worth mentioning that the maximum absolute difference we found was slightly below 10 km/h, which indicates a relatively small variation in speed between different circumstances within the dataset for this case study. In Section 4.5, we investigate this observation in more detail.

Next, we look a little deeper in the roads that have a high variation in the predicted speed between circumstances. We divide the roads in three categories: roads with a low, moderate, or high variation in speed between circumstances. The roads with a high variation are therefore the roads that are expected to be more vulnerable to changes in the circumstances. The variation will be measured by the standard deviation of the predicted speed of a road over the six different circumstances. In Figure 9, we show a heatmap of the average speed prediction under different circumstances, divided over the three categories of variation.

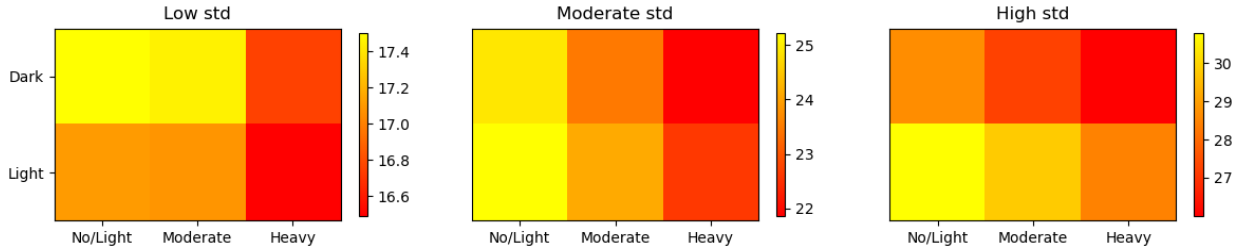


Figure 9: Average speed predictions across circumstances, categorized by variation (standard deviation, std) in road speeds (left: low variation (std < 1.5), middle: moderate variation (std $\in[1.5,2.5]$), right: (std > 2.5)). Vertical axes: whether it is dark or not; Horizontal axis: previous rainfall in the last hour.

We observe that for the heatmap on the left, the predicted speeds are very close to each other. Although it seems that the

speed prediction for when it is dark is larger than the speed prediction during the day, this effect is negligible. For the two plots on the right, the heatmaps that corresponds to the roads with the higher variations, we do see a similar, more significant, effect that indicates that driving after heavy rain, and driving in darkness, have a negative effect on the predicted speed. The speed is predicted to be lowest when both of these phenomena occur, *i.e.*, driving in darkness after heavy rain. Still for these right plots, the speeds are not very far apart, which again indicates a relatively small variation in speed between different circumstances within the dataset for this case study.

Thirdly, we conduct an analysis on the variations in travel time between actual origin-destination pairs (OD pairs) based on different circumstances (whether it is daytime and how much rain has fallen in the past hour). To perform this analysis, we utilize the dataset of 14,846 OD pairs from GPS trajectories, as mentioned and used in Stienen et al. (2022). We choose to only consider the routes that have a shortest distance of at least 2.5 kilometers. In total, this means that we consider 8,405 different OD pairs. For each circumstance, we calculate the optimal travel time for these OD pairs by considering the individual travel times of each road in the specific circumstance. Subsequently, we compare these times with the travel time required to drive the route that is optimized based on distance alone. Note that this is currently the conventional method for determining an optimal route. The resulting differences between these times, in percentages, observed across all OD pairs, are presented in Figure 10. In our analysis, we consider only those routes that have a strictly positive difference in at least one of the six circumstances (2,495 OD pairs).

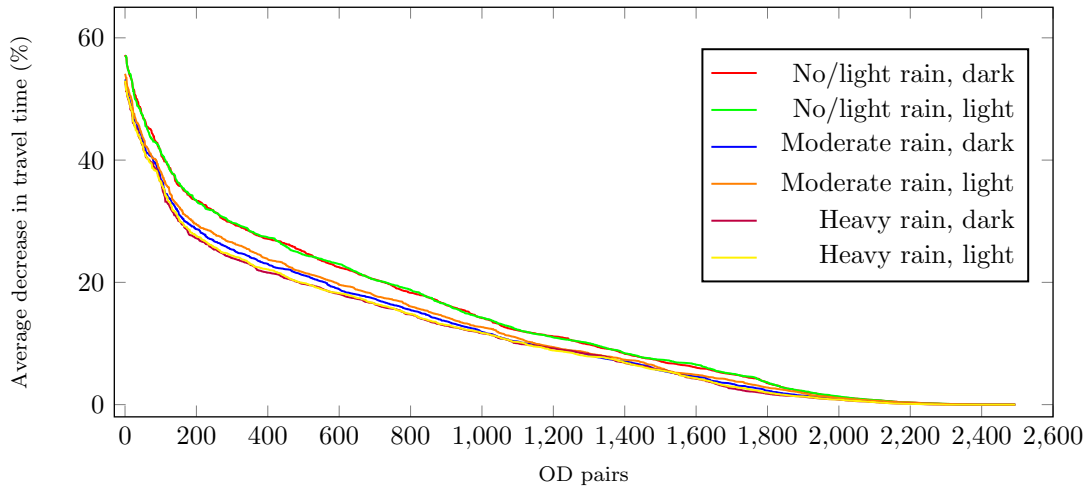


Figure 10: Comparison of travel time differences for various origin-destination (OD) pairs.

For the considered routes, we observe an average decrease of 12.3% in travel time when using the route based on travel time optimization instead of using the route based on distance optimization. Moreover, we observe that, for the considered routes, over 23.8% of the OD pairs, an average travel time decrease of at least 20% can be expected when using the route that is optimized based on travel time.

Moreover, the circumstances with the lowest travel time per road (*e.g.*, when there is *no/light rain* in the last hour) seem to experience a higher percentage decrease in travel time. We see a similar effect when we look at the absolute differences. First, this indicates that the travel time of the route that is optimized based on distance is closest to the travel time that is obtained

when optimizing the travel time in a situation with higher travel times per road (*e.g.*, when there is *heavy rain* in the last hour). Second, this also suggests that there are roads that can be traversed relatively quick under these favorable conditions, changing the usual travel times one would expect based on their length.

We do note, however, that the differences between the circumstances are again small, and that this also confirms what we saw in the previous figures, that there is only little difference between the different circumstances. In the next section, we will do an ablation analysis in order to examine the influence and impact of the different sources of information.

Finally, we also look at the average predicted speed, and see if we can observe a relation between this predicted speed and a map of the region. In Figure 11, we show for each road whether the predicted average speed is below 15 km/h (red edges), between 15 and 30 km/h (orange edges), and above 30 km/h (green edges).

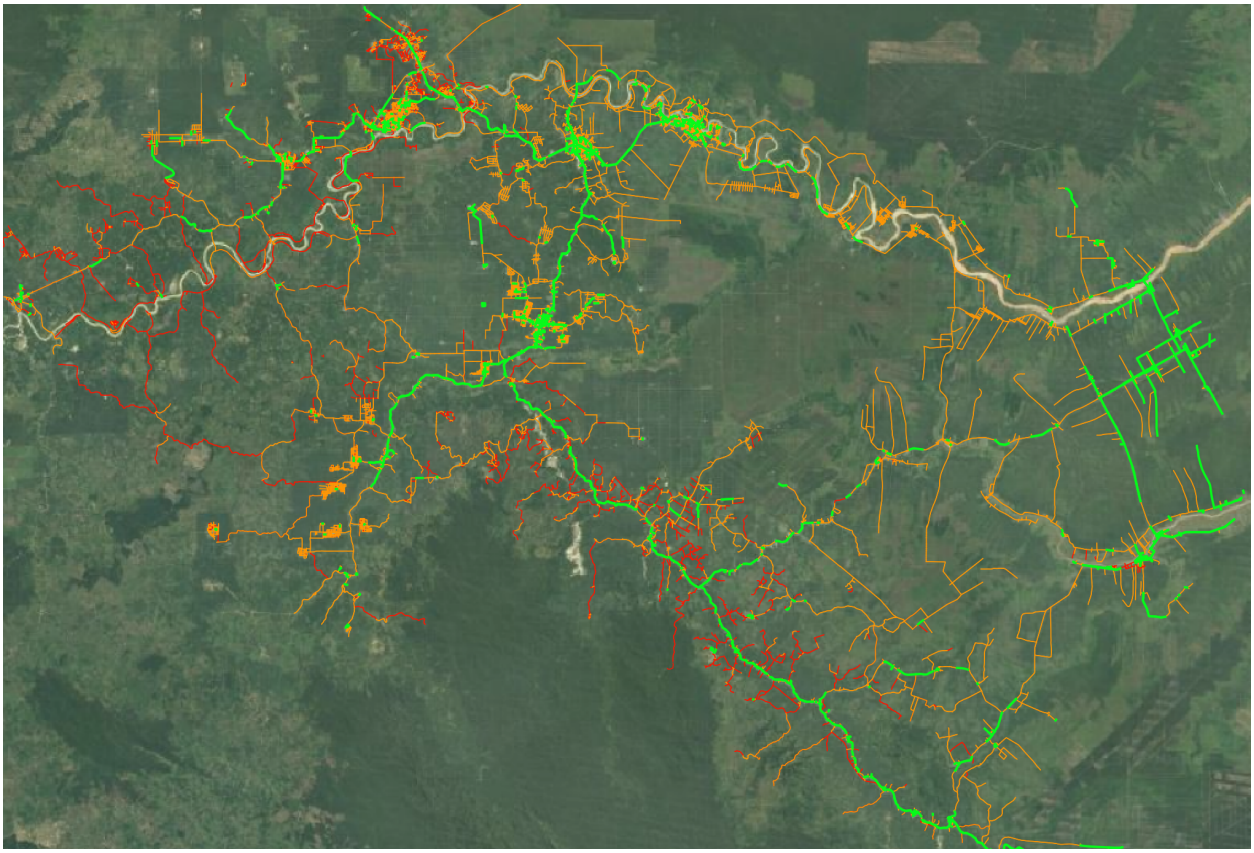


Figure 11: Speed predictions for the entire road network. red: Speed prediction $<$ 15 km/h, orange: 15 km/h $<$ Speed prediction $<$ 30 km/h, green: Speed prediction $>$ 30 km/h.

We observe that the green edges form indeed the core road network of the area. Then there are many orange edges, and a few clusters of red edges. Comparing these edges with the results presented in Stienen et al. (2022), we observe that edges that were not present in OSM before tend to be red. This makes sense, as these used to be unknown and are therefore expected to be of less quality. These observations are therefore in line with our expectations.

4.5 Ablation analyses

We conclude this section with several ablation analyses. As observed in the previous subsections, the inclusion of information regarding previous rainfall and darkness may have a negligible effect on the predictions. It seems that, in the dataset we use, there is not enough variation present between different circumstances to exploit the full potential of the proposed model. To further investigate this, we assess the impact of all the information used in making the predictions. First, we examine the effect of the satellite image of the road. Then, we determine whether including information about the location, the rainfall, the darkness, and the satellite information (SWIR information) play an important role in the predictive performance.

We start with the effect that the images of the roads have on the prediction accuracy. Note that choosing to use the same procedure as before, but using *blank* images, may result in overfitting the data, since we designed the network for the model including information about the images. Therefore, as a baseline model, we fit a linear model on the tabular data that is not obtained from the satellite image. Recall that this tabular data consists of the location (lat/lng), the amount of rainfall in the last hour, and whether or not it is dark outside. We refer to this set of features as the **Info** set. Additionally, we also examine whether the linear model performs any better when including the tabular information of the satellite image (minimum/maximum/mean SWIR). To increase possible performance, we also choose to add the average color of the image as tabular data of the satellite image. We refer to this set of features as the **Sat** set. In Table 10, we show the results obtained when combining these different sets in a linear model.

	Training loss		**Test loss**			
			Known roads		Unknown roads	
	MSE	RMSE	MSE	RMSE	MSE	RMSE
Original model	68.0	8.2	68.2	8.2	75.3	8.7
Info	126.0	11.2	131.4	11.5	129.3	11.4
Sat	117.5	10.8	120.8	11.0	124.0	11.1
Info + Sat	113.6	10.7	118.0	10.9	118.3	10.9

Table 10: Training and test losses when using a linear model to predict the average travel speed. The features of the linear model are the location, the rainfall in the last hour, and whether it is dark or not (**Info**), and the SWIR measurements and the average color (**Sat**).

We observe that without the satellite images, we perform significantly worse. Even if we include all the possible features, we only achieve an MSE of 118.0 for known and 118.3 for unknown roads in the test set. This means that the original model indeed uses information in the image that is not only color. This may be the width or the amount of buildings in the neighborhood.

Next, we examine the effect of the other information sources, while we keep using the satellite image of the road. We distinguish six different scenarios, which are characterized by including/excluding specific parts of the available information. The different scenarios are shown in Table 11.

For each of these scenarios, we determine the optimal weights and biases using the previously obtained optimal architecture from the grid search, which was based on using all available information. Subsequently, we evaluate the performance using the test set.

Symbol	Description
A	Do not include the location info
B	Do not include the rain feature
C	Do not include the dark feature
D	Do not include the dark and rain feature
E	Only include satellite image and info
F	Only include satellite image

Table 11: Scenarios and descriptions of different information sources used for predicting.

Additionally, we re-do the same grid search procedure for each scenario to identify the *optimal* architectures using the MSE as the evaluation metric, just as we did previously. For this grid search, we now use various RTX 3090 and 4090 GPUs to conduct our experiments. Then, we evaluate the performance of the best models on the test set by running them on a local CPU machine. Recall that slight variations might occur between the results obtained via the GPUs and the CPU. Similar to previous procedures, we carry out all calculations with three different seeds and report the average results. The results, for all the scenarios, are presented in Table 12.

		Training loss		**Validation loss**		**Test loss**			
						Known roads		Unknown roads	
		MSE	RMSE	MSE	RMSE	MSE	RMSE	MSE	RMSE
Original model		68.0	8.2	69.8	8.4	68.2	8.2	75.3	8.7
A	Filled in	73.6	8.6	73.3	8.6	74.5	8.6	76.8	8.8
	Optimal	92.7	9.6	84.1	9.2	89.3	9.4	87.1	9.3
B	Filled in	62.3	7.9	67.6	8.2	63.5	7.9	69.4	8.3
	Optimal	67.1	8.2	67.0	8.2	69.1	8.3	69.3	8.3
C	Filled in	66.6	8.2	70.3	8.4	67.6	8.2	73.9	8.6
	Optimal	68.7	8.3	71.0	8.4	69.2	8.3	72.2	8.5
D	Filled in	77.5	8.8	74.6	8.6	73.8	8.6	76.9	8.8
	Optimal	76.0	8.7	74.5	8.6	75.2	8.7	76.2	8.7
E	Filled in	82.0	9.1	78.1	8.8	81.9	9.0	82.6	9.1
	Optimal	115.2	10.7	108.5	10.4	114.5	10.7	113.3	10.6
F	Filled in	80.3	9.0	82.0	9.1	82.3	9.1	83.8	9.2
	Optimal	76.8	8.8	75.5	8.7	74.0	8.6	79.0	8.9

Table 12: Comparison of predictive performance in different scenarios with varying information sources.

One of the first phenomena that draws the attention is that, in some cases, the performance of the optimized architecture is worse than the performance we obtain if we use the original architecture, such as in scenario E. Further analyses show that this issue indeed comes from the different sources of randomness between the GPU and CPU environments. Once we obtain the optimal model through the GPU-based grid search, running it on the CPU results in the model rapidly getting stuck in a local optimum. It seems that this is most obvious in scenario A and E, in which we exclude the location information. A potential reason for this could be that these location variables play a crucial role, and their omission might lead to an inadequate model structure itself.

The second phenomenon observed is that similar results can be obtained by using only a subset of the available information in the data. In scenario B and C, where we exclude the rain, and darkness feature respectively, we even achieve slightly better results on the test set. However, considering the practical implications and relating these numerical differences to actual driving speeds, such small variations seem relatively insignificant. Note that including all information does lead to one of the lowest MSE, indicating that including the information does not worsen the performance.

As hypothesized earlier, the findings in this table confirm that, in this specific case study, we may omit the information about rainfall in the last hour, or the information about daylight and still achieve similar results. However, it is important to acknowledge that in a case study with significant variation in speeds under different circumstances, it is highly likely that using all information that is available performs best.

4.6 Conclusion

To conclude, the proposed method can be used to significantly enhance the information about travel speeds in road networks. We show that for a region in Indonesia, we are able to predict the travel speed with an average RMSE of 8.5 km/h. Potential lines for further research are to include additional features that may have an influence on the speed one is expected to drive on a given road. Moreover, the data that is used (the GPS trajectories) may (still) contain errors or inaccuracies. Further research may focus on obtaining more, accurate, GPS trajectories, in order to verify results. Finally, further research can focus on transfer learning, in which differences between road networks of different regions are compared and taken into account when creating new models.

References

Abdollahi, M., Khaleghi, T., and Yang, K. (2020). An integrated feature learning approach using deep learning for travel time prediction. *Expert Systems with Applications*, 139(C):112864.

BMKG (2022). Badan Meteorologi, Klimatologi, and Geofisika (BMKG), rain classification used. Accessed via `https://bmkg.go.id/cuaca/probabilistik-curah-hujan.bmkg?mm=50&hour=24&gen=k21gzb1b6hhghnqfgb#`. Accessed: 2022-11-15.

Brewer, E., Lin, J., Kemper, P., Hennin, J., and Runfola, D. (2021). Predicting road quality using high resolution satellite imagery: A transfer learning approach. *PLOS ONE*, 16(7):e0253370.

Cadamuro, G., Muhebwa, A., and Taneja, J. (2018). Assigning a grade: Accurate measurement of road quality using satellite imagery. *arXiv preprint arXiv:1812.01699*.

ESA (2022). Copernicus Sentinel data. Accessed through Google Earth Engine, processed by ESA. Accessed: 2022-08-08.

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., and Moore, R. (2017). Google Earth Engine: Planetary-scale geospatial analysis for everyone. *Remote Sensing of Environment*, 202:18–27.

He, S., Bastani, F., Jagwani, S., Park, E., Abbar, S., Alizadeh, M., Balakrishnan, H., Chawla, S., Madden, S., and Sadeghi, M. A. (2020). Roadtagger: Robust road attribute inference with graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10965–10972.

Huang, L., Yang, Y., Chen, H., Zhang, Y., Wang, Z., and He, L. (2022). Context-aware road travel time estimation by coupled tensor decomposition based on trajectory data. *Knowledge-Based Systems*, 245:108596.

Iddianozie, C. and Mcardle, G. (2021). Transferable graph neural networks for inferring road type attributes in street networks. *IEEE Access*, 9:158331–158339.

Jenelius, E. and Koutsopoulos, H. N. (2013). Travel time estimation for urban road networks using low frequency probe vehicle data. *Transportation Research Part B: Methodological*, 53:64–81.

Jia, Y., Wu, J., Ben-Akiva, M., Seshadri, R., and Du, Y. (2017). Rainfall-integrated traffic speed prediction using deep learning method. *IET Intelligent Transport Systems*, 11(9):531–536.

Jiang, Y. and Li, X. (2013). Travel time prediction based on historical trajectory data. *Annals of GIS*, 19(1):27–35.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Liu, Z., Li, Z., Li, M., Xing, W., and Lu, D. (2016). Mining road network correlation for traffic estimation via compressive sensing. *IEEE Transactions on Intelligent Transportation Systems*, 17(7):1880–1893.

Lv, Z., Xu, J., Zheng, K., Yin, H., Zhao, P., and Zhou, X. (2018). LC-RNN: A deep learning model for traffic speed prediction. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 3470–3476.

Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., and Wang, Y. (2017). Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction. *Sensors*, 17(4):818.

OpenStreetMap (2023). Road network (part of) Sumatra. Accessed: 2023-03-17.

OpenWeather (2022). Weather forecasts, nowcasts and weather history. Accessed: 2022-3-26.

PemPem (2023). `https://www.pempem.org`. Accessed: 2023-5-10.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Stienen, V., den Hertog, D., Wagenaar, J., and de Zegher, J. F. (2022). Enhancing digital road networks for better operations in developing countries. *CentER Discussion Paper*, 2022-014.

Tedjopurnomo, D. A., Bao, Z., Zheng, B., Choudhury, F., and Qin, A. K. (2020). A survey on modern deep neural network for traffic prediction: Trends, methods and challenges. *IEEE Transactions on Knowledge and Data Engineering*, 34(4):1544–1561.

Xie, Z., Lv, W., Huang, S., Lu, Z., Du, B., and Huang, R. (2019). Sequential graph neural network for urban road traffic speed prediction. *IEEE Access*, 8:63349–63358.

Yao, B., Chen, C., Cao, Q., Jin, L., Zhang, M., Zhu, H., and Yu, B. (2017). Short-term traffic speed prediction for an urban corridor. *Computer-Aided Civil and Infrastructure Engineering*, 32(2):154–169.

Yu, H., Wu, Z., Wang, S., Wang, Y., and Ma, X. (2017). Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. *Sensors*, 17(7):1501.

Zhang, J., Zheng, Y., Qi, D., Li, R., and Yi, X. (2016). DNN-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 1–4.

Zhou, Z., Yang, Z., Zhang, Y., Huang, Y., Chen, H., and Yu, Z. (2022). A comprehensive study of speed prediction in transportation system: From vehicle to traffic. *iScience*, 25(3):103909.