Vrije Universiteit Amsterdam

# Predicting the Color Change of Traffic Signals Using Image Processing and Machine Learning

Bachelor Thesis

**Author:**

**Ronan Hochart (2614393)**

**Supervisor:**

**Kees Verstoep**

**Daily Supervisor:**

**Kees Verstoep**

**Second Reader:**

**Henri E. Bal**

DATE 2023

# *Abstract*

In this thesis research is conducted on traffic light signals in the Netherlands to predict what color a signal will show in 3 seconds. This is done by gathering data from live intersections around the Netherlands and training classification models on the gathered data. This thesis will go into how the data was gathered and processed as well as which models were trained and their respective results. From this analysis it can be seen that a random forest classifier will have better results when compared with a logistic regression and a decision tree model in predicting what signal a traffic light will show in three seconds. The highest impact feature was observed to be the duration a signal stayed a certain color with vehicle presence and duration also having an observable impact. Further improvements are also discussed in both how to improve the current methodology by using regression based models as well as the possible different options for data gathering that would provide more encompassing and accurate data for future research.

# Chapter 1

# Introduction

## 1.1 Background

When living in the Netherlands and specifically in the provinces of North Holland and South Holland, traffic is no stranger to anyone trying to get to any destination via any vehicle. The high population density of these areas as well as all the tightly packed cities with numerous modes of transportation often cause a high level of interaction between vehicles throughout the day. To deal with this, an intricate network of roundabouts and intersections governed by traffic lights are used. However even if these systems, particularly traffic lights, are calibrated for the most efficient and maximum throughput of traffic it won't always feel that way on the ground. A common scenario that can play out is a vehicle moving towards a traffic light while it is green, and right before reaching it it flips to red, causing immediate frustration. The exact same situation can play out in reverse where right after slowing down to a stop the light flips green, causing a waste of energy as the vehicle has to re-accelerate to get moving again. This type of profile, namely the phase changing of green to red is called "Signal Phase and Timing" or SPaT and the focus of this paper will be attempting to predict an intersections SPaT profile a set amount of time into the future.[19]

Both of these negative side effects also have other larger repercussions. The former causing frustration or other negative emotions has a moderate link to driving capabilities as well as an illicit link to the safety of the driver and those around them.[12] The latter causing vehicles to decelerate to a full stop and immediately start moving again is directly linked to a waste in energy. For bikers this causes them to simply lose personal energy and get tired more quickly, but for drivers this has a much bigger impact on the environment as quickly decelerating and then accelerating has a significant effect on a vehicle's fuel consumption.[7] This results in using much more fuel than otherwise would be needed by a gradual stop, or simply never stopping at all and just slowly cruising to the light until it flips green.

This creates the possibility of a system to exist that can predict, based on contextual information of the environment presented, how long a traffic light will stay green or red or if a traffic light will flip its signal in the near future. This information can then be presented to a user so they know if they need to slightly speed up to make the light on time, or if they are better off already slowing down to conserve energy. This system can also be adapted to bikes, as they use a very similar system as road vehicles with traffic lights that can be seen ahead of time. The hopes of this system would be to lessen user frustration on the road, be it bike or car, and also lower energy waste. Both of these outcomes would make the road safer, more pleasant and lower the environmental impact of wasted energy constantly stopping and going at traffic junctions.

Research in this field is already being conducted in many different aspects, from attempting to have vehicles communicate directly with one another to having vehicles communicate with static infrastructure.[14][17] This is only possible due to the introduction of Intelligent Traffic Controllers (iVRI) to Dutch intersections that have the ability to monitor their environment and react to traffic flow as well as communicate in an IoT system with other devices on top of making their internal data available.[1] However these tasks often require high levels of access to either vehicles or the infrastructure they are trying to interface with, therefore this paper will dive into a more hands off approach which would not require any high level access and could be applied to any existing or future system.

To tackle this task, it can be split into three individual components. The first component will deal with data processing and gathering related to traffic lights and when they turn green or red; other contextual data such as the weather and time of day will be gathered as well. The second component will deal with training a machine learning algorithm based on the data gathered, so it is able to make predictions based on a set of input values. The final component would be on getting some simulated real-time input data and making a prediction based off of it. This final component could then be continued in the future with the ability to have real input data from advanced image processing of a live scene be used to complete the full system.

Another important aspect of this research is that it is to provide a proof of concept to the idea. Traffic lights will vary wildly depending on country, local environment and the date they were built in terms of their shape and types of light they emit. Additionally the perspective of the camera will have an impact on how the application is written in order to gather and process the data in real-time.[6] Taking these facts into account, this paper will focus on traffic lights found around the Netherlands. Specific intersections will be chosen to create a training data set and therefore the exact location of the traffic light will be predetermined and static, allowing for a simple crop to be used to isolate just the traffic light's state. This has the benefit of not having to design an algorithm to be able to pick out and recognize traffic lights among a chaotic image.

## 1.2 Research Questions

The aim of this research is to discover what factors, if any, impact the time between different lights turning on and off. This information can then be used to try and create a prediction algorithm in order to tell when a light will switch between red or green a set amount of time into the future. In order to do this, various machine learning algorithms will be tested and compared. The parameters used to tune these algorithms will also be tested as they can have a significant influence on the precision of the resulting model. The following research questions will be addressed:

- What set of features is the most optimal to create a precise model for predicting the future state of a signal?
- Which classification model is the best at predicting the future state of a signal?

## 1.3 Related Work

Being able to monitor and process traffic lights has been heavily researched in the past few years due to the advent of autonomous driving. This research is conducted to not only increase road safety but also to make it possible for autonomous vehicles to work at all. A considerable amount of the research done is based on the best method to gather and then process the data. The type of camera, the lighting conditions as well as the type of traffic light are all important factors to take into account for an algorithm to reliably be able to read the status of a traffic light.[6][21]These papers also describe the difference between the two major structures traffic lights are set up as, either suspended or supported. The former type is high in the air and has the advantage of a static background especially when viewed from a lower point of view. The latter type suffers from being low to the ground and having contrasting backgrounds. For the purpose of this paper only suspended traffic lights will be used.

Once the data and images have been collected, the next step is to make them usable for data processing. Different papers have used different methods to achieve this, such as splitting the RGB spectrum in order to more easily eliminate the surrounding environment[21] or creating a mask using color thresholding allowing for reliable outdoor video processing.[18]For this paper the latter method will be used and adapted by creating a mask focused just on the top part (red light) of a traffic signal. This mask will then be able to tell if the red light is on or off, which by definition also gives the green lights status. There is still the yellow light which can be taken into account, but since the yellow light only appears momentarily after a green light it can be ignored for the purposes of this research as it acts essentially as a green light in terms of being able to cross an intersection legally.

There is also a technically easier but still limited method of gathering this data and that is utilizing APIs. Due to the advent of more and more iVRIs being installed, their internal data is becoming more available. There are already papers using this access as a proof of concept for similar applications.[19][10] However the availability of this data is currently very limited and there are examples online that are no longer supported or possible to use anymore due to the lack of maintenance or the accessibility to this data being removed. This is a current limitation of this API based method that this paper will avoid by using image recognition.

# Chapter 2

# Image Processing

## 2.1 Foundation

The starting point of this project is gathering sufficient data to be used in training models using different machine learning algorithms. This means there is a need to translate a visual image of a traffic light into text data representing whether it is green or red, that can then be fed into an algorithm. Other data must also be added in order to give context to the traffic signal; this data will be the date, time, weather conditions, temperature, if a vehicle is present at the traffic light and for how long it is present for. Weather and temperature are both chosen to see if different weather conditions have an impact on traffic flow due to people preferring to use different modes of transport or different routes depending on the conditions. These choices would then have an impact on the traffic conditions and volume. The feature of a vehicle waiting and for how long is to observe any impact this might have on an iVRI's decision making to change the signal. After multiple models are trained they will be tested based on a series of metrics such as accuracy and the ROC-AUC on which can perform the best at predicting on if a traffic signal will flip three seconds into the future. Three seconds was chosen as it is the ideal maximum time for a vehicle to either start decelerating or keep its current speed to pass an intersection. [15][9] All of the code referenced in the paper can be found at a github repository with an additional sample dataset.[13]

## 2.2 Data Collection

In an ideal world, the data collection would be able to go through a direct API and get instant real time information of when a traffic light switches between red and green. However as this service is not widely available to the public, instead image processing had to be used. There exists research and projects based on iVRIs API interfaces however their access is typically restricted and have the risk of breaking if they are not maintained properly. A substantial amount of the work being done as well is focusing on vehicles directly communicating with the infrastructure in real time.[3]

Instead this project will be using image recognition, there were different options to getting the initial images. A camera could be setup and constantly record/monitor a traffic signal, but this can be difficult to do as you would need to find a safe spot to install a camera with good access to power as well as a steady supply of SD cards in order to keep storing the data. A simpler option that was found that will be used for this research, is finding relevant live-streams on *YouTube*. For a live-stream to be relevant it must have a traffic light in frame and be located in the Netherlands. Additionally, the higher resolution the image, and the closer the traffic light array is to the camera, the better. Once a live-stream was found that met these requirements a method had to be developed to gather its data.

The two intersections used for this papers research are both located in the Dutch province of Gelderland using a website showcasing live streams of multiple intersections throughout the province.[2] These two intersections were chosen because they represent both a three-way and a four-way intersection allowing for some research into if there is a difference or not in traffic signal behavior for these two types. The data for these intersections was gathered in 2022 for the three-way and 2023 for the four-way intersections. The three-way intersection contains two signals for the minor "T-Road" and three signals for each direction of the major "Through Road". The four-way intersection contains three distinct signals for each of its directions. However due to the limitation on how the data was gathered only one direction worth of signals is being analyzed in this paper.


(A) Sample image of the 3 way crossing used

(B) Sample image of the 4 way crossing used

FIGURE 2.1: Sample raw images taken from live webcam feeds on YouTube

### 2.2.1 Python Script

A python script was used to capture each frame from the live-stream, process it, and then categorize that frame as either a green light or a red light. The main library used to do the processing was *OpenCV2*. Once a frame was captured by the script, a predetermined crop was used to isolate the traffic light, specifically the red light at the top of the array. This small section of the frame was then converted from RGB to HSV and two different masks were created to isolate the red color. If the light was on, and red was present, a large mask would be formed, and if it was not the image would remain purely black. This worked reliably because the actual traffic array is black and has a high contrast with the light being emitted.

Additional data such as the weather and temperature were tagged along with the status of the light, and then the duration of the light was calculated once it had flipped to green or red. This created a string of information that was stored in a CSV file in the format seen in figure 2.1. This CSV file would be the basis of the machine learning process and what an algorithm would be fed in order to create a model.

| Date | Time | Duration of Light (Seconds) | Temperature °C | Conditions | Light Color |
|---|---|---|---|---|---|
| 5/26/2021 | 08-18-16.542 | 11 | 9.68 | broken clouds | Red |
| 5/26/2021 | 08-18-24.563 | 9 | 9.68 | broken clouds | Green |
| 5/26/2021 | 08-19-40.837 | 74 | 9.68 | broken clouds | Red |
| 5/26/2021 | 08-19-50.754 | 8 | 9.68 | broken clouds | Green |
| 5/26/2021 | 08-20-47.002 | 60 | 9.68 | broken clouds | Red |
| 5/26/2021 | 08-21-01.806 | 12 | 9.7 | broken clouds | Green |
| 5/26/2021 | 08-22-07.820 | 64 | 9.68 | broken clouds | Red |
| 5/26/2021 | 08-22-18.975 | 12 | 9.7 | broken clouds | Green |
| 5/26/2021 | 08-23-30.263 | 69 | 9.73 | broken clouds | Red |
| 5/26/2021 | 08-23-44.709 | 12 | 9.73 | broken clouds | Green |
| 5/26/2021 | 08-25-01.518 | 77 | 9.72 | broken clouds | Red |
| 5/26/2021 | 08-25-16.439 | 12 | 9.72 | broken clouds | Green |
| 5/26/2021 | 08-26-07.598 | 51 | 9.73 | broken clouds | Red |
| 5/26/2021 | 08-26-17.430 | 12 | 9.73 | broken clouds | Green |

FIGURE 2.2: First 15 entries in the CSV file

The basic structure and components of the python script are quickly summarized bellow.

- The *URL* was not a simple YouTube link, but a m3u8 link. This link is essentially the raw video stream and is needed for the Opencv2 library to take frames from the stream. However thanks to the pafy library this m3u8 link can easily be generated from a regular YouTube link.
- The *read* function reads from the video stream and creates a frame that will be the image the script processes.
- The *cvtColor* function will take the frame and transform it from RGB to HSV. This shift in color space is used to make the following process easier.
- The *inRange* function uses the cropped frame as well as a set of upper and lower bounds to be able to determine if there is red in the image. Two individual masks

are made due to red having two distinct hue saturation levels.[4] These masks are then combined for a unified mask which will be used to determine if the red light is on or off.

- The *findContours* function uses a specified algorithm to find the contours of a binary image file. All contours found are then returned as an image array.
- The *contourArea* function will compute a contour area given the image array provided. This area's size will be what determines if the light was on or off as a large area will signify a large amount of light was detected and therefore the light is active.
- The *norm* function will compute the similarity between two crops of the frame where vehicles are expected to be present. If the crops are similar no vehicle is present and if they are showing a high level of dissimilarity then a vehicle is present. These crops are updated periodically to ensure lightning and environmental changes don't impact the vehicle detection.

# Chapter 3

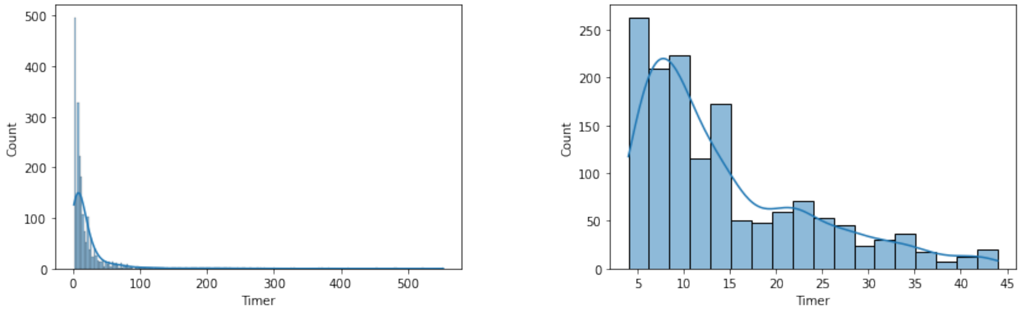
# Data Processing and Analysis

## 3.1 Outliers

With any data set there will be outliers that can have a negative impact on the accuracy of any model applied to the dataset[11]. It is therefore imperative to identify and remove the outliers without removing too much as to compromise the dataset as a whole. The data collected in this paper used live image monitoring. This means any bug in the system or random obstruction the camera feed (such as rain, a random bird or simply a unfortunately placed shadow) can create a false reading. These false readings will create abnormally large or small duration's for a particular color of light. However this also means they are quite simple to detect as they are not only rare but also very obvious when they deviate far from the average light duration.

For the purposes of this dataset, first the distribution of the data had to be identified. For this the distribution of the "Timer" value in the dataset was plotted and a skewed distribution was observed. This means an Interquartile Range (IQR) detection method can be used to detect and eventually trim out the outliers from the data. The way this method works is by taking the median value of the data and establishing two markers around it, named Q1 and Q3. If a feature from the dataset is viewed as a list then Q1 would be 25% through the list and Q3 would be 75%. The IQR is then the difference between Q1 and Q3. A lower bound can then be created from the Q1 marker value by subtracting the product of 1.5 with the IQR value. The same can be done for an upper bound by instead using the Q3 marker value and adding instead of subtracting it with the product of the same equation.[20][8]

Another example that would impact the viability of the data gathered is the actual recording of whether or not a vehicle was present at the intersection and waiting for




(A) Before outlier trimming (B) After outlier trimming

FIGURE 3.1: Distribution of the "Timer" feature in the data set

a light to flip. Initially the design of the python script and therefore data gathering used a subsection of a camera feed to detect whether or not a vehicle was present. This was done by comparing a reference image of the static background of the road to a live image, if the two images were seen as being too far apart in similarity, there must be an object in the frame and therefore a car is waiting or passing. However over time this was seen as not a practical solution as any object or obstruction would be labeled as a waiting vehicle, while it might be any other disturbance. Therefore a change was made to the python script to augment it with vehicle detection capabilities. For this the OpenCV library was used again, using its cascade classifier with a pre-trained classifier for cars.

## 3.2 Metrics

The goal of this paper is to be able to predict if a signal will flip three seconds into the future and if so into what color. The first metric that could be used would be the accuracy metric. Accuracy would be an ideal metric to showcase how reliable the model is at predicting the correct color the signal would flip or not flip to. Following the same logic the F1-Score will be calculated to ensure there is no imbalance problem that would cause the accuracy to stay relatively high due to a large amount of true negatives for example.

## 3.3 Data Pre-processing

The initial data gathering consolidated all of the signal time data into single rows. This means a single row in the database would represent how long a signal was red or green for. This data was too limited for the type of research this paper wants to conduct. Therefore a python script was developed that would take a single row in the database and expand it into multiple rows depending on how long a signal stayed a certain color. [13] This script would then create a column which would become the target of the models prediction, this column would note the signals color in three seconds time given the current conditions. These 3 seconds could be altered to any custom number but for the purposes of this research three seconds was chosen.

# Chapter 4

# Prediction

This section will dive into different classification models to see which model and which parameters of these models will have the greatest chance of making an accurate and precise prediction. The reason classification is being chosen over regression is that the goal of the current models is to predict, given a set of features, weather or not a light is green or red in three seconds time. This is a binary decision and therefore fits more into a classification problem. It is notable however that regression, with the goal of predicting how much time is left until a signal flip, is also a possible avenue of research. The features used to make these predictions are as follows: the date and time, timer of a light, current signal color, weather condition, a car being present or not and how long a car has been waiting at the light. Each classification will then be rated on three different metrics: the accuracy, the F1-score and the AUC. The accuracy metric represents the proportion of correct predictions. The F1-Score is a combination of the precision and recall performance of the model. The AUC metric is the area of the curve under an ROC curve ranging from a score of 0 to 1. An ROC curve plots the true positive rates of the model with the false positive rate. These three metrics should provide a good amount of clarity on the performance of each model.

## 4.1 Decision Tree

The first classification model to test will be a decision tree classifier. This is a supervised learning type of classifier which attempts to make predictions based on how a previous set of "questions" were answered. The sklearn library has a built-in implementation of this model and was used for this thesis. A variety of different hyperparameters were tested using the inbuilt GridSearchCV method to end up on the criterion of entropy with a maximum node depth of 11.

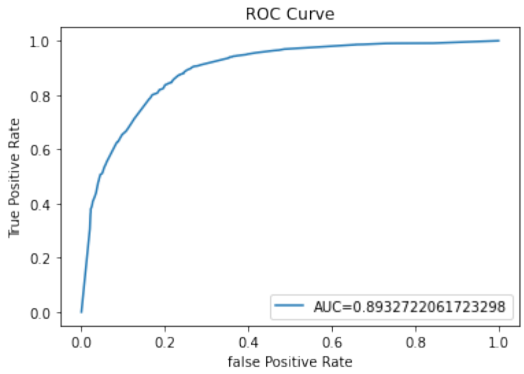



FIGURE 4.1: Decision tree model ROC curve

## 4.2 Logistic Regression

The second model to test will be a logistic regression model. This model is also an example of supervised learning which is commonly used in binary classification problems such as this one. This will again be taken from the sklearn library and a similar hyperparameter tuning methodology using the GridSearchCV was used here.

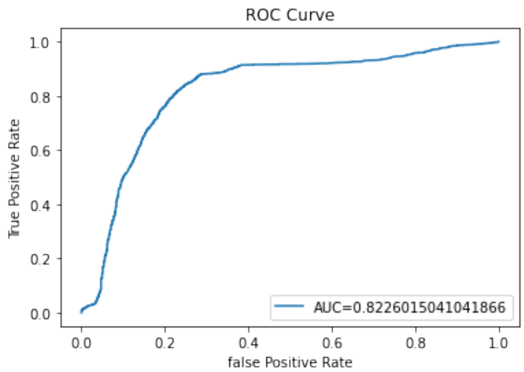



FIGURE 4.2: Logistic Regression model ROC curve

## 4.3 Random Forest

The third model to test will be a random forest model. Like the previous examples this model is also an example of supervised learning. By the nature of random forest being a combination of many decision trees it should outperform the decision tree model. The same sklearn library and steps was used for this model as the previous two.

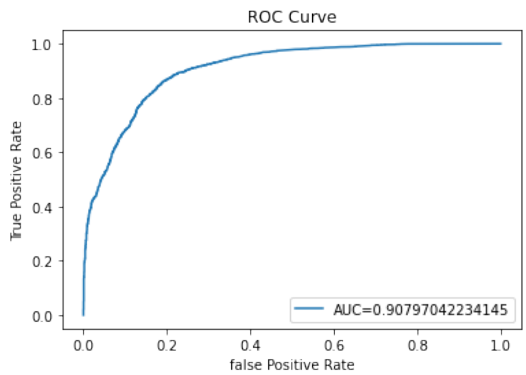



FIGURE 4.3: Random Forest model ROC curve

## 4.4 Result Comparison

Comparing the evaluation metrics selected for both models the following figure is produced. This analysis is based on data gathered at the three-way intersection over a selection of hours spanning multiple weeks during 2022. The four-way intersection data was found less viable to analyse due to the cameras position when gathering data making it impossible to get a consistent and reliable reading of the traffic state at the intersection. This was caused by two factors, the first was the camera was too close to the intersection blocking the view of traffic which was waiting at the visible signal. This would make it quite challenging to gauge exactly if a vehicle was waiting at the signal and for how long. The second factor was that for vehicles coming in the opposite direction the cameras angle meant that any vehicles taking a left turn would obstruct the vehicles currently waiting. This meant that if no vehicles were waiting to cross, but some vehicles were taking a turn, the python script would be forced to observe a vehicle in the designated spot and mark it as a vehicle waiting when in fact it was just a vehicle crossing the cameras view. This made this particular feature extremely riddled with outliers.

| Model | Accuracy | F1-Score | AUC |
|---|---|---|---|
| Decision Tree | 0.841 | 0.880 | 0.893 |
| Logistic Regression | 0.815 | 0.858 | 0.822 |
| Random Forest | 0.849 | 0.887 | 0.908 |

TABLE 4.1: Result comparison between the three different models

Looking at the three models and their respective metrics, we can see that at least for this task the decision tree model performed better then logistic regression. A possible explanation for this is due to some features in the dataset being more complex and categorical than others, such as the weather and datetime features. We can see from the F1 score being at a 0.880 that the decision tree was not excellent at being able to provide accurate and precise results, however the AUC score of 0.893 would seem to imply it was rather good. This discrepancy between both metrics is most likely a result of the input data set being very balanced being that for every green signal a red signal must have existed and been recorded. It can also be observed that the random forest model performed the best out of the three. Naturally random forest will outperform a simple decision tree as it is utilizing many internal decision trees to consolidate them into a single result. It will therefore outperform logistic regression in the same way a single decision tree has done which is a pattern seen in many similar classification problems.[5]

## 4.5 Feature Importance

In this section we will explore the importance of the different features selected to see how much of an impact they actually have on the models ability to predict. Due to the previous section seeing better results with the random forest model this model will also be used to compare different features to keep it consistent as well as using the three-way intersection data for these tests.

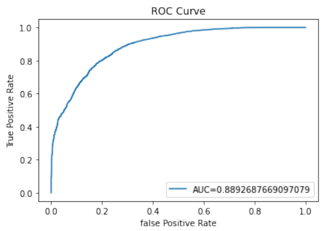
The five sets of features that will be tested are as follows:

- The presence of a vehicle and the duration of its wait
- The weather conditions and temperature
- The timer of the signal
- The color of the signal
- The timer and color of the signal

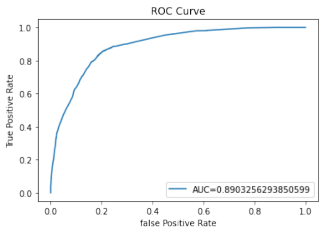
In order to test this the same data analysis steps as the previous sections will be replicated but with this feature removed. The first set of features to analyse will be if a vehicle is present or not and if so how long has this vehicle been waiting at the red signal. The Netherlands is equipped with smart traffic lights that adapt to traffic flow and therefore this feature could be impactful as traffic lights are constantly monitoring the traffic flow and adapting accordingly based on their programming.[16] The second set of features are related to the weather conditions and temperature, while not directly impacting a iVRI's decision making on whether or not to show a green or red signal, the current weather could have an impact on traffic volume or other road conditions that would have an impact on the traffic flow. The final feature is the actual timer of the signal for how long it has been red or green. This feature is not only likely the most impactful but also heavily based on traffic flow due to certain signals never going from red to green unless a vehicle is actually present.

| Removed Feature(s) | Accuracy | F1-Score | AUC |
|---|---|---|---|
| Vehicle Presence and Timer | 0.825 | 0.868 | 0.889 |
| Weather and Temperature | 0.840 | 0.879 | 0.890 |
| Timer of Signal | 0.829 | 0.872 | 0.860 |
| Color of Signal | 0.690 | 0.778 | 0.724 |
| Timer and Color of Signal | 0.673 | 0.793 | 0.629 |

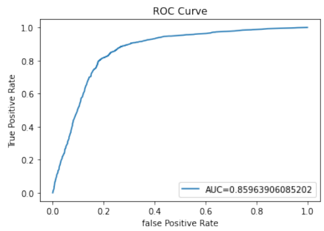
TABLE 4.2: Feature importance comparison using the Random Forest model



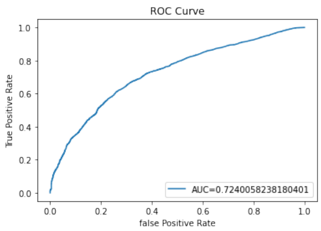

(A) Vehicle Presence Removed



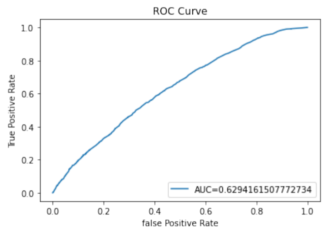

(B) Weather Presence Removed




(C) Signal Timer Removed




(D) Signal Color Removed




(E) Signal Timer and Color Removed

FIGURE 4.4: ROC curves of feature testing

From the table a few things can be observed. The first remark is that the most dominant feature is the current color of the signal. Without it the accuracy and precision of the model drops drastically. This is most likely because this feature is a more direct representation of the internal algorithm of the iVRI compared to the other features. Similarly the timer of the signal has a higher level of influence when compared with non signal based features. It can also be observed that if both signal based features, the color and timer, are removed they have an even larger impact on the performance of the model. Weather and temperature being removed have the lowest impact and seem almost negligible in their influence. Finally we can see a slight drop in performance when removing the vehicle presence and timer features, knowing what we know on how the signals operate a higher performance loss is expected however this slight loss instead can be explained at lower quality data gathering of the presence of vehicles due to the inability to record the actual volume of vehicles waiting.

# Chapter 5

# Discussion

## 5.1 Data

This thesis was focused on seeing if a model could be trained to predict the future state of a traffic light given a certain set of inputs as data. More data that was not collected in this paper that could help this prediction would be traffic volume and density. As the Netherland's traffic light system is being actively upgraded to iVRIs (intelligent traffic control systems), this factor is being taken into account for how frequent each signal changes color and for how long it maintains that color.[1] Not being able to track and gather data for how many vehicles are waiting in each section of an intersection will hamper the reliability of the data. This data however is difficult to obtain with the method used in this paper, as a single camera angle and webcam is often the only option for an intersection giving only clear view of 2 or maybe 3 avenues, but never all 4. Multiple cameras would need to be setup during the data gathering phase in order to obtain this data.

Another feature that could have been improved, is the red and green signal flip detection. Currently a python script was used to visually detect any change in color and determine if the signal was green or red. However, this had many downfalls when it comes to real life such as changing lighting conditions and potential obstructions. This created many false data entries that had to be trimmed during data pre-processing which lowered the amount of usable data as well as potentially still negatively impacting seemingly "normal" data. A simple solution to this problem is having a direct API with the actual traffic light in order to get its current signal status. The technology for this already exists with the implementation of iVRIs, and would be possible to do in the future. With a more robust dataset this problem could also be turned into a regression problem with the goal of predicting how much time is left until a signal would flip. This approach would lead to much more practicality when implemented in the real world.

## 5.2 Conclusion

In this paper an analysis of data gathered from a select few intersections in the Netherlands was used in an attempt to predict whether a traffic light signal was going to turn green or red 3 seconds in the future depending on a set of features. The goal was to not only find and create a model that could reliably predict a signal color 3 seconds in the future but also figure out which feature was the most dominant. Over the course of several months in both 2022 and 2023, traffic light and traffic data was gathered at two select intersections in the Netherlands. This data was then preprocessed to get rid of outliers caused by the way the data was gathered, visual image recognition that had real life constraints. Three different classification models were then trained on the data: A decision tree, logistic regression and random forest model. None of these models showed extremely good results, however random forest did show a higher level of true positives and accurate predictions. These models however could easily see some improvement if the data gather process was made easier with access to an API, and if more camera angles were offered to get additional important data such as traffic volume and density. From the feature importance testing it can be observed that the most dominant feature was the time a signal stayed a certain color. This however could be because of two reasons or most likely a combination of two reasons, either the system has a maximum time limit on a certain color signal or once a signal is a certain color for long enough, such as red, a build up of vehicles starts to happen causing an imbalance in the system as the other direction with the green signal is showing no build up and this incentives the signal to flip. Vehicle presence and duration also had an impact albeit to a lesser degree and the weather condition and temperature were observed to be negligible or even a hindrance in some cases.

## 5.3 Future work

Future work for this kind of project can go down two different avenues. Either a refinement of the current methodology, using API calls to actual traffic lights in order to make the data gathering process of this variable more accurate as well as adding in more features such as traffic volume and density. The other avenue for this type of work is to implement regression models. Regression would be more suited for the role of predicting how long is left until a traffic signal flips and not just if a traffic signal will flip in a static amount of time. This higher level of flexibility would allow this type of model to have better applications in the real world.

# Appendix A

## A.1 Setup of python code for image detection

The setup for the data gathering had some manual hard coded sections. Here is a list of steps that had to be performed before any data could be gathered and recorded:

- Find a YouTube livestream of an intersection
- Input the URL of the livestream into the appropriate variable inside the python script
- Select a frame from the livestream and find a crop for both the desired traffic light to be monitored as well as the specific area where a vehicles presense should be monintored and enter the crops x and y coordinates into the script

The two live-streams used for the data gathering:

- Three-way: https://www.youtube.com/watch?v=CakjTZ8V0Yw
- Four-way: https://www.youtube.com/watch?v=HrUMpcgLG78

## A.2 Setup jupyter notebook

The simple jupyter notebook only requires python 3 to run. The sklearn library as well as pandas, numpy and seaborn are required to make the application work.

# Bibliography

[1] (2019). Hoe staat het met de ivri? https://www.talking-traffic.com/en/news/hoe-staat-het-met-de-ivri.

[2] (2022). overzicht webcams https://webcam.gelderland.nl/.

[3] Boudhrioua, S. and Shatanawi, M. (2021). Implementation of absolute priority in a predictive traffic actuation schemes. *Periodica Polytechnica Transportation Engineering*, 49(2):182–188.

[4] Bourke, P. (1995). Colour spaces http://paulbourke.net/miscellaneous/colourspace/.

[5] Couronné, R., Probst, P., and Boulesteix, A.-L. (2018). Random forest versus logistic regression: A large-scale benchmark experiment. *BMC Bioinformatics*, 19.

[6] de Charette, R. and Nashashi bi, F. (2009). cvx. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 333–338.

[7] Ding, Y. (2003). Impact of stops on vehicle fuel consumption and emissions. *Journal of Transportation Engineering-asce - J TRANSP ENG-ASCE*, 129.

[8] H P, V., Poornima, B., and Sagar, B. (2018). *Detection of Outliers Using Interquartile Range Technique from Intrusion Dataset*, pages 511–518.

[9] Lee, J., Park, B. B., Malakorn, K., and So, J. J. (2013). Sustainability assessments of cooperative vehicle intersection control at an urban corridor. *Transportation Research Part C: Emerging Technologies*, 32:193–206.

[10] Madeline, L. (2021). Intersection control for cyclists with isignum.

[11] Perez, H. and Tah, J. H. M. (2020). Improving the accuracy of convolutional neural networks by identifying and removing outlier images in datasets using t-sne. *Mathematics*, 8(5).

[12] Pêcher, C., Lemercier, C., and Cellier, J.-M. (2011). *The Influence of Emotions on Driving Behavior*.

[13] Ronan, H. (2023). Predicting the phase timings of traffic signals using image processing and machine learning https://github.com/delay67/project2023.

[14] Saad, M. M., Khan, M. T. R., Srivastava, G., Jhaveri, R. H., Islam, M., and Kim, D. (2022). Cooperative vehicular networks: An optimal and machine learning approach. *Computers and Electrical Engineering*, 103:108348.

[15] Shafik, A. K., Eteifa, S., and Rakha, H. A. (2023). Optimization of vehicle trajectories considering uncertainty in actuated traffic signal timings. *IEEE Transactions on Intelligent Transportation Systems*, 24(7):7259–7269.

[16] Soni, S. and Calvert, S. (2020). A methodology for prediction accuracy assessment of intelligent traffic signal control algorithms with spat messages. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6.

[17] Szele Phd, A. and Kisgyörgy, L. (2021). Merging and crossing movements in the congested and the non-congested traffic flow. *Pollack Periodica*, 16.

[18] Tae-Hyun, H., In-Hak, J., and Seong-Ik, C. (2006). Detection of traffic lights for vision-based car navigation system. In *Proceedings of the First Pacific Rim Conference on Advances in Image and Video Technology*, PSIVT'06, page 682–691, Berlin, Heidelberg. Springer-Verlag.

[19] Van de Vyvere, B., Colpaert, P., Mannens, E., and Verborgh, R. (2019). Open traffic lights: A strategy for publishing and preserving traffic lights data. In *Companion Proceedings of The 2019 World Wide Web Conference*, WWW '19, page 966–971, New York, NY, USA. Association for Computing Machinery.

[20] Yang, J., Rahardja, S., and Fränti, P. (2019). Outlier detection: How to threshold outlier scores? In *Proceedings of the International Conference on Artificial Intelligence, Information Processing and Cloud Computing*, AIIPCC '19, New York, NY, USA. Association for Computing Machinery.

[21] Yu, C., Huang, C., and Lang, Y. (2010). Traffic light detection during day and night conditions by a camera. In *IEEE 10th INTERNATIONAL CONFERENCE ON SIGNAL PROCESSING PROCEEDINGS*, pages 821–824.