Using a Convolutional Neural Network to Predict Road Traffic Accident Risk from Geographic Data

William May

MSc Computer Science

The University of Bath

September 2024

Using a Convolutional Neural Network to Predict Road Traffic Accident Risk from Geographic Data

Submitted by: William May

Declaration

This dissertation is submitted to the University of Bath in accordance with the requirements of the degree of MSc Computer Science in the Department of Computer Science. No portion of the work in this dissertation has been submitted in support of an application for any other degree or qualification of this or any other university or institution of learning. Except where specifically acknowledged, it is the work of the author.

Abstract

This project describes a novel approach to road safety analysis, using a convolutional neural network (CNN) to predict crash risk from static road safety factors. A software application was developed to generate map-like images visualising these factors around geographic points of interest, integrating geographic data from OpenStreetMap (OSM) and Shuttle Radar Topography Mission (SRTM). Road traffic accident (RTA) data was acquired from annual road safety reports produced by the UK Department for Transport (DfT) and used to generate images around collision locations. An algorithm for randomly selecting waypoints from the Great Britain road network was developed, with bias towards geographic areas and way types with higher traffic volumes. This algorithm facilitates the generation of sets of images with a geographic distribution representative of real-world driving patterns. A set of 84,060 images representing collision locations and random waypoints were generated and used to train a CNN to classify images as high or low-risk. The model achieved good results on standard performance metrics, and feature analysis on test results revealed expected correlations between predicted crash risk and factors such as junction count and intersection proximity. However, counter-intuitive results were also observed, including a negative correlation between speed limit and predicted crash risk. Data availability and limitations in the negative sampling algorithm were implicated as causal. Opportunities to improve the real-world ability of the model were subsequently explored in depth. The project supports future work through the provision of foundational software and algorithms, while exploring the potential and challenges in applying computer vision to road safety analysis.

Contents

List of figures and tables

Acknowledgements

I would like to thank my project supervisor Raghubir Singh for providing regular feedback on my progress.

I would like to thank my employer, the Ministry for Housing, Communities and Local Government, for graciously allowing me a total of 10 days of study leave to work on this project after the past few months, without which it may have proven difficult to get this work done.

I would also like to thank my good friend Jonny Evans for his insights into the world of data science, including helping me to frame the model analysis, and for encouraging me to look beyond simple performance metrics in the quest for truth (and for his enthusiastic championing of F1 score!).

Finally, I would like to thank my long-suffering wife Kate, who has been putting up with my lack of contribution to the housework for years, but nevermore so than over the past couple of weeks! On a serious note, last year Kate and I were passengers in a car that was T-boned by a van at an intersection, with an impact speed of around 40 mph. We were both hospitalised; I was discharged later that same day, but Kate remained in hospital for four nights, owing to the more severe injuries she had suffered as a result of being closer to the point of impact. In reality, we were both fortunate to be alive. It was this experience that inspired me to start thinking more about road safety, making my choice of topic for this project an obvious one. So, here is my tribute to Kate, the love of my life, and to the incredible fortune that has meant we are still looking forward to sharing a life together.

Chapter 1

Introduction

1.1 Problem impact

Road traffic accidents (RTAs) represent a significant public health issue, causing substantial human suffering and economic loss. The World Health Organization (WHO) reported that, as of 2019, RTAs were the 12th leading cause of death worldwide across all age bands, and in 2021 alone were responsible for 1.19 million deaths (World Health Organization, 2023). The preventability of RTAs is emphasised by the fact that they rise to become the single leading cause of death worldwide among young people aged between 5 and 29 years.

Although RTAs disproportionately affect individuals in low-income countries, even in highly-developed countries like the United Kingdom (UK) the scale of the problem is enormous. In 2022, the UK Department for Transport (DfT) reported that RTAs were responsible for a total of 1,711 fatalities and a further 28,031 serious injuries (Department for Transport, 2023), with the average age of those killed being just 47.7 years.

The impact of RTAs extends beyond fatalities and injuries. The economic burden is staggering, encompassing lost productivity, medical costs and ambulance costs. While precise global figures are challenging to determine, the scale of the issue is clear. Estimates by the DfT of the economic cost of RTAs indicate that, excluding "human costs" and using 2012 prices, a fatal casualty has an average economic cost of £586,722, a serious casualty has an average economic cost of £36,237, and a "slight" casualty has an average economic cost of £3,397 (Department for Transport, 2021). Multiplying these numbers by 2022 UK RTA figures and accounting for the rise in inflation between 2012 and August 2024 puts the total economic cost for the UK in that year alone at £3.3 billion. This figure likely underestimates the true cost, as it does not account for unreported accidents or long-term societal impacts.

1.2 Research background

Road safety is a hugely complex issue involving multiple interacting factors (Wright *et al.*, 1976), which can be classed as either static or dynamic based on how they tend to vary with respect to geography. Static factors persist over a long period of time at a given location, and include the shape of the road network (e.g., road curvature, junctions, intersections), lane elements (e.g., number, width, directionality), speed limits and topography. Dynamic factors fluctuate over time at a given location, and include driver behaviour (e.g., speeding, distraction, impairment) and environmental conditions (e.g., weather, lighting).

Dynamic factors are the primary cause of RTAs (Treat *et al*., 1979; Dingus *et al.*, 2016); however, the interplay between dynamic and static factors is complex. A 2008 report by the American Association of State Highway and Transportation attributed 34% of RTAs in part to "roadway-related factors" (2008). Furthermore, studies have consistently shown that static factors, even when taken in isolation, significantly affect crash risk (Nilsson, 1982; Karlaftis, 2002; Islam *et al.*, 2019).

Understanding the effects of static factors on road safety in isolation from dynamic factors has major real-world implications for a variety of activities, including those involving long-term planning, such as road building. This is because it is inherently difficult to predict how dynamic factors will vary over time (Seaver *et al.*, 2000; Parr *et al.*, 2020). In addition, models built using dynamic factors may be harder to deploy in jurisdictions with limited data or data collection capacity, such as in developing countries (Mennecke *et al.*, 2001).

Much of the current literature relevant to the effects of static factors on road safety has limitations. Some studies do not leverage large datasets (e.g., Camacho-Torregrosa *et al.*, 2013), while others focus on roadway characteristics without considering the wider road network (e.g., Chen *et al.*, 2019). Many studies do not examine static factors in isolation (e.g., Wang *et al.*, 2013), or utilise discrete approximations of road junctions or networks that may not accurately capture real-world complexity in road geometry (e.g., Marshall *et al.*, 2011).

1.3 The opportunity

The limitations in existing research into static road safety factors, and particularly the under-utilisation of advanced machine learning techniques in the field (Silva *et al.*, 2020), combined with the continued advancement of computer vision techniques (Hassaballah and Hosny, 2019), provides us with the novel opportunity to apply computer vision to the analysis of static road safety factors. Specifically, RTA data can be integrated with geographic data to create map-like images of collision locations that can be used as inputs to a convolutional neural network (CNN). Using a CNN means there is no need to drastically reduce the dimensionality of the input data, which requires making assumptions about feature importance; instead, the model is able to work directly with high-dimensional representations of the geographic features implicated in crash risk, with geospatial integrity preserved to a much higher degree. The suggested approach aligns with recent trends in deep learning where end-to-end models are favoured over traditional feature engineering pipelines (Mirabello and Wallner, 2018; Dzieżyc *et al.*, 2020). The use of a CNN in this context could potentially uncover subtle interactions between geographic features that might be overlooked by conventional statistical methods or even human experts.

This opportunity is further amplified by the vast amount of geographically-labelled data available on RTAs in the UK, which should provide data in the volumes typically required to train a powerful computer vision model.

1.4 Project objectives

The primary aim of this project is to train a CNN for predicting crash risk. The CNN will function as a binary classifier, containing a sigmoid activation layer to output the probability of the positive class, which can be interpreted as a "crash risk index". This approach necessitates the creation of both positive and negative samples for training. Positive samples will be images centred around collision locations, obtained from RTA data provided by the DfT; negative samples will be images centred around random points on the UK road network, with bias towards higher-traffic areas and certain way types to ensure representativeness.

Specifically, the objectives of this project are:

- Develop a flexible and modular software system for generating map-like images around specific locations:
    - Create a command-line interface for customisable image generation
    - Integrate data from various sources including OpenStreetMap (OSM) for road geometry and features, and Shuttle Radar Topography Mission (SRTM) for topographical data
    - Visually encode static road safety factors in images in a way that emphasises variation while preserving data integrity
- Design and implement an efficient process for creating balanced datasets of collision and non-collision locations:
    - Retrieve and model DfT RTA data
    - Integrate RTA data with geographic data sources
    - Implement a conditional negative sampling algorithm using traffic data to facilitate the creation of sets of negative samples representative of real-world driving patterns
    - Optimise the image generation process for efficient production of large quantities of training data
- Train and evaluate a CNN for crash risk prediction:
    - Implement a CNN architecture capable of processing map-like images
    - Train the model using a balanced dataset of collision and non-collision locations
    - Evaluate the model's performance using appropriate metrics (e.g., accuracy, precision, recall, F1 score)

- Analyse the model's predictions to gain insights into the relationship between static road safety factors and crash risk:
    - Conduct a detailed feature analysis to identify the most influential static factors
    - Examine the model's performance across different scenarios
    - Compare the CNN's predictions with insights derived from conventional crash risk assessment techniques
- Discuss the implications of the findings for road safety:
    - Explore how the model's predictions can be applied in the real world
    - Discuss the limitations of the approach
    - Suggest directions for future work

1.5 Deliverables as evidence of fulfilling objectives

To demonstrate the successful completion of the project objectives, the following deliverables will be produced:

- A software application for generating map-like images visualising static road safety factors around specific locations
- A relational database schema for RTA data storage
- A set of scripts to facilitate local database setup and population
- A script that uses RTA data in conjunction with geographic data to efficiently generate a batch of positive training data samples
- A script that combines OSM and traffic data to efficiently generate a batch of negative training data samples
- A balanced dataset of positive and negative samples
- A script for training a CNN using these samples
- A trained and evaluated CNN capable of outputting crash risk index given an image produced by the software application
- A script for evaluating model performance on unseen data
- A comprehensive dissertation discussing software implementation, model development, model performance and potential real-world applications

Chapter 2

Developing the image generation software

2.1 Introduction

Developing a robust and performant software application for generating images is pivotal to the success of this project, as the images produced by the application are used as the training inputs for the crash risk prediction model. High-quality training data is essential for successful machine learning projects (Jain *et al.*, 2020). Furthermore, the application involves the spatial integration of data from a variety of sources, with any misalignment likely to distort the apparent relationship between features to such an extent that any model would be invalidated before training even begins. As such, a substantial portion of the time spent on this project was devoted to the development of this application.

2.2 Requirements

To ensure the software application meets the requirements of the project, a comprehensive set of requirements has been defined. These range from functional requirements, such as the implementation of a command-line interface through which users can interact with the software, to non-functional requirements such as performance optimisation.

- Command-line interface (CLI)
    - Implement a flexible CLI that accepts various arguments for customising image generation
    - Support arguments for location specification (latitude/longitude)
    - Allow configuration of visualisation parameters (extent, included way types, way colouring options)
    - Provide options for rendering additional data layers (elevation, traffic)
- Image generation
    - Generate map-like images visualising the static road safety factors surrounding specific locations
    - Implement consistent scaling and alignment of generated images
    - Clearly mark the central node on the image
    - Draw ways with appropriate line widths based on way type
    - Support various colour schemes for ways (e.g., by speed limit, random, custom colour)
    - Render one-way ways with a distinct visual signifier (e.g., dashed lines)

    - Render additional data layers (elevation, traffic) as background when specified
    - Allow for background smoothing through interpolation techniques
- Feature extraction
    - Calculate and output a comprehensive set of features based on the geographic data being visualised, including metrics such as average speed limit, elevation range, intersection count, etc.
    - Ensure feature extraction is optional and can be enabled/disabled via CLI
- Data acquisition and processing
    - Implement flexible OSM data retrieval (support both API and local database options)
    - Process raw OSM data to extract relevant information, including geometry, way type, lane characteristics and speed limits
    - Filter ways based on type (minor, medium, major) according to configuration
    - Implement flexible SRTM data retrieval (support both API and local database options)
    - Implement flexible traffic data retrieval (support both CSV and database options)
- Data structures
    - Implement classes for geographic entities including nodes and ways
    - Implement a bounding box class for simplifying data retrieval and feature extraction
    - Create a data structure to hold and process gridded data (elevation, traffic)
- Geospatial processing
    - Implement accurate conversion between different co-ordinate systems
    - Account for Earth's curvature in distance and area calculations
    - Develop algorithm to adjust nodes to the nearest point on the road network
    - Implement methods for data interpolation and smoothing
    - Utilise third-party libraries for geometric calculations (e.g., intersections)
- Performance optimisation
    - Support local storage of OSM data for faster retrieval
    - Support local storage of SRTM data for faster retrieval
    - Support multi-threading for batch image generation
- Error handling and logging
    - Implement comprehensive error checking and exception handling
    - Provide informative error messages for various failure scenarios
    - Implement a flexible logging system with different verbosity levels

- Log important events, warnings, and errors during the image generation process

2.3 Technologies

Python was the programming language used to develop this software application. Python is a robust and readable language that offers a rich ecosystem of libraries, both built-in and third-party, that are applicable to the specific challenges faced in this project (McClain, 2022). The built-in argparse library supported the creation of an easy-to-use and well-documented command-line interface. Built-in language features and NumPy handled low-level implementation details such as in-memory data structures and trigonometric operations. Matplotlib and Pillow provided visualisation capabilities, which are essential for image generation. Geospatial processing was assisted by Geographic Information System (GIS) libraries including Cartopy and Shapely, which offer highly specific tools for things like co-ordinate projection and identifying intersection points between geometric shapes. SciPy was used for interpolation, which is required for both background smoothing and traffic data processing. API calls were handled by the requests library, while local storage and modelling of data was handled by psycopg2 and SQLAlchemy. Beyond image generation, seamless integration with the model training and testing phase will be possible thanks to Python's support for popular and powerful machine learning frameworks such as TensorFlow.

2.4 Implementation

2.4.1 How it works

N.B., in the below section, custom classes specific to this software application are formatted like this: `CustomClass`

The process begins with the creation of a `Node` object, representing a geographic point, from a latitude and longitude provided by the user. The `Node` serves as the focal point of our visualisation, anchoring all subsequent data gathering and rendering processes.

The geographic area of interest is defined by creating a rectangle, or `BoundingBox`, centred around the `Node`. A `BoundingBox` is effectively a container for a set of four pairs of co-ordinates, representing the four cardinal directions: north, east, south and west. A configurable extent parameter determines how far in metres in each of these directions the `BoundingBox` extends from the `Node`. To convert metres to degrees, a constant approximation of 111,320 metres per degree of latitude is used (Lundbäck *et al.*, 2020). Given this approximation, latitudinal bounds are trivial to calculate. Calculation of longitudinal bounds, however, requires careful consideration of the spherical nature of the Earth: as the distance from the equator

increases, the distance represented by one degree of longitude decreases. This is handled by scaling the longitudinal extent by the cosine of the latitude, ensuring that the `BoundingBox` always represents a constant area, regardless of latitude.

Next, OSM data is fetched for the `BoundingBox`, with a buffer of 50 m applied in each direction to offer a stronger guarantee of no missing ways in the final visualisation. OSM is the richest data source used by our software application, providing not only way types and geometries, but also speed limits and lane characteristics. Two retrieval methods are available for OSM data, selectable via an environment variable. The default method is the Overpass API, which is slow but provides up-to-date data, with no geographic constraints or local storage requirements. The alternative involves setting up, populating and accessing a local database, providing fast, offline access that is particularly useful when generating training data in large quantities. Detailed setup instructions can be found in the code repository's README file. Both retrieval methods share a common interface: the `OSMClient` class. Abstracted functionality includes the parsing of speed limit data to handle inconsistencies. Ultimately, way data is processed into a list of `Way` objects.

The `WayCollection` class is used to organise, optimise and analyse OSM data. Before binding the list of `Ways` to itself, the `WayCollection` will process each `Way` into one or more new `Ways`, depending on the intersection between the `Way` and the `BoundingBox`. This process removes any `Way` segments that are not visible in the figure, ensuring that features such as junction count and total way length can be correctly extracted.

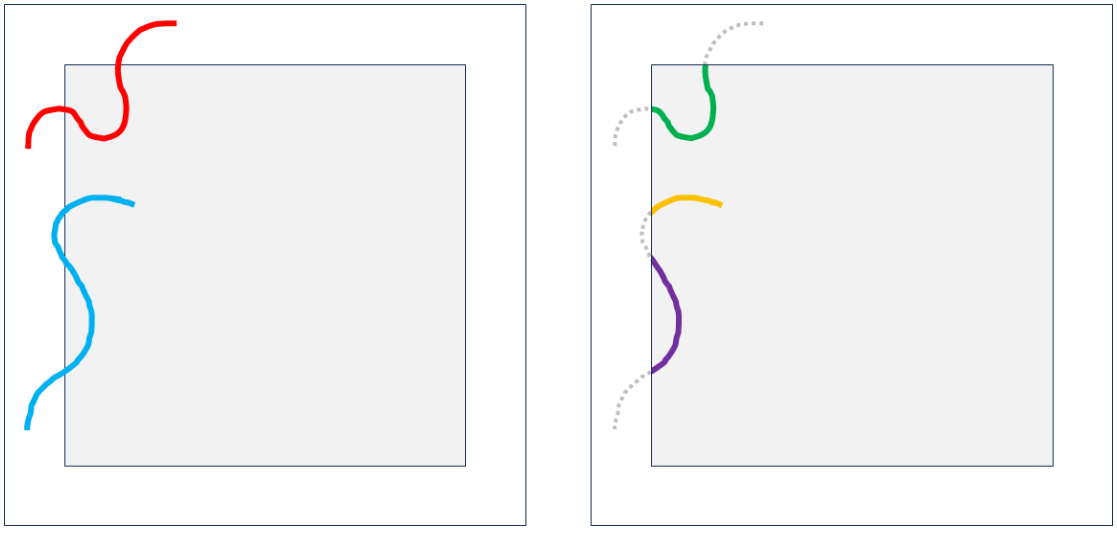

Figure 1: Visual representation of the effect of visible way processing by the WayCollection class, with the "before" state on the left, and the "after" state on the right. The grey inner boxes represent the geographic area shown in the final figure; the white outer boxes represent the buffered geographic area for which OSM data was fetched. The red way gives rise to the green way upon intersection with the

boundary of the inner box; the blue way gives rise to the yellow and purple ways. Discarded segments are shown as grey, dashed lines.

The `WayCollection` also merges collinear `Way`s, with a collinearity tolerance of 30°, in cases where way types are the same and the `Way`s do not have speed limits that differ. This is primarily done to more accurately distinguish junctions from intersections during feature extraction, as the splitting of real-world ways at intersection points is a common feature of OSM data.

The `Node` is then optionally translocated to the nearest point on the nearest `Way` in the `WayCollection`. This functionality was introduced to compensate for minor geographic discrepancies between data sources, and acts as a form of data cleaning, placing `Node`s directly on `Way`s. The `Way` to which the `Node` is translocated is also a valuable source of features, including number of lanes, speed limit and way type. After translocation, a new `BoundingBox` is created around the translocated `Node` and OSM data is re-fetched to ensure all `Way`s are accounted for in the newly-defined geographic area.

Next, depending on the configuration parameters used, either elevation or traffic data is fetched for the geographic area of interest, to be used to colourise the figure background. This data acquisition process is handled by a hierarchy of abstract and concrete descendants of the base `GridClient` class, with both an API and local database access available as retrieval methods, selectable via an environment variable. Regardless of the data type or acquisition method, the `BoundingBox` is buffered by 100 m in each direction before being used in the search query. Elevation data comes courtesy of SRTM and is naturally gridded, which is to say it is provided as a regular grid of data points, whereby each point represents the elevation at a specific location. Traffic data meanwhile is not naturally gridded, and has to be adapted into a grid through aggregation and interpolation; the exact process through which this is achieved will be discussed in the subsection “Generating negative samples” of the section “Generating training data”.

For SRTM data, the OpenTopography API is used to retrieve data from the GL1 dataset, the highest resolution dataset available from SRTM. The 1 represents the resolution in arc-seconds – equivalent to 30 m at the equator. At a latitude of 55°, which can be considered roughly representative of Great Britain, the east-west resolution increases to approximately 17.74 m. As with OSM data, the trade-offs between the API and local database access primarily relate to currency, speed and storage. Detailed setup instructions for the local SRTM database can be found in the code repository’s README file. After fetching, the grid data undergoes optional interpolation and smoothing to enhance visual quality. This process involves upsampling the data using linear interpolation to increase resolution, followed by Gaussian filtering for a smoother appearance. Grid data is returned from `GridClient` encapsulated within a

`BackgroundGrid` object, which like `WayCollection` is an intelligent data structure capable of both organising and analysing its own data.

The `FigureGenerator` class handles the actual visualisation, using Matplotlib and Cartopy. The figure's axes are initialised with a transverse Mercator map projection, chosen for its ability to accurately represent the geometry of shapes at a small scale, far from the equator. The `BackgroundGrid` object is rendered first, using a dark green to light green colour scale to highlight differences in elevation or traffic volume without interfering with the colour scheme used for `Way`s. The centre of the value range mapped to the colour scale depends on the data, eliminating the significance of global absolute values and helping to produce images that will allow the CNN to focus on local changes. Further, for elevation data the difference between the minimum and maximum values in the value range mapped to the colour scale is fixed relative to the plot extent, which is to say that for a fixed plot extent and within correspondingly fixed elevation bounds, the same visual difference in colour will always correspond to the same absolute change in elevation. This is achieved by dividing the plot extent by a fixed constant "grid colour range factor" to get a value X, finding the centre value Y in the grid and subtracting or adding X from or to Y to obtain the minimum and maximum values in the range, respectively. For example, with a plot extent of 100 m and a grid colour range factor of 10, the resulting value range would be ±10 m relative to the centre value. If no background grid rendering option is supplied by the user, a simple light grey background is used for the figure.

Next, the `WayCollection` is plotted. Each `Way` is represented as a line, with its appearance customised to convey additional information. Line width is used to indicate way type, with major ways such as motorways appearing thicker. `Way` colourisation is customisable, with the options including using a single specified colour for all `Way`s, randomly generating colours, or colouring `Way`s based on their speed limits. In the latter case, a custom blue-red colour map is employed, with blue used for low speed limits and red for high. One-way `Way`s are distinguished through line dashing, with the dash pattern adjusted based on the line width to optimise visibility.

The final thing that `FigureGenerator` does is mark the `Node` as a circle at the centre of the figure. The figure is then saved as a PNG file.

2.4.2 Configuration options

The visualisation process can be configured through command-line arguments:

- `extent` – defines the extent of the visualisation in metres in each cardinal direction, from the node
- `include-ways` – limits which way types are included. Available options include:

    - `MAJOR` – only major ways (e.g., motorway)
    - `MEDIUM` – major and medium ways (e.g., secondary)
    - `MINOR` – all ways (major, medium and minor e.g., footway)
- `way-colour` – specifies how ways should be colourised. Available options include:
    - `NONE` – apply no colour
    - `RANDOM` – apply random colours (useful for distinguishing between separate, collinear ways)
    - `OSM_SPEED_LIMITS` – colour ways according to OSM speed limits, where available
    - Any hex colour value
- `render-elevation` – colours the image background based on SRTM elevation data
- `render-traffic` – colours the image background based on traffic volume data
- `background-smoothing` – smooths the background render through grid interpolation. Only applicable when either `render-elevation` or `render-traffic` is set
- `raw-traffic` – specifies that raw traffic data should be used instead of interpolated traffic data. Only applicable when `render-traffic` is set
- `lat` – defines the latitude of the node
- `lon` – defines the longitude of the node
- `adjust-node` – translocates node to nearest point on nearest way
- `generate-features` – extracts geographic features and saves to JSON file

2.4.3 Guided example

To provide a concrete illustration of the software application's functionality, let's examine the output generated by a specific command.

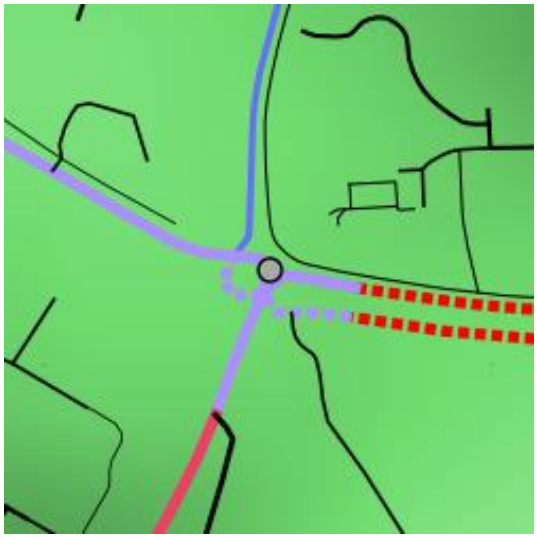

Figure 2: Map-like image generated by the software application using the command "python -m scripts.main —-lat 50.25488 —-lon -5.05630 --extent 100 --render-elevation --background-smoothing --way-colour OSM_SPEED_LIMITS".

The grey circle in the centre represents a node at the co-ordinates specified in the command. The mapped area is 200 m x 200 m, with the size defined by the extent parameter: the top edge of the image is aligned with the latitude 100 m north of the central node; the right edge is aligned with the longitude 100 m east of the central node; and so forth. The lines represent ways fetched using an OSM client. Thicker lines represent larger way types; dashed lines represent one-way ways. Line colour denotes speed limit: a blue-red colour gradient is used, from low to high speed limits. Black is used where no speed limit data is available. The darker green areas in the image background represent areas of lower elevation, while the lighter green areas represent areas of higher elevation.

2.4.4 Further examples

A variety of scenarios and outputs are documented below, to demonstrate the software.

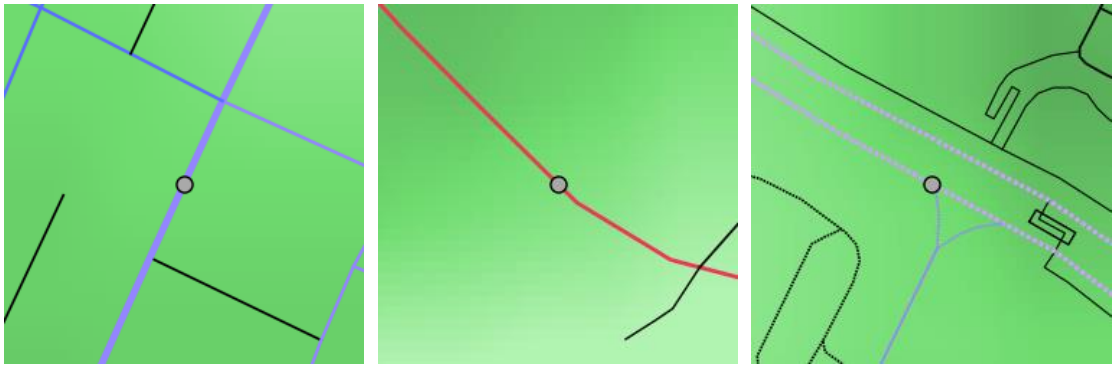

Figure 3: Example outputs, showing how the software is able to visualise static road safety factors in the area immediately surrounding a given location. From left to right, the co-ordinates used are as follows: 53.65774, -2.63200; 51.07897, -1.45825; 51.52590, -2.60595.

The software is capable of rendering both small and large areas centred around custom co-ordinates, with precise integration of elevation data at any scale.

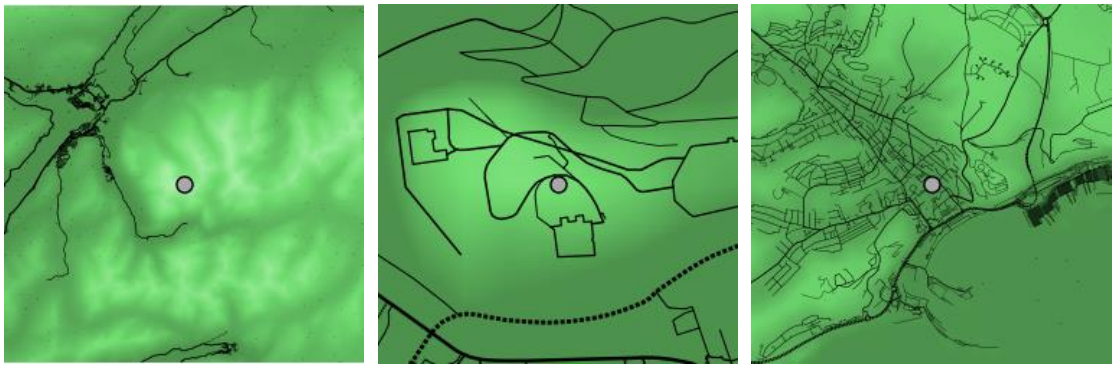

Figure 4: Examples of outputs that demonstrate the flexibility of the software. From left to right, the commands used are as follows: "python -m scripts.main --lat 56.7969 --lon -5.0036 --extent 10000 --render-elevation --include-ways MEDIUM"; "python -m scripts.main --lat 55.9486 --lon -3.1999 --extent 150 --render-elevation --background-smoothing"; "python -m scripts.main --lat 51.1279 --lon 1.3134 --extent 2000 --render-elevation --background-smoothing". The image on the left shows the area around Ben Nevis; the image in the centre shows the area around Edinburgh Castle; and the image on the right shows the area around the White Cliffs of Dover.

The high degree of configurability, including variable way colourisation and way type inclusion, allows the software to be used to generate meaningful images in a variety of scenarios and at a variety of scales.

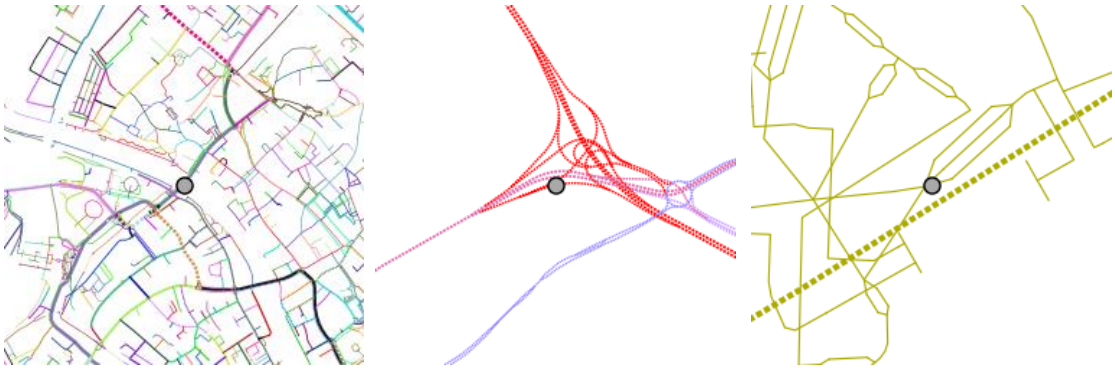

Figure 5: Further example outputs, demonstrating various configuration options. From left to right, the commands used are as follows: "python -m scripts.main --lat 53.9599 --lon -1.0873 --extent 500 --way-colour RANDOM"; "python -m scripts.main --adjust-node --lat 52.5095 --lon -1.8669 --extent 750 --way-colour OSM_SPEED_LIMITS --include-ways MAJOR"; "python -m scripts.main --lat 51.5115 --lon -0.128 --extent 25 --way-colour #AAAA00". The image on the left shows the centre of York; the image in the centre shows Spaghetti Junction in Birmingham; and the image on the right shows a small 50 m x 50 m area in the centre of Manchester.

Chapter 3

Generating training data

3.1 Introduction

We want our model to learn how the spatial relationships between various static road safety factors surrounding a specified location influence the likelihood of a collision occurring at that location. The problem can be framed as a binary classification task, distinguishing between locations where collisions are likely to occur and those where they are not. To train a binary classifier, labelled sets of positive and negative samples are required.

3.2 Generating positive samples

Positive samples in the context of binary classification are data samples that belong to the class of interest. In the case of our project, a positive sample is a map-like image visualising the static road safety factors around a known collision location. The image generation software application defined in the preceding section is able to take a latitude and longitude and generate such a visualisation. Therefore, given co-ordinate-labelled collision data, it is possible to use the software to generate positive samples.

Collision data was taken from annual RTA reports produced by the DfT. These reports are comprehensive and high-quality sources of information, with data going back as far as 1979 (Department for Transport, 2024). In the latest dataset (covering the year 2022) alone, 105,957 collisions were recorded, each with a precise latitude and longitude, along with rich supplementary data, both numerical (e.g., number of vehicles) and categorical (e.g., local authority, road type, light conditions). The richness of the UK's RTA data, along with the country's high availability of OSM and traffic data, make it an outstanding option for this analysis.

DfT RTA data was downloaded alongside an RTA data guide, which is an Excel file containing a single table with the fields "table", "field name", "code/format" and "label". This table defines schemata for a set of three tables "Accident", "Vehicle" and "Casualty", which are the primary entities in DfT RTA data. The table also provides mappings of IDs to labels for the data categories present in the DfT RTA data itself. These mappings are essential for correctly interpreting DfT RTA data, where categorical data points are supplied as IDs rather than labels.

Acquisition and relational modelling of DfT RTA data were facilitated by a set of custom Python modules. An SQLAlchemy model was defined for each data category present in the DfT

RTA data guide. Reference data from the DfT RTA data guide was loaded into these models and persisted to a MySQL database. Once reference data had been loaded into the database, the same process was applied for accident (collision), vehicle and casualty data. A comprehensive set of foreign key constraints were used to ensure data integrity. Collisions missing either a latitude or a longitude were discarded.

Initial data quality analysis revealed that for the 2022 DfT RTA dataset, only 22 of the 106,004 raw collision records (0.02%) were missing latitude or longitude. No duplicate rows were found. A naïve, automated comparison with the reference data in the DfT RTA data guide demonstrated that the majority of values map correctly to valid data categories with minimal parsing.

With RTA data modelled and loaded into a relational database, the image generation software was extended through a new module specific to positive sample generation, that adds the command line option `collision-id` for specifying a collision whose latitude and longitude is used as the centre point for the visualisation. In case no collision ID is passed, a random collision is retrieved from the database. A separate script for batch sample generation was also created, sharing functionality with the single sample script but adding concurrency and progress logging.

The batch positive sample generation script was run to generate positive samples. The extent parameter was set to 50 m, on the basis that any geographic feature further than 50 m from the node was considered significantly less likely to affect crash risk, and would likely serve only as unhelpful noise. Interpolated elevation data was used to colourise image backgrounds. All way types were included to accurately represent the complexity of road networks at a small scale, and reduce the chance of a collision location being adjusted to an incorrect way. Ways were colourised according to OSM speed limit data where available. A total of 83,877 positive samples were generated in 4 hours and 8 minutes; 42,030 of these samples were designated for training, and 41,847 were designated for testing. The rough 50/50 split was used to limit model training time and provide an extensive testing resource.

3.3 Generating negative samples

Negative samples in the context of binary classification are data samples that do not belong to the class of interest. In the context of this project, negative samples are locations where collisions have *not* occurred. By randomly selecting waypoints from the road network in Great Britain, with selection probability weighted by traffic volume and way type, a distribution of negative samples that accurately reflects real-world driving patterns can be created. The underlying principle here is that an essential prerequisite for a collision occurring at a given

waypoint is a vehicle passing through that waypoint, which we will call a "vehicle passage". As such, with all other things being equal, collisions are more likely to occur on ways with higher vehicle throughput. In theory, given an infinitely long period of time, dividing the number of collisions into the total number of vehicle passages at a given location should result in a value indicative of the crash risk at that location.

Recording vehicle passage data at *all* waypoints on the UK network is practically impossible given current technologies. However, the DfT does provide a dataset of 4,960,860 raw traffic counts recorded over 24 years (from 2000 to 2023) at 60,075 unique locations. This data can be aggregated by location to get average traffic counts.

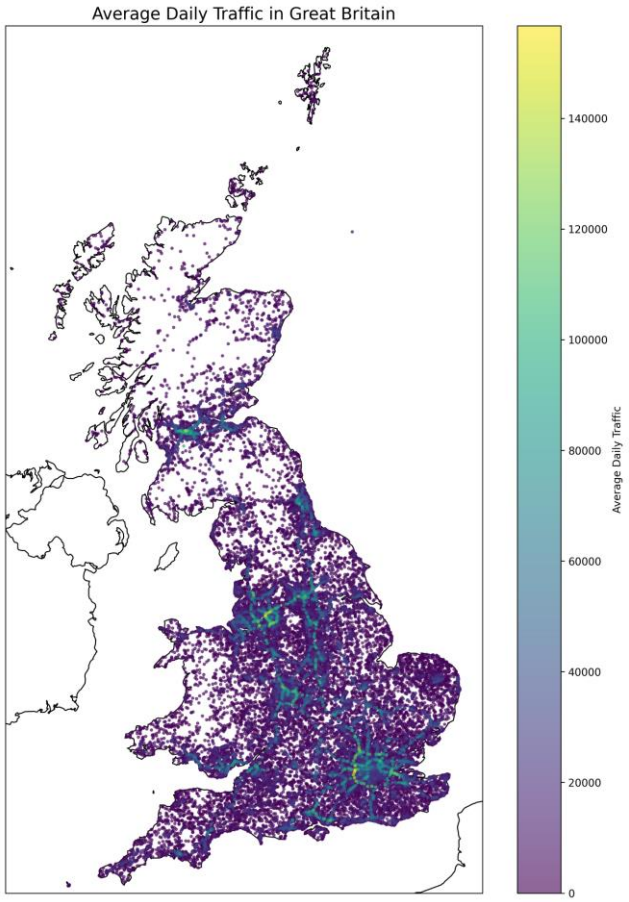

Figure 6: Average traffic counts in Great Britain. Counts are represented as circles of fixed size, with a yellower hue indicating a higher value.

While this average traffic count dataset provides valuable information, further transformation is required before the data can be used within a negative sampling algorithm. The fundamental problem is that traffic counts are recorded at specific geographic points, which are discrete, dimensionless entities in space which offer no direct information about traffic volume at any other point along the continuous road network. This problem is compounded by the sparse

distribution of traffic counts across suburban and rural areas, as well as the fact that traffic volume can vary significantly over short distances (Thomas *et al.*, 2008).

To ensure that any given way in Great Britain is associated with traffic volume data, an interpolated grid is used. Raw, point-based data is transformed into a continuous surface of polygons or cells, allowing for assignment of traffic volumes to ways based on their geometric intersection with traffic volume-labelled cells. A regular grid covering the geographic extent of Great Britain is first constructed, where each cell represents an area of 250 m x 250 m. This area size was chosen to balance granularity, which is naturally limited anyway by the sparseness of the raw data, with computational efficiency. Traffic volumes are assigned to each grid cell by taking the average of the average traffic counts recorded within the geographic bounds of the cell. For the vast majority of grid cells which completely lack traffic data, an interpolation technique known as Inverse Distance Weighting (IDW) is applied. IDW is a deterministic method for multivariate interpolation that estimates values for unknown points based on the values of known points, with the influence of known points diminishing as distance increases (Amini *et al.*, 2019). Parallel processing is used to optimise the performance of the computationally expensive interpolation process.

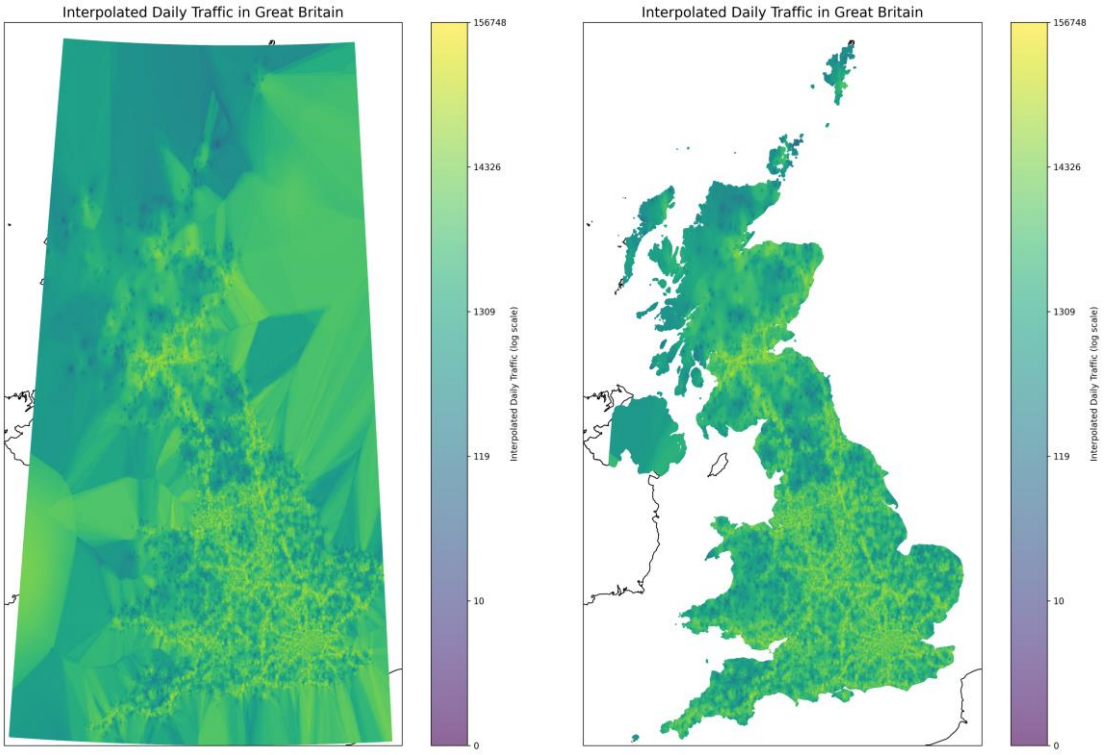

Figure 7: Interpolated traffic data. On the left, the full extent of the grid is colourised, including marine areas, where colourisation appears heavily distorted thanks to the lack of data to guide interpolation. The overall distortion in the shape of the grid is a result of the OSGB projection used in visualisation. On the right, a UK GeoJSON shapefile was used to "mask" the grid data, providing a more intuitive

representation of traffic volume across Great Britain. No traffic data counts are recorded for Northern Ireland; its colourisation here is simply a byproduct of the UK shapefile filtering and the fact that the grid is a simple rectangle based on the minimum and maximum latitude and longitude values of Great Britain.

The negative sampling process relies on integration of OSM and traffic data. OSM data for Great Britain can be downloaded from a website like GeoFabrik and imported into a PostGIS database using the command line tool osm2pgsql, with a custom "style" file allowing import to be limited to relevant data, which saves time and storage space. PostGIS natively supports complex geometric operations, which will facilitate the spatial joining of OSM to traffic data based on intersection between the line geometries of ways and the polygon geometries of traffic grid cells. To support this, the Python script used to add traffic grid data to the PostGIS database also creates and indexes a geometry column on the table, populating this column with polygon data calculated from the latitude and longitude bounds of the corresponding cells.

The negative sampling algorithm is multi-stage. Firstly, a sample of % of the available OSM ways are taken, using the Bernoulli sampling method for truly random row-level sampling. Secondly, sampled ways are spatially joined to traffic grid cells. In the case of a way intersecting multiple traffic grid cells, traffic volume is averaged across those cells. Thirdly, sampled ways are filtered based on a combination of way length, traffic volume and way type. We will discuss this filtering behaviour in more detail in the paragraphs below. Fourthly and finally, a random point is selected on each filtered way. This is done by generating a random point within the bounding box of the way and then projecting it onto the way.

The way length filter component is important because it ensures that when selecting random points on selected ways in the final query, the probability of selecting any given point on the road network as a whole is uniform; without this filter, shorter ways would be overrepresented. The way type filter component allows us to bias way selection according to way type, which is important because of the low fidelity of traffic data, with sparse traffic counts and relatively large grid cells that each incorporate numerous ways of different types. This helps to mitigate localised variation in traffic volumes.

Way type weights can be based either on traffic volume or collision frequency, depending on the characteristics desired in the negative sample set. Biasing way selection towards ways based on traffic volume is more naturalistic and potentially representative of real-world driving patterns; biasing way selection towards ways based on collision frequency may produce a negative sample set that more closely matches the positive sample set, helping a model learn to discriminate based on highly localised geometric features. Regardless of methodology, tools

from our image generation software application, specifically the OSM client and the node adjustment algorithm, are leveraged in the generation of way type weights.

Two Python scripts were created to leverage this algorithm – a script for generating a single negative sample, in which a single filtered way is randomly selected, and a script for generating multiple negative samples, which leverages parallel processing and batch retrieval of filtered ways for performance optimisation purposes. The batch negative sample generation script was run to generate negative samples. The same configuration parameters that were used for positive sample generation were re-used for negative sample generation, which is crucial to ensure that any differences between positive and negative samples are genuine and not merely artefacts of the generation process. Way type weights for biasing negative sampling were calculated based on traffic volume. A total of 84,960 negative samples were generated in 4 hours and 34 minutes; 42,478 of these samples were designated for training, and 42,482 were designated for testing.

Chapter 4

Training the convolutional neural network

4.1 Introduction

Having described the process through which training data was generated for a binary classification CNN in detail, we now turn our attention to the training itself. The training process exists to allow the CNN to learn how the presence of and interactions between a variety of static road safety factors influences crash risk. Our end goal is not to produce a model capable of outputting the real-world probability of a collision occurring at a particular location; a numerical index value that can be used as a point of comparison suffices. Given the fact that RTAs are exceedingly rare events in the real world, with the UK DfT estimating that there were only approximately five fatalities per billion vehicle miles travelled in 2022 (Department for Transport, 2023), training the model as a direct probability estimator would be particularly challenging due to the difficulty of representing the real-world distribution in the training data, as well as the inevitable class imbalance concerns that would arise (Niaz *et al.*, 2022).

4.2 Data preparation

As our training data is generated by custom software specially designed for the task, image preprocessing in the context of the model training phase is not necessary. The software ensures images are created with consistent dimensions, colour schemes and feature representations. Images are sized at 250 pixels square to balance fidelity with memory and storage considerations. Data augmentation is limited to horizontal and vertical flipping, which introduces variation on a per-epoch basis without compromising geospatial integrity. These transformations may enhance the model's generalisability to right-hand traffic systems, which are prevalent in the majority of countries worldwide.

A balanced dataset with an equal number of samples of each class gives the model exposure to a number of positive instances large enough to facilitate the learning of features characteristic of collisions, while also providing a contrastive sample set for class separability. This is beneficial for model generalisation (Dharmasaputro *et al.*, 2022) and feature importance understanding (Zaza *et al.*, 2023). The balance will mean that the penalty for misclassification of either class is equivalent, which prevents bias towards the majority class (Dablain *et al.*, 2022).

An 80:20 validation split gives us a sufficient volume of training data for model learning and feature extraction, while reserving enough validation data for robust performance evaluation.

This ratio is considered standard for balanced model training and validation in machine learning tasks.

4.3 Model architecture

The model architecture used is primarily comprised of a series of convolutional layers, which are the fundamental distinguishing feature of CNNs. Convolutional layers enable hierarchical feature learning, from low-level patterns through to high-level concepts (Qi *et al.*, 2017). A non-linear activation function is applied to the output of each convolutional layer, creating decision boundaries of arbitrary complexity that enable the network to learn complex patterns (Sharma *et al.*, 2017). ReLU is chosen as the activation function for its computational efficiency and effectiveness in mitigating the vanishing gradient problem in deep networks (Alkhouly and Hefny, 2021). Max pooling layers are interspersed throughout the network, reducing spatial dimensions to mitigate overfitting and promote translation invariance (Mouton *et al.*, 2020), which allows the network to recognise features regardless of their exact position in the image. The architecture culminates in a series of three dense (fully-connected) layers, with the final layer utilising a sigmoid activation function to produce a single output value between 0 and 1, representing the probability of the input belonging to the positive class.

Mixed precision training is employed to optimise performance and memory usage. This technique utilises both 16-bit and 32-bit floating-point representations of model parameters, activation values, backpropagation gradients and intermediate computations during training, accelerating the process while maintaining model accuracy (Micikevicius *et al.*, 2017).

The decision was taken not to use a pre-trained model because of two main factors. Firstly, pre-trained CNNs are typically trained for object recognition from photographs, while our training data samples are synthetic, map-like images that exhibit fundamentally different visual characteristics. This disparity could lead to negative transfer, where features learned by the pre-trained model are not applicable and may even have an adverse effect on learning (Wang *et al.*, 2019). Secondly, training data is not scarce in our case, as we have access to extensive RTA data for generating positive samples and a theoretically infinite pool of negative samples.

4.4 Hyperparameters

The training process is configured with a batch size of 16, a maximum of 25 epochs, and an early stopping “patience” value of 5. The relatively small batch size is chosen due to GPU memory constraints, which are a common limitation when working with CNNs, while the epoch limit was chosen based on preliminary experiments indicating that convergence typically occurs around this point. Early stopping is implemented to prevent overfitting (Ying, 2019), halting training when validation loss ceases to reduce for five consecutive epochs. Model

checkpointing ensures that the best-performing model iteration is saved, based on validation loss.

The Adam optimiser is utilised for its adaptive learning rate capabilities, which can lead to faster convergence and better performance across a wide range of deep learning tasks (Zhang, 2018). Binary cross-entropy serves as the loss function, appropriate for this binary classification problem and aligning with the sigmoid activation in the output layer (Ruby and Yendapalli, 2020).

4.5 Training execution

The model was trained on 84,060 images equally divided between the positive and negative classes. Training was performed on a Dell XPS 15 laptop with an NVIDIA GeForce RTX 3050 Ti Laptop GPU, and took 4 hours, 1 minute and 30 seconds.

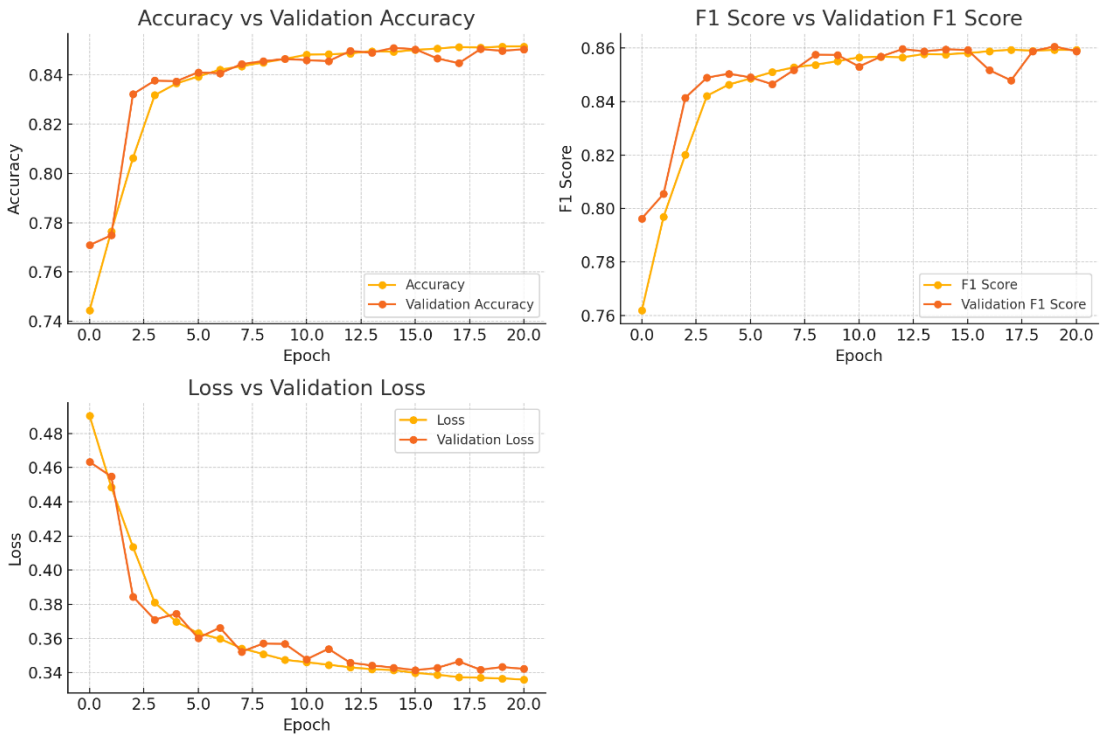

Figure 8: Training and validation metrics for the CNN model over 21 epochs.

The training accuracy rapidly increases and stabilises around the fifth epoch, closely followed by the validation accuracy, indicating strong model generalisation without overfitting. Training and validation F1 scores show a similar pattern. Given the equal number of positive and negative samples in the training dataset, it is unsurprising to see close correlation between accuracy and F1 score. Overall, the model converged effectively, with minimal overfitting.

Chapter 5

Analysing the model

5.1 Introduction

The real-world capability of the CNN developed in this project cannot be fully captured by traditional performance metrics alone, as these metrics are not only dependent on the accurate representation of features in the generated images but also critically on the contrast between the geographic distributions of positive and negative samples. The geographic distribution of negative samples is shaped by the assumptions made during the negative sampling process; therefore, the model's performance reflects the validity of these assumptions. Poor assumptions during negative sampling could, paradoxically, result in stronger performance metrics, thereby making these metrics less reliable indicators of the model's true effectiveness.

An alternative way to measure the abilities of the model is to extract categorical and numerical features from training samples, analyse how these features in isolation affect the model's predictions, and then frame findings in the context of existing research into the effects of those same features on crash risk. The idea will be to determine whether the model is fundamentally sound in its understanding of key static road safety factors.

5.2 Model performance overview

With the above caveats in mind, the model was tested against 84,329 unseen test samples, comprising 41,847 positive samples and 42,482 negative samples.

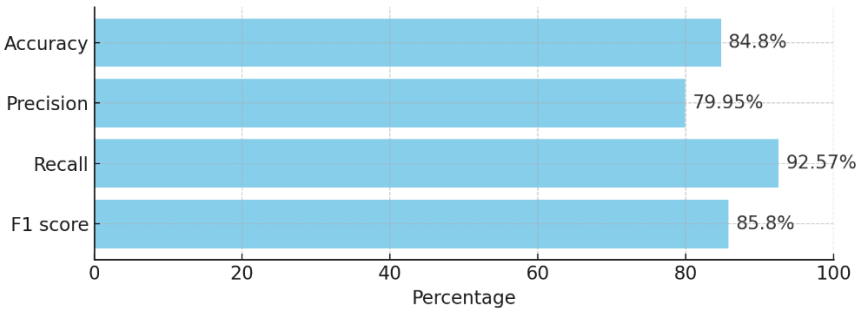

Figure 9: Performance metrics: accuracy: 84.8%; precision: 79.95%; recall: 92.57%; F1 score: 85.8%.

5.3 Feature analysis

A variety of static road safety factors were selected as features, based on their documented importance in crash risk as well as the technical complexity in their extraction. These features are as follows:

- **Average speed limit**: the average speed limit in miles per hour (mph) across all ways in the image. Ways without speed limit are ignored
- **Elevation range**: the difference in metres between the highest and lowest elevation points in the image
- **Presence of roundabout**: whether a roundabout is present within the image
- **Intersection count**: the number of intersections within the image. An intersection is a point at which two ways cross
- **Intersection proximity**: the distance in metres between the central node and the nearest intersection, if one exists in the image
- **Junction count**: the number of junctions in the image. A junction is a point at which two ways touch but do not cross
- **Junction proximity**: the distance in metres from the central node to the nearest junction, if one exists in the image
- **Node way curvature**: the curvature of the way on which the central node is located, calculated as the ratio of the length of the way to the straight-line distance between its start and end nodes. This means that a straight line would have a curvature of 1.0, with higher values indicating more curvature
- **Node way lanes**: the number of lanes on the way where the central node is located
- **Node way one-way**: indicates whether the way at the central node is one-way
- **Node way speed limit**: the speed limit in mph of the way on which the central node is located
- **Node way type**: describes the type of way at the central node (e.g., "motorway")
- **Way count**: the total number of ways in the image
- **Way length**: the total length in metres of all ways in the image; approximates “density”

Feature extraction is built in to the image generation software, with the main data structures offering interface methods that expose these features while abstracting away the implementation detail. Generated features are saved as metadata JSON files during batch sample generation. During model testing, these metadata files were associated with their true labels and predicted crash risk indices to create a table for feature analysis.

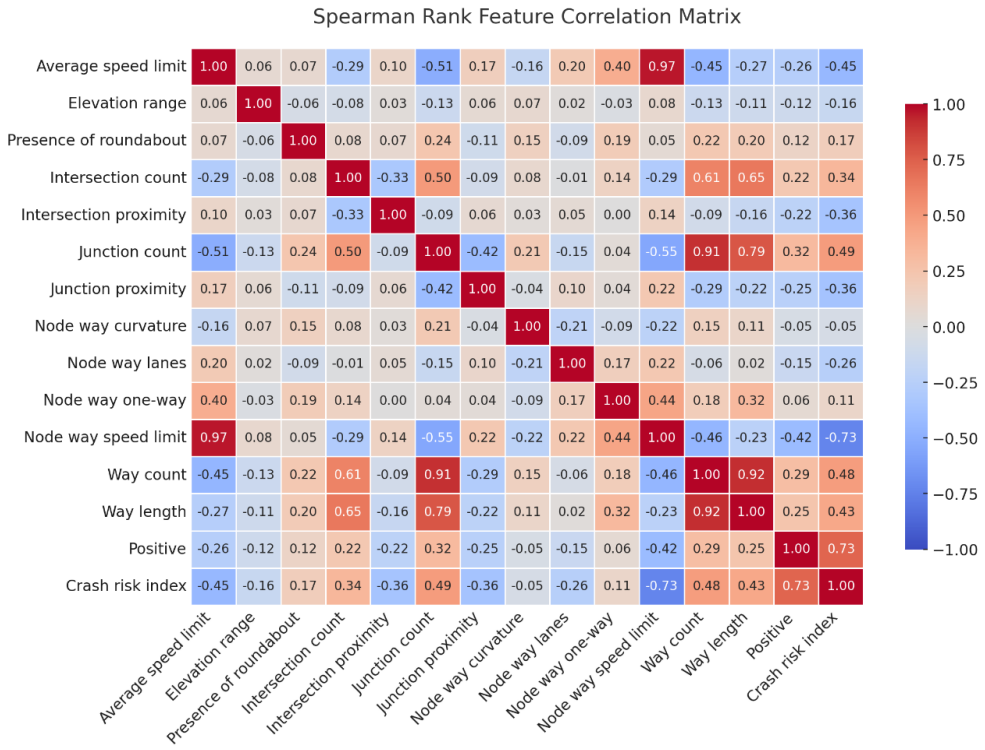

Figure 10: Spearman's rank feature correlation matrix showing the correlation between evaluated features, the indicator "Positive" (a binary value representing whether the sample is positive or negative) and the crash risk index. Spearman's rank was used owing to its ability to handle outliers (present in e.g., intersection count) or features with nearly uniform distributions (e.g. node way curvature).

The correlation feature matrix reveals that junction count, way count and way length are the three features that have the strongest positive correlation with crash risk index, providing an immediate indication that the model is learning to identify positive samples through the presence of denser, more interconnected networks of ways. This tracks with reality because such configurations are more challenging to navigate, with greater interaction between vehicles and an increased likelihood of collisions at conflict points (Wu *et al.*, 2019). Strong negative correlations between crash risk index and each of the two speed limit factors – average speed limit and node way speed limit – are counter-intuitive, owing to the documented increased risk of collisions at higher speeds (Aarts and Van Schagen, 2006), making this a candidate for deeper investigation. Meanwhile, the moderate negative correlation between junction proximity and crash risk index, as well as intersection proximity and crash risk index, indicates that crash risk index decreases as proximity increases, which is to say that the further we get from a junction or intersection, the lower the predicted crash risk. This again tracks with reality because of the increased decision-making demands on drivers given reduced proximity to junctions and intersections, and the potential for sudden braking. It may also be a function of the fact that junctions and intersections are natural conflict points: geographic imprecision in DfT RTA data may scatter collisions occurring at these points around these points in our training data, meaning

that the further we move from a junction or intersection, the less likely it is that a collision will be recorded as occurring nearby, thus reducing the predicted crash risk. Features such as elevation range, node way curvature and node way one-way do not appear to factor significantly into the model's predictions.

The intuitive covariance between features related to way density, including way length, way count, junction count, junction proximity, intersection count and intersection proximity, is clearly apparent from the feature correlation matrix. To isolate the underlying effects of these features, multiple regression analysis was performed using Ordinary Least Squares (OLS) regression, with Multiple Imputation by Chained Equations (MICE) used to handle missing data values.

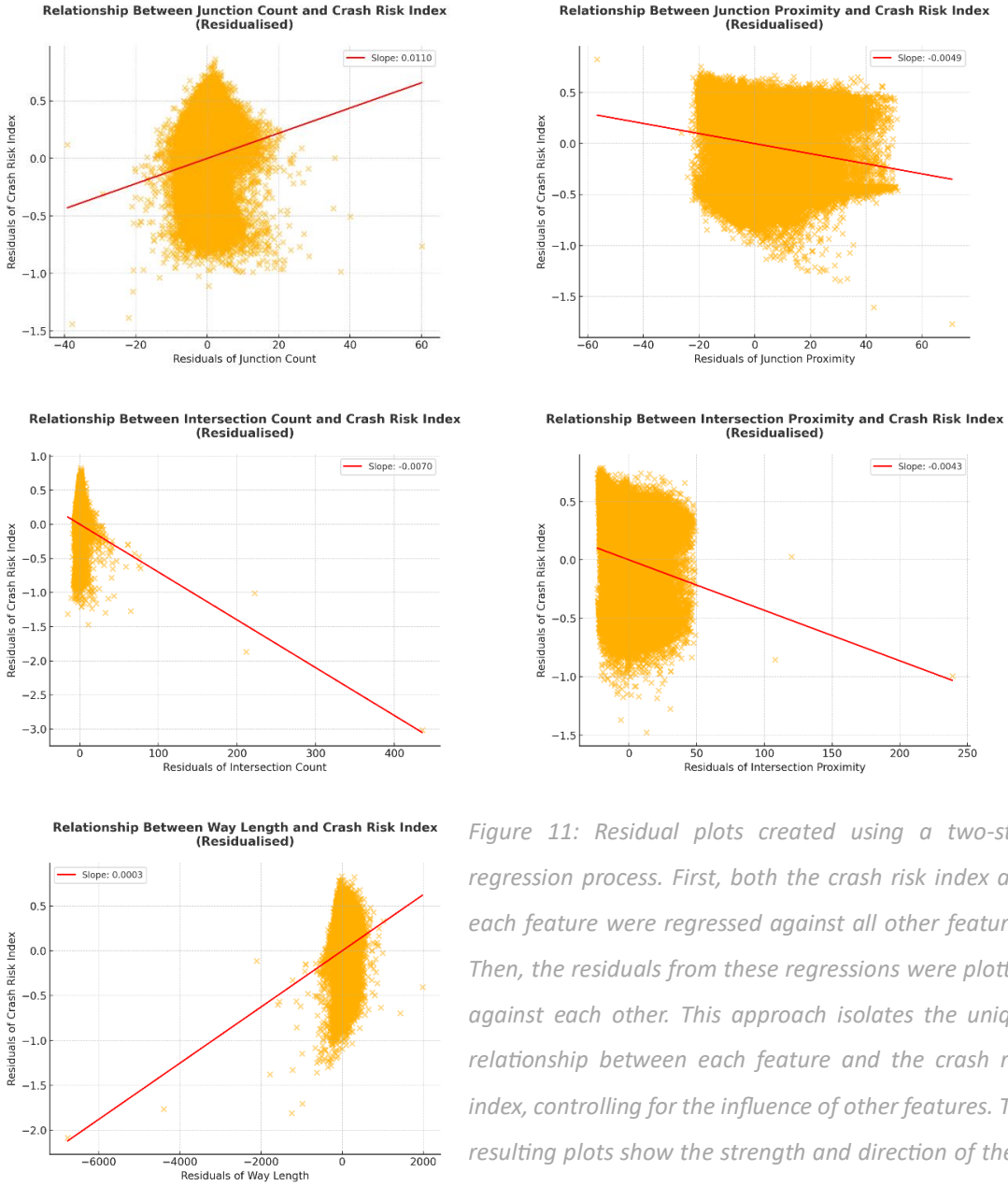

Figure 11: Residual plots created using a two-step regression process. First, both the crash risk index and each feature were regressed against all other features. Then, the residuals from these regressions were plotted against each other. This approach isolates the unique relationship between each feature and the crash risk index, controlling for the influence of other features. The resulting plots show the strength and direction of these isolated relationships.

Analysing the effect of each feature in isolation reveals some interesting insights, with statistically significant relationships found between each of the features and crash risk index. Total way length has only a marginal effect on predicted crash risk (coefficient: 0.000313), indicating that our model is not simply predicting higher risk based on the total amount of road present, but rather considering more complex interactions between features. The feature that correlates most strongly with crash risk index is junction count, with a meaningful effect size (coefficient: 0.010956) that underscores the salience of connection points between ways in model learning. Moreover, the negative correlation observed between crash risk index and both junction proximity (coefficient: -0.004946) and intersection proximity (coefficient: -0.004319) is retained even after controlling for the effects of covariant features. The importance of intersection proximity in particular will be spoken about in the next subsection, "Prediction and error analysis".

The negative correlation between speed limits and crash risk index indicates that the model appears to have learnt to associate higher speed limits with lower crash risk, which is not immediately consistent with intuition or research findings. While it is true that higher-speed ways such as motorways are relatively safe (Hovenden *et al.*, 2020), by artificially manipulating the speed limit feature in a sample image to observe the effect on model prediction (feature sensitivity analysis), we can observe that the model's bias in relation to speed limit does not appear to be legitimate.

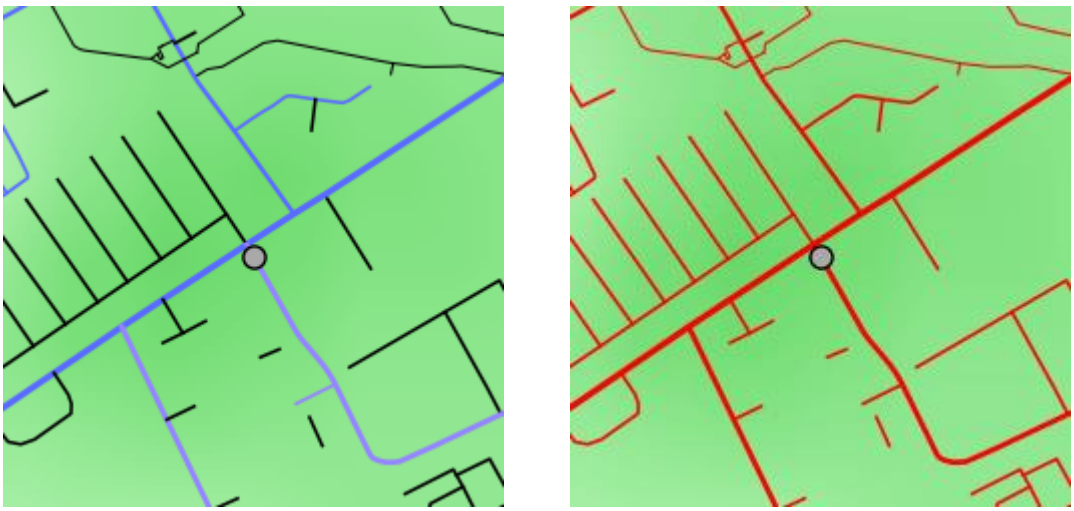

Figure 12: On the left, an image generated using our positive sample generator, with ways colourised according to OSM speed limit data. On the right, the same image but artificially manipulated to colourise all ways as red, the colour used to indicate a speed limit of 70 mph in the training data. The model predicted a crash risk index of 0.9453 for the image on the left, and 0.6064 for the image on the right.

The bias is likely an unwanted artefact of deficiencies in the negative sample generation algorithm, although due to the nature of the problem that that algorithm exists to resolve, it is

inherently difficult to confirm how far real-world traffic flow across the network is being misrepresented. We can however see in the testing data that lower speed limits are much more common in the node ways of positive samples than negative samples.

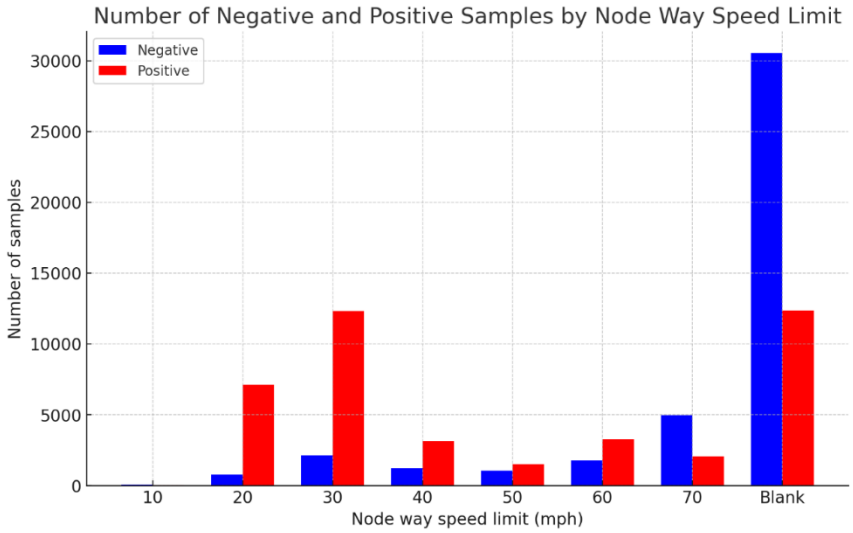

Figure 13: Number of negative and positive samples by node way speed limit in the testing data.

Exacerbating the problem with speed limits, there is an inescapable correlation in the sample data between higher speed limits and smaller values for way length, junction count and intersection count, and higher values for junction and intersection proximity, all feature directions which have been shown to independently exert a negative effect on crash risk index.

Node way speed limit (mph)	Average way length	Average junction count	Average junction proximity	Average intersection count	Average intersection proximity
10	488.18	7.46	17.15	2.03	22.39
20	422.47	6.62	14.44	2.05	19.04
30	404.39	6.06	14.87	1.61	21.07
40	401.17	4.74	17.95	1.39	23.04
50	353.94	2.58	23.05	0.95	24.15
60	213.11	1.58	19.28	0.22	21.64
70	336.50	0.77	31.97	0.48	29.14

Figure 14: Correlation between speed limits and features associated with way density in testing data.

Put simply, there are literally no real-world examples available of high speed limits in dense, residential areas, due to centralised regulation of road infrastructure based on the implied crash risk of such configurations, and this means that our model is not given the data it needs to learn that such configurations are dangerous. Human intervention here has led to a form of survivorship bias, a common problem in data science (Slaper *et al.*, 2019; Pasqualetti *et al.*, 2023). The final thing to note in relation to speed limits is that collision severity is not visually encoded in sample images and is therefore not accounted for by our model. The ambivalence

to collision severity is not an oversight but a defining characteristic of the model, which functions only as a predictor of the risk of *any collision occurring*; however, incorporating collision severity as a feature might help to produce a more usable model. Refinement of the negative sampling algorithm to more accurately calculate traffic flow may also help to mitigate or eliminate counter-intuitive effects.

5.4 Prediction analysis

Although the residual analysis in the preceding section did not show intersections to be any more important as a predictor than way length or junctions, it is clear from examining the test samples with the highest crash risk indices that they are an influential factor on model learning. 95.9% (71 / 74) of samples with a crash risk index of >=0.999 have at least one intersection, compared to 33.7% (28,415 / 84,329) in the general population, while the average intersection proximity of those 71 high-risk samples with intersections is only 6.19 m, compared to 22.37 m among samples with intersections in the general population.

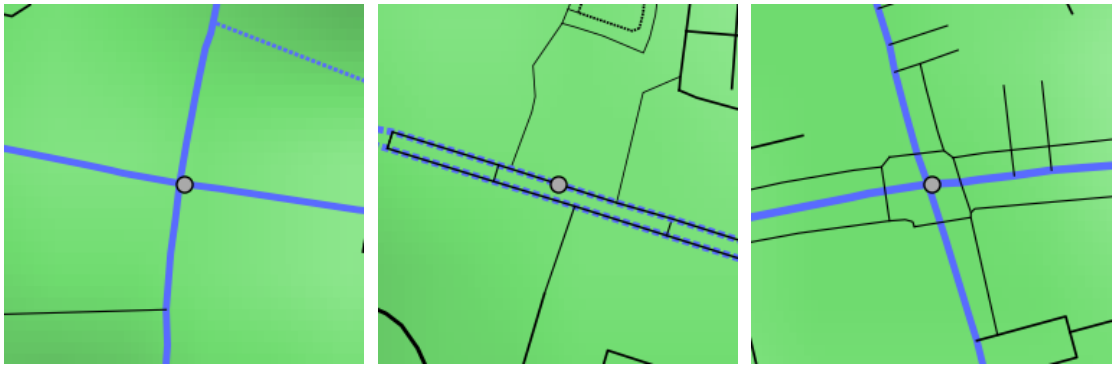

Figure 15: The three test samples with the highest predicted crash risk indices. All are positive instances. From left to right: collision ID 2022010377492 (CRI 1.000), collision ID 2022010404565 (CRI 0.999) and collision ID 2020010229063 (CRI 0.999).

Intersections also appear in many of the most extreme false positive cases. In the most extreme false positive case (see below), an intersection appears in fact to be the only feature of note. From this, it appears that the model is learning that ways that intersect at an angle near to the perpendicular are more dangerous. Another common cause of false positives is extreme way density around the central node, which is apparent for example in the negative test sample with the second-highest crash risk index (see below).

Figure 16: The three negative samples with the highest predicted crash risk indices. From left to right: 55.69197, -2.86194 (CRI 0.999), 51.51087, -0.08672 (CRI 0.998), 51.51486, -0.12492 (CRI 0.998).

At the other end of the predicted crash risk spectrum, the test samples with the lowest indices share a fascinating similarity. The model appears to have learnt to associate collisions with straight, unintersected lines passing through the central node and extending at a very specific angle in the roughly northeast direction. Although straight roads are associated with lower crash risk (Chen, 2006), there is no obvious reason why this specific angle should be. This is an example of how even a sophisticated neural network architecture can misinterpret feature importance, especially given limited data.

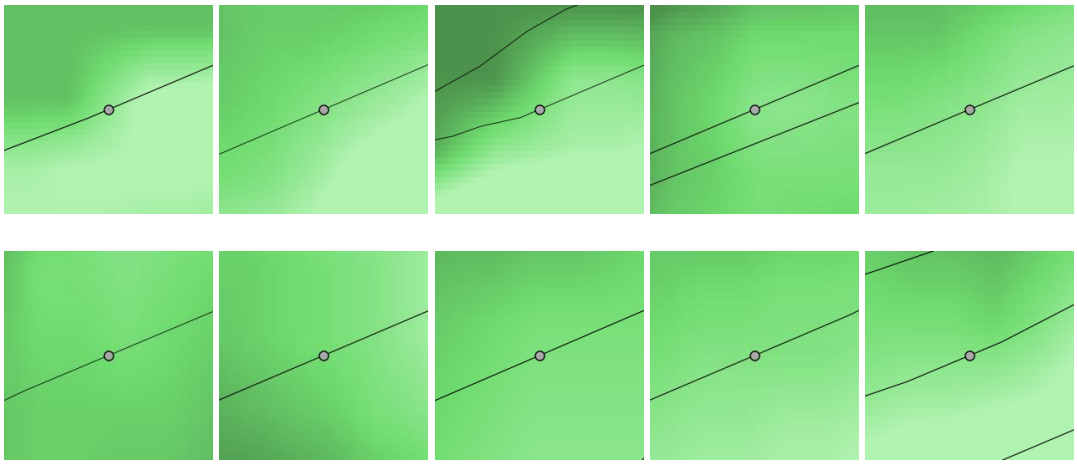

Figure 17: The ten test samples with the lowest predicted crash risk indices. From left to right: 53.74966, -2.79472 (CRI 0.0001584); 50.90361, -1.71187 (CRI 0.0001597); 54.72848, -3.48690 (CRI 0.0001622); 52.78997, -1.58001 (CRI 0.0001622); 51.98218, 1.24466 (CRI 0.0001622); 51.27020, -1.61454 (CRI 0.0001636); 53.82936, -2.17033 (CRI 0.0001647); 53.68375, -2.55163 (CRI 0.0001647); 53.49972, -2.47852 (CRI 0.0001647); 53.38850, -1.52072 (CRI 0.0001647).

Chapter 6

Reflection

6.1 Summary of work

In this project a software application was created for generating map-like images representing the static road safety factors around specific locations, by integrating various data sources including OSM and SRTM. RTA data was collected, analysed, modelled and stored, with the precise geographic labels in the data enabling us to use the software application to visualise collision locations. An algorithm was created for efficiently selecting random waypoints from the Great Britain road network, as represented in OSM data. Traffic count data from local authorities was used to bias waypoint selection to higher traffic geographic areas and way types, in the former case via the creation of an interpolated grid of traffic data whose polygonal cells can be spatially joined to ways through intersection. This process was created to facilitate the generation of a set of map-like images whose geographic distribution mirrors real-world driving patterns (negative samples), to be contrasted against images generated around collision locations (positive samples). Positive and negative samples were generated in large batches using performance-optimised batch generation scripts, and then used to train a CNN as a binary classifier. To facilitate a more effective analysis of model performance, the image generation software application is able to calculate numerical features (e.g., intersection proximity) from the geographic data that is being visualised, during image generation. The model was tasked with predicting crash risk for an unseen set of feature-labelled samples, with the predicted indices stored alongside the features and true labels. Correlation between individual features and crash risk indices was analysed, as well as inter-feature correlation, to get information on feature importance and covariance. Multiple regression was used to isolate the underlying feature importance of covariant features. Finally, individual crash risk prediction cases were examined, focusing on extremes and misclassified samples, in a further attempt to gain an intuitive understanding of the inner workings of the model.

6.2 Critical analysis of objective fulfilment

The objectives of the project were threefold – to create a flexible and extensible software application for generating images visualising the static road safety factors around specific locations; to train a CNN to predict crash risk using this application; and to analyse the abilities of this model.

The first objective – to create a flexible and extensible software application for generating images visualising the static road safety factors around specific locations – was achieved through methodical, iterative engineering, with functional and non-functional requirements used to guide the software development process at all stages. With extension and reuse in mind, the application was designed to be highly modular. Polymorphic interfaces and inheritance allow for the integration of diverse data sources, segmented not only based on content (OSM vs SRTM), but also by retrieval method (API vs local database access). A naturalistic ontology of custom geographic data structures can be easily manipulated in common gateways thanks to the expressive interfaces that they expose. Low-level implementation details are handled by leveraging common geospatial libraries in most cases, ensuring efficiency and robustness and increasing maintainability. Overall performance can be optimised hugely by following well-documented steps to store frequently accessed data sources locally. The fulfilment of functional requirements was evidenced by rigorous output validation throughout development, as well as quality assurance of batch-generated samples.

The second objective – to train a CNN to predict crash risk using this application – was also achieved, with a model trained using sample images generated by our software application. This model performed well on standard performance metrics, although as discussed in the introduction to the section “Analysing the model”, this is not evidence in itself of real-world applicability.

The third and final objective – to analyse the abilities of this model – was achieved through a detailed global and local analysis of the model’s predictions. This analysis revealed that the model appears to have successfully learnt from features that have been shown by other studies to affect crash risk, including junction density and intersection proximity. The analysis also exposed imperfections in the model, with counter-intuitive results observed during feature analysis confirmed as aberrant through granular, instance-level prediction analysis, including feature sensitivity analysis. Potential biases in the sample data were identified, particularly in relation to speed limit, with our negative sampling algorithm implicated. The algorithm in its current form, whilst meritorious, is likely too rudimentary to model real-world traffic flow sufficiently well for the model to be used in a practical setting.

6.3 Data limitations

Before proceeding with calls for further work, it is important to acknowledge the importance of data in machine learning, and the consequent significance of the limitations inherent in the data sources relevant to static road safety factor analysis. All of the data sources used in this project were imperfect in ways that impacted the performance of our crash risk prediction model to a variable degree.

Although the UK DfT's RTA datasets are clean, comprehensive and extensive, there are also issues. For example, running all collisions from the 2020, 2021 and 2022 DfT RTA datasets through the image generation software with the node translocation distance limit of 5 m strictly applied results in 16.1% of collisions erroring and being discarded, which is to say that 16.1% of collisions are labelled with geographic co-ordinates that do not appear to fall within 5 m of an OSM way. This indicates either imprecision in recording or misalignment between data sources. Further, vehicle directionality data associated with collisions is not useful in its current form: intercardinal categories (e.g., east, southeast) are used that do not appear to map obviously to way geometries around collision locations.

OSM data is crowd-sourced, making it patchy and at times unreliable (Teimoory *et al.*, 2021). Way tags indicating speed limit (tag: maxspeed), directionality (tag: oneway) and number of lanes (tag: lanes) are unavailable for the majority of ways and the crowd sourcing means that even where tags are present, they cannot always be trusted.

	All ways		Minor ways		Medium ways		Major ways	
Category	**Count**	**% Total**	**Count**	**% Total**	**Count**	**% Total**	**Count**	**% Total**
Tag: junction	59397	0.8%	11	0.0%	25696	0.6%	33323	11.8%
Tag: lanes	470899	6.7%	1425	0.1%	263673	6.5%	204135	72.2%
Tag: maxspeed	1153233	16.3%	3663	0.1%	823819	20.2%	262127	92.7%
Tag: oneway	589901	8.4%	7879	0.3%	368252	9.0%	169017	59.7%
All	7062703	100.0%	2464655	100.0%	4077717	100.0%	282913	100.0%

Figure 18: Tag availability by way type. Minor way types include pedestrian, track, bus_guideway, escape, raceway, road, busway, footway, bridleway, steps, corridor, path, and via_ferrata. Medium way types include secondary, secondary_link, tertiary, tertiary_link, unclassified, residential, living_street, and service. Major way types include motorway, motorway_link, trunk, trunk_link, primary, and primary_link.

Further, the categorisation of way types is ambiguous, with the definitions of way types such as "primary", "secondary" and "tertiary" open to subjective interpretation. Even more pertinently, a variety of persistent geographic features that are likely to affect crash risk fall entirely outside the jurisdiction of OSM, owing to perceived importance and practical limitations. These might include factors such as actual road width (in metres), presence and quality of road markings, or the amount of roadside foliage affecting visibility at junctions. Ultimately, there will always be a gap between the complexity of the real world and the data used to model it, but continually striving to close this gap is essential for building models that are better at solving problems.

The issue with SRTM data is fidelity. The highest-resolution SRTM dataset has a resolution of 1 arc-second, which equates to approximately 30 m at the equator. East-west resolution increases at higher latitudes, but north-south resolution remains constant. While this resolution

is suitable in most cases, it does mean that highly localised changes in elevation may not be accurately captured.

Traffic data was a particular problem for this project, considering the importance in binary classification of the contrasting effect from negative samples, and the paramountcy of traffic data in informing the geographic distribution of the negative sample set used for model training. Raw traffic counts are geographically discrete and sparsely distributed, forcing the use of techniques such as interpolation to map traffic data to geometries including ways and waypoints, introducing inaccuracy. It is noted however that there is a sufficiently large quantity of raw traffic counts to suppose that a more sophisticated approach to negative sampling might yield a strong representation of traffic flow at any given waypoint.

6.4 Suggested future work

From a practical perspective, although the image generation software application built for this project is robust and was designed to be flexible enough to support integration with domain-specific software applications, the crash risk prediction model needs further interrogation and possibly re-designing before it can be used in a real-world setting. The software and the model are intended to work co-operatively: the images generated by the software do not offer obvious value without the model; while the model, as a result of its having been trained on images produced by the software, will always remain dependent on the software for the encoding of domain-specific data in a way that it can interpret. However, assuming future work leads to a more robust model, then a variety of interesting applications become possible, including tools for urban planning, road safety auditing, and road maintenance prioritisation.

As a thought experiment, let's imagine an engineering consultancy has been commissioned to add a new roundabout to an existing road to help manage traffic flow. An employee of this consultancy could interact with a frontend layer that sits on top of our image generation software, entering polyline data that represents the roundabout through a user interface. The image generation software, enhanced to dynamically integrate polyline data with existing OSM data in-memory, could render a visualisation of the altered road network including the roundabout, input it into the crash risk prediction model, and return the image alongside the crash risk index to the frontend layer for display. The employee could iteratively make adjustments to the polyline data, observing the effects on crash risk index until a result under a certain acceptable threshold of risk is achieved.

Regarding future work on improving the model, there are numerous potential options that vary significantly in complexity. Although data is paramount, it is possible that further hyperparameter tuning, for example incorporating zoom as a method of data augmentation, may

result in a stronger model (Agustin *et al.*, 2020). Adding channels to the input tensor could help us represent geographically layered information in a naturalistic way, allowing us for example to leverage OSM's layer tag to help distinguish between ways that actually intersect at ground level and ways that cross at different elevations, e.g., overpasses. Additional tensor channels could also be used to incorporate more geographic features, with the natural segregation of layers mitigating concerns about visually encoding features in a distinct way. Non-geographic data points such as collision severity could also be incorporated during training to enable the model to produce a crash risk index that more closely aligns with human notions of risk. Missing OSM data could be compensated for using a process of data imputation. The algorithm by which random waypoint selection is biased to mimic real-world traffic flow (negative sampling) could be refined, perhaps using machine learning techniques, or via improvements to data pre-processing, such as adjusting raw traffic count data to handle chronological bias (e.g., historic traffic counts could be scaled up or down based on general patterns over time). Finally, the use of datasets from non-UK countries could be explored.

6.5 Concluding remarks

This project has demonstrated both the potential and challenges of applying computer vision to the analysis of static road safety factors. While we have successfully developed and operationalised a software application for image generation that should find future use, and trained a crash risk model that exhibits promising performance characteristics, the work also highlighted limitations in relevant data sources that may inhibit such a model from being practically applicable in the near-term. It is hoped that the work done on this project can serve as both a theoretical and practical foundation for future work in the domain of road safety analysis, with the ultimate aim of helping to reduce the devastating impact RTAs have on human lives.

References

Aarts, L. and Van Schagen, I., 2006. Driving speed and the risk of road crashes: A review. *Accident Analysis & Prevention*, *38*(2), pp.215-224.

Agustin, T., Utami, E. and Al Fatta, H., 2020, November. Implementation of data augmentation to improve performance CNN method for detecting diabetic retinopathy. In *2020 3rd International Conference on Information and Communications Technology (ICOIACT)* (pp. 83-88). IEEE.

Alkhouly, A.A., Mohammed, A. and Hefny, H.A., 2021. Improving the performance of deep neural networks using two proposed activation functions. *IEEE Access*, *9*, pp.82249-82271.

American Association of State Highway and Transportation Officials (AASHTO), 2008. *Highway Safety Manual*. Washington, D.C.: AASHTO.

Amini, M.A., Torkan, G., Eslamian, S., Zareian, M.J. and Adamowski, J.F., 2019. Analysis of deterministic and geostatistical interpolation techniques for mapping meteorological variables at large watershed scales. *Acta Geophysica*, *67*(1), pp.191-203.

Camacho-Torregrosa, F.J., Pérez-Zuriaga, A.M., Campoy-Ungría, J.M. and García-García, A., 2013. New geometric design consistency model based on operating speed profiles for road safety evaluation. *Accident Analysis & Prevention*, 61, pp.33-42.

Chen, S., Rakotonirainy, A., Sheehan, M., Krishnaswamy, S. and Loke, S., 2006. Assessing crash risks on curves. In *2006 Australasian road safety research, policing and education conference proceedings* (pp. 1-9). Able Video and Multimedia Pty Ltd.

Chen, S., Saeed, T.U., Alinizzi, M., Lavrenz, S. and Labi, S., 2019. Safety sensitivity to roadway characteristics: A comparison across highway classes. *Accident Analysis & Prevention*, 123, pp.39-50.

Dablain, D., Krawczyk, B. and Chawla, N., 2022. Towards a holistic view of bias in machine learning: Bridging algorithmic fairness and imbalanced learning. *arXiv preprint arXiv:2207.06084*.

Department for Transport, 2021. *A valuation of road accidents and casualties in Great Britain: Methodology note*. London: Department for Transport. Available at: https://www.gov.uk/government/publications/road-accidents-and-safety-statistics-quality-and-methodology (Accessed: 1 September 2024).

Department for Transport, 2023. *Reported road casualties Great Britain, annual report: 2022*. London: Department for Transport. Available at: https://www.gov.uk/government/statistics/reported-road-casualties-great-britain-annual-report-2022 (Accessed: 1 September 2024).

Department for Transport, 2024. *Reported road casualty statistics: background quality report*. Updated 30 May 2024. Available at: https://www.gov.uk/government/publications/reported-road-casualty-statistics-background-quality-report.

Dharmasaputro, A.A., Fauzan, N.M., Kallista, M., Wibawa, I.P.D. and Kusuma, P.D., 2022, January. Handling missing and imbalanced data to improve generalization performance of machine learning classifier. In *2021 International Seminar on Machine Learning, Optimization, and Data Science (ISMODE)* (pp. 140-145). IEEE.

Dingus, T.A., Guo, F., Lee, S., Antin, J.F., Perez, M., Buchanan-King, M. and Hankey, J., 2016. Driver crash risk factors and prevalence evaluation using naturalistic driving data. *Proceedings of the National Academy of Sciences*, 113(10), pp.2636-2641.

Dzieżyc, M., Gjoreski, M., Kazienko, P., Saganowski, S. and Gams, M., 2020. Can we ditch feature engineering? End-to-end deep learning for affect recognition from physiological sensor data. *Sensors*, 20(22), p.6535. https://doi.org/10.3390/s20226535.

Hassaballah, M. and Hosny, K.M., 2019. *Recent Advances in Computer Vision: Theories and Applications*. 1st ed. Studies in Computational Intelligence, vol. 804. Springer.

Hovenden, E., Zurlinden, H. and Gaffney, J., 2020. Safety on heavily trafficked urban motorways in relation to traffic state. *Journal of road safety*, *31*(1), pp.51-65.

Islam, M.H., Hua, L.T., Hamid, H. and Azarkerdar, A., 2019, November. Relationship of accident rates and road geometric design. In *IOP Conference Series: Earth and Environmental Science* (Vol. 357, No. 1, p. 012040). IOP Publishing.

Jain, A., Patel, H., Nagalapatti, L., Gupta, N., Mehta, S., Guttula, S., Mujumdar, S., Afzal, S., Mittal, R.S., and Munigala, V., 2020. Overview and importance of data quality for machine learning tasks. *In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20)*, pp.3561-3562. https://doi.org/10.1145/3394486.3406477.

Karlaftis, M.G. and Golias, I., 2002. Effects of road geometry and traffic volumes on rural roadway accident rates. *Accident Analysis & Prevention*, 34(3), pp.357-365.

Lundbäck, M., Persson, H., Häggström, C., and Nordfjell, T., 2020. Global analysis of the slope of forest land. *Forestry: An International Journal of Forest Research*, 94(1), pp.54-69. https://doi.org/10.1093/forestry/cpaa021.

Marshall, W.E. and Garrick, N.W., 2011. Does street network design affect traffic safety? *Accident Analysis & Prevention*, 43(3), pp.769-781.

McClain, B.P., 2022. *Python for Geospatial Data Analysis*. Available at: https://books.google.com.

Mennecke, B.E. and West Jr, L.A., 2001. Geographic information systems in developing countries: issues in data collection, implementation and management. *Journal of Global Information Management (JGIM)*, 9(4), pp.44-54.

Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G. and Wu, H., 2017. Mixed precision training. *arXiv preprint arXiv:1710.03740*.

Mirabello, C. and Wallner, B., 2018. rawMSA: End-to-end deep learning makes protein sequence profiles and feature extraction obsolete. *bioRxiv*. https://doi.org/10.1101/394437.

Mouton, C., Myburgh, J.C. and Davel, M.H., 2020, December. Stride and translation invariance in CNNs. In *Southern African Conference for Artificial Intelligence Research* (pp. 267-281). Cham: Springer International Publishing.

Niaz, N.U., Shahariar, K.N. and Patwary, M.J., 2022, March. Class imbalance problems in machine learning: A review of methods and future challenges. In *Proceedings of the 2nd International Conference on Computing Advancements* (pp. 485-490).

Nilsson, G., 1982. Effects of speed limits on traffic accidents in Sweden.

Parr, S., Wolshon, B., Renne, J., Murray-Tuite, P. and Kim, K., 2020. Traffic impacts of the COVID-19 pandemic: Statewide analysis of social separation and activity restriction. *Natural hazards review*, 21(3), p.04020025.

Pasqualetti, F., Barberis, A., Zanotti, S., Montemurro, N., De Salvo, G.L., Soffietti, R., Mazzanti, C.M., Ius, T., Caffo, M., Paiar, F. and Bocci, G., 2023. The impact of survivorship bias in glioblastoma research. *Critical Reviews in Oncology/Hematology*, p.104065.

Qi, C.R., Yi, L., Su, H. and Guibas, L.J., 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, *30*.

Ruby, U. and Yendapalli, V., 2020. Binary cross entropy with deep learning technique for image classification. *Int. J. Adv. Trends Comput. Sci. Eng*, *9*(10).

Seaver, W.L., Chatterjee, A. and Seaver, M.L., 2000. Estimation of traffic volume on rural local roads. *Transportation research record*, 1719(1), pp.121-128.

Sharma, S., Sharma, S. and Athaiya, A., 2017. Activation functions in neural networks. *Towards Data Sci*, *6*(12), pp.310-316.

Silva, P.B., Andrade, M. and Ferreira, S., 2020. Machine learning applied to road safety modeling: A systematic literature review. *Journal of traffic and transportation engineering* (English edition), 7(6), pp.775-790.

Slaper, T.F., 2019. What regions should we study? How survivorship bias skews the view. *Indiana Business Review*, *94*(2), pp.1-12.

Teimoory, N., Ali Abbaspour, R. and Chehreghan, A., 2021. Reliability extracted from the history file as an intrinsic indicator for assessing the quality of OpenStreetMap. *Earth Science Informatics*, *14*(3), pp.1413-1432.

Thomas, T., Weijermars, W. and van Berkum, E., 2008. Variations in urban traffic volumes. *European Journal of Transport and Infrastructure Research*, 8(3), pp.251-263. ISSN: 1567-7141. Available at: http://www.ejtir.tbm.tudelft.nl.

Treat, J.R., Tumbas, N.S., McDonald, S.T., Shinar, D., Hume, R.D., Mayer, R.E., Stansifer, R.L. and Castellan, N.J., 1979. *Tri-level study of the causes of traffic accidents: final report. Executive summary*. Indiana University, Bloomington, Institute for Research in Public Safety.

Wang, C., Quddus, M.A. and Ison, S.G., 2013. The effect of traffic and road characteristics on road safety: A review and future research direction. *Safety science*, 57, pp.264-275.

Wang, Z., Dai, Z., Póczos, B. and Carbonell, J., 2019. Characterizing and avoiding negative transfer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11293-11302).

World Health Organization (2023) *Global status report on road safety 2023*. Geneva: World Health Organization. Available at: https://www.who.int/teams/social-determinants-of-health/safety-and-mobility/global-status-report-on-road-safety-2023 (Accessed: 1 September 2024).

Wright, P.H. and Baker, E.J., 1976. Factors Which Contribute to Traffic Accidents. *Transportation Planning and Technology*, 3, pp.75-79.

Wu, P., Meng, X., Song, L. and Zuo, W., 2019. Crash risk evaluation and crash severity pattern analysis for different types of urban junctions: fault tree analysis and association rules approaches. *Transportation research record*, *2673*(1), pp.403-416.

Ying, X., 2019, February. An overview of overfitting and its solutions. In *Journal of physics: Conference series* (Vol. 1168, p. 022022). IOP Publishing.

Zaza, S., Atemkeng, M. and Hamlomo, S., 2023, October. Wine feature importance and quality prediction: A comparative study of machine learning algorithms with unbalanced data. In *International Conference on Safe, Secure, Ethical, Responsible Technologies and Emerging Applications* (pp. 308-327). Cham: Springer Nature Switzerland.

Zhang, Z., 2018, June. Improved adam optimizer for deep neural networks. In *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)* (pp. 1-2). Ieee.