Backpropagating to the Future

Evaluating Predictive Deep Learning Models

Patrick Ribu Gorton

Thesis submitted for the degree of
Master in Informatics: Robotics and Intelligent Systems
60 credits

Department of Informatics
Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Spring 2020

Backpropagating to the Future

Evaluating Predictive
Deep Learning Models

Patrick Ribu Gorton

Backpropagating to the Future

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Abstract

Predicting the future using deep learning is a research field of increasing interest. The majority of contributions concern architectural designs for predictive models, however, there is a lack of established evaluation methods for assessing their predictive abilities. Images and videos are targeted towards human observers, and since humans have individual perceptions of the world, evaluation of videos should take subjectivity into account. With the absence of appropriate evaluation methods, measuring the performance of predictive models and comparing different model architectures is challenging.

In this thesis, I present a protocol for evaluating predictive models using subjective data. The evaluation method is applied in an experiment to measure the realism and accuracy of predictions of a visual traffic environment. These predictions are generated by a proposed model architecture, which produces discrete latent representations of the environment. Application of the evaluation method reveals that the proposed deep learning model proves to be capable of producing accurate predictions ten seconds into the environment's future. The predictive model is also shown to be robust in terms of processing different image types for describing the environment.

The proposed evaluation method is shown to be uncorrelated with the predominant approach for evaluating predictive models, which is a frame-wise comparison between predictions and ground truth. These findings emphasise the importance of using subjective data in the assessment of predictive abilities of models, and open up a new alternative of evaluating predictive deep learning models.

Acknowledgements

This thesis was written for my Master of Science Degree at the Department of Informatics, the University of Oslo.

I want to thank my supervisor Kai Olav Ellefsen for his great support and enthusiasm related to my thesis. I would also like to thank Oslo Metropolitan University for providing me with a student scholarship, and to the ROBIN research group for covering expenses of my experiments. A special thanks to Raquel Dagar Ellingsen for being a great girlfriend and for sharing invaluable methodological insight.

The spring of 2020 became a rather unusual semester, due to a global pandemic that probably will leave its mark for years to come. Luckily, my social interaction and the academic community was maintained thanks to my friend and fellow student Vemund S. Schøyen. I am also very appreciative for my talented mother, Kirsten Ribu, who has shown great excitement for my work.

Lastly, I would like to thank three, small intelligent agents who never bothered to ask about my work, but provided me with comfort in stressful times.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Humans constantly use information from past experience to perform predictive processing, which in turn can improve future behaviour. This bridging over different temporal points with past considerations is suggested to be the core capacity which makes our cognitive brain so versatile and efficient (Bubic, Yves von Cramon, & Schubotz, 2010). The use of predictive machine learning models has been around for some time, for instance for predicting stock market movements (Coupelon, 2007), or the next word in a sentence (Sutskever, Martens, & Hinton, 2011). In recent years, similar methods have also become more common for visual tasks, allowing computers to learn internal models of physical environments, and predict images of how the environments will evolve in the future.

Equipping an intelligent agent with the ability to predict future states and results of potential actions may improve its performance and robustness in environments comprised of complex physical systems (Ha & Schmidhuber, 2018). In addition, research has shown that the type of image used to represent an environment may influence the model's ability to predict future states (Luc, Neverova, Couprie, Verbeek, & Lecun, 2017).

A number of visual predictive models based on deep learning face challenges because they are either highly domain specific and targeted towards

simple environments, or try to model complex environments but tend to lose image detail quickly. Moreover, these models are evaluated by computers quantifying the level of numerical resemblance between predicted and true states, like videos.

But the ultimate receivers of images and videos are human observers, and not all differences between images are equally significant to humans (Moorthy, Wang, & Bovik, 2011). The interpretation of images and video is highly subjective. Why then is model evaluation mainly performed using objective, numerical methods? It is apparent that the research field of video prediction using deep learning is in demand of alternative methods for the evaluation of predictive models.

1.2 Research Questions

In this thesis, I attempt to develop an appropriate method for evaluating predictive models using subjective data. This leads to the following research questions:

1. How can subjective data be used to evaluate predictive deep learning models, and which model properties should be assessed in the evaluation?

2. To what degree does the type of image representing the environment influence a model's ability to predict a meaningful future?

1.3 Scope and Delimitations

Scope

The main focus of this thesis is the design of a new method for evaluating the performance of predictive deep learning models. The method uses subjective data, collected both to reveal aspects of an unfamiliar environment that are considered important to human observers, and to recognise similar aspects in video predictions. The evaluation method is tested using long-term

predictions of different image types, generated by a proposed deep learning model. The proposed deep learning model operates within a traffic environment containing numerous objects moving simultaneously, including the point of view. The evaluation method is compared to the existing evaluation approach within the research field.

Delimitations

The potential workload associated with this thesis is substantial. Therefore, I devote attention to experiments that best answer the research questions and impose the following delimitations.

The human observers in the experiments are obtained by convenience sampling and through a crowdsourcing service. Also, the method evaluates only the performance of the proposed model, not other state-of-the-art models. The traffic environment used to train and test the proposed model architecture is a computer simulation of a finite town and a fixed number of object classes. In the proposed model architecture, I disregard the implementation of an intelligent agent, but focus rather on predicting long-term future video. The distinct image types representing the environment are RGB and semantic segmentation.

1.4 Contributions

In this thesis, I bring two contributions to the field of visual prediction using deep learning. The main contribution is a protocol for evaluating predictive models using subjective data. In addition, I propose a deep learning model architecture that is capable of predicting accurate long-term futures of complex visual environments.

1. A protocol for evaluating video predictions using subjective data. This protocol is found in section 4.5, and is summarised by figure 4.8. Results in section 3.4.4.5 demonstrate that this method leads to very different interpretations of model performance as opposed to standard

evaluation approaches; a finding which reveals significant limitations of standard evaluation approaches.

2. An adaptation of World Models by Ha and Schmidhuber (2018) for predicting accurate long-term video predictions. The proposed model architecture adopts a vector quantised-variational autoencoder (van den Oord, Vinyals, & Kavukcuoglu, 2017) in place of the variational autoencoder (Kingma & Welling, 2014) originally used by the authors of World Models. It produces discrete latent representations in place of continuous latent representations, and is trained with a gradient-based optimisation method. Implementation of the model architecture is found in section 3.3.

1.5 Thesis Structure

- Chapter 2 includes an overview of theory and techniques relevant for this thesis.
- Chapter 3 reviews the research methodology related to design and implementation of the proposed model architecture, and development and testing of the evaluation method.
- Chapter 4 reviews the results of the model evaluation, which reveals the performance of the proposed model architecture and utility of the proposed evaluation method. In addition, it presents a refined protocol for evaluating predictive models with subjective data.
- Chapter 5 presents conclusions of the results and findings in this thesis and ideas for future are presented and presents ideas for future work.

Chapter 2

Background

The following chapter aims to give the reader knowledge about the relevant theories, methods and techniques used in this thesis. The reader will find that *prediction* is a recurring topic throughout the complete text; thus, the chapter begins by presenting this very subject (2.1). The definition of intelligent agents (2.2) is followed by a comprehensive presentation of fundamental theory related to deep learning (2.3, 2.3.5), a topic which gets considerable attention. Highly relevant to the experiments in the thesis are methods within deep learning used to learn alternative representations of data (2.4), as well as sequences of data (2.5). Moreover, since applying deep learning involves working with significant amounts of data, some knowledge of how to preprocess such data (2.6) and also quantify the level of resemblance between data (2.7) is reviewed. Finally, recognising the existing research on visual prediction with deep learning, the last section presents a literature review of the research field of visual prediction with deep learning (2.8).

2.1 Prediction and Predictive Models

A weather forecast might anticipate a sunny afternoon, but your experience suggests that the dark cloud in the sky is a sure indication of rain. A fortune teller might tell you to expect thrilling economic times in the near future, but with a lack of faith in supernatural powers, you rather trust the stock

market for financial gain. These are all examples of predicting future states, though some have more scientific support than others. Bubic et al. (2010) refers to predictive processing as 'any type of processing which incorporates or generates not just information about the past or the present, but also future states of the body or the environment'.

Neuroscientific research suggests that the brain runs an internal model of the world that continually generates predictions about what is expected to be perceived (Leinweber, Ward, Sobczak, Attinger, & Keller, 2017). Central to this idea is *predictive coding*, a theory which postulates that the internal model is created and updated by comparing predicted sensory input to actual sensory input (Friston, 2005).

Mathematical and statistical models have also for long been used to describe past and future behaviour of various processes. Such models are usually characterised as either deterministic or probabilistic. A deterministic model does not include random elements so that each time the model is run with the same initial conditions, it will give the same results. On the other hand, a probabilistic model does include random elements. Even with the same initial conditions, the model is likely to give different results each time it is run.

Recent advances in artificial intelligence have enabled machines to predict future events of environments that resemble the real world. These methods may further be divided into various subcategories. Some methods focus on predicting the direct consequence of a series of states, i.e. for sensorimotor control (Dosovitskiy & Koltun, 2016) or action recognition (H. Wang & Schmid, 2013). Other methods predict a continuation of items or states, like language (Vaswani et al., 2017) or video prediction (Srivastava, Mansimov, & Salakhutdinov, 2015). The focus of this thesis is on the latter, namely predicting future visual states of an environment in the form of videos.

2.2 Intelligent Agents

The ability to plan and execute goal specific actions in varied and unknown environments is a central requirement of intelligent agents (Fragkiadaki,

Agrawal, Levine, & Malik, 2015). An intelligent agent is an autonomous entity which, in a similar manner to humans, perceives sensory input from its environment, makes decisions, and carries out actions that will affect the environment. An agent may be regarded as intelligent if it possesses abilities such as responding to environmental changes in a timely fashion, taking initiative in order to satisfy its objective and socially interacting with other agents or humans (Wooldridge & Jennings, 1995).

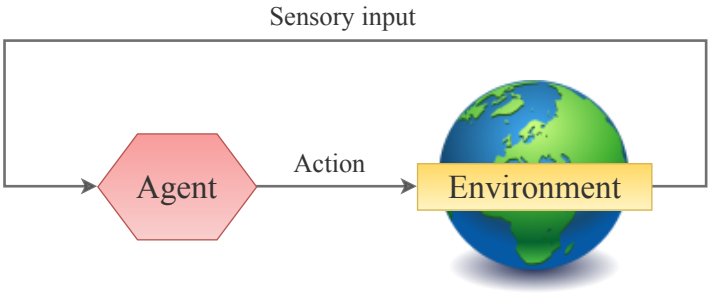


Figure 2.1: The scheme of an intelligent agent.

Interacting with the world requires a common sense understanding of how it operates at a physical level. For example, humans can quickly decide if we can cross an area without falling, or how an object will behave if we push it. Making such judgments does not require us to apply laws of physics, instead we rely on experience and intuition, built up through interaction with the world (Lerer, Gross, & Fergus, 2016). Just like humans benefit from performing predictive processing, so do intelligent agents. A visual predictive model of physics gives the agent the ability to generate potential future states of the world in response to an action without actually performing that action (Fragkiadaki et al., 2015). Recent work has shown that agents equipped with internal predictive models like those studied in this thesis, efficiently learn to interact with environments (Ha & Schmidhuber, 2018; Hafner et al., 2019).

2.3 Deep Learning

Deep learning (DL) is a subcategory of machine learning (ML), which concerns the design of algorithms that make computers able to learn from em-

pirical data, and use this knowledge to make decisions. Machine learning can be divided into three main classes: supervised, unsupervised, and reinforcement learning. Supervised learning is concerned with learning input-output mappings, unsupervised learning aims to find hidden structure in data, and reinforcement learning deals with goal-directed behaviour (Dosovitskiy & Koltun, 2016). Within unsupervised learning there is also what is called self-supervised learning, which is autonomous supervised learning. When using supervised learning, one must prefabricate labels, or rather target variables, for the system to learn a mapping $y = f(x)$. The use of self-supervised learning systems eliminates the need of prefabricating such labels, because the process rather extracts and uses naturally relevant context as supervisory signals (Singh, 2018). Self-supervised learning is applicable for instance for extracting features from images, modeling the order of words in a sentence, or the sequence of images in a video. In the latter cases, the next item in the sequence would be this supervisory signal, or label. Due to the thesis' objective of predicting videos with deep learning, attention is mainly devoted to self-supervised learning.

2.3.1 Artificial Neural Networks

Deep learning models focus heavily on biologically inspired methods, such as neural network models. Early developments of these models were highly influenced by neuroscience and cognitive sciences (Barrett, Morcos, & Macke, 2019). An artificial neural network (ANN) is a composition of artificial neurons, or nodes, which loosely model the neurons in a biological brain. These nodes are connected to one another in layers, where they compute and pass on new combinations of the network's input. Their weighted connections loosely resemble the behaviour of brain synapses between biological neurons. Deep neural networks (DNN), which have multiple so-called hidden layers, can under the right circumstances capture more complex functions than shallower networks (Kriegeskorte & Douglas, 2018).
Figure 2.2 shows a *fully connected network*, sometimes referred to as a dense neural network, which means that all nodes between two consecutive layers

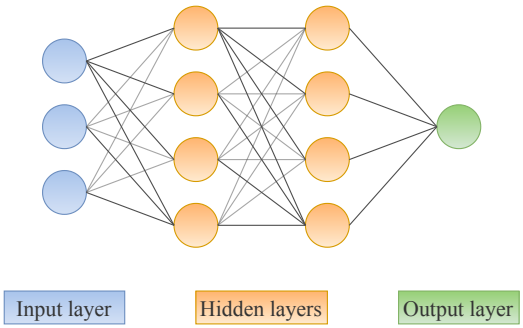


Figure 2.2: A simple artificial neural network illustrating the neurons, layers and weighted connections.

are connected. Throughout this chapter, various network structures will be discussed which may be combined as building blocks to compose more advanced ANNs than what a fully connected network can offer alone. The node activations in a given layer, l, of a fully connected network are calculated using the following expression and parameters.

$$a_k^{[l]} = \sum_{j=1}^{n^{[l-1]}} w_{jk}^{[l]} a_j^{[l-1]} + b_k^{[l]} \tag{2.1}$$

$a_k^{[l]}$ activation of node k in layer l
$w_{jk}^{[l]}$ weight from node j in layer $l-1$ to node k in layer l
$b_k^{[l]}$ bias of node k in layer l

where $n^{[l-1]}$ are the number of nodes in the layer, and w and b are trainable parameters that are adjusted using some optimisation method. Now, what does it mean that w and b are trainable parameters? As mentioned in the introduction to this chapter, ANNs learn to approximate some function $y = f(x)$ by observing data. Approximating this unknown function is the training objective, and is done as follows: The input samples x, or observations, are propagated forward through the network such that the outputs $\hat{y}$ are linear combinations of x. The outputs $\hat{y}$ are then compared to the target variables y by the means of a *loss function*. The loss function gives a response value, and the optimisation method changes the network's parameters w and b such that the loss value decreases, which in turn brings the network's outputs $\hat{y}$

closer to the target variables y. This procedure is called *training* a neural network, and is further described in section 2.3.5.

2.3.2 Convolutional Networks

When using machine learning to process data with a grid-like topology, such as images which are grids of pixels, convolutional neural networks (CNN) are typically used (Goodfellow, Bengio, & Courville, 2016, p. 326). For simplicity, this section considers images as the default input to CNNs. CNNs are a special kind of ANNs which contain one or more convolutional layers. A convolutional layer is a filter kernel which extracts *features* from an input by means of convolution, a process for linear spatial filtering closely related to correlation (Gonzalez & Woods, 2008). The convolution of image I and filter kernel K at image location i, j is defined as

$$(I * K)(i, j) = \sum_m \sum_n I(m, n)K(i - m, j - n) \tag{2.2}$$

where m and n are the dimensions of the kernel. The filter kernel is convolved with the entire image, thus creating a complete feature map of the image. In the above equation, the input I and kernel K have only one channel, however the convolution process may be, and usually is, extended to data with multiple channels. The different layers of a CNN form a hierarchical structure, where each layer learns to search for different features. For example, the first layer may look for horizontal or vertical lines, while the second layer uses this information to detect corners, and the subsequent layers detect more complex patterns such as texture or objects (Zeiler & Fergus, 2014). A convolutional layer shares the same filter coefficients, or parameters, for all positions in the image it processes. This results in a useful property called *translation equivariance*, which means that the CNN will detect the position of an object or structure even if it is not fixed. More precisely, it means that if an image I is shifted, e.g. by one pixel to the right such that $I'(x, y) = I(x + 1, y)$, its representation after the convolution will be shifted correspondingly. Parameter sharing is an important distinction

between fully connected layers and convolutional layers, and the same property is also found in other architectures such as recurrent neural networks (section 2.5).

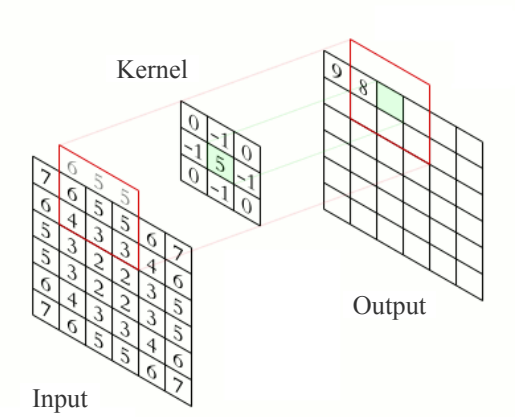


Figure 2.3: Convolving an input with a filter kernel. Each element of the image is added to its local neighbours, weighted by the kernel coefficients. Adapted from Wikimedia Commons, by Michael Plotke, January 28 2013, retrieved from `https://commons.wikimedia.org/wiki/File:2D_Convolution_Animation.gif`.

In traditional image analysis, feature extraction methods are designed manually, while CNN's, being machine learning systems, effectively develop suitable filter parameters to extract features themselves through training (section 2.3.5). These filter parameters are analogous to the parameters described in section 2.3.1. Today's CNNs are based on work introduced in the late 90's, namely 'Object Recognition with Gradient-Based Learning' (Yann LeCun, 1999). In the last decade there has been tremendous developmental progress for CNNs, as well as other network architectures. Many classification benchmarks have been broken, allowing new possibilities for artificial intelligence. This is especially the case regarding image analysis and computer vision (Barrett et al., 2019).

2.3.3 Residual Blocks

While it is shown that a neural networks' depth is of great significance, especially with regard to performance on visual recognition tasks (Long, Shel-

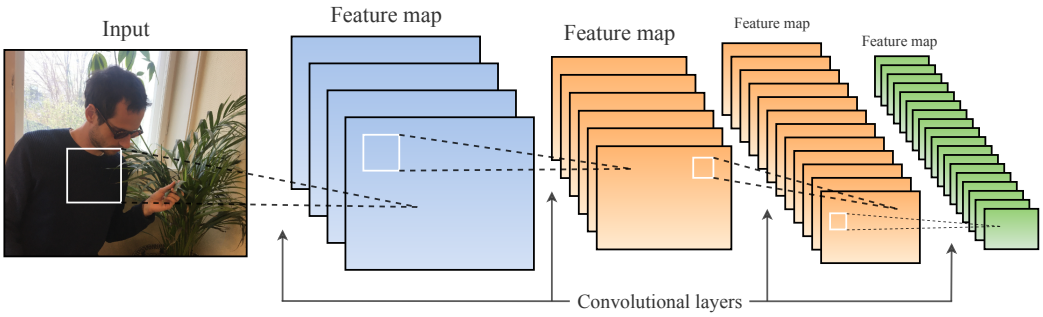


Figure 2.4: An example of the hierarchical structure of a typical convolutional neural network. The number of channels may vary among inputs and feature maps.

hamer, & Darrell, 2015; Simonyan & Zisserman, 2015), very deep networks come at the cost of being difficult to optimise. He et al. (2016) demonstrated within an image classification task that as network depth increases, the classification accuracy of images saturates and then degrades quickly. The authors investigated this topic, and documented the power of *residual connections* in deep learning networks that have great numbers of layers. Their proposed model architecture, called 'ResNet', was successfully optimised with various significant depths ($\leq$ 1000 layers), while comparable architectures without residual blocks could not be optimised. A residual block is a set of layers that includes a residual connection, allowing the flow of unaltered information from the first to the last layer in the block, i.e. retaining the input. This property is useful because, depending on the task, some parts of a deep neural network may be unnecessary or even impairing. Consider a conventional block of neural network layers that attempts to fit the mapping $y = \mathcal{F}(x)$. If the block is not able to fit this mapping, it would be more useful to retain the input x and let a subsequent layer process it. On the other hand, the layers $\mathcal{F}(x)$ in a residual block learn the deviation from the input x, in other words the residual $\mathcal{R}(x) = \mathcal{F}(x) - x$. The residual block as a whole performs the mapping $y = \mathcal{F}(x) + x$.

The reason deep networks with residual blocks are easier to optimise is related the loss function, which depends on the model's architecture. The loss function defines a *loss landscape* in which the optimisation method searches

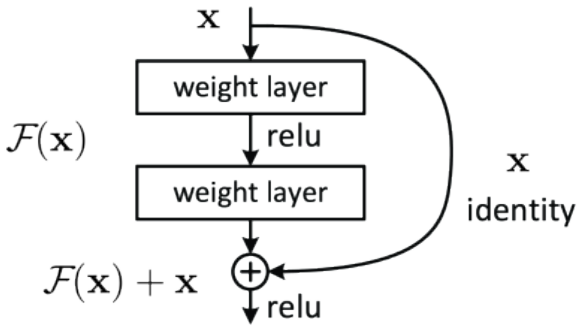


Figure 2.5: Residual learning: a building block. A residual block retains the input to the first layer and adds it to the output of the last layer. Figure by He et al. (2016).

for the global optima, i.e. finding model parameters that yield the lowest possible loss. However, loss landscapes are often rugged which might result in getting stuck in one of many local optima. Li et al. (2017) demonstrated that the loss landscape of a deep model with residual connections is much smoother than that of a similar network without the residual connections.

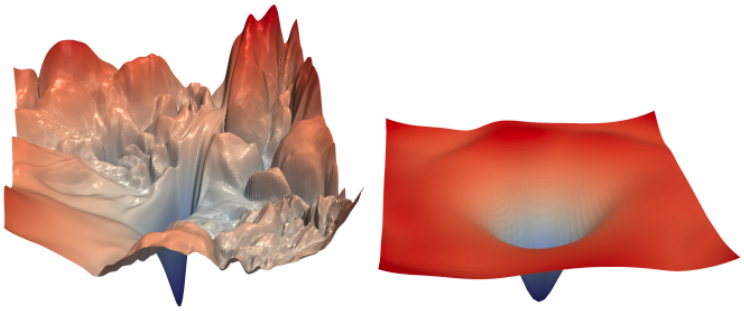


Figure 2.6: The loss landscapes of a 56-layered ResNet model without (left) and with (right) residual connections. Adapted from Li et al. (2017)

A smooth landscape aids the optimisation method in converging towards a low loss, possibly at reduced time compared to its rugged counterpart. Keep in mind that the loss landscapes in figure 2.6 are not actually three-dimensional. Li et al. (2017) use specialised visualisation techniques to transform the actual high-dimensional loss landscapes into visually interpretable landscapes.

2.3.4 Nonlinear Activation Functions

In the previous sections, various layers, or building blocks used in deep learning models were discussed. A layer performs a linear operation $f(x)$ on an input, meaning that combining more layers will only result in a deeper linear model. Usually, there is a desire for deep learning models to be universal function approximators and not only linear functions. To make linear models represent nonlinear functions of x, some nonlinear transformation $\phi(x)$ is applied to x before passing it to a consecutive layer (Goodfellow et al., 2016, p. 165). This type of transformation may be referred to as a nonlinear activation function, or simply a nonlinearity. Recall that the activation of a node in layer l is computed as

$$a_k^{[l]} = \sum_{j=1}^{n^{[l-1]}} w_{jk}^{[l]} a_j^{[l-1]} + b_k^{[l]}$$

Nonlinearity is introduced to this particular network layer by applying a nonlinear activation function to all its activations a_k

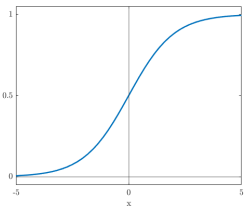
$$\phi(a_k^{[l]}) = \phi(\sum_{j=1}^{n^{[l-1]}} w_{jk}^{[l]} a_j^{[l-1]} + b_k^{[l]}) \tag{2.3}$$

where $\phi(a)$ is some chosen nonlinear activation function applied to activations a. The following are some examples of nonlinear activation functions that are used in deep learning, and the models implemented in this thesis.

Sigmoid

The sigmoid function, denoted $\sigma(x)$, stems from logistic regression and squashes an input x to a value between values $[0, 1]$. This makes it applicable to classification problems where the target variables are either 0 or 1.

$$\text{Sigmoid}(x) = \sigma(x) = \frac{1}{1 + \exp(-x)} \qquad (2.4)$$

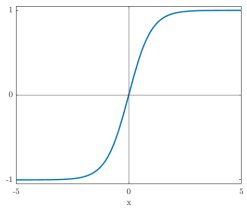


The sigmoid activation function may suffer from drawbacks which include unwanted effects during a gradient-based training process (section 2.3.5), such as vanishing gradients and slow loss convergence (Nwankpa, Ijomah, Gachagan, & Marshall, 2018).

Hyperbolic tangent

The hyperbolic tangent function is in some circumstances preferred to the sigmoid function because it yields better training performance for multi-layer neural networks (Olgac & Karlik, 2011). Nonetheless, the function does not solve the vanishing gradient problem described above. It does, however, produce a zero centred output which aids the gradient-based training process (Nwankpa et al., 2018), equivalent to a scaled and shifted sigmoid function.

$$\begin{aligned} \tanh(x) &= \frac{e^x - e^{-x}}{e^x + e^{-x}} \\ &= 2\sigma(2x) - 1 \end{aligned} \qquad (2.5)$$

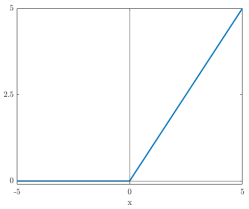


Rectified linear unit

Ever since the rectified linear unit (ReLU) activation function was proposed by G. Hinton (2010), it has been widely used for many deep learning applications (Nwankpa et al., 2018). The ReLU activation function remains very close to being linear, thus preserving properties associated with the

ease of training linear models, and is the default activation function recommended for use with most feedforward neural networks (Goodfellow et al., 2016, p. 170).

$$\text{ReLU}(x) = \max(0, x) \tag{2.6}$$



2.3.5 Training Deep Neural Networks

Deep neural networks learn some goal or property through many iterations of observing data and adjusting parameters. This learning process is usually referred to as *training* a model, and is done by using a chosen optimisation method. When training a model, one must first establish some way of evaluating its performance concerning the training objective. Such a measure is most often referred to as a *loss function* (see section 2.7). A loss function is necessary because it lets the optimisation method know how it may change a model's parameters to improve its performance. There exist various optimisation methods used to train deep learning models. These methods may be divided into gradient-based methods such as Adam (Kingma & Ba, 2015), and non-gradient-based methods such as evolutionary algorithms (Such et al., 2018). Most of the gradient-based methods are variants of *gradient descent* (Cauchy, 1847), which takes a model's parameters in the gradient direction that minimises the loss. Computing all gradients of the loss function with respect to the model's parameters is effectively done by the use of the *back-propagation* algorithm (Rumelhart, Hinton, & Williams, 1986), which applies the chain rule of derivatives iterating backwards from the last network layer to the input samples. The *stochastic gradient decent* (SGD) algorithm is perhaps the most fundamental and popularised gradient-based optimisation method used in practice. SGD is a stochastic approximation of gradient des-

cent, which approximates the gradients of the whole dataset but based on mini-batches of input data. The gradient descent parameter updates may be expressed as follows

$$\theta' = \theta - \eta \nabla_\theta \mathcal{L} \tag{2.7}$$

where θ are the model's parameters, θ' the updated parameters, η is the learning rate scaling the gradient step, and $\nabla_\theta \mathcal{L}$ are the derivatives of the loss function with respect to the parameters.

2.3.5.1 Generalisation

While the goal of learning is to approximate some function based on the data which is put into the model and their corresponding target variables, it is essential that the model also learns to generalise to unseen data. Checking how well a model is generalising may be done by presenting the model with two sets of data during training; a *training set* and a *validation set*. The training set is used by the optimisation method to update parameters and improve the training loss. The validation set is at regular intervals used to measure the model's ability to generalise beyond what it learns from the training set. If a model performs well on the training set, but poorly on the validation set, it is likely *overfitting*, meaning it has learned the data's variance, such as noise, too well. On the other hand, *underfitting* happens when the model is incapable of capturing the complexity and underlying pattern of the data. This may be due to the number of training steps or training data being too small or the complexity of the model being insufficient. The validation set may also be used by the developer to tune a model's hyperparameters, such as the learning rate. In any case, both the training and validation set should represent the same data distribution.

2.3.5.2 Regularisation Techniques

It is quite common to experience overfitting when working with deep learning models. Various regularisation techniques may be imposed on the model when training in order to reduce these symptoms, thus helping the model

to generalise. Acquiring more data for the model to train on is perhaps the best alternative, though this is often unfeasible, as gathering data may be costly. There exist well-documented techniques that may be applied to a model during training, with a few of them listed below. Which regularisation techniques to apply to a model depends on the problem, model architecture and dataset of choice.

Early stopping

Early stopping involves stopping a training procedure when the validation loss starts increasing rather than decreasing (Yao, Rosasco, & Caponnetto, 2007). Though this is a very effective method for preventing a model from overfitting, it may restrict the model's expressiveness and desired performance.

Dropout

Dropout (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014) is a technique that randomly deactivates different neurons in a network layer at each training step, the rate of which is determined by probability hyperparameter. The idea behind dropout is to motivate the model to not rely on any single feature. Effectively, dropout trains the ensemble of all subnetworks that are formed by removing individual units (Goodfellow et al., 2016, p. 255).

Batch normalisation

Batch normalisation makes normalisation a part of the model architecture and performs normalisation for each batch of data (Ioffe & Szegedy, 2015). As a result, the model becomes less sensitive to learning rates and parameter initialisation. Batch normalisation is somewhat similar to dropout in the way that it induces random components or noise to a model's features (Goodfellow et al., 2016, p. 314). Though its primary objective is to improve upon model optimisation, the noise resulting from normalisation can have a regularising effect, and sometimes makes dropout unnecessary.

Data augmentation

Data augmentation involves modifying a dataset to increase its size. Ways of modification may be adding noise to the data, or if the data are images apply transformations such as crop, flip and rotation. Data augmentation is especially valuable when data is scarce and collecting more data is challenging.

2.4 Representation Learning

The performance of a machine learning model is highly dependent on the representation it attempts to learn, but also the representation of the provided input. This input may be raw data from a dataset or the output of another model. So what makes one representation superior to another? According to Goodfellow et al. (2016, p. 525), 'a good representation is one that makes a subsequent learning task easier'. Examples of such subsequent learning tasks could be classification tasks or prediction tasks. Simply choosing a reasonable representation is no uncomplicated procedure (Bengio, Courville, & Vincent, 2013); however, unsupervised and self-supervised learning allows deep learning models to discover useful data representations by observing large amounts of this data. Among deep learning methods for representation learning, the focus of this thesis is on *autoencoders*.

2.4.1 Autoencoders

An autoencoder (AE) is an encoder-decoder network that attempts to decompose and reconstruct its input through one or more intermediate stages. The network is trained using self-supervised learning, meaning that the data labels are contained within the data itself. AEs were originally designed for unsupervised *feature extraction* and *dimensionality reduction* (Bourlard & Kamp, 1988; Kramer, 1991). As the name implies, an AE consists of two parts; an encoder $h = f(x)$ and a decoder $\hat{x} = g(h)$, where h is a latent representation holding features of x, and $\hat{x}$ is the reconstruction of x.

Learning useful representations, or features, with AEs is usually done by constraining the latent representation h to have a smaller dimension than the input x, forcing the autoencoder to capture only the most evident properties of the training data, rather than focusing on fine details (Goodfellow et al., 2016, p. 500). This means that AEs do not learn to copy the data perfectly, but instead produce reconstructions that closely resemble the training data. The learning objective is to minimise the reconstruction error

$$\mathcal{L}_{recon} = \mathcal{L}(x, g(f(x))) \tag{2.8}$$

where $\mathcal{L}$ is some loss function measuring dissimilarity between x and $g(f(x)) = \hat{x}$. Bottlenecked models with $dim_h < dim_x$ are called *undercomplete autoencoders*. In addition to undercomplete autoencoders, there exist other categories such as *sparse autoencoders* and *denoising autoencoders*. Sparse AEs motivate the presence of sparse latent representations, which can be useful for classification problems. Denoising AEs receive a corrupted sample as input and aims to predict the original, i.e. the uncorrupted sample. In this thesis, we focus on undercomplete autoencoders, and subcategories of these.

2.4.2 Variational Autoencoders

Generative models are models that represent probability distributions over multiple variables in some way (Goodfellow et al., 2016, p. 651). Simply put, a generative model aims to learn an approximation of a dataset's true distribution, and use this distribution to generate new data. According to G. E. Hinton et al. (1995), the goal of generative models is 'to learn representations that are economical to describe but allow the input to be reconstructed accurately'.

A variational autoencoder (VAE) (Kingma & Welling, 2014) is a type of generative model, which tries to learn the distribution of a given dataset using self-supervised learning. Learning the true distribution of a dataset may be impractical or even impossible if the dimensionality of the data is large, which is why a VAE instead attempts to learn an approximation of

the true distribution.

While traditional autoencoders are powerful compression systems, their latent variables are typically sparsely populated, making it unlikely to sample random latent variables that can be decoded into valid outputs (Spinner, Körner, Görtler, & Deussen, 2018). Similarly to undercomplete autoencoders, VAEs can reconstruct input samples, but with the possibility of adding some variation, and also generate completely new samples not seen during training. VAEs do this by learning latent representations that contain useful properties of the data, but simultaneously make sure that all latent representations are clustered together. This clustering is achieved by forcing the latent variables z to fit a prior probability distribution $p(z)$, which is usually a standard normal distribution $\mathcal{N}(0, 1)$. The encoder takes as input some data point x and outputs two parameters μ and σ for a posterior distribution $q(z|x) = \mathcal{N}(\mu, \sigma^2)$. Latent variables z are then sampled from $q(z|x)$ and used by the decoder to reconstruct $\hat{x} \approx x$, maximising the likelihood that the model will generate the observed data. This process of estimating parameters for a probability distribution is called *maximum likelihood estimation* (MLE), and is widely used in statistics.

By forcing all latent variables to stay close together we obtain the ability to interpolate between samples. In the case where the data consists of images, this means that a value change in the latent domain should yield a meaningful change in the image domain (Ha & Schmidhuber, 2018). For example, Hou et al. (2017) showed the possibility of interpolating between faces, such as the transition from a non-smiling woman to a smiling woman, or the transition from a man without eyeglasses to a man with eyeglasses.
To make sure that the parameters μ and σ do not become sparsely populated, the prior $p(z)$ is used as the target distribution during model optimisation. $q(z|x) = \mathcal{N}(\mu, \sigma^2)$ is then forced to become as similar as possible to $p(z) = \mathcal{N}(0, 1)$, thus restricting the shape of the latent space and enabling sampling of $p(z)$ to generate new data at inference. The two distributions are brought closer together by minimising the Kullback-Leibler divergence

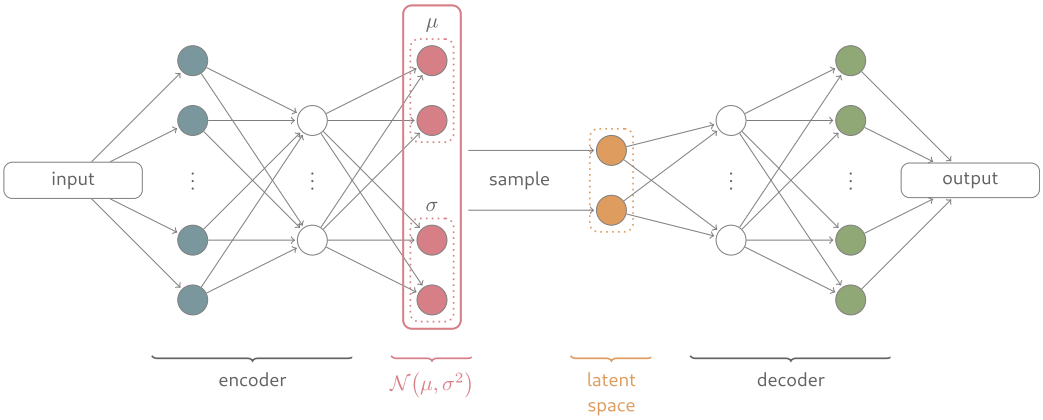


Figure 2.7: We see a VAE encoding an input and outputting parameters μ and σ for a normal distribution $\mathcal{N}(\mu, \sigma^2)$. Latent variables $\mathbf{z}$ are then sampled from $\mathcal{N}(\mu, \sigma^2)$ and used by the decoder to reconstruct the input. Figure by Spinner et al. (2018).

$$D_{KL}(q(z|x)||p(z)) = \sum_{i=1}^{N} q(z_i|x_i) \cdot \log \frac{q(z_i|x_i)}{p(z_i)} \tag{2.9}$$

A VAE can be trained by minimising the sum of the data reconstruction loss $\mathcal{L}_{recon}$ and the KL divergence $\mathcal{L}_{KL}$

$$\mathcal{L}_{recon} = -\mathbb{E}_{z\sim q(z|x)}[\log(p(x|z))] \tag{2.10}$$

$$\mathcal{L}_{KL} = D_{KL}(q(z|x)||p(z)) \tag{2.11}$$

$$\mathcal{L}_{VAE} = \mathcal{L}_{recon} + \mathcal{L}_{KL} \tag{2.12}$$

where $-\mathbb{E}_{z\sim q(z|x)}[\log(p(x|z))]$ is the negative expected log-likelihood of the observations x.

Training VAEs

To train a VAE using a gradient based optimisation method, the gradients with respect to the loss terms in equation 2.12 must be computed. While $\mathcal{L}_{KL}$ is a differentiable expression, this is not the case for $\mathcal{L}_{recon}$ in its current form. This is due to sampling of latent variables not being a differential operation. However, Kingma and Welling (2014) solved this issue by introducing a re-

parametrisation trick, which allows rewriting the expectation with respect to $q(z|x)$ such that the Monte Carlo estimate of the expectation is differentiable with respect to μ and σ. Using the fact that any normal distribution may be expressed in terms of the standard normal distribution as follows

$$\mathcal{N}(\mu, \sigma^2) \sim \mu + \sigma^2 \cdot \mathcal{N}(0, 1) \tag{2.13}$$

This property is then used to sample ϵ from the standard normal distribution and create a latent sample

$$z = \mu + \sigma^2 \cdot \epsilon, \qquad \epsilon \sim \mathcal{N}(0, 1) \tag{2.14}$$

The random component ϵ may be treated as a constant for each sample, and thus the gradients over μ and σ may now be derived.

2.4.3 Vector-Quantised Variational Autoencoders

VAEs applied to complex datasets of natural images have a tendency to produce blurry and somewhat unrealistic images due to uninformative latent features (Zhao, Song, & Ermon, 2017). In recent years, there have been a number of research contributions related to VAEs that attempt to overcome these issues. The vector quantised-variational autoencoder, abbreviated VQ-VAE, is one such contribution.

The VQ-VAE combines the VAE framework with *vector quantisation* to obtain discrete latent representations (van den Oord et al., 2017). While learning representations with continuous features has been the norm for much previous work, the authors argue that discrete representations are better suited for learning to model language, speech, images and video, due to the fact that their properties are indeed discrete. For example, Ferrone and Zanzotto (2020) define natural language as 'inherently a discrete symbolic representation of human knowledge', and image content may be described by language, such as with image captioning systems (Vinyals, Toshev, Bengio, & Erhan, 2015). van den Oord et al. (2017) also argue that discrete representations are a natural fit for complex reasoning, planning and pre-

dictive learning. By itself, the VQ-VAE is deterministic, meaning that an input x will always result in the same reconstruction $\hat{x}$, though in the original paper it is paired with an autoregressive model that learns a prior distribution over the discrete representations. The learned prior distribution allows the VQ-VAE to add variations to existing samples, or generate new samples, making the combination a generative model. Even so, the use of such an autoregressive model is not a requisite, and neither part of this thesis.

The core component of the VQ-VAE is the *embedding space* defined as $e \in \mathbb{R}^{K \times D}$, where K is the size of the embedding space, and D is the dimension of each embedding vector e_i. This embedding space is comparable to a latent space in a VAE. However, rather than manually deciding a distribution shape for the embedding space as one would do in a regular VAE, the embeddings are randomly initialised and learn to hold suitable features for producing valid reconstructions of all samples in the dataset.

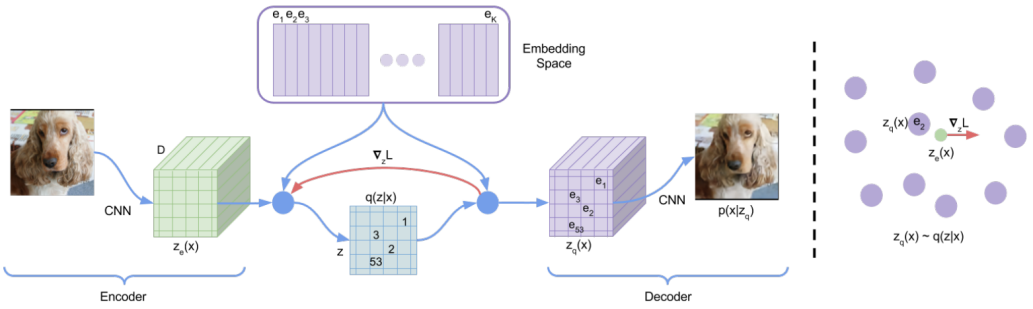


Figure 2.8: Overview of the VQ-VAE process. The left part of the figure shows an image being encoded an mapped to discrete embeddings, before it is decoded into a reconstruction of the image. The right part of the figure is a visual interpretation of the embedding space, which shows how the encoder output is brought closer to the embeddings. Figure by van den Oord et al. (2017).

As can be seen in figure 2.8, the encoder receives an image x as input and outputs an encoded image $z_e(x)$. The vector quantisation layer searches the embedding space e for the closest embedding vectors $z_q(x)$ through nearest neighbour lookup, and returns indices z to these vectors. The decoder finally maps $z_q(x)$ to $\hat{x}$, reconstructing the original image.

Training VQ-VAEs

As mentioned above, the vector quantisation layer performs a nearest neighbour lookup to find the embedding vectors $z_q(x)$ that are closest to the encoder output $z_e(x)$. This operation is not differentiable, and will stop the flow of gradients during backpropagation. The solution is to copy the gradients directly from the decoder input $z_q(x)$ to the encoder output $z_e(x)$, as shown in the left part of figure 2.8. However, this means that the embeddings e_i do not receive any gradients from the reconstruction loss $\mathcal{L}_{recon} = \log p(x|z_q(x))$, but are instead learned by using vector quantisation. This involves iteratively moving the embeddings e_i closer to the encoder outputs $z_e(x)$ with a predefined step size, minimising the euclidian distance between $z_e(x)$ and e_i. Alternatively, exponential moving averages can be to update the embedding vectors. van den Oord et al. (2017) define the complete learning objective as follows

$$\begin{aligned} \mathcal{L}_{\text{VQ-VAE}} &= \mathcal{L}_{recon} + \mathcal{L}_{commit} + \mathcal{L}^*_{embed} \\ &= \log p(x|z_q(x)) + \beta||z_e(x) - sg[e]||_2 + ||sg[z_e(x)] - e||^*_2 \end{aligned} \tag{2.15}$$

where sg is the stop-gradient operator defined as identity during forward pass and has zero partial derivatives. $\mathcal{L}^*_{embed}$ is marked because the embeddings are trained using vector quantisation, i.e. not the same optimisation scheme used to train the encoder and decoder. It is possible that the embeddings e_i do not train as fast as the encoder, so the term $\mathcal{L}_{commit}$ is added to help the encoder commiting to an embedding, controlled by a hyperparameter β. The decoder optimises only $\mathcal{L}_{recon}$, while the encoder optimises both $\mathcal{L}_{recon}$ and $\mathcal{L}_{commit}$.

2.5 Sequence Learning

Modelling data sequences is a classical problem in stastics and machine learning. There exist various methods within machine learning that model sequences of data. Recurrent neural networks (RNN) are an example of such

methods for time series regression and classification.

2.5.1 Recurrent Neural Networks

While deep feed-forward networks may be considered universal function approximators, recurrent neural networks are universal approximators of dynamical systems (Schäfer & Zimmermann, 2006). RNNs differ from fully connected networks in the way that they share parameters across different parts of the model, a property similar to the one found in CNNs (section 2.3.2). This sharing of parameters allows an RNN to learn from and generalise across sequences of arbitrary lengths (Goodfellow et al., 2016, p. 363). Consider, for example, a traffic environment, in which the signs, regulations, and structuring of lanes are approximately the same across various parts of a sequence. A model trying to learn such an environment may benefit from owning a limited set of parameters that contain information about these general rules, applicable to all steps in the sequence. Equation 2.16 represents a simple recurrent function

$$x_{t+t} = f_\theta(x_t) \tag{2.16}$$

where $f_\theta(x_t)$ is a function of an input x at step t, with a set of parameters θ shared over every time step. A recurrent neural network extends the above function and works by receiving as input not only the current input example but also the internal states from when it processed previous examples. These internal states, normally called *hidden states*, represent information from all previous steps, making RNNs good at modelling rather long sequences. The general form of an RNN may be expressed as follows

$$h_t = f_\theta(h_{t-1}, x_t) \tag{2.17}$$

where h_t is the hidden state at the current step, h_{t-1} is the previous hidden state, and x_t is the current input. As a result, recurrent networks can recognise, predict, and generate dynamical patterns, and are commonly used in tasks where data occurs as time-series events, such as in natural language

processing (Vinyals et al., 2015) or videos (Srivastava et al., 2015).

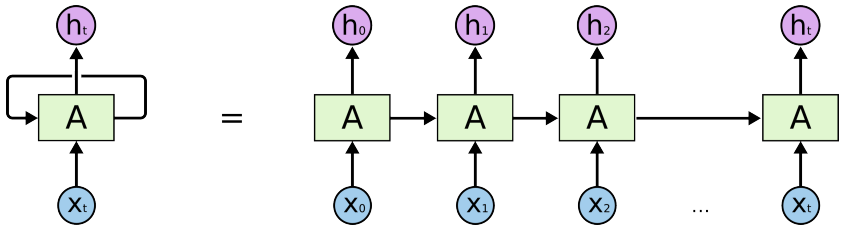


Figure 2.9: The structure of a recurrent neural network whips maps an input x_t to a hidden state h_t. All units share the same set of parameters. Figure from Colah's Blog, by Christopher Olah, August 27 2015, retrieved from `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

The hidden states h may be used for subsequent tasks directly, or transformed to outputs y by an output layer, or weight matrix W_{hy}. A traditional recurrent neural network multi-dimensional data may be more precisely expressed as follows

$$\begin{aligned} h_t &= \tanh(W_{hh}h_{t-1} + W_{hx}x_t + b) \\ y_t &= W_{hy}h_t \end{aligned} \tag{2.18}$$

x_t : Input vector
h_t : Hidden state vector
y_t : Output vector
W : Weight matrices
b : bias

where W_{hh}, W_{hx} and W_{hy} are the weight matrices used to transform the previous hidden state h_{t-1}, the input x_t and obtain the output y_t, respectively. The hyperbolic tangent function applies a nonlinear transformation to the RNN and scales the hidden states within the value range [-1,1].

The main challenge with regular RNNs is that they struggle to preserve long-range dependencies. The cause of this problem is related to what is called *exploding gradients* and *vanishing gradients*. These effects appear when the number of steps in a sequence increases, and as a result the RNN's gradient values progressively amplify or decrease when backpropagating through time. The consequence could be that early time steps yield gradients that

either ruin or do not contribute to learning. It is therefore said that RNNs suffer from short term memory and can only learn sequences of limited length, which is why there exist various sub-classes of RNNs, designed specifically to deal with these issues. The most common sub-classes are the long short-term memory network (LSTM) (Hochreiter & Schmidhuber, 1997) and gated recurrent units (GRU) (Cho et al., 2014).

2.5.2 Long Short-Term Memory Networks

One of the most effective types of sequence models used in practical applications are called *gated RNNs* (Goodfellow et al., 2016, p. 397). Long short-term memory networks are such a type of RNNs, explicitly designed to deal with the challenges related to learning long-term dependencies. Hochreiter and Schmidhuber introduced the LSTM in 1997, which has since then been improved and popularised in subsequent work. LSTM networks are shown to work well on a large variety of problems, such as handwriting recognition (Graves et al., 2009), machine translation (Sutskever, Vinyals, & Le, 2014) and image captioning (Vinyals et al., 2015). LSTM units differ from traditional RNNs in the way that they contain *cells* that control the flow of gradients, which leads to faster learning and more successful runs (Hochreiter & Schmidhuber, 1997). Each cell has an internal recurrence in addition to the outer recurrence of the RNN (Goodfellow et al., 2016, p. 399).

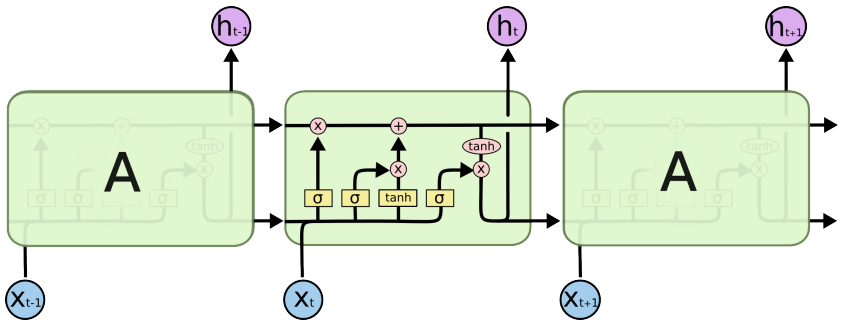
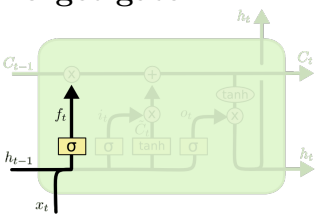


Figure 2.10: The structure of a LSTM network. Figure from Colah's Blog, by Christopher Olah, August 27 2015, retrieved from `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

The core component of the cell is the *cell state*, which is controlled by

three *gating units*. By adding and removing information to this cell state, the LSTM network may learn what aspects of the data that are essential to remember in order to preserve long-range dependencies. First, it decides what information to discard from the cell state, using a *forget gate*. Following this, an *input gate* creates candidate values for an updated cell state. Finally, an *output gate* uses the cell state to output a new hidden state. Through this process of four steps, the LSTM network determines what parts of past events that are useful to remember.

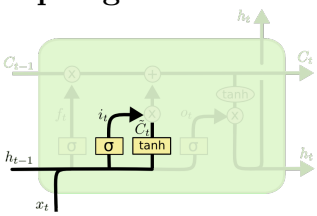
Forget gate



Decides which parts and how much of the cell state to forget, f_t

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.19}$$

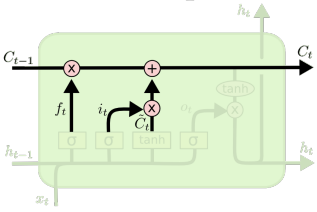
Input gate



Decides which parts and how much of the cell state to update, i_t, and creates candidate values for the new cell state, $\hat{C}_t$

$$\begin{aligned} i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \hat{C}_t &= \tanh(W_c \cdot [h_{t-1,x_t}] + b_c] \end{aligned} \tag{2.20}$$

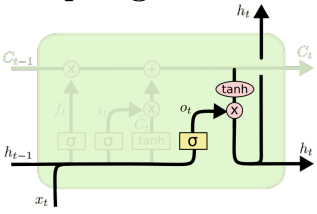
Cell state update



Applies f_t to the cell state, making it forget certain information, and updates the cell state with i_t and the candidate values $\hat{C}_t$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \hat{C}_t \tag{2.21}$$

Output gate



Decides what information the new hidden state h_t will contain

$$\begin{aligned} o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t &= o_t \cdot \tanh(C_t) \end{aligned} \tag{2.22}$$

Figure 2.11: The different gating functions in an LSTM unit. Adapted from Colah's Blog, by Christopher Olah, August 27 2015, retrieved from `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

where the W's denote the weight matrices for the forget gate, input gate, candidate cell values and output gate, respectively. The b's are the gating functions' biases, and h_t and x_t is the hidden state and the input at time step t. The functions σ and tanh are the sigmoid and hyperbolic tangent activation functions (section 2.3.4). Due to the LSTM's increased complexity compared to regular RNNs, they possess a greater number of learnable parameters, meaning they are somewhat more computationally expensive.

2.6 Preprocessing Data

Before applying a machine learning model to a chosen set of data, it may be beneficial to first transform the data in ways that improve its compatibility with the model. Such transformations may involve normalising or scaling data for more efficient learning or altering its representation to better suit a specific task.

2.6.1 Data Normalisation

When working with large amounts of data, which is typical in the field of deep learning, it is common practice to adjust the data samples such that their values all lie within a similar range. This type of adjustment, called data normalisation, ensures no samples deviate considerably from the rest and may help neural networks train faster (Sola & Sevilla, 1997). Two common methods for data normalisation are the *standard score* and *min-max feature scaling*.

The standard score compares an observation to a theoretical deviate, such as a standard normal deviate. Here, the population mean μ and standard

deviation σ are known or estimated parameters of the chosen dataset

$$x' = \frac{x - \mu}{\sigma} \tag{2.23}$$

Feature scaling brings all values into a given numerical range, typically $[0, 1]$.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{2.24}$$

2.6.2 One-Hot Encoding

One-hot encoding involves converting a set of categorical data to binary representations. Not all machine learning models can operate on such categorical data directly, so depending on the task at hand, giving the data more appropriate representations may be necessary. A one-hot representation of data is a group of bits where maximum one value is high, and the rest are low. One-hot encoding a categorical data sample may be seen as a two-step process:

1. Integer encoding: From category to integer
2. One-hot encoding: From integer to one-hot representation

First of all, the number of possible categorical values or classes should be limited to a fixed set, n. Integer encoding assigns integer values to the data categories, for example by assigning categories 'pedestrian' and 'vehicle' with integer values 4 and 10. This intermediate representation could be applied to the machine learning model already but likely results in poor performance and predictions that lay halfway between categories. Next, the integer values are one-hot encoded to the group of bits, which are $1 \times n$ vectors holding only 0's except in the value position of class $c \in [0, n-1]$, where it is 1. Say for example that the maximum number of classes n is 10, then the one-hot encodings of classes 3 and 7 become

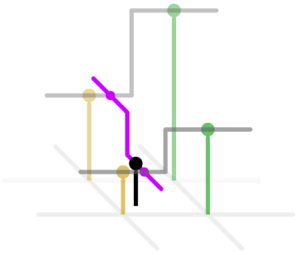
$$c = 3 \xrightarrow{one-hot} [0, 0, 0, 1, 0, 0, 0, 0, 0, 0]$$

$$c = 7 \xrightarrow{one-hot} [0, 0, 0, 0, 0, 0, 0, 1, 0, 0]$$

2.6.3 Image Scaling

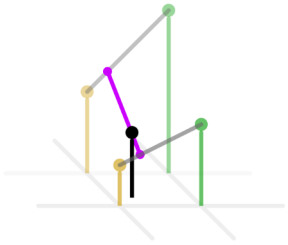
Typically, the capacity of deep learning models for image processing is tailor-made to defined image sizes. Rather than increasing the capacity of a model to fit large images, images may be downscaled and applied to a smaller model. A smaller model would pay less attention to image detail, but contains fewer parameters than a larger model, meaning it occupies less computational memory. The scaling factor should be chosen such that the scaling process retains the image's most essential content and structures. Resizing images results in new images that are either larger, meaning new pixel values must be assigned or smaller in which case some pixels must be removed or altered. This altering of pixel values is done by means of an interpolation method. Some conventional interpolation methods when resizing images are the nearest neighbour, bilinear and bicubic interpolation.

Nearest neighbour



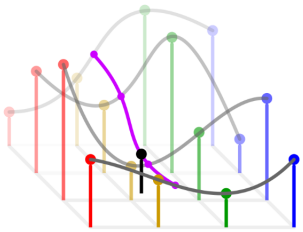
The nearest neighbour algorithm selects the value of the pixels' closest neighbour. This interpolation technique is the quickest among the three, and is typically used when computational speed is a criterion.

Bilinear



Bilinear interpolation is performed using linear interpolation between points in two directions. It involves more computational steps than the nearest neighbor algorithm, but produces a smoother result.

Bicubic



Bicubic interpolation is normally accomplished using either Lagrange polynomials or cubic splines. The interpolated surface is smoother than that of both bilinear interpolation and nearest neighbor interpolation.

Figure 2.12: Black coloured points correspond to the interpolated point and neighbouring samples, respectively. Their heights correspond to their values. Adapted from Wikipedia, retrieved from `https://en.wikipedia.org/wiki/Bicubic_interpolation`.

2.6.4 Semantic Segmentation

Semantic segmentation is not exactly a way of preprocessing data, but rather a process consisting of compound methods for dividing regions of an image by category. Thoma (2016) defines semantic segmentation as 'the task of clustering parts of images together which belong to the same object class'. The process is analogous to an image classification task at the pixel level, thus instead of assigning the overall image with a class label, one or more classes are assigned to specific regions in the image. Semantic segmentation has several use-cases, such as detecting objects in traffic scenes or tumours in patients. Semantically segmented images are usually obtained through a semantic segmentation algorithm, often in the form of a deep learning model. However, such images may be synthetically generated, for example, using a computer simulator (Dosovitskiy, Ros, Codevilla, López, & Koltun, 2017). Figure 2.13 illustrates an RGB image turned into a semantic segmentation map.

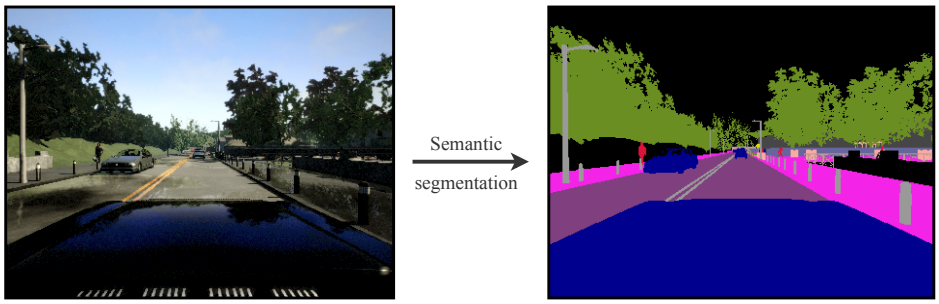


Figure 2.13: Classifying objects in an image with semantic segmentation. Adapted from the CARLA Documentation, retrieved from `https://carla.readthedocs.io/en/stable/cameras_and_sensors/`.

2.7 Similarity Measures

Measuring similarity between images or other types of data is a process regularly used in conjunction with machine learning. For example, during model optimisation, loss functions (2.3.5) are similarity metrics that aid the optimisation method to adjust the model's parameters. Or, such similarity metrics can be used to evaluate model performance at inference. If the similarity metric is to be used as a loss function during gradient-based optimisation, it must be differentiable. Though, if it is simply to be used as an evaluation metric at inference, it is not required. This thesis largely deals with the processing of images, and therefore, the following similarity metrics are mainly targeted toward usage on images.

2.7.1 Mean Squared Error

Mean squared error (MSE) is a simple metric for investigating the squared distance between points in a defined space, and may be used to indicate the pixel-level similarity between images. Given two images Y and $\hat{Y}$, the MSE quantifies the squared difference between all their pixel values. For two images containing a single channel the expression can be written as follows

$$\text{MSE}(Y, \hat{Y}) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2.25}$$

where n is the number of elements in either of the images. The lower the MSE value, the greater is the resemblance between Y and $\hat{Y}$.

2.7.2 Cross-Entropy

Entropy is a measure of the uncertainty associated with a given distribution, or a set of data. The cross entropy of a set Y relative to a set $\hat{Y}$ is given as

$$H(Y, \hat{Y}) = \sum_{i=1}^{n} Y_i \log \frac{1}{\hat{Y}_i} = -\sum_{i=1}^{n} Y_i \log \hat{Y}_i \tag{2.26}$$

Cross entropy is useful as a loss function in for example multi-class classification problems, or regression over discrete variables (van den Oord et al., 2016).

2.7.3 Binary Cross-Entropy

Binary cross entropy (BCE) is typically used within single-class classification problems. BCE measures the distance between a predicted sample $\hat{Y}$, and its true value Y, which is either 0 or 1.

$$\text{BCE}(Y, \hat{Y}) = -\frac{1}{n} \sum_{i=1}^{n} (y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log(1 - \hat{y}_i)) \tag{2.27}$$

2.7.4 Peak Signal-to-Noise Ratio

This method measures the Peak signal-to-noise ratio (PSNR), in decibels, between two images. The ratio is a quality measurement between an original and a compressed image. The higher the PSNR, the better the quality of the compressed image.

$$\text{PSNR}(Y, \hat{Y}) = 10 \log_{10} \frac{R}{\text{MSE}(Y, \hat{Y})} \tag{2.28}$$

where R is the maximum possible pixel value of the image.

2.7.5 Structural Similarity Index

While MSE and PSNR are simple to implement, they are not indicative of similarity as perceived by humans (Moorthy et al., 2011). The structural similarity index (SSIM) attempts to overcome this problem by taking into account structural information in images. The method assesses image quality based on the perceived change in structural information between two images (Z. Wang, Simoncelli, & Bovik, 2003). The SSIM index is calculated within *windows* of two images, moved across to cover all image locations. Using SSIM requires to set three hyperparameters, but assuming that they are all set to 1, the similarity between two windows, w and $\hat{w}$, both of size $n \times n$, is calculated as such

$$\text{SSIM}(w, \hat{w}) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{2.29}$$

where

$$\begin{aligned} \mu_x &: \quad \text{the mean of} \quad x \\ \mu_y &: \quad \text{the mean of} \quad y \\ \sigma_x &: \quad \text{the variance of} \quad x \\ \sigma_x &: \quad \text{the variance of} \quad y \\ \sigma_{xy} &: \quad \text{the covariance of} \quad x \quad \text{and} \quad y \\ c_1, c_2 &: \quad \text{stabilising coefficients} \end{aligned}$$

For details related to the hyperparameters and the stabilising coefficients, please refer to the original paper on the structural similarity index metric by (Z. Wang et al., 2003).

2.7.6 Intersection over Union

Also known as the Jaccard index, this specific metric is applied to categorical data, such as images of semantic segmentation. As the name implies, it quantifies the intersection of two categorical images, Y and $\hat{Y}$, divided by

their union.

$$\mathrm{IoU}(Y, \hat{Y}) = \frac{Y \cap \hat{Y}}{Y \cup \hat{Y}} \tag{2.30}$$

2.8 Visual Prediction with Deep Learning: A Literature Review

While deep learning models may be inspired by human cognitive processes (2.3.1), Lake et al. argue that for intelligent machines to better mimic human intelligence, they should build causal models of the world that support explanation and understanding, rather than merely solving pattern recognition problems (Lake, Ullman, Tenenbaum, & Gershman, 2016). Also, for an agent to be able to predict how a visual world will change over time, it should possess some understanding of object structure and the possible transformations objects may undergo (Lotter, Kreiman, & Cox, 2016). Recent developments in deep learning have enabled computers and robots to learn such internal models of physical systems from large sets of observations of the systems in question (P. W. Battaglia, Pascanu, Lai, Rezende, & Kavukcuoglu, 2016; Chang, Ullman, Torralba, & Tenenbaum, 2017; Watters et al., 2017). In turn, these internal models may be used to perform predictions of future states of physical environments. Castelló (2018) presents a comprehensive survey on deep learning methods for future frame prediction. Below is a compact review of some of these methods, including others, grouped by their methodological similarity.

2.8.1 Learning Physical Reasoning

By providing a deep learning model with prior knowledge to the environments' physics, i.e. what may be considered interactable objects, it can take advantage of these constraints to rapidly learn new tasks, adapt to changes, and naturally generalise reasoning to new scenes (Chang et al., 2017). This type of technique may predict how balls move on a pool table (Fragkiadaki

et al., 2015), how a tower of blocks is likely to collapse (Lerer et al., 2016) and even the motion of humans (Alahi et al., 2016). These studies are methodically similar in the way they consider only relevant objects in synthetic images, retrieve their locations and translate them to positional coordinates in a defined space, making it easier for a network to capture the essence of their movements effectively. However, retrieval and use of such information imply that the methods are supervised. Deterministic prediction of future velocities and locations of these objects is then made using a model's learnt intuition of physics, which in turn are used to create synthetic images of the future states. The models do not, however, generate visual states of the future directly, e.g. by means of a decoder. Therefore the networks need not retain image quality detail throughout a predicted sequence, as opposed to methods discussed next.

2.8.2 Pixel-Level Prediction

An opposite approach to predicting future states of physical environments is by pixel-wise processing of images. This is typically done by feeding one or more images into a recurrent (2.5.1) or convolutional (2.3.2) network which outputs a new image, assumed to be the next in the sequence. Lotter et al. (2016) investigated the ability to do this with frames in video sequences from car dashboard cameras, by training a model on large sets of such videos. They describe a neural network architecture, namely 'PredNet', inspired by the theory of predictive coding (2.1). Each layer in the network makes local predictions, forwarding only deviations from those predictions to subsequent network layers. Though giving good results for a few predicted time-steps, their method is prone to accumulation of pixel-level noise. As predictions are made deeper into the future, small errors in pixel space will exceedingly amplify, quickly spoiling the desired image quality (Wichers, Villegas, Erhan, & Lee, 2018).

An entirely different network architecture, more robust to the frame-wise accumulation of noise, predicts future scenarios in video games by analysing long sequences of gameplay (Ha & Schmidhuber, 2018). This article,

Figure 2.14: PredNet predicting two frames into the future of a dashboard camera video sequence. Adapted from Lotter et al. (2016).

namely 'World Models', received considerable attention because the researchers, among other things, created the first agent to solve the Car Racing environment in the OpenAI Gym (Brockman et al., 2016). Their proposed predictive model is probabilistic and composed of three parts. First, a visual component, or more specifically a VAE, compresses each observed input frame into latent representations. Then, a memory component predicts how these latent representations evolve, and finally, a controller component controls the agent. The visual component may decode the predicted latent representations to gain a visual interpretation of the future. The memory model is an LSTM combined with a mixture density network (Bishop, 1994) which produces distributions of predictions, allowing the network to be creative, and model various scenarios rather than just one trajectory.

Common to these methods is the fact that they train predictive models in a self-supervised manner, meaning the next element in a sequence is provided as a supervisory signal during training.

2.8.3 Masked Video Prediction

Also the following models are self-supervised. In semantic segmentation, image regions are masked and labelled by category, dividing the image into several objects or classes (2.6.4). Luc et al. (2017) show that using their method, predicting future semantic segmentation frames yields better results than pre-

dicting RGB images, and allows predicting deeper into the future. Focusing on traffic environments, the researchers explore the ability for deep learning to predict segmentation maps of not yet observed video frames that lie up to a second or further in the future. By turning an image into a segmentation map, the complexity of the environment is drastically reduced, making the predictive model able to estimate future states of the environment more accurately. The model developed in the study is an autoregressive CNN that learns to generate multiple segmented frames of traffic environments iteratively, and their results show that directly predicting future segmentations is substantially better than predicting and then segmenting future RGB frames (Luc et al., 2017). The intuition behind this is that it is easier to learn the dynamics of an environment that contains only high-level information (2.8.1), rather than complex and undefined content. Though Luc et al. (2017) show good results, their method fails to maintain the structure of objects when predicting more than a second or two into the future.

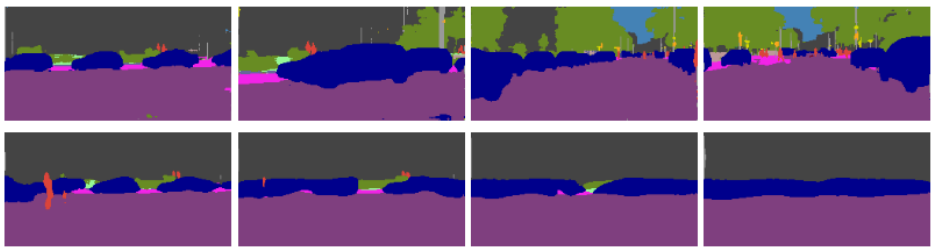


Figure 2.15: Long-term video prediction with semantic segmentation by (Luc et al., 2017). Ground truth segmentations at 1, 4, 7, and 10 seconds into the future (top row), and corresponding predictions of the autoregressive model (bottom row). Adapted from Luc et al. (2017).

Another line of research uses the latter idea to create a method that first estimates a mask of the object of primary interest, before predicting how that structure evolves in the future, and finally reconstructs the future frames (Wichers et al., 2018). In their research, the domain is restricted to human movement observed from a fixed position, which creates an environment with low complexity and only one transforming object at a time. The method trains an encoder to capture the high-level information, like a moving foreground object, while making sure that the decoder can reconstruct

the future frame. The method allowed the researchers to avoid compounding errors which occurs in recursive pixel-level prediction, and predict a large number of convincing future frames.

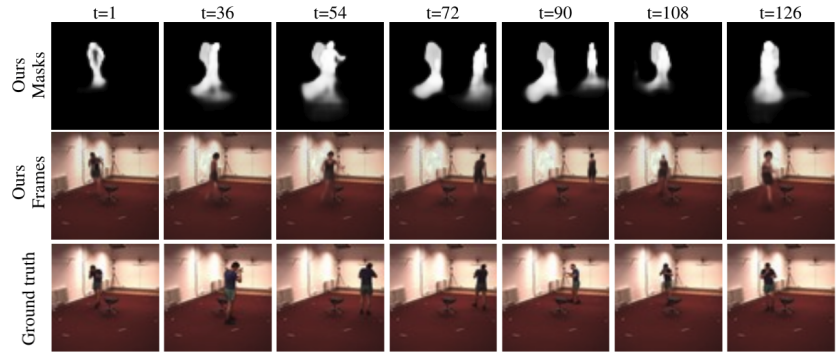


Figure 2.16: Long-term predictions of high-level human structures, reconstructed into photo-realistic images. Adapted from the results of Wichers et al. (2018).

2.8.4 Evaluating Predictive Models

There is currently a growing interest in the field of predicting the future using deep learning with an increasing number of contributions. When reviewing the literature I discovered that the focus in most of this work is mainly on designing models that are superior to the current state of the art, and that there is insufficient focus on scientific evaluation of these models. In the work discussed above, researchers evaluate and report results of their model, and the predominant approach of doing this is to perform a numerical frame-wise comparison between ground truth and predicted states or sequences. Moorthy et al. (2011) emphasises the importance of assessing image and video quality with methods that take into account how humans perceive visual stimuli. Among the reviewed research, the only contributions that describe other means of model evaluation in addition to a frame-wise comparison are by Lerer et al. (2016) and Wichers et al. (2018). Lerer et al. (2016) predict whether a tower of blocks will fall or not, and compare the outcome to human judgement in a small-scale experiment. Wichers et al. (2018) employ a crowd-sourced human preference evaluation to determine whether or not objects

are of correct colour and shape throughout a prediction. Castelló (2018) has made a similar remark about the shortcomings related to evaluating predictive deep learning models, and states that 'there is in general a lack of common test ground for this kind of models'.

Chapter 3

Research Methods

Section 2.8 reviews some of the most successful methods for predicting the future of visual environments using deep learning. While there exist a fair amount of research contributions to the field, predicting deep into the future of complex environments based on self-supervised learning still remains a challenge. This is largely due to the accumulation of errors in each predicted step, which with time, leads to significant inaccuracy. There is also a lack of ways to determine how good video predictions generated by deep learning models are, comparable to how humans would perceive them, as well as an understanding of how such predictions could be used to benefit the decision making of an intelligent agent. The following chapter argues for the choice of methods used to answer the research questions (section 3.1) and reviews the implementation of deep learning models (section 3.3), its learning environment (section 3.2) and a proposed method for evaluating predictive models (section 3.4).

3.1 Considerations

The amount of research related to deep learning is vast, and there exists a number of successful and less successful documented deep learning models for various tasks. One of the thesis' research contributions is a deep learning model architecture for visual prediction, and to narrow the possibilities

related to the design of this model architecture, I define the following criteria:

- The model must be applicable to complex visual environments, which includes a moveable point of view and numerous moving objects of different shapes and sizes.
- Given a complex visual environment, the model must be able to predict long-term future states while persevering object structures and avert exposure to cumulative noise.
- The model must learn an understanding of the environment in a self-supervised fashion.
- The model must be trained using a gradient-based optimisation method.

3.1.1 The Model Architecture

Among the types of methods discussed in section 2.8, the one that best fits the above criteria is perhaps World Models by Ha and Schmidhuber (2018). This model is comprised of three components; a visual component, a memory component and a controller component. The controller component is essentially an agent's decision-making tool. However, in this thesis, I have chosen to disregard the implementation of any intelligent agent and focus on predicting long-term futures. In the original paper, the authors used a VAE (section 2.4.2) for the visual component and an LSTM network (section 2.5.2) for the memory component.

A VAE produces continuous latent variables; however, van den Oord et al. (2017) argue that discrete representations are a natural fit for complex reasoning, planning and predictive learning. A VQ-VAE (section 2.4.3) outputs such discrete latent variables using a vector quantisation layer while maintaining the ability to reconstruct the images from these representations. The VQ-VAE is shown to output sharp images and does not assume the data's distribution, as opposed to a VAE which attempts to fit the latent variables to a defined distribution, e.g. a normal distribution. In conclusion, I take inspiration from World Models when creating the basis for the

proposed predictive model, but implement only a visual component and a memory component, and replace the VAE with a VQ-VAE. Risi and Stanley (2019) have documented a similar architecture, although, it is not trainable with a gradient-based optimisation method.

3.1.2 Learning to Predict Within an Environment

When selecting an environment for the model to operate within, I require it to be complex in terms of possessing properties similar to the real world. Such properties involve an element of uncertainty as well as numerous objects moving simultaneously, including the point of view. Lotter et al. (2016) state that 'the visual world is alive with movement, driven both by self-motion of the viewer and the movement of objects within the scene'. Ha and Schmidhuber (2018) use video games such as Car Racing (Brockman et al., 2016) and ViZDoom (Wydmuch & Kempka Michałand Jaśkowski, 2018) as their domain to test their model. Especially ViZDoom, a research-related implementation of the classic video game 'Doom' from 1993, is a challenging environment that would be interesting to study. However, it is targeted toward reinforcement learning mainly, i.e. controlling of an agent.

The Cityscapes dataset (Cordts et al., 2016), which is a traffic environment for semantic urban scene understanding, is a popular choice of domain among others (Lotter et al., 2016; T.-C. Wang et al., 2017; Luc et al., 2017). An advantage of such traffic environment datasets, both real and synthetic, is how they provide various image representations such as semantic segmentation, depth, RGB and optical flow. In my experiments, I use the the CARLA traffic environment simulator (Dosovitskiy et al., 2017) because it can provide unlimited amounts of data of various sensory types. More details on this simulator are found in section 3.2.

3.1.3 Effects Associated with Image Types

Experimenting with different image types, more specifically RGB and semantic segmentation images, requires the model architecture to be capable of processing them both. Luc et al. (2017) proposed a quite different model

architecture for a similar prediction task; an autoregressive CNN. Using this autoregressive CNN, the researchers demonstrated the utility of representing the visual environment with semantic segmentation images as opposed to RGB images. However, the model which I propose interprets the environment somewhat differently than an autoregressive CNN. Therefore, I investigate if also the proposed model's predictive abilities are improved by using semantic segmentation as the visual input, which is the effect Luc et al. (2017) describe. All details related to the model implementation are found in section 3.3. Determining if the model accurately and meaningfully predicts the future using either of these image types is closely related to the main research question, as discussed below.

3.1.4 Evaluating Video Predictions

Evaluating video predictions is a field that has not been investigated in-depth, and there is a lack of well-established standards for measuring the quality and plausibility of such predictions (Castelló, 2018). Metrics that measure quality and similarity between two images, such as PSNR, SSIM and IOU (section 2.7), are usually the same metrics used to evaluate video predictions. A frame-wise comparison does not take into account contextual information specific to the environment (discussed further in section 3.4.4). According to Moorthy et al. (2011), the ultimate receivers of images and videos are human observers, and not all differences in images are equally significant for humans. Since many similarity metrics (section 2.7) for frame-wise comparison do not share the same visual perception as humans (Moorthy et al., 2011), I argue that the research field of visual prediction with deep learning is in demand of a new, flexible but at the same time domain-specific evaluation method for evaluating predictions.

An evaluation method without knowledge of the events of interest possibly fails to evaluate the quality of a long-term video prediction appropriately. Nevertheless, the frame-wise comparison is the prevalent approach to evaluating video predictions, and to comply with the norms of the research field, this thesis makes use of the same evaluation metrics. However, even

though the frame-wise comparison between consecutive ground truth and predicted frames may be useful to indicate certain things such as temporal accumulation of noise, it does not measure a prediction's main events or whether it is a plausible trajectory.

Especially challenging is exploring how the quality of video predictions are impacted by the type of image data that is used. It would not be fair to apply any of the similarity metrics discussed in section 2.7 to such distinct data types and expect a reasonable comparison. Generally speaking, evaluating unsupervised models is challenging because vague properties such as *meaningfulness* is difficult to measure analytically (Wortman Vaughan, 2018). Vaughan also argues that crowdworkers' opinions may be well suited to indicate coherency between categorical data.

3.1.5 Context is Important

What does it mean that a model produces *meaningful* outputs, and how can this property be measured analytically? Whether a piece of information is considered meaningful depends completely on what that piece of information is meant to represent, in other words; meaningful information is contextual (Cox, 2014). The proposed predictive model attempts to learn the general dynamics and rules of the specific traffic environment, and use this knowledge to predict accurate future scenarios that are, ideally speaking, deceivingly similar to the true scenarios. The context is the CARLA traffic environment, thus meaningfulness is related to this specific context.

If a human is to describe video, or more specifically an event from the environment, she or he would presumably do so with words, by category, or by how realistic she or he perceived it to be. A prediction may be identical to a true scenario, categorically speaking, even though objects are not completely overlapping in space or time. Take for instance a scenario where in the ground truth there are four vehicles passing in the left lane, but in the prediction there are only two. Although the prediction failed to foresee all the vehicles, it is still very accurate in relation to the context. A frame-wise comparison might on the other hand consider this a poor prediction because

of the dissimilarity in the image regions where the vehicles are missing. I argue that since a frame-wise comparison does not take the required context into consideration, it might be an unreliable way of evaluating the predictive model.

3.1.6 A Mixed Methods Research Design

These considerations lead to the idea of conducting surveys to help evaluate the content of video predictions. In her book, Willig (2013, p. 28) argues that sometimes the most appropriate way to answer a research question requires the use of two or more research methods. She suggests that combining qualitative and quantitative methods within the same study may help answer related questions. This approach is referred to as *mixed methods research* design (MMR, section **??**), a methodology for conducting research that involves collecting, analysing and integrating quantitative and qualitative research to expand and strengthen a study's conclusions (Schoonenboom & Johnson, 2017). Mixed methods research may be used when the integration of these research types provides a better understanding of the research problem than either of each alone.

Describing the contents of video clips, or more specifically video predictions, is a subjective matter which should not be left to one single person or machine to do, in order to preserve a shared opinion. This indicates that using a quantitative method alone might not suffice for the research this thesis. I suggest first conducting a simple qualitative survey to learn more about aspects of the environment that are considered important to humans, followed by a quantitative survey designed on the basis of these results in order to collect a larger amount of data for further analysis. Finally, by using numerical and statistical methods on the collected data, we may learn how good the proposed model is at predicting the future of the established environment, and to what degree the choice of image type will help predict a meaningful and accurate future.

3.2 The Dataset

The data used to train and evaluate the proposed model is obtained by using the CARLA simulator. CARLA is an open-source simulator designed for autonomous driving research. It features open digital assets such as urban layouts, buildings and vehicles that were created for this purpose and can be used freely (Dosovitskiy et al., 2017). The simulation platform supports flexible specification of sensor suites and environmental conditions. For the experiments in this thesis, CARLA 0.8.4 was set up to have a car drive around in a predesigned town while storing images from two camera feeds. The agent manoeuvring the vehicle was CARLA's built-in autopilot, following arbitrary routes while obeying all traffic rules and regulations defined by the simulator's default configuration.

The data generation was conducted in individual sequences with lengths of 500 frames. The initialisation of new sequences involves assigning vehicles and pedestrians to the town. The numbers units were chosen randomly between 150 and 250. The agent vehicle spawned in an arbitrary position within the city, before beginning to manoeuvre. The weather condition was randomly set to be either default, clear noon, or cloudy noon upon sequence start.

Figure 3.1: Using the CARLA simulator, a vehicle automatically drives around in an environment while gathering data of two image types.

The agent used built-in camera sensors to gather RGB images and corres-

ponding semantic segmentation labels, synchronised to represent the same sequence. These images were stored with a height and width of 256 pixels, at a frame rate of 15 frames per second. The RGB images have three colour channels; red, green and blue, while the semantically segmented images have 13 class channels, each representing one of 13 object. In total, 1479 sequences were recorded for each image type, equalling a total of 2×7395500 images. All the specification details on simulator conditions and object classes are summarised in table 3.1.

CARLA specifications			
Simulator conditions		Object classes	
CARLA version	0.8.4	0	None
Environment	Town 02	1	Buildings
Agent controller	Built-in autopilot	2	Fences
Image dimensions	256 x 256 pixels	3	Other
Frame rate	15 FPS	4	Pedestrians
Sequence length	500	5	Poles
Weather condition	Random $x \in$ [default, clear noon, cloudy noon]	6	Road lines
		7	Roads
		8	Sidewalks
no. Sequences	1479	9	Vegetation
no. Pedestrians	Random $x \in [150, 250]$	10	Vehicles
no. Vehicles	Random $x \in [150, 250]$	11	Walls
		12	Traffic signs

Table 3.1: Table summarising the CARLA simulator conditions and object classes which were used when creating the dataset.

3.3 Implementation of the Model

The predictive deep learning model presented in this thesis is a system comprised of two components. A visual component first compresses what it sees into discrete, abstract representations. A memory component then uses these

abstract representations of the visual environment to learn an internal model of the world's behaviour, that is used to predict future states of the environment based on recent events. The work of Ha and Schmidhuber (2018), World Models, inspires this setup, which builds upon the idea that intelligent agents benefit from running an internal model of the world it operates within. The proposed architecture's main difference from World Models is its use of discrete image representations. Also, both components are deterministic, which makes the model as a whole deterministic. The model implementation is open-sourced and may be found at `https://github.com/plaffa/discrete-world-models`.

3.3.1 The Visual Component

The visual component is a vector quantised-variational autoencoder (section 2.4.3). The VQ-VAE is an encoder-decoder network that learns discrete latent representations of its inputs, while maintaining the ability to reconstruct the images from these representations. Here, the network is designed to receive images of height and width 128×128, and output latent representations of size 8×8. Two versions of this network are created, with minor differences; one with 3 input channels to support RGB images, and the other with 13 input channels to support semantically segmented images, hereby referred to as VQ-VAE$_{\text{RGB}}$ and VQ-VAE$_{\text{SEG}}$, respectively. In both cases, the encoders output discrete 5-bit latent variables of size 8×8, meaning each element in the latent space contains an integer in the range $[0, 31]$.

Autoencoders are essentially compression systems, and given an 8-bit image I_{RGB}, and a 1-bit image I_{SEG}, achievable bit compression ratios are

$$\text{VQ-VAE}_{\text{RGB}}: \quad \frac{128 \times 128 \times 3 \times 8}{8 \times 8 \times 5} \approx 1229 : 1$$

$$\text{VQ-VAE}_{\text{SEG}}: \quad \frac{128 \times 128 \times 13 \times 1}{8 \times 8 \times 5} \approx 666 : 1$$

One could argue that since the latent space contains discrete variables, the VQ-VAE is somewhat restricted when it comes to the variety of images it

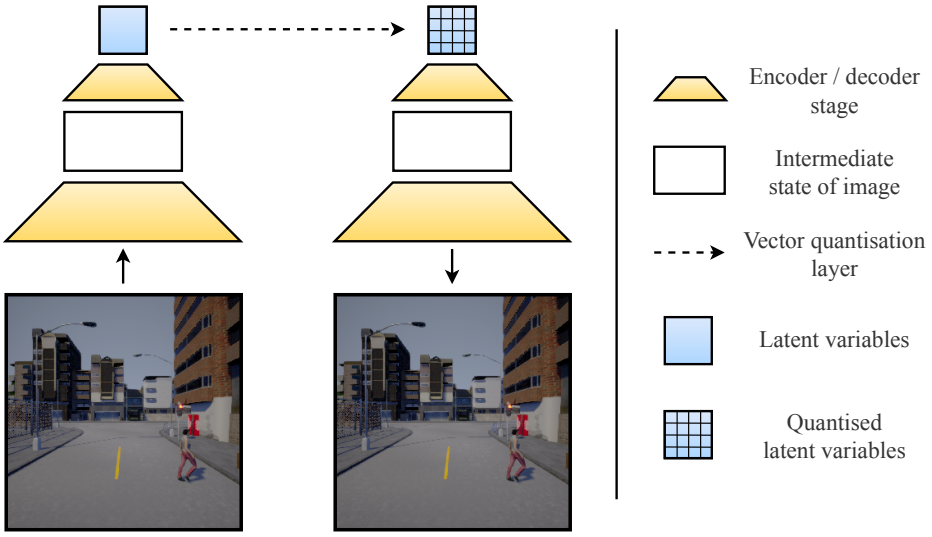


Figure 3.2: An overview of the visual component's structure (left) and an explanation of its modules (right).

may express. However, this is probably not an issue since with the given specifications there exist up to 32^{64} discrete latent space combinations, which theoretically yield more than $2.135 \cdot 10^{96}$ unique images.

The training objective is given by equation 2.15. For the VQ-VAE$_{\text{RGB}}$, $\mathcal{L}_{recon}$ is an MSE function (section 2.7.1), while for the VQ-VAE$_{\text{SEG}}$ it is a BCE function (section 2.7.3). Pixel-wise class weighting is integrated to these loss functions, using class weights w_c proportional to the distribution of classes c in each mini-batch of semantic segmentation data

$$w_c = 1 - N_c/N$$

where N_c is the number of elements (pixels) of class c and N is the total number of elements (pixels) constituted by all classes in one mini-batch of semantic segmentation data. The weighting of the loss functions involves using a priori information such as labels inherited in the semantic segmentation data. Therefore, the models are not strictly speaking self-supervised. However, when working with unbalanced semantic segmentation datasets, meaning some classes occur more often than others, it is advised to balance the class frequencies by introducing weights (Ronneberger, Fischer, & Brox,

2015). For an evaluation comparison to be fair across models tailored to distinct image types, I argue that the same weighting should apply to VQ-VAE_{RGB}. The weighting also helps both models pay more attention to detail, i.e. classes of small size or classes that seldom occur.

I use the released VQ-VAE implementation in the Sonnet library [1] [2] for inspiration, with some slight architectural modification. The full details on the VQ-VAE architecture are summarised in figure 3.6.

3.3.2 The Memory Component

The memory component learns an internal model (section 2.1) of the environment. It consists of two fully connected layers (section 2.3.1) and two LSTM layers (section 2.5.2) that receive as input a sequence of latent representations produced by the visual component and outputs a statistically probable continuation of these representations. This means that it does not learn to model a series of images directly, but rather a sequence of latent variables that represent these images. The architecture of the memory component remains the same no matter which image type is used.

The training objective is to minimise the cross-entropy (section 2.7.2) between a predicted sequence of latent representations, $\hat{Y}$, and a target sequence of latent representations, Y, given an input sequence X:

$$
\begin{aligned}
X &= [x_t, x_{t+1}, x_{t+2}, \cdots, x_{k-1}], \\
Y &= [x_{t+1}, x_{t+2}, x_{t+3}, \cdots, x_k], \\
\hat{Y} &= [\hat{x}_{t+1}, \hat{x}_{t+2}, \hat{x}_{t+3}, \cdots, \hat{x}_k],
\end{aligned}
$$

where x_t and $\hat{x}_t$ are 8×8 latent representations at time step t and k is the number of latent representations in a sequence. Nearly all sequences in the dataset are dominated by large, still objects in the image domain, such as roads and buildings. As a result, consecutive latent representations share highly similar characteristics. A simple way for the memory component to

[1] `https://github.com/deepmind/sonnet/blob/v2/sonnet/src/nets/vqvae.py`

[2] `https://github.com/deepmind/sonnet/blob/master/sonnet/examples/vqvae_example.ipynb`

achieve low loss values is then to learn to predict a copy of the input sequence. To help the memory component learn the dynamics of small, rarely occurring objects in these sequences, the cross-entropy loss function is weighted by a matrix $\sigma_{n\times m}$ containing the standard deviations of the latent representations' elements across all sequences in a mini-batch of data

$$\sigma = \sqrt{\mathrm{Var}(X)} = \begin{bmatrix} \sigma_{1,1} & \sigma_{1,2} & \sigma_{1,3} & \cdots & \sigma_{1,m} \\ \sigma_{2,1} & \sigma_{2,2} & \sigma_{2,3} & \cdots & \sigma_{2,m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{n,1} & \sigma_{n,2} & \sigma_{n,3} & \cdots & \sigma_{n,m} \end{bmatrix}$$

where X is a mini-batch of sequences of latent representations and n and m are the height and width of the latent representations, which in this case is 8×8. Weighting the loss function with σ forces the memory component to pay more attention to regions in the latent space that contain dynamic properties.

In order to preserve the latent representations' discrete characteristics, they are one-hot encoded (section 2.6.2) and flattened before they are sent to the memory component, as shown in figure 3.3.

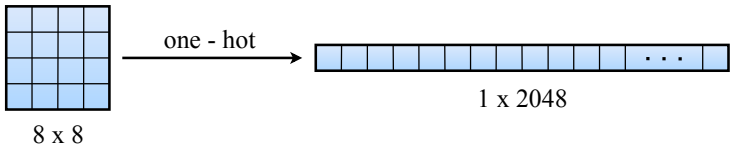


Figure 3.3: One-hot encoding and flattening of a latent representation.

The flattened representations are fed through a fully connected layer with 2048×1024 nodes, before entering the LSTM for recurrent processing. The predicted hidden states of the LSTM are fed through another fully connected layer with 1024×2048 nodes, which is followed by a 32-dimensional argmax operation and reshaping to get the desired 8×8 latent space shape. The memory component can receive sequences of arbitrary lengths, and predict as many future steps as desirable, which makes it applicable for producing both short-term and long-term predictions. The complete architecture of the

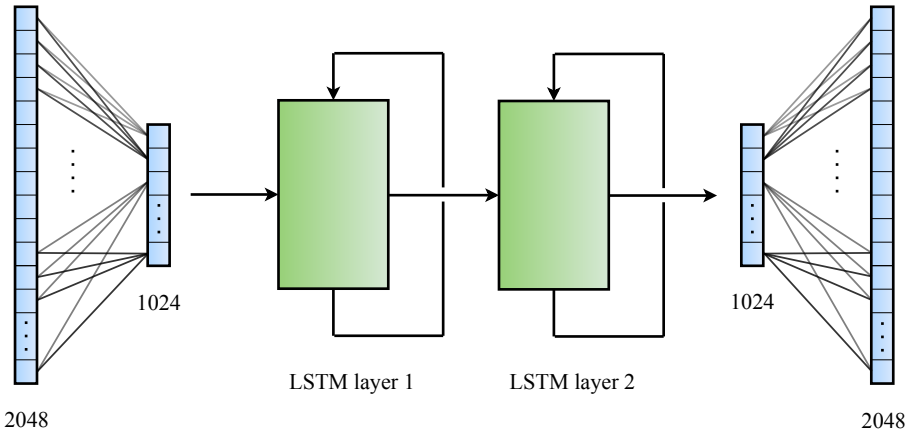


Figure 3.4: An overview of the memory component consisting of a fully connected input layer, two LSTM layers and a fully connected output layer.

memory component is summarised in figure 3.6.

3.3.3 Putting the Components Together

The visual and memory components work together to perceive, compress and predict future states of the environment based on recent events. Figure 3.5 illustrates the complete predictive model, consisting of the two components put together.

3.3.4 Model Training Procedure

There are two instances of the model with minor differences; $\text{model}_{\text{RGB}}$ and $\text{model}_{\text{SEG}}$. The individual components and models as whole are implemented in PyTorch 1.1.0 (Paszke et al., 2019), and use the Adam optimisation method (Kingma & Ba, 2015) for learning parameters. The components are trained separate from one another, similar to (Ha & Schmidhuber, 2018).

Training the Visual Component

Both $\text{VQ-VAE}_{\text{RGB}}$ and $\text{VQ-VAE}_{\text{SEG}}$ are trained on images scaled from size 256×256 to size 128×128, using nearest neighbour interpolation for semantic segmentation images and bicubic interpolation for RGB images. The images

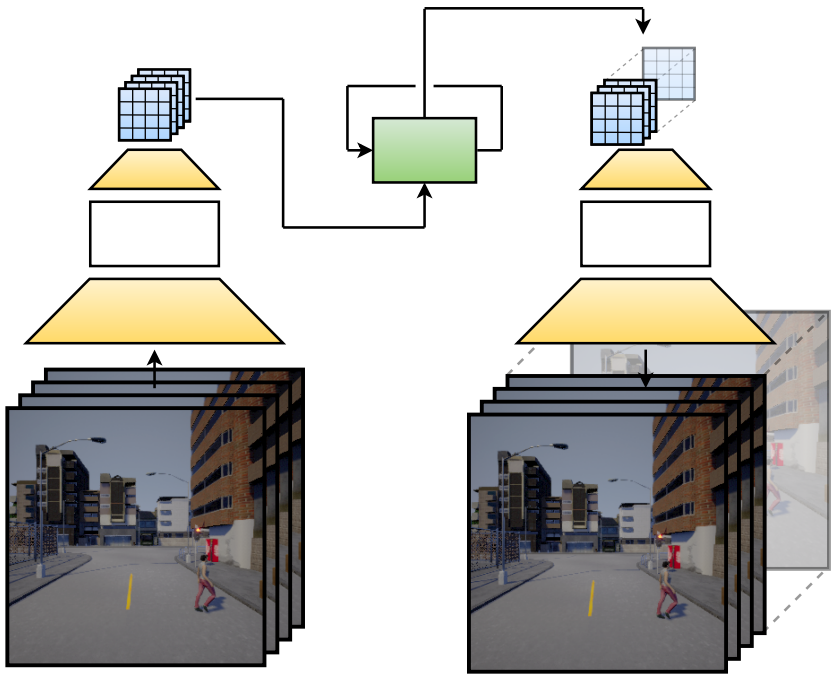


Figure 3.5: The complete predictive deep learning model comprised of the two components. The visual component compresses a sequence of images into a corresponding sequence of latent representations. The memory component uses this as a condition to predict an arbitrary number of future states, that are decoded by the visual component.

used to train the VQ-VAEs are from 514 of the 1479 sequences described in section 3.2. Both VQ-VAEs are trained with a learning rate of 3e−4 for 80 epochs with shuffled mini-batch sizes of 64.

Training the Memory Component

After training the VQ-VAEs, the remaining 965 sequences described in section 3.2 are divided into a training set and a validation set used to train the memory component. The training set contains 773 sequences, while the validation set contains 192 sequences. This validation set is also used for evaluation purposes in the next section, 3.4. From these datasets, the visual components extract latent representations which are stored locally for efficient training of the memory component. The sequences are originally of length 500 and frame rate 15 FPS (section 3.2), but the frame rate is re-

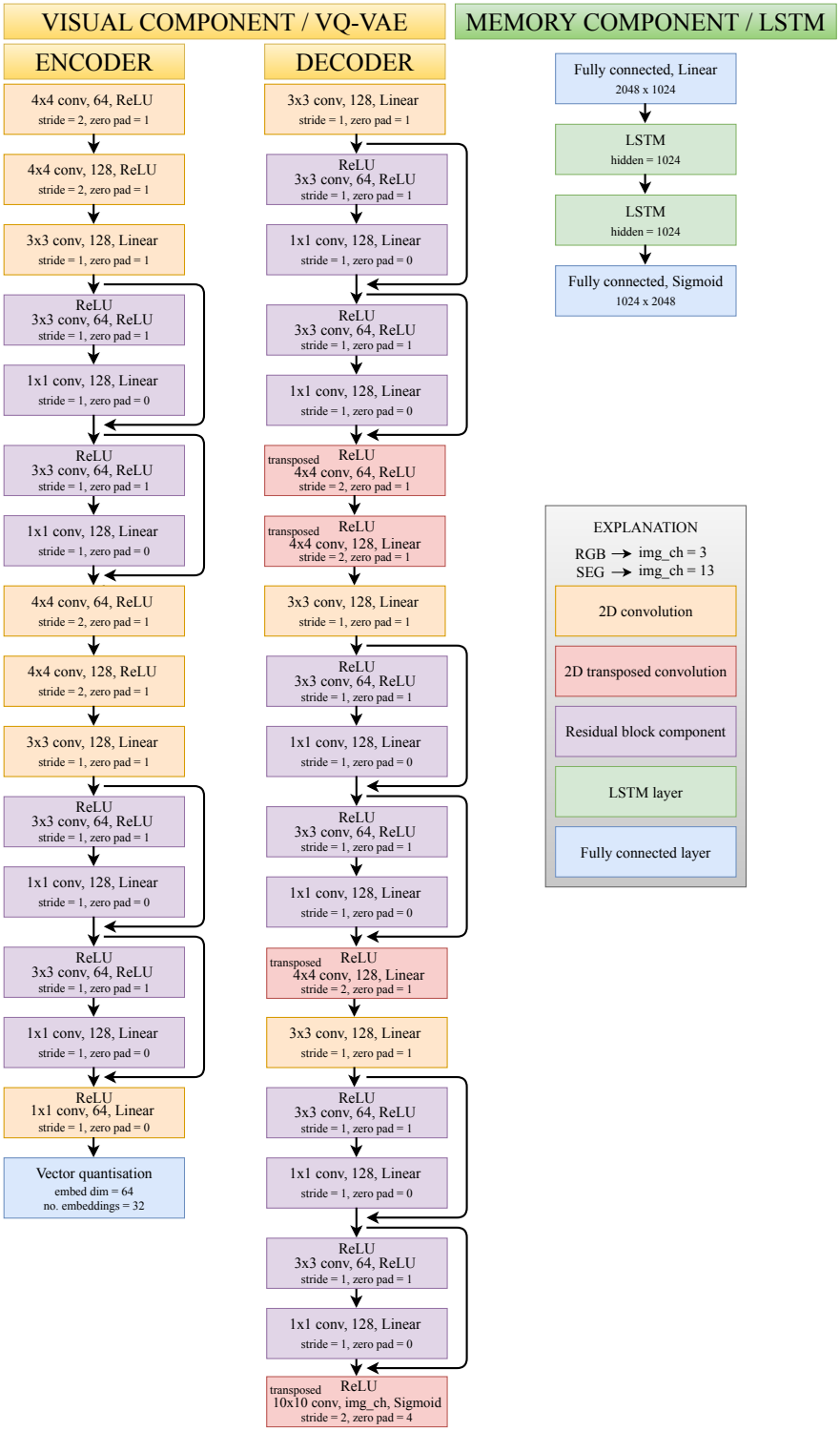


Figure 3.6: The component architectures that comprise the proposed deep learning models.

duced to 5 FPS by splitting the sequences into groups of three of lengths 164. The memory component is trained on sequences from the training set [3] with a learning rate of 3e−4 using shuffled mini-batch sizes of 64. All layers use dropout regularisation (section 2.3.5.2) with a probability of $p = 0.3$. To reduce overfitting (section 2.3.5.1), the training is interrupted by early stopping (section 2.3.5.2) when there is no further improvement on the validation set [4].

3.4 The Proposed Evaluation Method

The method used to evaluate model performance is based on a mixed methods research design consists of two surveys; a qualitative survey S_1 and a quantitative survey S_2. Conducting S_1 establishes which aspects of the environment (section 3.2) that should be considered important. This information is used within S_2 to collect a large amount of data related to the predictive abilities of the proposed models. The S_2 submissions are then analysed using numerical and statistical methods that assess model performance.

Both surveys are web-based, in which participants observe short video clips and perform certain tasks related to these videos. The University of Oslo's 'Nettskjema' service is used for this purpose. All videos have a frame rate of 5 FPS and a length of 50 frames, which equals a 10-second duration in the environment. However, for visual purposes the videos' playback speed is increased by a factor of two. The predictive models' visual components (section 3.3.1) process all videos such that ground truth and predicted videos appear visually similar so as not to create a bias in the mixed methods research. The layouts of S_1 and S_2 may be found at `https://plaffa.github.io/surveys.html`.

[3] training sequences with 164 steps at a frame rate of 5 FPS
[4] validation sequences with 164 steps at a frame rate of 5 FPS

3.4.1 The Qualitative Survey

In the qualitative survey, S_1, seven participants obtained by convenience sampling observed and created descriptions for 52 ground truth video clips from the specified traffic environment. M. Battaglia (2008) defines convenience sampling as 'a type of nonprobability sampling in which people were sampled simply because they are "convenient" sources of data for researchers'. The ground truth videos were sampled from the validation dataset described in section 3.3.4, though first inspected to make sure they represent a variety of events. More specifically, a video corresponds to the first 150 frames of a sampled sequence, but to reduce the frame rate from 15 FPS to 5 FPS only every third frame is selected, resulting in a length of 50 frames.

Among the 52 videos, 26 are of image type RGB and the other 26 of semantic segmentation, presented in a random order. All sequences are processed by the visual component (section 3.3.1), and are thus subject to some quality change. Before beginning the survey, the participants were thoroughly informed about the thesis' topic, and that the descriptions they create should mainly relate to traffic, roads and interaction between vehicles. Furthermore, they were shown how the same video may be represented with different image types, but they did not know whether they were going to see ground truth or predicted sequences. For convenience, the descriptions they created could consist of up to four words, which suggested a clip's main events. New participants were recruited until theoretical saturation among the descriptions was achieved, as suggested by Willig (2013, p. 71). This means that as long as new video descriptions can be identified, the data collection should continue, which for this experiment was achieved with seven participants. The submitted video descriptions usually portrayed one or two main events per video, and the most recurring descriptions among these were accepted for the quantitative survey, S_2, as shown in table 3.2 below.

3.4.2 The Quantitative Survey

In the quantitative survey, S_2, the eight categories obtained from S_1 are used by new participants to describe new videos. In addition, another category,

Recurring descriptions selected as categories			
c_1	Driving straight	c_5	Breaking for traffic
c_2	Left turn	c_6	Approaching vehicle
c_3	Right turn	c_7	Oncoming vehicle(s)
c_4	Standing still	c_8	Following vehicle

Table 3.2: The eight most recurring descriptions created by participants of S_1 to be used as categories for S_2.

Summary of the qualitative survey S_1			
Research type	Qualitative	**Format**	
		Survey	
no. Participants	7	**Gender distribution**	**Driving licence**
		57% female, 43% male	71% YES, 29% NO
no. Videos	52	**Type**	
		Ground truth	RGB and semantic segmentation
no. Categories	8		

Table 3.3: Summary of the qualitative survey, S_1.

'undefined', is added to be used in cases where the participants of S_2 consider none of the other categories to be appropriate. The videos of S_2 are 52 unique sets of video sequences. This means that for both image types, RGB and semantic segmentation, there are 52 ground truth videos and 52 corresponding video predictions, representing the same sequences across image types. I define a *test set* as the collection of these $4 \times 52 = 208$ videos, selected as follows:

1. With a discrete uniform distribution, select 52 random numbers from the interval $[0, 191]$, corresponding to the sequences of the validation set (section 3.3.4).

2. Delimit the sequences to frame numbers 225-405 (not to overlap with the videos of S_1), i.e. to a total of 180 frames. Reduce the frame rate

from 15 FPS to 5 FPS by selecting every third frame, resulting in a length of 60 frames.

3. For both image types, select the 10 first frames from each of these 52 sequences to be used as *conditions* to the predictive models.

4. For each condition, select the 50 subsequent images in the sequence as the ground truth videos.

5. Apply each condition to the predictive models and save the predicted sequences, each comprising of 50 predicted frames

Owing to the large number of videos in the test set, S_2 is divided into four subsets with 52 videos each, so as not to tire the participants. However, they may complete as many of the subsets as they wish, though only the same subset once. To reduce a potential ordering effect (Strack, 1992), the order of all videos is randomised across the four subsets. The participants do not know when they observe a ground truth video or prediction. Every video is followed by two inquiries, shown in tables below. The first inquiry records a participant's category of choice for the video on a nine-level nominal scale, and the second inquiry records its degree of perceived realism in the video on a five-level ordinal scale.

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9
Category	o	o	o	o	o	o	o	o	o

Table 3.4: A participant of S_2 selects one of the nine possible categories to describe a video's event.

	Unrealistic	Somewhat unrealistic	Neither	Somewhat realistic	Realistic
Realism	o	o	o	o	o

Table 3.5: A participant of S_2 reports how realistic it perceives a video to be on a five-level ordinal scale.

As discussed in section 3.1, properties such as *meaningfulness* and *realism* are challenging to measure analytically, and should thus be well defined not to

confuse the participants and make sure they all have a similar understanding of what the property means. The Lexico dictionary service defines realism as 'the quality or fact of representing a person or thing in a way that is accurate and true to life'. This definition was extended to fit the research problem, and provided in S_2 as follows:

> In the context of these videos, realism is thought of as how realistic the events are in comparison to real life events. This should not consider the quality of the images, but rather whether you think it is likely that the scenario or trajectory could happen in a real traffic environment. A realistic video should also have a relatively continuous flow, and stable transitions between consecutive images.

In total, S_2 received 453 submissions (not all were from unique participants), of which approximately 95% were from users of the Amazon Mechanical Turk service (Crowston, 2012). These users received a small payment for completing the survey, while the remaining proportion of submissions were from participants obtained by convenience sampling and did not receive any payment.

Summary of the quantitative survey S_2			
Research type	Quantitative	**Format**	
		Survey	
no. Submissions	453	**Gender distribution**	**Driving licence**
$S_{2.1}$	110		
$S_{2.2}$	106	30.3% female	96.3% YES
$S_{2.3}$	112	69.7% male	3.7% NO
$S_{2.4}$	125		
no. Videos	208	**Type**	
$S_{2.1}$	52		RGB and
$S_{2.2}$	52	Ground truth	semantic
$S_{2.3}$	52	and prediction	segmentation
$S_{2.4}$	52		

Table 3.6: Summary of the quantitative survey, S_2. $S_{2.i}, i \in [1, 2, 3, 4]$ are the four subsets of S_2.

3.4.3 Outlier Detection

One drawback of using crowdworkers' opinions is that one cannot guarantee the quality of each submission. It could be tempting to believe that most of the participants understand the task in S_2, wish to contribute and thereof provide their honest opinions on the topics. However, since the participants are rewarded with a payment, some may be motivated by other aspects, such as completing the survey quickly only to claim the reward and consequently ignore its content altogether. This is unfortunate as these participants induce a response bias, i.e. a nonrandom deviation of the answers from their actual value (Lavrakas, 2008). Submissions that deviate considerably from the population trend may be considered outliers and should be identified and rejected to assure that they do not influence the result. There exist numerous methods for rejecting outliers in data. One approach is to accept only samples that are within some factor of the standard deviation from the population mean, and discard the rest. Though this method may be used, the mean and standard deviation are quite sensitive to extreme values in the data, making it somewhat unreliable. A more robust method utilises the median absolute deviation (MAD) to detect outliers (Leys, Ley, Klein, Bernard, & Licata, 2013). Using this method, we can determine whether a sample x_i is an outlier by using the following expression

$$\frac{|x_i - \text{median}(\underline{x})|}{\text{MAD}(\underline{x})} > T \tag{3.1}$$

where $\underline{x} = \{x_1, x_2, \cdots, x_n\}$ is the population data and $x_i \in \underline{x}$ is a sample from the population data. $\text{MAD}(\underline{x})$ is the median of all samples' absolute deviations from the population median, multiplied by an empirically derived constant k

$$\text{MAD}(\underline{x}) = k \cdot \text{median}(|\underline{x} - \text{median}(\underline{x})|) \tag{3.2}$$

If the calculated value exceeds the threshold T, the sample x_i is flagged as an outlier. This formulation of the method is applicable when x_i is a scalar, but in our case, the samples are vectors of category frequencies. Each sample represents a submission provided by a participant, or more specifically, a

vector containing the number of times the participant chose each category, as shown in the example table 3.7.

Participant	c_1	c_2	c_3	c_4	$\cdots$	c_n
$\underline{p}_1$	24	7	6	3	$\cdots$	0
$\underline{p}_2$	2	5	23	13	$\cdots$	1
$\underline{p}_3$	11	14	18	8	$\cdots$	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	5
$\underline{p}_m$	21	3	4	9	$\cdots$	0

Table 3.7: An example of what the summarisation of survey submissions may look like. Here, $\underline{p}_i, i \in [1, 2, \cdots, m]$ represents a participant's submission to a subset of S_2, i.e. the participant's frequency count of each category c_j for $j \in [1, 2, \cdots, n]$. m is the total number of submissions to the subset and n is the number of categories.

The complete set of submissions X is a matrix with shape $m \times n$, where X_p is a row of X containing the frequencies of categories from participant p. Then, a row of the transpose of X, i.e. X_c^T, contains the frequencies of category c for all participants. Outlier detection using Mahalanobis' distance (Mahalanobis, 1936) is an acknowledged way of detecting outliers among such multivariate samples. However, that involves computation of the inverse covariance matrix of X, $\text{Cov}(X)^{-1}$, which may be unstable or even impossible if encountering singularity issues. Therefore, we stick with the MAD based method, and reformulate it by using the euclidian distance to make equation 3.1 applicable to vectors.

$$\frac{\sqrt{\sum_{c=1}^{n}(X_{pc} - \text{median}(X_c^T))^2}}{\text{MAD}^*(X)} > T \tag{3.3}$$

where

$$\text{MAD}^*(X) = k \cdot \text{median}(\sqrt{(X - \text{median}(X))^2}) \tag{3.4}$$

and $\text{median}(X) = [\text{median}(X_1^T), \text{median}(X_2^T), \cdots, \text{median}(X_n^T)]$ are the medians of the columns of X. Finally, equation 3.3 is used to create a function

that determines whether the submission by participant p is an outlier

$$\text{outlier}(X_p) = \begin{cases} \text{True}, & \text{if } \frac{\sqrt{\sum_{c=1}^{n}(X_{pc} - \text{median}(X_c^T))^2}}{\text{MAD}^*(X)} > T \\ \text{False}, & \text{otherwise} \end{cases} \tag{3.5}$$

Now, this function should indicate participants who have responded in such a manner that deviates from the population trend. As discussed at the beginning of the section, participants that complete the survey quickly may induce problems. Spending an unreasonably long time completing the survey could also be suspicious. The described function (equation 3.5) does not take the participant's time spent into consideration, and it might suggest an outlier even though the elapsed time is reasonable. Therefore, the function is used in conjunction with another criterion; the participants' time spent, t_p, must be within one standard deviation of the population mean

$$\mu_{time} - \sigma_{time} \leq t_p \leq \mu_{time} + \sigma_{time} \tag{3.6}$$

If a participant exceeds both these limits, there is good reason to believe that it is an outlier and thus remove it from the set.

3.4.4 Analysing Survey Submissions

The result of conducting both surveys, S_1 and S_2, and refining S_2 by removing submission outliers, is a collection of subjective data about all videos in the test set (section 3.4.2). More specifically, data has been gathered about the categorical preference of each video on a nine-level nominal scale, in addition to their degree of realism on a five-level ordinal scale. The following section describes how the obtained data may be used to evaluate the predictive models. Considering that the MMR design is a candidate approach to evaluating predictive deep learning models, its conclusions are compared to the recognised evaluation approach, a frame-wise comparison.

3.4.4.1 The Overall Distribution of Categories

The total amount of submissions to S_2 constitute a distribution of event categories. More specifically, the submissions constitute a distribution of categories for each group of video, RGB_{GT}, RGB_{PD}, SEG_{GT} and SEG_{PD}. Using a goodness of fit test, it is possible to compare an observed distribution associated with predictions to a probability distribution associated with ground truth. These distributions are of unknown shape, and the data is nominal; therefore, we require a non-parametric test that can handle 2×9 contingency tables. The Chi-square goodness of fit test ("Chi-square Goodness of Fit Test", 2008) fits these criteria. Using this test, we investigate whether or not there is a significant difference between the distributions of categories derived from predictions and those derived from the ground truth. If no significant difference between the distributions is identified, it could mean that the predictive models succeed in predicting traffic events that follow the same distribution as the environment.

One remark related to using this test is that the expected counts of categories should be five or more in at least 80% of the categories, and no category should have an expected frequency of less than one (McHugh, 2012). However, this assumption is most likely to be met if the sample size equals at least the number of cells multiplied by five (McHugh, 2012). This means that each subset of S_2 should receive more than $2 \times 9 \times 5 = 90$ submissions, which, seen from table 3.6 is indeed satisfied.

3.4.4.2 Inter-Rater Agreement

The task in S_2 is highly subjective, and it is not to be expected that all respondents submit identical opinions. Estimating the inter-rater agreement (section **??**), i.e. to what extent the respondents agree upon categories, will indicate which methods that are most reliable for evaluating model performance. The subsets of S_2 do not receive an equal amount of submissions (see table 3.6), hence there is some potential missing data. The Krippendorff's alpha test is a test that computes the inter-rater agreement, corrects for agreement by chance and takes into account potential missing data (Hayes &

Krippendorff, 2007). Krippendorff's alpha values, α, indicate the following:

$\alpha = 1$: Perfect agreement.

$\alpha = 0$: Absence of agreement. Statistically unrelated opinions.

$\alpha < 0$: Disagreement exceeding what can be expected by chance.

Alpha values less than 0.667 signify poor agreement and are considered unacceptable (Krippendorff, 2004, p. 241-243).

3.4.4.3 Realism

The proposed model should produce realistic predictions, therefore, realism is investigated as a property of its own. More specifically, what is being recorded in S_2 is the participants' degree of perceived realism in the videos, with the definition in section 3.4.2 as a reference. Whether or not the models predict realistic videos should be detectable with another statistical test: This data (realism) is recorded on an ordinal scale, and again the distribution is unknown, therefore, another non-parametric test is required. A Wilcoxon-Mann-Whitney test (Neuhäuser, 2011) is therefore used to compare outcomes in realism between ground truth and prediction. This is a non-parametric test used to investigate whether two independent samples are from populations of the same distribution. The test may also be used to investigate whether the participants prefer videos with a specific video type, in terms of how realistically they are perceived.

3.4.4.4 Prediction Accuracy

The predictive models attempt to predict accurate future events of the traffic environment. Therefore, it is natural to investigate their *prediction accuracy*. In order to investigate this property, two approaches are discussed. If the level of inter-rater agreement is high, the opinions provided in S_2 are reliable and may be used as labels for the ground truth videos.

A classification task

If there is significant agreement between raters, it may be appropriate to view S_2 as a classification task with one label for each ground truth video. Since the participants unknowingly categorise both predictions and ground truth videos, the ground truth responses may be used to determine the labels. The categories associated with predictions are matched against their ground truth counterparts to measure the single-label classification accuracy, or rather, the prediction accuracy of the model.

If there is less agreement between the raters, it may be more appropriate to treat S_2 as a multi-label classification task. The reason not all participants who rate a video agree upon a category could be related to the fact that the videos extend over a long period (10 seconds), and that more than one notable categorical event may occur during this duration. In this multi-label classification task, a video may belong to more than one true category. The label(s) of a given ground truth video is chosen in the following way:

Single-label classification: The category with the most votes is the true label assigned to the ground truth video.

Multi-label classification: All categories with the number of votes exceeding one standard deviation of the median are true labels assigned to the ground truth video.

Once the classification scores are derived, a statistical test comparing two proportions of correctly classified videos may detect which, if any, image type that yields more accurate predictions than the other. For this purpose, a two-sided z-test of equality for two proportions (Finkelstein & Levin, 1990) should be appropriate.

Pairwise comparison of categorical distributions

If the inter-rater agreement is low, the models may perform poorly on the classification tasks. Panda et al. (2018) argues that with a consensus approach, two groups of raters can be treated in the same way as if they were only two raters, and then any inter-rater agreement method can be used to

assess agreement between these two groups. The task is then to measure the rate of agreement, or similarity, between two such distributions, across all video pairs in S_2. The Chi-square goodness of fit test could be used for this also, but with suspicion of not fulfilling the assumption about 80% of the expecteds having to be ≤ 5 Mcfarlane et al. (2008) suggest using vector space methods for assessing inter-rater agreement where the data is high-dimensional. One advantage of the classification task is the fact that the model performance is quantified as the proportion of correct classifications, i.e. a value ranging from zero to one. Model evaluation based on measuring similarity between groups of high-dimensional data calls for a metric with such property.

The cosine similarity does exactly this. This measure of similarity treats the categorical distributions as positive vectors in an n-dimensional space, where n is the number of categories. A cosine similarity value of one indicates complete similarity between two positive vectors, meaning the vectors are parallel. A value of zero indicates complete dissimilarity, meaning the vectors are perpendicular. Cosine similarity is defined by the dot product and magnitude of the two vectors, as shown in equation 4.5.1.

$$\cos\theta = \frac{\underline{a} \cdot \underline{b}}{\|\underline{a}\|\|\underline{b}\|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}} \tag{3.7}$$

As with the classification scores, another statistical test may detect which, if any, image type that yields more accurate predictions than the other when comparing categorical distributions. Classification accuracy measures the proportion of *correctly* classified predictions, while cosine similarity gives a measure of *how correct* the prediction is compared to the ground truth in terms of their distributions of categories. Therefore, a Wilcoxon-Mann-Whitney test would be more appropriate than a two-proportion test to detect whether one of the image types is associated with better predictions.

3.4.4.5 Comparing the MMR to a Frame-Wise Comparison

After conducting the above analysis on the submissions to S_2, the proposed evaluation method (the mixed methods research design) should be compared

to the acknowledged evaluation method (the frame-wise comparison). The use of scatter plots allow a visual interpretation of this comparison, and correlation coefficients may indicate the relationship between the evaluation methods. Pearson correlation ("Pearson's Correlation Coefficient", 2008) is a statistic that measures the linear correlation between two variables. For the sake of the comparison, these two variables are the results of the prediction accuracy discussed above and the results of the frame-wise comparison. The Pearson correlation coefficient, r, indicates the strength and direction of the linear relationship between these variables.

Strength

The correlation coefficient ranges from 1 to +1. The closer the absolute value is to 1, the stronger is the relationship between the variables, and an absolute value of 1 signifies a perfect linear relationship. Correlation coefficients close to 0 indicate that there is no linear relationship between the variables.

Direction

The direction of the linear relationship is interpreted from the sign of the correlation coefficient. A positive coefficient means that the variables tend to increase or decrease together, while a negative coefficient indicates that the variables move in opposite directions.

If the two evaluation methods measure similar properties of the video predictions, we would expect a strong, positive relationship between them.

3.4.5 Refining the Evaluation Method

I have proposed a method for evaluating video predictions and predictive deep learning models. This involves designing, conducting and analysing the mixed methods research design for evaluation of predictive models. The final result is an improved and updated evaluation method based on the experiences gained during the process. I present the updated evaluation

method in the format of a protocol, which is my main contribution to the field of visual prediction with deep learning.

Chapter 4

Results and Discussion

The following chapter reviews the results of the work in this thesis, revealing the performance of the proposed models, and the utility of the mixed methods research design. Throughout the chapter, the research questions from section 1.2 are discussed, which concern evaluating predictive models by measuring the *realism* and *accuracy* of predictions they produce. Also, I investigate the importance of *image type* for representing the visual environment in video prediction tasks. Below is a summary of the predictive models, the methods used to evaluate model performance, and the data from which the results are derived.

- The predictive models used in all experiments are described in section 3.3.
- Evaluation of model performance is assessed using two different approaches; a prevalent approach which is a quantitative frame-wise comparison between video predictions and ground truth videos, and a candidate approach which is our proposed mixed methods research design (section 3.4).
- The data (videos) used to derive the following results are from the test set described in section 3.4.2. Note that all videos are processed by the visual components (section 3.3.1) such that a frame-wise comparison

is reasonable, and ground truth and predicted videos appear visually similar so as not to create a bias in the mixed methods research.

A qualitative review of some video prediction samples gives an intuition of what the models have learnt. A quantitative frame-wise comparison between video predictions and ground truth videos is performed, before finally reviewing the results based on the proposed evaluation method comprised of the mixed methods research design.

4.1 A Qualitative Review of Video Prediction Samples

To help fully understand the results presented in this chapter, samples from the data described above are inspected. A *sample* consists of the condition used to generate a video prediction, the video prediction itself and its corresponding ground truth, as shown in figure 4.1. The condition is comprised of ten input frames, which the model uses as context to predict a future trajectory. Prediction of future states is made purely in the latent space by the memory component (section 3.3.2). The predicted latent variables are then decoded by the visual component to create the sequence of visual states, i.e. the video prediction.

Figure 4.1 shows four samples, commented from top to bottom:

1. (RGB) The model succeeds in predicting some of the oncoming traffic in the left lane, in addition to the right turn towards the end of the sequence.

2. (RGB) The model predicts the agent approaching a vehicle and a full stop due to intersecting traffic. The ground truth video shows deceleration but no full stop.

3. (SEG) The model predicts a right turn followed by traffic clog, while the ground truth shows a left turn followed by traffic clog.

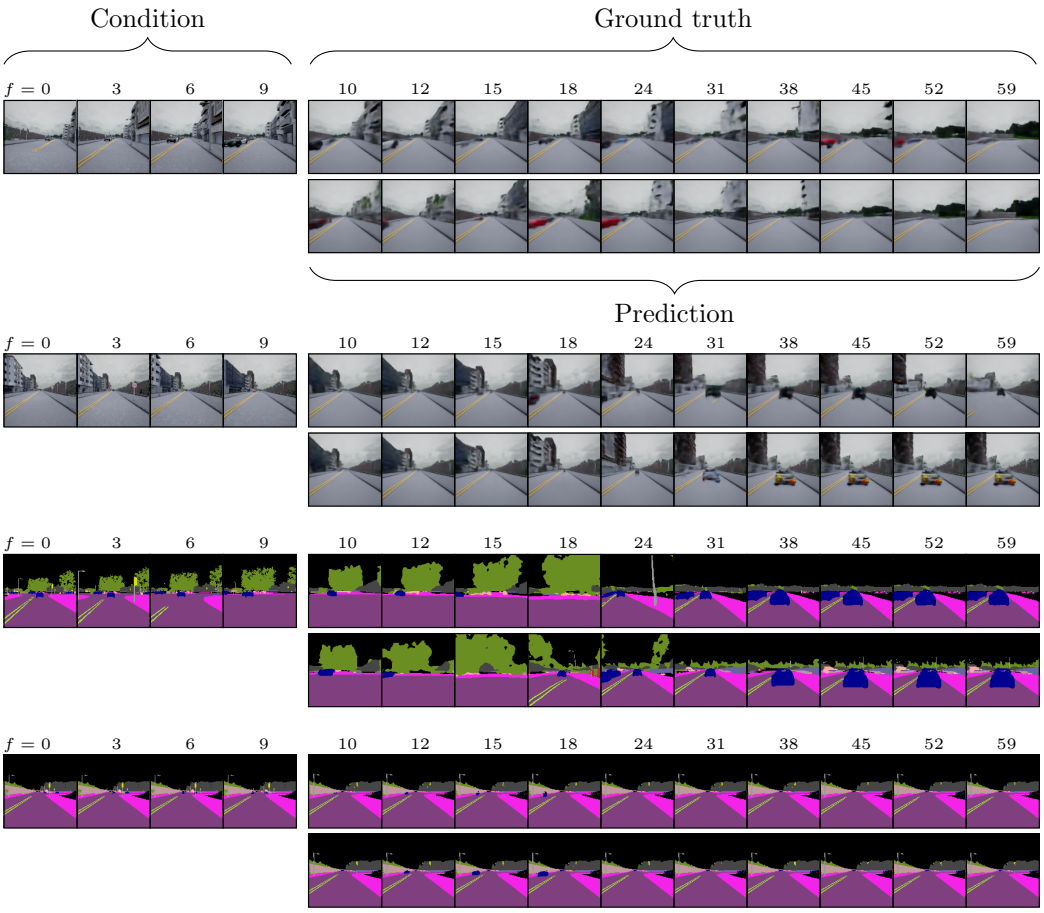


Figure 4.1: Examples of two RGB *samples* (top) and two semantic segmentation *samples* (bottom). The top row contains a sequence of ground truth images from the validation set, where the initial 10 frames are the condition used by the model to predict future states. The bottom row contains the corresponding predicted sequence generated by the model. The frame number in a sample is denoted by f. Recall that the ground truth ($f = 10 - 59$) are also encoded and decoded by the model's visual component (section 3.3.1).

4. (SEG) The model predicts a standstill by an intersection, while one vehicle passes in the left lane. The ground truth shows the same, except that the passing vehicle is a motorcycle.

Looking closer at sample 2, it can be seen that the cars in frames $f = 31$ are quite different. The car in the ground truth image is black, whereas in the predicted image it is mostly yellow and red. Something similar is observed in sample 4, where the passing vehicle in the ground truth sequence

is a motorcycle, while it in the prediction is a car. This characteristic is interesting because from one perspective the objects are the same, namely cars. However, comparing the images with a similarity metric such as mean square error might indicate little resemblance between the vehicles due to their difference in pixel intensity.

This observation raises a relevant question; with what detail should a model capture the environment to meaningfully predict its future? For example, is predicting the specific type or colour of a vehicle more useful than predicting the fact that it is a vehicle? The next section examines to what extent measuring pixel-level similarity between prediction and ground truth can be used to evaluate the performance of the models.

Another property of the models worth noting, seen in figure 4.1, is their ability to maintain the structure of objects over many predicted frames. This property is not achieved in related research such as (Luc et al., 2017), whose methods struggle to generate accurate long-term predictions of complex environments (see figure 2.16). The reason the proposed models are able to maintain object structure is believed to be due to the use of autoencoders, or more specifically VQ-VAEs in the visual components. The latent representations produced by the visual components enable the models to predict precise future states and create meaningful image reconstructions, even over many consecutive time steps. This property supports the assumptions (section 3.1) when it comes to designing a suitable model architecture for this prediction task.

More of these samples may be found in video format at `https://plaffa.github.io/samples.html`. Having gained some knowledge about what the models have learnt, it is time to investigate how good these predictions are overall, by using the first evaluation approach, which is the frame-wise comparison.

4.2 Model Evaluation Approach 1: A Quantitative Frame-Wise Comparison

This section reviews the frame-wise comparison between ground truth videos and video predictions. As mentioned in section 3, related research within the field of video prediction suggests this approach to be the acknowledged way of evaluating predictive models. Taking inspiration from Luc et al. (2017), the video prediction task is divided into three scopes; short-term, mid-term and long-term prediction.

Short-term prediction is here defined to involve predicting the next 5 frames of a sequence, mid-term to be the next 20 frames, and long-term to be the next 50 frames. Frame-wise comparison with four different similarity metrics is then applied to evaluate the video predictions, at all three scopes. The similarity metrics quantify the resemblance between consecutive pairs of prediction and ground truth frames. Tables 4.1, 4.2 and 4.3 report average structural similarity (SSIM), peak signal-to-noise ratio (PSNR), mean square error (MSE) and intersection over union (IoU) for short-term, mid-term and long-term prediction respectively, computed using built-in functions of MATLAB R2018b. Average similarity scores are based on the similarity between all pairwise frames throughout a scope's total sequence length. In addition, IoU for moving objects (IoU-MO) is included (Luc et al., 2017). Moving objects in the test set are pedestrians and vehicles. Note that IoU and IoU-MO apply only for semantic segmentation images, so they are not reported for RGB.

In addition to reporting results for the predictive models ($\text{model}_{\text{RGB}}$ and $\text{model}_{\text{SEG}}$), a naive baseline model is included. This baseline model is a repeated copy of the last conditional input frame, and gives a basis for comparison of the results. Tables 4.1, 4.2 and 4.3 report image similarity between prediction and ground truth for the four models, where bold values indicate the best scores.

As can be seen from tables 4.1, 4.2 and 4.3, both $\text{model}_{\text{RGB}}$ and $\text{model}_{\text{SEG}}$ are superior to their baselines at all prediction scopes, and all average similarity scores. This indicates that the models have at least learnt to predict a

Short-term prediction					
Method	SSIM	PSNR	MSE	IoU	IoU-MO
$\text{Baseline}_{\text{RGB}}$	0.79	21.56	797.4	-	-
$\text{Baseline}_{\text{SEG}}$	0.82	(18.49)	(1157.9)	0.82	0.25
$\text{Model}_{\text{RGB}}$	0.83	**23.25**	**605.4**	-	-
$\text{Model}_{\text{SEG}}$	**0.86**	(19.99)	(854.6)	**0.86**	**0.32**

Table 4.1: Average scores based on frame-wise comparison for short-term predictions (next 5 frames). SSIM, PSNR and IoU scores indicate quality (higher is better). MSE is a measure of error (lower is better).

Mid-term prediction					
Method	SSIM	PSNR	MSE	IoU	IoU-MO
$\text{Baseline}_{\text{RGB}}$	0.78	20.73	989.7	-	-
$\text{Baseline}_{\text{SEG}}$	0.81	(18.12)	(1288.3)	0.79	0.22
$\text{Model}_{\text{RGB}}$	0.82	**22.95**	**709.6**	-	-
$\text{Model}_{\text{SEG}}$	**0.85**	(19.80)	(899.3)	**0.85**	**0.27**

Table 4.2: Average scores based on frame-wise comparison for mid-term predictions (next 20 frames). SSIM, PSNR and IoU scores indicate quality (higher is better). MSE is a measure of error (lower is better).

Long-term prediction					
Model	SSIM	PSNR	MSE	IoU	IoU-MO
$\text{Baseline}_{\text{RGB}}$	0.75	19.95	1187.8	-	-
$\text{Baseline}_{\text{SEG}}$	0.78	(17.67)	(1499.9)	0.76	0.20
$\text{Model}_{\text{RGB}}$	0.79	**21.87**	**888.5**	-	-
$\text{Model}_{\text{SEG}}$	**0.83**	(18.92)	(1159.7)	**0.82**	**0.23**

Table 4.3: Average scores based on frame-wise comparison for long-term predictions (next 50 frames). SSIM, PSNR and IoU scores indicate quality (higher is better). MSE is a measure of error (lower is better).

more accurate future of the environment than simply copying the last conditional frame. As will be explained below, some values are given in parentheses

because application of these metrics on semantically segmented (SEG) images is somewhat questionable. Summary of each of the similarity scores in tables 4.1, 4.2 and 4.3:

SSIM: $\text{Model}_{\text{SEG}}$ outperforms $\text{model}_{\text{RGB}}$ and the baselines.

PSNR: $\text{Model}_{\text{RGB}}$ outperforms $\text{model}_{\text{SEG}}$ and the baselines.

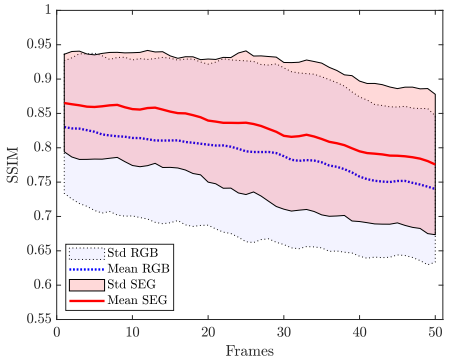
MSE: $\text{Model}_{\text{RGB}}$ outperforms $\text{model}_{\text{SEG}}$ and the baselines.

IoU: $\text{Model}_{\text{SEG}}$ outperforms $\text{baseline}_{\text{SEG}}$, also for IoU-MO.

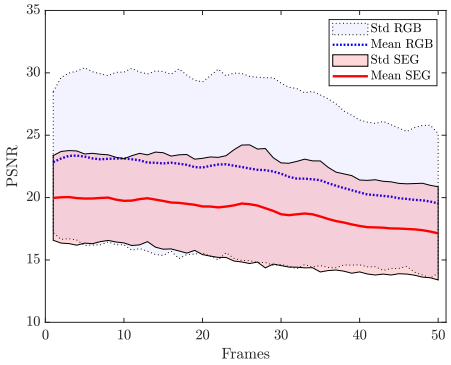
Figures 4.2 and 4.3 show scores from the frame-wise comparisons with respect to all 50 frames in long-term predictions. The curve represents the average score, and the regions surrounding them show the standard deviation. Figure 4.2a (SSIM) indicates that the degree of average structural similarity between prediction and ground truth decreases with time, at a similar rate for both models. However, $\text{model}_{\text{SEG}}$ generally achieves a higher SSIM score than $\text{model}_{\text{RGB}}$, possibly due to the homogeneity of object regions in semantic segmentation images.

A similar tendency is seen in figure 4.2b (PSNR), though the rate of descent is less prominent, and this time $\text{model}_{\text{RGB}}$ receives a higher overall score. The reason $\text{model}_{\text{RGB}}$ receives higher PSNR scores than $\text{model}_{\text{SEG}}$ is unsurprising and related to figure 4.2c (MSE). Calculation of PSNR makes use of MSE (see section 2.7), such that low MSE values consequently correspond with high PSNR values. While RGB images consist of small transitions in pixel intensity, semantically segmented images contain distinct colours in separate class regions (see figure 4.1 for intuition). Class regions that fail to overlap across two semantic segmentation images will yield a larger impact than that of two RGB images. In other words, the use of MSE and thus PSNR is targeted towards RGB images and comparing the same metrics applied to other data types such as semantic segmentation is not equitable.

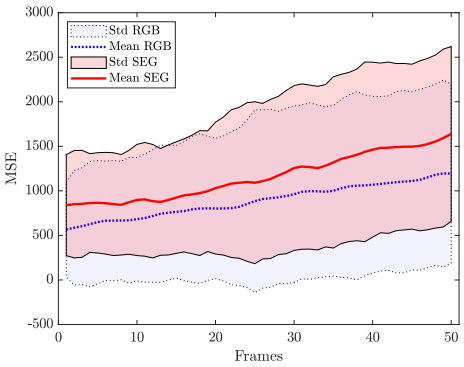
Figure 4.3 shows the two similarity metrics IoU and IoU-MO for semantic segmentation sequences. IoU-MO scores (moving classes; vehicles and pedestrians) are significantly lower than IoU computed across all classes. Because



(a) Structural similarity (SSIM). Higher value is better.



(b) Peak signal-to-noise ratio (PSNR). Higher value is better.



(c) Mean squared error (MSE). Lower value is better.

Figure 4.2: Frame-wise comparison between ground truth and video predictions. RGB and semantic segmentation model. The curves represent the similarity measures averaged across the 52 videos. The coloured areas represent the standard deviations from the mean curve.

a scene is dominated by large, static objects, such as roads and the sky, small, dynamic objects, such as vehicles and pedestrians, contribute less to the similarity score. Therefore, IoU, as well as MSE, PSNR and SSIM, may indicate a good prediction, simply because large, static objects are easier to predict.

IoU-MO more accurately describes the model's ability to predict dynamic objects. However, there are two challenges with IoU-MO. Firstly, IoU-MO is only applicable to semantic segmentation images, not RGB images. Secondly, IoU-MO scores merely measure to what extent dynamic objects in prediction

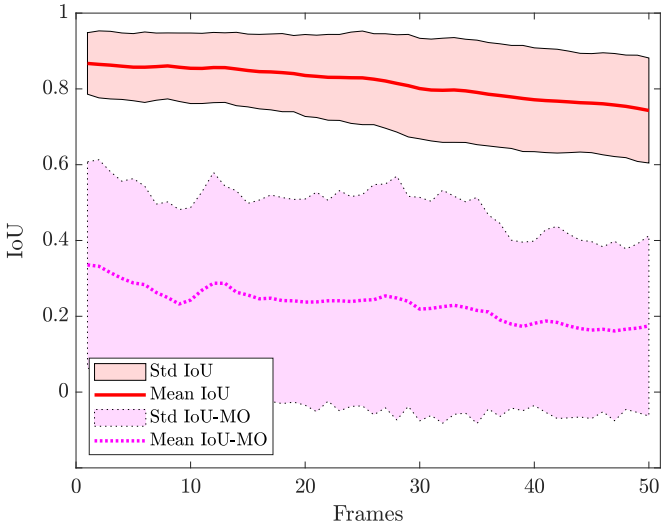


Figure 4.3: Intersection over union for all objects (IoU) and moving objects (IoU-MO) for semantic segmentation sequences. Higher value is better.

and ground truth overlap. A condition (10 frames) does not contain sufficient information to accurately predict the future on the pixel-level. Therefore, one should not require a prediction to *overlap* with the ground truth for it to be *meaningful*.

Following the reasoning above, none of the four metrics is ideal to evaluate a model's ability to predict the future. Still, SSIM is perhaps the most reliable metric, because it is less sensitive to pixel-level mismatch compared to MSE and PSNR, and it is applicable to both image types. Besides, SSIM (and IoU) scores are easier to interpret than MSE and PSNR, because they yield values in the range $[0, 1] \subset \mathbb{R}$, i.e. percentage similarity.

A final overview of average SSIM is shown in figure 4.4, which illustrates the performance of both predictive models compared to their baselines. The regions of greatest distinction between the predictive models and their baselines are the first 10 frames in a sequence, where both baselines show great decline in structural similarity with ground truth.

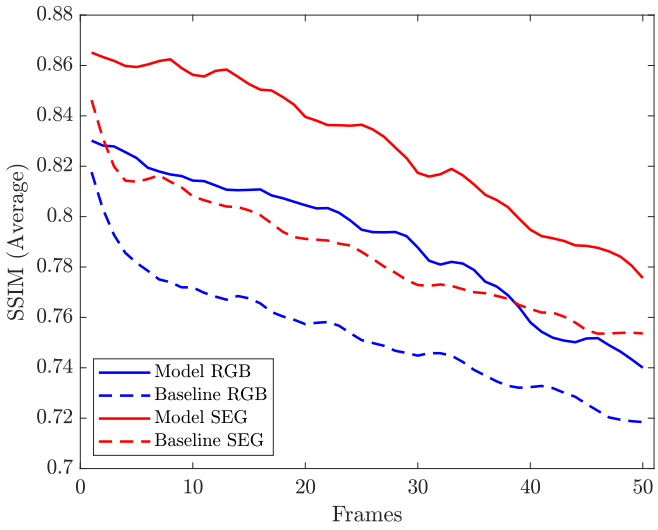


Figure 4.4: Average SSIM.

4.3 Model Evaluation Approach 2: The Mixed Methods Research Design

The mixed methods research design involves studying responses from two surveys; a qualitative survey, S_1, and a quantitative survey, S_2. Conducting the qualitative survey establishes areas of focus related to the prediction task in this thesis, formulated as a set of traffic events (section 3.4.1). These events are categorical choices used by participants in the quantitative survey, S_2 (section 3.4.2). Full details on the implementation and outcome of the mixed methods design is found in section 3.4. Evaluation of the predictive model is based on the responses to S_2 by using methods described in section 3.4.4.

4.3.1 Preliminary Analysis

The analysis of the mixed methods research begins by identifying possible outliers among the responses to S_2. The method used to detect outliers is based on median absolute deviation as described in section 3.4.3, with parameters $k = 1.0$ and $T = 2.0$. The amount of time a participant spends

on a subset of S_2 should not exceed one standard deviation from the mean, which across the four subsets equals a valid time interval between 5.6 and 20.7 minutes. Any submission to S_2 violating both these criteria is considered an outlier. Among the 453 responses to S_2, 24 (5.3% of the population) were identified as outliers and removed from the set. Table 4.4 shows the distribution among categories across the four survey subsets, produced by the remaining 429 responses.

Distribution among selected categories										
Video type	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	Total
RGB_{GT}	2053	265	426	691	420	434	406	543	167	5404
RGB_{PD}	2006	351	515	514	306	394	385	645	173	5289
SEG_{GT}	1678	278	456	762	522	447	404	593	217	5357
SEG_{PD}	1773	403	668	691	306	390	372	612	158	5373
Total	7510	1297	2065	2658	1554	1665	1567	2393	715	-
Proportion	35.05%	6.05%	9.64%	12.41%	7.25%	7.77%	7.31%	11.17%	3.34%	100%

Table 4.4: The distribution among the 9 categories related to ground truth video and video prediction for both image types, RGB and SEG (semantic segmentation).

The first step is to investigate whether the distribution among the video predictions' categories is similar to that of the ground truth categories. The following null and alternative hypothesis are:

$\boldsymbol{H_0}$: There is no significant difference between the distribution of categories for video predictions (observed) and ground truth video (expected), with respect to the 52 videos.

$\boldsymbol{H_A}$: There is a significant difference between these distributions

The Chi-square goodness of fit test with an alpha level of 0.05 and 8 degrees of freedom is used on the data in table 4.4, computed using the statistical software R (R Core Team, 2018). With sample sizes of ∼5300 the test has a statistical power of 0.99, i.e. a 99% chance of detecting an effect if it exists. Table 4.5 summarises the Chi-square goodness of fit test. Since both p-values are lower than the significance level of 0.05, the null hypotheses are rejected with 95% confidence. This concludes that both $\text{model}_{\text{RGB}}$ and $\text{model}_{\text{SEG}}$ generate video predictions with distributions among event categories that are different from that of the ground truth videos. This result might suggest

Chi-square goodness of fit test				
Model	alpha	df	p-value	Conclusion
$\text{Model}_{\text{RGB}}$	0.05	8	$< 2.2e-16$	Reject H_0
$\text{Model}_{\text{SEG}}$	0.05	8	$< 2.2e-16$	Reject H_0

Table 4.5: Specifications and results related to the Chi-square goodness of fit performed on the data in table 4.4.

that the models have learnt some general understanding of the environment, rather than to copy the data perfectly, as suggested by the visual inspection of samples (section 4.1). However, the result is only intermediate, because the overall distributions have merely been determined unequal, whereas the level of dissimilarity has not been determined. Sections 4.3.2 and 4.3.3 investigate the latter, but first the realism of video predictions produced by the models must be assessed, i.e. the participants' degree of perceived realism among the videos in part two of the survey, with the definition of realism from section 3.4 in mind. Realism related to the 52 videos in a subset of S_2 is recorded on a five-level ordinal scale, with an overall distribution across all subsets shown in table 4.6.

Distribution among the degree of perceived realism					
Video type	Unrealistic	Somewhat unrealistic	Neither	Somewhat realistic	Realistic
RGB_{GT}	3.03%	7.58%	13.51%	39.78%	36.10%
RGB_{PD}	4.47%	7.83%	13.37%	38.52%	35.81%
SEG_{GT}	13.00%	12.67%	13.53%	34.73%	26.08%
SEG_{PD}	13.36%	12.54%	13.03%	33.67%	27.39%
Overall	8.49%	10.16%	13.36%	36.68%	31.35%

Table 4.6: The participants' degree of perceived realism among the videos in part two of the survey, divided into the four video types.

Table 4.6 shows that the participants mainly interpret all video types as *somewhat realistic* or *realistic*. By treating the ordinal scale as numbers within the range 1-5, it is clear that the participants favour RGB videos over segmented videos in terms of how realistic they are perceived. As can be seen in table 4.7, RGB (ground truth and predictions) videos are generally perceived as more realistic than segmented videos. This may be due to the

nature of RGB images falling more natural to people, as opposed to semantic segmentation which is a representation of the world people rarely relate to.

General tendency of the degree of perceived realism		
Video type	Median	Average
RGB_{GT}	Somewhat realistic (4)	Somewhat realistic (3.99)
RGB_{PD}	Somewhat realistic (4)	Somewhat realistic (3.94)
SEG_{GT}	Somewhat realistic (4)	Neither / Somewhat realistic (3.49)
SEG_{PD}	Somewhat realistic (4)	Neither / Somewhat realistic (3.50)

Table 4.7: The general tendency of the degree of perceived realism when treating the ordinal scale as numbers in the range 1-5.

Wilcoxon-Mann-Whitney tests with significance levels of 0.05 are used to compare outcomes in realism between ground truth and prediction for RGB and segmented videos. The tests reveal that for both image types, no significant difference between the levels of realism in ground truth and predictions can be identified. This result may suggest that video predictions produced by both $\text{model}_{\text{RGB}}$ and $\text{model}_{\text{SEG}}$ are perceived just as realistic as the ground truth videos.

4.3.2 Video Classification

It has been established that both $\text{model}_{\text{RGB}}$ and $\text{model}_{\text{SEG}}$ generate video predictions with distributions among event categories unequal to that of environment. While this result is somewhat explanatory in terms of model performance, it has not yet been determined *how accurate* the predictions are. Therefore, the evaluation objective is treated as two separate video classification tasks, as described in section 3.4.4.4. With the submissions to S_2, there are usually one or two true labels in the multi-label classification task, on rare occasions three. The classification scores are summarised in table 4.8.

At first thought, the classification scores in table 4.8 are not remarkably impressive. For both models, the average accuracy is quite low, and the standard deviation is quite high. However, because long-term futures are uncertain, it is difficult to determine whether these classification scores are

Video classification						
Model	Accuracy: Single-label			Accuracy: Multi-label		
	Average	Median	Std.	Average	Median	Std.
$\text{Model}_{\text{RGB}}$	32.8%	32.1%	20.2%	48.0%	53.8%	20.60%
$\text{Model}_{\text{SEG}}$	35.8%	34.8%	21.2%	47.6%	54.1%	19.9%

Table 4.8: Classification scores for $\text{model}_{\text{RGB}}$ and $\text{model}_{\text{SEG}}$ with the single-label and multi-label classification tasks, reported accuracy in terms of average, median and standard deviation.

good or bad. Nevertheless, a way of deriving a concrete measure of model performance according to a defined objective has been determined. This is not tailored to a specific type of data but may instead be used with both RGB images and semantically segmented images.

Which model is better at predicting correct events?

The next step is to examine whether there is a significant difference in classification performance between the two models (RGB and SEG), using a two-proportion test: A two-sided z-test of equality for two proportions, without continuity correction and a significance (alpha) level of 0.05, computed using the statistical software R. To investigate whether the proportions of correctly classified video predictions are equal for both models, the following null and alternative hypotheses is formulated:

$\boldsymbol{H_0}$: The proportion of correctly classified video predictions with $\text{model}_{\text{RGB}}$ is equal to the proportion of correctly classified video predictions with $\text{model}_{\text{SEG}}$

$\boldsymbol{H_A}$: The two proportions are unequal

With sample sizes of over 5300 (see table 4.4) the test has a statistical power of 0.99, i.e. a 99% chance of detecting an effect if it exists. Figure 4.5 shows box plots of the classification results, including p-values from the two-proportion tests. The p-value from the single-label proportion test

is $1.401e{-}4$, i.e. less than the significance level of 0.05, meaning that the null hypothesis may be rejected and concluding with 95% confidence that $\text{model}_{\text{SEG}}$ is superior to $\text{model}_{\text{RGB}}$ at the single-label video classification task. The p-value from the two proportions test for multi-label classification is 0.1995, i.e. higher than the significance level, and the null hypothesis may not be rejected. This means that for the multi-label classification task, no significant difference in accuracy proportion between $\text{model}_{\text{RGB}}$ and $\text{model}_{\text{SEG}}$ is identified.

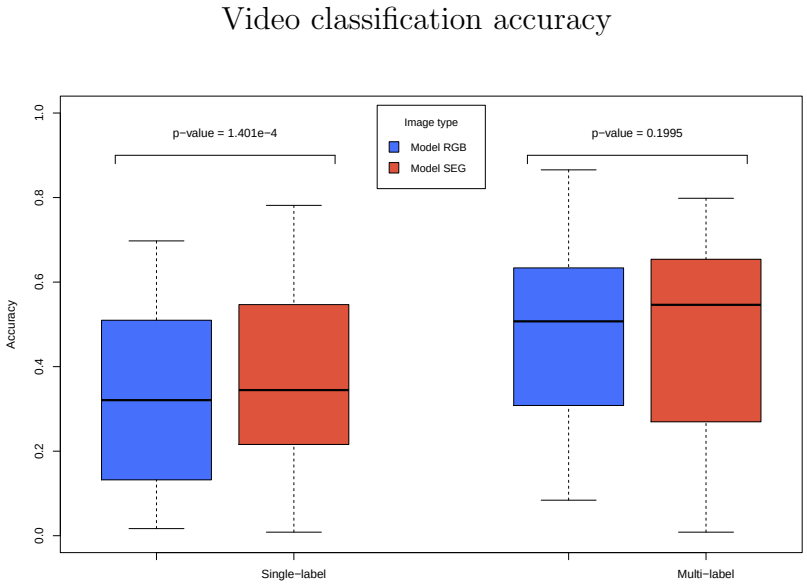


Figure 4.5: Box plots showing the classification results from the single-label and multi-label video classification tasks.

Is there a biased category?

When studying table 4.4, category c_1 ('driving straight') is chosen much more frequently than any other category, namely ∼35% of the time. The reason could be related to the position of this particular categorical choice within the survey compared to the other categories (see the link provided in section 3.4 for the survey layout), or because much of the events in the videos incidentally involves driving straight forward. Also, category c_9 ('un-

defined') is chosen very seldom, only ∼3% of the time, possibly because it is unnecessary. Including these somewhat biased categories, especially c_1, may lead to unreliable results, the video classification scores were computed again but now without categories c_1 and c_9. Note that removing the two categories also means losing a number of category votes, which in turn may result in more uncertain classification scores. $\text{Model}_{\text{RGB}}$ now gets an average classification accuracy of 29.6% at the single-label task, and 36.6% at the multi-label task. $\text{Model}_{\text{SEG}}$ maintains a higher single-label classification accuracy than $\text{model}_{\text{RGB}}$, with an accuracy of 34.0%. At the multi-label classification task, $\text{model}_{\text{SEG}}$ scores 37.7%. Performing two new two-proportion tests yields p-values of $1.547e{-}4$ and 0.016 for the single-label and multi-label tasks, respectively. In other words, $\text{model}_{\text{SEG}}$ remains superior to $\text{model}_{\text{RGB}}$ at the single-label video classification task. Still, no significant difference in classification performance between the models can be identified at the multi-task video classification task. In conclusion, though categories c_1 and c_9 are identified as high and low frequent categorical choices, removing them makes no noticeable difference in terms of classification accuracy.

Explaining accuracy with inter-rater agreement

The considerable increase in classification accuracy using multiple labels does suggests that the events in the videos may be better described using multiple categories rather than a single category. Initially, somewhat poor classification scores were observed in general, and it is therefore necessary to investigate why this is the case. An examination of the extent to which participants agree on the video's categories was conducted by calculating the inter-rater agreement of responses to S_2 in SPSS (IBM Corp., 2019). The Krippendorff's alpha test (section 3.4.4.2) is here used to calculate the inter-rater agreement, and these alpha (α) are reported in table 4.9 below. For convenience, the total set of videos is divided into four groups, RGB_{GT}, RGB_{PD}, SEG_{GT} and SEG_{PD}, and quantify the degree of agreement between participants with respect to each group.

The reliability coefficients (α) from the Krippendorff's alpha test help to

Inter-rater agreement on choice of category					
Agreement	RGB_{GT}	RGB_{PD}	SEG_{GT}	SEG_{GT}	Average
Total number of ratings	5405	5289	5357	5373	(5356)
Krippendorff's alpha (α)	0.122	0.131	0.106	0.164	(0.131)
Degree of agreement	poor	poor	poor	poor	(poor)

Table 4.9: Inter-rater agreement using Krippendorff's alpha.

understand why the classification accuracies are somewhat low. Alpha values less than 0.667 signify poor agreement (section 3.4.4.2), thus, as shown in table 4.9, the agreement between participants is poor within all groups of videos. This might create uncertainty related to the use of ground truth labels to describe the *true scenario* of a video. At this point, it is necessary to question the survey layout in terms of the tools the participants have at their disposal to describe videos in the survey. Mcfarlane et al. (2008) suggest that inter-rater agreement may be low if the number of possible categories to choose from is large. Therefore, owing to the fact that participants in S_2 describe a ten-second long video using only one of nine possible categories, broad agreement between raters of video predictions and raters of their ground truth counterparts cannot be expected. Nevertheless, the collected data based on ten-second long videos and nine categorical choices can be used in an alternative way to evaluate model performance if the agreement between individual raters is low.

4.3.3 Pairwise Comparison of Categorical Distributions

A video's event content is better described using up to several categories, as opposed to using a single category. Therefore, an alternative evaluation objective that is more appropriate than the video classification task at measuring the categorical similarity between two videos. As argued in section 3.4.4.4, a consensus approach enables two groups of raters to be treated in the same way as if they were only two raters. Table 4.10 shows how videos

are described by categorical distributions following this logic.

Categorical distributions for an arbitrary sample										
Video type	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	Total
RGB_{GT}	33.1%	2.4%	4.0%	41.9%	3.2%	4.8%	8.9%	0.8%	0.8%	100.0%
RGB_{PD}	20.0%	3.2%	3.2%	50.5%	4.2%	7.4%	5.3%	3.2%	3.2%	100.0%

Table 4.10: Category distribution for an arbitrary sample (one video pair) shown as an example. The proportions are rounded to the nearest decimal.

Agreement between categorical distributions is here quantified using cosine similarity. As with the classification task, the cosine similarity is measured after removing categories c_1 and c_9 due to the suspicion of c_1 inducing a bias and c_9 being rarely used. Eliminating these categories adjusts the categorical distributions accordingly.

Inter-group agreement: Cosine similarity between categorical distributions						
Model	$\cos\theta$ with all categories			$\cos\theta$ with $\{c_2, c_3, c_4, c_5, c_6, c_7, c_8\}$		
	Average	Median	Std.	Average	Median	Std.
$\text{Model}_{\text{RGB}}$	0.7988	0.8649	0.1969	0.7630	0.8012	0.2040
$\text{Model}_{\text{SEG}}$	0.7992	0.8660	0.2023	0.7730	0.8601	0.2053

Table 4.11: Inter-group agreement assessed as cosine similarity between categorical distributions for all video pairs. Average, median and standard deviation of similarity between distributions are reported using all categories and a subset of categories.

Table 4.11 summarises the degree of similarity between categorical distributions of video predictions and ground truth for all 52 videos. By visual inspection, it seems $\text{model}_{\text{RGB}}$ and $\text{model}_{\text{SEG}}$ perform equally well, perhaps except for $\text{model}_{\text{SEG}}$ having a somewhat larger median similarity value when ignoring categories c_1 and c_2. To test whether categorical similarity with one model tends to be larger than with the other, the following null and alternative hypotheses are formulated to be tested with Wilcoxon-Mann-Whitney tests:

$\boldsymbol{H_0}$: The distributions of categorical similarity scores of $\text{model}_{\text{RGB}}$ and $\text{model}_{\text{SEG}}$ are equal

$\boldsymbol{H_A}$: The distributions of scores of the two models are unequal

Note that in the transition from category votes by groups to categorical distributions, the sample size is reduced to the number of videos, i.e. 52. The Wilcoxon-Mann-Whitney tests give a p-value of 0.8561 when including all categories, and 0.6996 with the subset of categories, as shown with box plots in figure 4.6. With significance levels of 0.05, the null hypotheses may not be rejected, meaning there is no real evidence that one model performs better than the other. In essence, comparing categorical distributions with cosine similarity arrives at the same conclusion as the multi-label video classification does. Using the proposed model architecture, and if the content of a video is described by multiple categories or by a categorical distribution, representing the visual environment with either RGB images or semantic segmentation makes no significant difference in terms of how accurate the generated video predictions are.

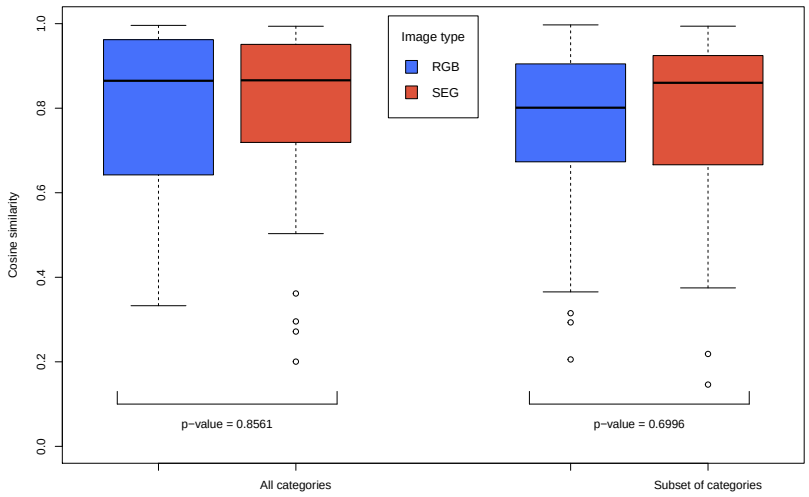


Figure 4.6: Box plots showing the cosine similarity for categorical distributions with all categories and a subset of categories.

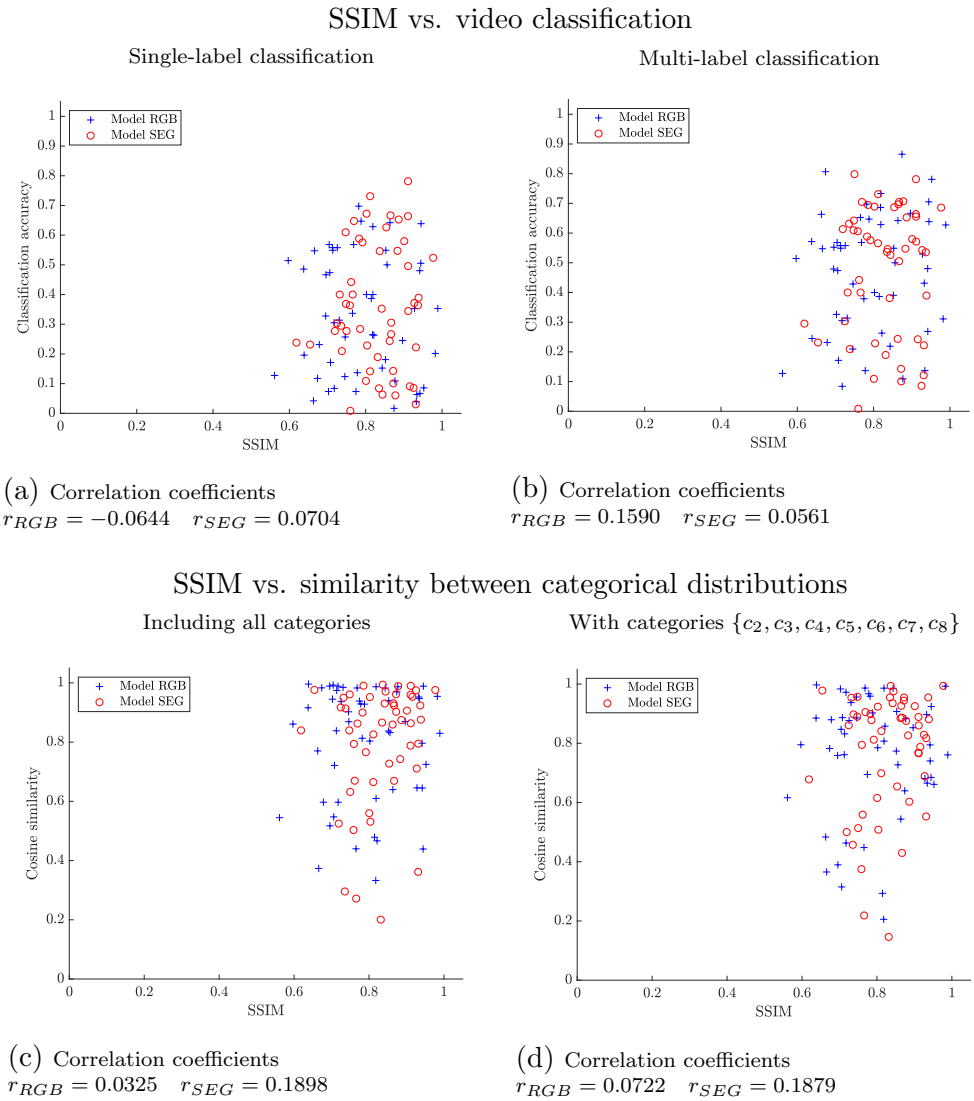
4.4 Comparing the Two Evaluation Approaches

The results based on the two approaches for assessing model performance have been presented. Measuring the correlation between approach 1 (the frame-wise comparison) and approach 2 (the proposed mixed methods research design) will reveal to what extent the approaches measure similar properties of the predictive models. As argued in the end of section 4.2, SSIM is the most appropriate metric for frame-wise comparison across image types. Therefore, only SSIM is used in this final comparison, excluding MSE, PSNR and IoU. Figure 4.7 shows the relation between results using approach 1 and approach 2 when evaluating each of the 52 video predictions. Coefficients r describe the Pearson correlation between these outcomes (section 3.4.4.5).

In all four plots shown in figure 4.7, the linear relationship between the two model evaluation approaches is either nonexistent or very weak (section 3.4.4.5). These weak relationships are interpreted from the correlation coefficients, given in greater detail below. The null hypothesis is that there is no statistically significant relationship between the variables, i.e. $r = 0$, which is tested with significance levels (α) of 0.05.

$$\{\text{SLC vs. SSIM}\}_{RGB} : r(50) = .06, p = .672 \rightarrow \text{not significant}$$
$$\{\text{MLC vs. SSIM}\}_{RGB} : r(50) = .16, p = .257 \rightarrow \text{not significant}$$
$$\{\cos\theta_{\text{all}} \text{ vs. SSIM}\}_{RGB} : r(50) = .03, p = .833 \rightarrow \text{not significant}$$
$$\{\cos\theta_{\text{sub}} \text{ vs. SSIM}\}_{RGB} : r(50) = .07, p = .621 \rightarrow \text{not significant}$$
$$\{\text{SLC vs. SSIM}\}_{SEG} : r(50) = .07, p = .621 \rightarrow \text{not significant}$$
$$\{\text{MLC vs. SSIM}\}_{SEG} : r(50) = .06, p = .672 \rightarrow \text{not significant}$$
$$\{\cos\theta_{\text{all}} \text{ vs. SSIM}\}_{SEG} : r(50) = .19, p = .177 \rightarrow \text{not significant}$$
$$\{\cos\theta_{\text{sub}} \text{ vs. SSIM}\}_{SEG} : r(50) = .19, p = .182 \rightarrow \text{not significant}$$

where $r(df)$ is the correlation with $df = N - 2$ degrees of freedom, and $N = 52$, i.e. the total amount of samples per image type in the test set.



(a) Correlation coefficients
$r_{RGB} = -0.0644 \quad r_{SEG} = 0.0704$

(b) Correlation coefficients
$r_{RGB} = 0.1590 \quad r_{SEG} = 0.0561$

(c) Correlation coefficients
$r_{RGB} = 0.0325 \quad r_{SEG} = 0.1898$

(d) Correlation coefficients
$r_{RGB} = 0.0722 \quad r_{SEG} = 0.1879$

Figure 4.7: Scatter plots showing the relationship between SSIM and cosine similarity for all 52 videos in survey part two. The correlation between the evaluation methods outcomes is given for each image type by $r_i, i \in [RGB, SEG]$.

SLC, MLC and $\cos\theta_c$ are the single-label and multi-label classification scores and cosine similarity, where c may include all categories, *all*, or the subset of categories, *sub*.

The null hypothesis may not be rejected in any of the cases, which means that no statistically significant linear relationship is detected between any of the variables, and correlation coefficients $r \neq 0$ cannot be expected with the

current sample sizes.

In the previous section, 4.3.3, it was shown that the proposed evaluation method's two ways of deriving *prediction accuracy* agree quite well, especially when comparing cosine similarity with multi-label video classification. What is clear, however, is that the prediction accuracy does not agree with the frame-wise comparison. For one thing, all four scatter plots suggest that the SSIM scores are biased toward values > 0.6, meaning that SSIM values within a rather small range of operation $[0.6, 1] \subset \mathbb{R}$ correspond to video predictions of quality or accuracy somewhere between 'poor' and 'good'. On the other hand, the prediction accuracy based on the proposed method (classification accuracy and cosine similarity) makes use of the entire scale $[0, 1] \subset \mathbb{R}$.

The two evaluation approaches are supposedly *uncorrelated*, which in practice means that the frame-wise comparison evaluates video predictions differently to how humans perceive them. This finding emphasises the shortcomings related to using a frame-wise comparison for judging the quality of videos, which according to Moorthy et al. (2011) is a task with a highly subjective nature, 'coupled with the human visual system's peculiarities'. Also, it supports the argument that there is indeed a need for developing new ways of evaluating predictive deep learning models to assess how meaningful their predictions are.

4.5 Refining the Proposed Evaluation Method

The results in this chapter and the considerations below lead to the development of an improved version of the evaluation method. The main contribution of this thesis is presented; a refined protocol for evaluating predictive models using subjective data, shown in figure 4.8. The term *predictive models* refers mainly to deep learning models that predict future visual states of environments. It is believed that the protocol can be applicable for other and similar models and prediction tasks.

4.5.1 Considerations

A number of challenges and shortcomings were encountered during the process of developing and testing the proposed evaluation method. These challenges are mainly related to the design of the qualitative survey S_1 and quantitative survey S_2, and were identified when discovering the poor agreement (section 4.3.2) between participants in S_2. Mcfarlane et al. (2008) argues that the low inter-rater agreement may be explained by the significant number of possible categories to choose from when conducting S_2 as a participant. This brings us back to the design and analysis of S_1 (section 3.4.1).

Modifications to the qualitative survey

In S_1, the participants' task is to create descriptions for videos from the environment. I suggest that in addition to creating the descriptions, the participants should be inquired about how many distinct events they believe the environment (videos) is capable of expressing, N. When sampling new participants to achieve theoretical saturation (section 3.4.1), this number, N, should contribute to the level of saturation. After collecting enough submissions, m, the median($[M_1, M_2, \cdots, M_m]$) will aid the researcher when selecting categories for use in S_2, for example by not exceeding the suggested number. Also, the participants in S_1 could inform about other aspects of the environment they consider essential, and which are not interpretable from the descriptions.

Modifications to the quantitative survey

As long as the sample size (number of participants) is large, model prediction accuracy derived by measuring the cosine similarity between categorical distributions is a reliable measure, even if the number of categories to choose from is significant. That said, there is room for improvement, especially regarding the way participants of S_2 may use the given selection of categories. It appears that the videos in S_2 are somewhat ambiguous, and participants tend to struggle when selecting a category because several categories may be equally plausible.

If the categorisation task appears challenging, it should be adjusted early in the process. Therefore, experience suggests recruiting only a limited amount of participants for S_2 in an initial session. With the limited number of submissions, compute the inter-rater agreement (reliability) between raters by using e.g. Krippendorff's alpha or Fleiss' kappa (Panda et al., 2018). If there is strong agreement among between participants (for example $\alpha_{Krip} > 0.667$, see section 3.4.4.2), proceed sampling participants for S_2. The model evaluation objective may be treated as a classification task according to section 3.4.4.4.

If on the other hand there is low agreement between the participants, the number of categories, N, should be decreased. If reducing N is problematic, a solution is to allow the participants to assign more than one category to a video, thus allowing for a hierarchy of categories with k levels. The main category describes the main event, and optional subcategories describe possible subevents. A matrix representing the categorical distributions of a video would be

$$\begin{matrix}\text{The distribution}\\ \text{of categories}\\ \text{(main and subcategories)}\\ \text{for one video}\end{matrix} \rightarrow \begin{bmatrix} c_{1,1} & c_{1,2} & c_{1,3} & \cdots & c_{1,N} \\ c_{2,1} & c_{2,2} & c_{2,3} & \cdots & c_{2,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{k,1} & c_{k,2} & c_{k,3} & \cdots & c_{k,N} \end{bmatrix}$$

where $\underline{c}_1 = [c_{1,1}, \cdots, c_{1,N}]$ is a vector with the distribution of main categorical events, and $\underline{c}_j = [c_{j,1}, c_{j,2}, \cdots, c_{j,N}]$ is a vector with the distribution of subcategory $j \in \{2, 3, 4, \cdots, k\}$. The researcher must then decide upon the level of the hierarchy, k. This new way of categorising videos in S_2 allows retaining the use of the cosine similarity as a measure of categorical similarity between a ground truth video, GT, and a predicted video, PD

$$\cos\theta = \frac{\underline{a} \cdot \underline{b}}{\|\underline{a}\| \|\underline{b}\|}$$

where $\underline{a} = [\underline{c}_1, \underline{c}_2, \underline{c}_3, \cdots, \underline{c}_k]^{GT}$ is comprised by concatenating vectors $\underline{c}_i^{GT}$ and $\underline{b} = [\underline{c}_1, \underline{c}_2, \underline{c}_3, \cdots, \underline{c}_k]^{PD}$ is comprised by concatenating vectors $\underline{c}_i^{PD}$,

where $i \in \{1, 2, 3, \cdots, k\}$. Other measures of similarity or reliability may also be applicable, such as the Kupper Hafner statistic (Kupper & Hafner, 1989), a statistic for assessing inter-rater agreement for multiple attribute responses. This however, is left to be investigated in later work.

4.5.2 The Evaluation Protocol

The protocol for evaluating predictive models using subjective data is shown figure 4.8 below.

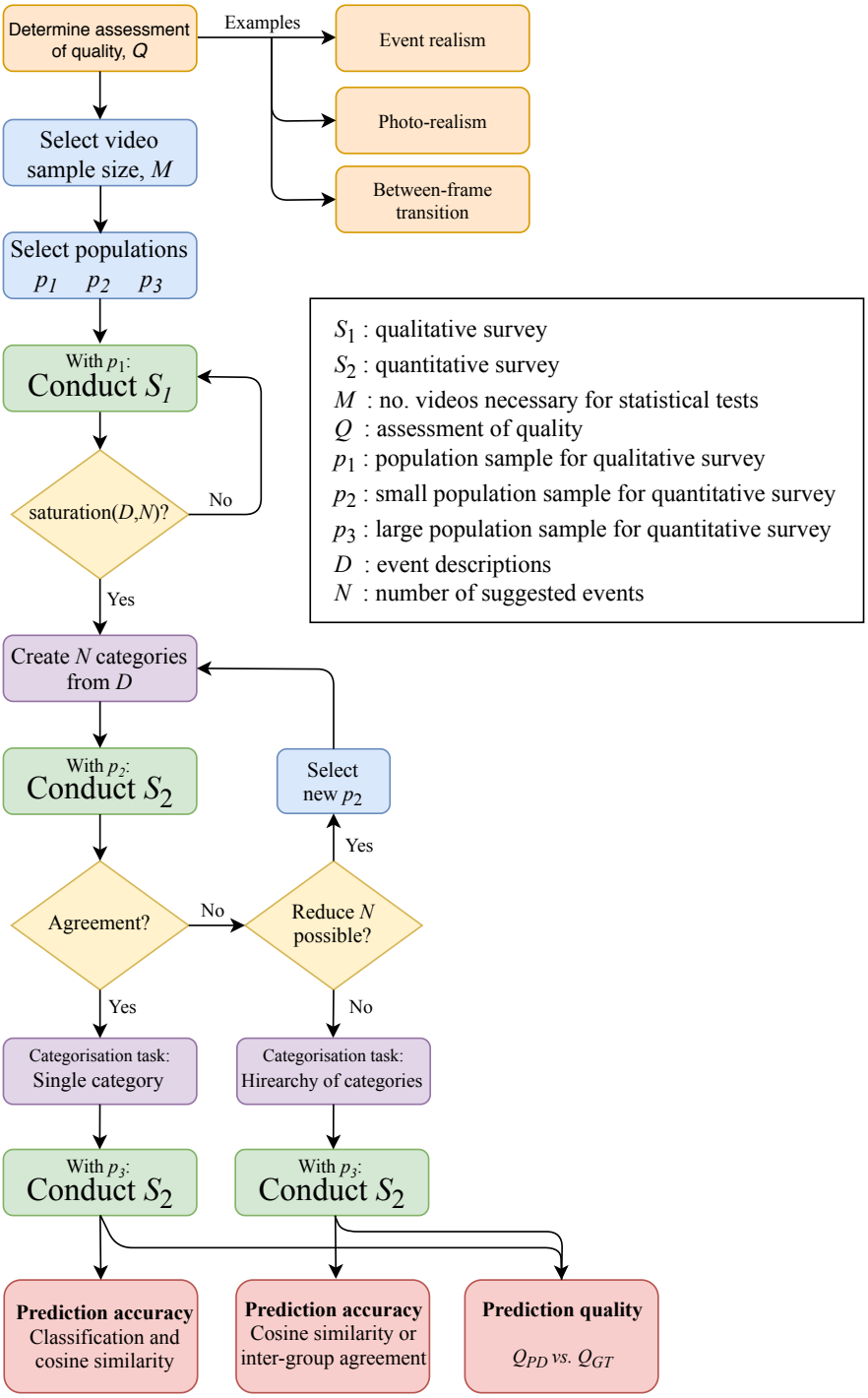


Figure 4.8: A protocol for evaluating predictive models with subjective data. p_1, p_2 and p_3 denote populations of human evaluators.

Chapter 5

Conclusions and Future Work

In this chapter, the conclusions of the results and analysis from chapter 4 are presented in section 5.1, and ideas for future work in section 5.2.

5.1 Conclusions

The field of video prediction using deep learning faces a challenge of great significance, namely the general lack of common test ground for predictive deep learning models (Castelló, 2018). While there exist acknowledged numerical methods for evaluating such models, their use is questionable because they perceive videos in different way of the human visual system.

The work in this thesis presents a new method for evaluating predictive models using subjective data, which measures mainly two properties of predictive models; the realism and the accuracy of the predictions they produce. The evaluation method is based on a mixed methods research design, which involves conducting surveys to gather subjective data about videos generated by predictive models, with minimal prior information about the environment. The data collection is followed by statistical analyses that reveal the quality and prediction accuracy of the models. Evaluating video predictions with the proposed method is still in the early stages; however, it has been tested in order to discover the potential for improvement. With these improvements, the main contribution of the thesis is a refined protocol

for evaluating predictive models using subjective data.

5.1.1 The Proposed Evaluation Method

The proposed evaluation method was tested by applying it in the evaluation of two proposed deep learning models for video prediction of a simulated traffic environment. These models have similar architecture, but each model accepts a distinct image type that represents the environment. In this work, the image types were RGB and semantic segmentation. It was found that both models can produce long-term video predictions that to human observers appear equally realistic to ground truth videos.

The prediction accuracy of the two models was measured by video classification and by comparing categorical distributions for each pair of video consisting of prediction and ground truth. The model using semantic segmentation images performs marginally better than the model using RGB images at the single-label classification task. However, when including more categories in the analyses, as in the multi-label classification task and when comparing categorical distributions, no statistically significant difference in terms of prediction accuracy is identified between the two models. The models produce accurate long-term (10 seconds) predictions of the environment, shown by the considerable cosine similarity between categorical distributions of ground truth and video predictions.

The proposed evaluation method was compared to the currently acknowledged evaluation method in the field of video prediction with deep learning, which is a frame-wise comparison between ground truth and video predictions. Analyses reveal that the two evaluation methods are uncorrelated, and thus measure different properties of a video. Furthermore, this confirms that the acknowledged evaluation method, i.e. the frame-wise comparison, evaluates video predictions differently to how humans perceive them, a discovery consistent with the findings of (Moorthy et al., 2011). This implies that future contributions to the research field in the form of model architectures most certainly should evaluate the model's performance using subjective data, if not only as a part of the evaluation. As a result of the work

throughout this thesis, I have contributed to the research field with a protocol for performing exactly this type of model evaluation. I therefore encourage others to either use it in their research as is, or as a source of inspiration.

5.1.2 The Predictive Model and Image Type

As for the proposed deep learning model architecture, I show that the 'World Models' architecture by (Ha & Schmidhuber, 2018), which uses continuous latent variables to learn an internal model of the environment, can be modified to use discrete latent variables. By replacing the visual component's VAE with a VQ-VAE, and modifying the memory component to perform regression over discrete variables, I have documented a new architecture similar to World Models. Other than it using discrete variables, the main difference with the proposed architecture is that it disregards the implementation of an agent.

Based on the results in section 4.2, I conclude that the proposed model architecture is robust to the type of image representing the environment. Also, whether the model architecture receives RGB images or semantic segmentation images for interpretation of the environment makes no significant difference in terms of the realism and accuracy of predictions. This finding is different from the results by (Luc et al., 2017), who by using an autoregressive CNN architecture and a quantitative evaluation method obtain superior prediction results using semantic segmentation images opposed to RGB images.

5.2 Future Work

5.2.1 The Evaluation Protocol

Having presented a protocol for evaluating predictive models with subjective data, the next natural step is to see this protocol being used to evaluate a variety of model architectures that predict future visual states of a variety of environments. In particular, it would be interesting to see a comparison

between the model architecture proposed in my work and the autoregressive CNN by Luc et al. (2017). Also, since World Models (Ha & Schmidhuber, 2018) inspires both my work and the work by (Risi & Stanley, 2019), an intriguing idea is using the evaluation protocol for a comparison between all three implementations on a video prediction task.

The evaluation protocol can be used to measure other model properties that did not make it to the scope of this thesis. Because some events may occur more frequently than others within an environment, the prediction accuracy can be measured within each separate category to evaluate the model's *expressiveness*. This investigates whether a model has the ability to predict all categories equally well, or if some categories are harder to predict than others.

5.2.2 Image Types

In this work, only the image types RGB and semantic segmentation were used to investigate to what degree image types influence a model's predictive abilities. For future work, the use of other image types should be worth investigating. Also, it would be interesting to see the development of a single predictive model that accepts various image types, as opposed to having k models for the k image types.

5.2.3 The Predictive Model

I acknowledge that there is room for improvement for the proposed predictive model, especially regarding the choice of loss functions and the weighting of these. Also, training this model end-to-end with a gradient-based optimisation method remains a challenge due to the inhibited flow of gradients in the VQ-VAE. Nevertheless, the model has proven to be appropriate for predicting long-term futures of a visual environment, and its capabilities due to its use of discrete variables lead to other interesting ideas for future work:

The current predictive model is deterministic, designing a probabilistic variant could be possible by for example introducing multinomial sampling in the output of the memory component. Another idea involves the use of

transformers (Vaswani et al., 2017), which are sequence models shown to learn sequences of discrete variables effectively. An interesting experiment would, therefore, be to replacing the memory component's LSTM network with a transformer network.

Bibliography

Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., & Savarese, S. (2016). Social LSTM: Human Trajectory Prediction in Crowded Spaces. In *2016 ieee conference on computer vision and pattern recognition (cvpr).* doi: 10.1109/CVPR.2016.110

Barrett, D. G., Morcos, A. S., & Macke, J. H. (2019, 4). Analyzing biological and artificial neural networks: challenges with opportunities for synergy? *Current Opinion in Neurobiology*, *55*, 55–64. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0959438818301569` doi: 10.1016/J.CONB.2019.01.007

Battaglia, M. (2008). *Encyclopedia of Survey Research Methods.* Thousand Oaks: SAGE Publications, Inc. Retrieved from `http://sk.sagepub.com/reference/survey` doi: 10.4135/9781412963947

Battaglia, P. W., Pascanu, R., Lai, M., Rezende, D., & Kavukcuoglu, K. (2016, 12). Interaction Networks for Learning about Objects, Relations and Physics. Retrieved from `http://arxiv.org/abs/1612.00222`

Bengio, Y., Courville, A., & Vincent, P. (2013, 8). Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828. doi: 10.1109/TPAMI.2013.50

Bishop, C. M. (1994). *Mixture density networks* (Tech. Rep.).

Bourlard, H., & Kamp, Y. (1988). Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, *59*(4-5), 291–294. doi: 10.1007/BF00332918

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang,

J., & Zaremba, W. (2016). *OpenAI Gym.*

Bubic, A., Yves von Cramon, D., & Schubotz, R. I. (2010). *Prediction, cognition and the brain* (Vol. 4). doi: 10.3389/fnhum.2010.00025

Castelló, J. S. (2018). A Comprehensive survey on deep future frame video prediction..

Cauchy, A. M. (1847). *Methode générale pour la résolution des systèmes d'équations simultanées* (Vol. 25). doi: 10.1017/CBO9781107415324.004

Chang, M. B., Ullman, T., Torralba, A., & Tenenbaum, J. B. (2017). A Compositional Object-Based Approach to Learning Physical Dynamics.
doi: 10.1002/anie.201610406

Chi-square Goodness of Fit Test. (2008). In *The concise encyclopedia of statistics* (pp. 72–76). New York, NY: Springer New York. Retrieved from `https://doi.org/10.1007/978-0-387-32833-1_55` doi: 10.1007/978-0-387-32833-1{_}55

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Emnlp 2014 - 2014 conference on empirical methods in natural language processing, proceedings of the conference* (pp. 1724–1734). doi: 10.3115/v1/d14-1179

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., ... Schiele, B. (2016). The Cityscapes Dataset for Semantic Urban Scene Understanding. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 3213–3223. doi: 10.1109/CVPR.2016.350

Coupelon, O. (2007). *NEURAL NETWORK MODELING FOR STOCK MOVEMENT PREDICTION 1 Neural network modeling for stock movement prediction A state of the art* (Tech. Rep.). Retrieved from `http://olivier.coupelon.free.fr/Neural_network_modeling_for_stock_movement_prediction.pdf`

Cox, S. (2014). *What is Meaningful Information? - Voice - Two*

Twelve. Retrieved from http://www.twotwelve.com/voice/what-is-meaningful-information.html

Crowston, K. (2012). Amazon Mechanical Turk: A Research Tool for Organizations and Information Systems Scholars. In A. Bhattacherjee & B. Fitzgerald (Eds.), *Shaping the future of ict research. methods and approaches* (pp. 210–221). Berlin, Heidelberg: Springer Berlin Heidelberg.

Dosovitskiy, A., & Koltun, V. (2016). Learning to Act by Predicting the Future.
doi: 10.1111/j.1365-2265.2008.03465.x

Dosovitskiy, A., Ros, G., Codevilla, F., López, A., & Koltun, V. (2017). *CARLA: An Open Urban Driving Simulator* (Tech. Rep.).

Ferrone, L., & Zanzotto, F. M. (2020). Symbolic, Distributed, and Distributional Representations for Natural Language Processing in the Era of Deep Learning: A Survey. *Frontiers in Robotics and AI*, *6*(January). doi: 10.3389/frobt.2019.00153

Finkelstein, M. O., & Levin, B. (1990). Statistical Inference for Two Proportions. In *Statistics for lawyers* (pp. 156–201). New York, NY: Springer New York. Retrieved from https://doi.org/10.1007/978-1-4612-3328-2_5 doi: 10.1007/978-1-4612-3328-2{_}5

Fragkiadaki, K., Agrawal, P., Levine, S., & Malik, J. (2015). Learning Visual Predictive Models of Physics for Playing Billiards.
doi: 10.1016/j.anbehav.2003.09.019

Friston, K. (2005). A theory of cortical responses. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *360*(1456), 815–836. doi: 10.1098/rstb.2005.1622

Gonzalez, R. C., & Woods, R. E. (2008). *Digital image processing.* Upper Saddle River, N.J.: Prentice Hall. Retrieved from http://www.amazon.com/Digital-Image-Processing-3rd-Edition/dp/013168728X

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning* (Tech. Rep.).

Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., & Schmidhuber, J. (2009). A Novel Connectionist System for Uncon-

strained Handwriting Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(5), 855–868. doi: 10.1109/TPAMI.2008.137

Ha, D., & Schmidhuber, J. (2018). World Models. doi: 10.5281/zenodo.1207631

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., & Davidson, J. (2019). Learning Latent Dynamics for Planning from Pixels. In *International conference on machine learning* (pp. 2555–2565).

Hayes, A. F., & Krippendorff, K. (2007). Answering the Call for a Standard Reliability Measure for Coding Data. *Communication Methods and Measures*, *1*(1), 77–89. Retrieved from `https://doi.org/10.1080/19312450709336664` doi: 10.1080/19312450709336664

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 770–778. doi: 10.1109/CVPR.2016.90

Hinton, G. (2010). Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair. In *Proceedings of icml* (Vol. 27, pp. 807–814).

Hinton, G. E., Dayan, P., Frey, B. J., & Neal, R. M. (1995). The "wake-sleep" algorithm for unsupervised neural networks. *Science*, *268*(5214), 1158–1161. Retrieved from `https://science.sciencemag.org/content/268/5214/1158` doi: 10.1126/science.7761831

Hochreiter, S., & Schmidhuber, J. (1997). *Long Short-Term Memory* (Vol. 9; Tech. Rep. No. 8). Retrieved from `http://www7.informatik.tu-muenchen.de/~hochreithttp://www.idsia.ch/~juergen`

Hou, X., Shen, L., Sun, K., & Qiu, G. (2017). Deep feature consistent variational autoencoder. In *Proceedings - 2017 ieee winter conference on applications of computer vision, wacv 2017* (pp. 1133–1141). doi: 10.1109/WACV.2017.131

IBM Corp. (2019). *IBM SPSS Statistics for Windows.* Armonk, NY: IBM Corp. Retrieved from `https://hadoop.apache.org`

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. In *3rd international conference on learning representations, iclr 2015 - conference track proceedings.*

Kingma, D. P., & Welling, M. (2014). Auto-encoding variational bayes. In *2nd international conference on learning representations, iclr 2014 - conference track proceedings.*

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, *37*(2), 233–243. doi: 10.1002/aic.690370209

Kriegeskorte, N., & Douglas, P. K. (2018). *Cognitive computational neuroscience.* doi: 10.1038/s41593-018-0210-5

Krippendorff, K. (2004). *Content Analysis: An Introduction to Its Methodology (second edition).* Sage Publications.

Kupper, L. L., & Hafner, K. b. (1989, 5). On Assessing Interrater Agreement for Multiple Attribute Responses. *Biometrics*, *45*(3), 957–967. Retrieved from `http://www.jstor.org/stable/2531695` doi: 10.2307/2531695

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2016). *Building Machines That Learn and Think Like People* (Tech. Rep.). Retrieved from `https://arxiv.org/pdf/1604.00289.pdf`

Lavrakas, P. J. (2008). Response Bias. *Encyclopedia of Survey Research Methods*. doi: https://dx-doi-org.ezproxy.uio.no/10.4135/9781412963947.n486

Leinweber, M., Ward, D. R., Sobczak, J. M., Attinger, A., & Keller, G. B. (2017, 9). A Sensorimotor Circuit in Mouse Cortex for Visual Flow Predictions. *Neuron*, *95*(6), 1420–1432. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0896627317307791?via%3Dihub` doi: 10.1016/j.neuron.2017.08.036

Lerer, A., Gross, S., & Fergus, R. (2016). Learning Physical Intuition of Block Towers by Example.
doi: 10.1016/j.neucom.2015.11.100

Leys, C., Ley, C., Klein, O., Bernard, P., & Licata, L. (2013). Do not use

standard deviation around the mean, use absolute deviation around the median. *Experimental Social Psychology*, 4–6.

Li, H., Xu, Z., Taylor, G., Studer, C., & Goldstein, T. (2017, 12). Visualizing the Loss Landscape of Neural Nets. Retrieved from `http://arxiv.org/abs/1712.09913`

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *07-12-June*, 3431–3440. doi: 10.1109/CVPR.2015.7298965

Lotter, W., Kreiman, G., & Cox, D. (2016). Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning. , 1–18. Retrieved from `http://arxiv.org/abs/1605.08104` doi: 10.1063/1.1727962

Luc, P., Neverova, N., Couprie, C., Verbeek, J., & Lecun, Y. (2017). Predicting Deeper into the Future of Semantic Segmentation. In *Proceedings of the ieee international conference on computer vision.* doi: 10.1109/ICCV.2017.77

Mahalanobis, P. C. (1936). *On the generilised distance in statistics.*

Mcfarlane, D. J., Ancker, J. S., & Kukafka, R. (2008). A vector space method to quantify agreement in qualitative data. *Symposium A Quarterly Journal In Modern Foreign Literatures*, 455–459.

McHugh, M. L. (2012, 6). The Chi-square test of independence. *Biochemia Medica*, *23*(2), 143–149. doi: 10.11613/BM.2013.018

Moorthy, A. K., Wang, Z., & Bovik, A. C. (2011). Visual Perception and Quality Assessment. *Optical and Digital Image Processing: Fundamentals and Applications*, 419–439. doi: 10.1002/9783527635245.ch19

Neuhäuser, M. (2011). Wilcoxon-Mann-Whitney Test. In M. Lovric (Ed.), *International encyclopedia of statistical science* (pp. 1656–1658). Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from `https://doi.org/10.1007/978-3-642-04898-2_615` doi: 10.1007/978-3-642-04898-2{_}615

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). *Activation Functions: Comparison of trends in Practice and Research for Deep*

Learning. Retrieved from http://arxiv.org/abs/1811.03378

Olgac, A., & Karlik, B. (2011). Performance Analysis of Various Activation Functions in Generalized MLP Architectures of Neural Networks. *International Journal of Artificial Intelligence And Expert Systems*, *1*, 111–122.

Panda, M., Paranjpe, S., & Gore, A. (2018). Measuring Intergroup Agreement and Disagreement. , 19.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc. Retrieved from http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Pearson's Correlation Coefficient. (2008). In W. Kirch (Ed.), *Encyclopedia of public health* (pp. 1090–1091). Dordrecht: Springer Netherlands. Retrieved from https://doi.org/10.1007/978-1-4020-5614-7_2569 doi: 10.1007/978-1-4020-5614-7{_}2569

R Core Team. (2018). *R: A Language and Environment for Statistical Computing.* Vienna, Austria. Retrieved from https://www.r-project.org/

Risi, S., & Stanley, K. (2019). Deep neuroevolution of recurrent and discrete world models. In (pp. 456–462). doi: 10.1145/3321707.3321817

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In N. Navab, J. Hornegger, W. M. Wells, & A. F. Frangi (Eds.), *Medical image computing and computer-assisted intervention – miccai 2015* (pp. 234–241). Cham: Springer International Publishing.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). *Learning representations by back-propagating errors* (323rd ed.). Nature.

Schäfer, A. M., & Zimmermann, H. G. (2006). Recurrent Neural Networks Are Universal Approximators. In S. D. Kollias, A. Stafylopatis,

W. Duch, & E. Oja (Eds.), *Artificial neural networks – icann 2006* (pp. 632–640). Berlin, Heidelberg: Springer Berlin Heidelberg.

Schoonenboom, J., & Johnson, R. B. (2017). How to Construct a Mixed Methods Research Design. *KZfSS Kölner Zeitschrift für Soziologie und Sozialpsychologie*. doi: 10.1007/s11577-017-0454-1

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *3rd international conference on learning representations, iclr 2015 - conference track proceedings.* Retrieved from `http://www.robots.ox.ac.uk/`

Singh, D. (2018). Self-supervised learning gets us closer to autonomous learning. , 1–5. Retrieved from `https://hackernoon.com/self-supervised-learning-gets-us-closer-to-autonomous-learning-be77e6c86b5a`

Sola, J., & Sevilla, J. (1997). Importance of input data normalization for the application of neural networks to complex industrial problems. *Nuclear Science, IEEE Transactions on*, *44*, 1464–1468. doi: 10.1109/23.589532

Spinner, T., Körner, J., Görtler, J., & Deussen, O. (2018, 10). *Towards an Interpretable Latent Space – An Intuitive Comparison of Autoencoders with Variational Autoencoders.*

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, *15*(56), 1929–1958. Retrieved from `http://jmlr.org/papers/v15/srivastava14a.html`

Srivastava, N., Mansimov, E., & Salakhutdinov, R. (2015). Unsupervised Learning of Video Representations Using LSTMs. In *Proceedings of the 32nd international conference on international conference on machine learning - volume 37* (p. 843–852). JMLR.org.

Strack, F. (1992). "Order Effects" in Survey Research: Activation and Information Functions of Preceding Questions. In N. Schwarz & S. Sudman (Eds.), *Context effects in social and psychological research* (pp. 23–34). New York, NY: Springer New York. Retrieved from

https://doi.org/10.1007/978-1-4612-2848-6_3 doi: 10.1007/978-1-4612-2848-6{_}3

Such, F. P., Madhavan, V., Conti, E., Lehman, J., Stanley, K. O., & Clune, J. (2018). Deep Neuroevolution: Genetic Algorithms Are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning. Retrieved from http://bit.ly/http://arxiv.org/abs/1712.06567

Sutskever, I., Martens, J., & Hinton, G. (2011). *Generating Text with Recurrent Neural Networks* (Tech. Rep.). Retrieved from https://www.cs.utoronto.ca/~ilya/pubs/2011/LANG-RNN.pdf

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems 27* (pp. 3104–3112). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

Thoma, M. (2016). A Survey of Semantic Segmentation.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., ... Kavukcuoglu, K. (2016). WaveNet: A Generative Model for Raw Audio. In *Arxiv.* Retrieved from https://arxiv.org/abs/1609.03499

van den Oord, A., Vinyals, O., & Kavukcuoglu, K. (2017). Neural discrete representation learning. In *Advances in neural information processing systems* (Vol. 2017-Decem, pp. 6307–6316).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is All you Need. In I. Guyon et al. (Eds.), *Advances in neural information processing systems 30* (pp. 5998–6008). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *07-12-June*, 3156–3164. doi: 10.1109/CVPR.2015.7298935

Wang, H., & Schmid, C. (2013). Action Recognition with Improved Trajectories. In *The ieee international conference on computer vision (iccv).*

Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Tao, A., Kautz, J., & Catanzaro, B. (2017). High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs.
doi: 10.1145/2050100.2050101

Wang, Z., Simoncelli, E. P., & Bovik, A. C. (2003, 11). Multiscale structural similarity for image quality assessment. In *The thrity-seventh asilomar conference on signals, systems computers, 2003* (Vol. 2, pp. 1398–1402).
doi: 10.1109/ACSSC.2003.1292216

Watters, N., Tacchetti, A., Weber, T., Pascanu, R., Battaglia, P., & Zoran, D. (2017, 6). Visual Interaction Networks. Retrieved from `http://arxiv.org/abs/1706.01433`

Wichers, N., Villegas, R., Erhan, D., & Lee, H. (2018, 6). Hierarchical Long-term Video Prediction without Supervision. Retrieved from `http://arxiv.org/abs/1806.04768`

Willig, C. (2013). *Introducing Qualitative Research in Psychology* (3rd. ed.).

Wooldridge, M., & Jennings, N. R. (1995). Intelligent agents: theory and practice. *The Knowledge Engineering Review*, *10*(02), 115. Retrieved from `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.119.2204&rep=rep1&type=pdfhttp://www.journals.cambridge.org/abstract_S0269888900008122` doi: 10.1017/S0269888900008122

Wortman Vaughan, J. (2018). *Making Better Use of the Crowd: How Crowdsourcing Can Advance Machine Learning Research* (Vol. 18; Tech. Rep.). Retrieved from `http://jmlr.org/papers/v18/17-234.html`.

Wydmuch, M., & Kempka Michałand Jaśkowski, W. (2018). ViZDoom Competitions: Playing Doom from Pixels. *IEEE Transactions on Games*.

Yann LeCun, L. B. Y. B., Patrick Haffner. (1999). Object Recognition with Gradiant-Based Learning. (0), 302.

Yao, Y., Rosasco, L., & Caponnetto, A. (2007). On Early Stopping in Gradient Descent Learning. *Constructive Approximation*, *26*(2), 289–315. Retrieved from `https://doi.org/10.1007/s00365-006-0663-2`

doi: 10.1007/s00365-006-0663-2

Zeiler, M. D., & Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), *Computer vision – eccv 2014* (pp. 818–833). Cham: Springer International Publishing.

Zhao, S., Song, J., & Ermon, S. (2017). Towards Deeper Understanding of Variational Autoencoding Models. Retrieved from `http://arxiv.org/abs/1702.08658`