Colourising and Classifying Greyscale Images Using an Unsupervised Deep Convolutional Neural Network

George Monk 17666456@students.lincoln.ac.uk

A report submitted in partial fulfilment of the requirement for Bsc(Hon) Computer Science

School of Computer Science

University of Lincoln.

2021

Acknowledgements

Thank you to all of my friends and family for keeping me at least (somewhat) sane during the Covid-19 epidemic.

Abstract

Given a greyscale image, this project explores the possibility of a deep convolutional neural network producing viable colourisations that are acceptable to the human eye. As such is the nature of past bias and the human visual system, certain considerations must be made to accommodate what is expected comparative to what is produced. In particular, exploration into feature generation and application has been undertaken, in an attempt to better segment images into colourised regions. The final network is of a U-net style autoencoder, and the Cifar-10 and Flickr Faces datasets have been used to train the network, and test its capabilities. Overall, the results gained from such training and testing demonstrate the efficacy of the network and colourisation using this method. In addition, literature research has been provided, giving a summary of much similar work and an explanation into the considerations of both neural networks and colour regarding this problem space.

1. Introduction

Image colourisation is a subject of deep and intense curiosity for many interested in machine learning problems. A difficult challenge steeped in a curiosity for the unknown, the premise of colourisation is one that is both easy to imagine and hard to implement. Given the sentimental value of many old photographs; historical, personal, or perhaps a combination of both, seeing such items in colour provides a window into a world barely glimpsed at before. It can breathe life into the lost, and grant some of the present to the past.

This is not just because colour can be pleasing to the eye, but also because it allows the viewer to draw parallels between their life and that from years past. A sense of greater empathy can be brought about through the simple provocation of emotion upon seeing green grass and a blue sky (O’Connor, 2011): a feeling that the image is somewhat more alive than it was before. Colourisation also sees deep practical use through the expansion of colour dimensionality. Medical and infrared imaging becomes much clearer and more evident when areas of interest can be detected and summarily colourised (Kong, Lei, and Ni, 2011), although this is not the topic of this report or project. Thus is the main focus and goal of this project: to extrapolate colour from greyscale images through using a deep convolutional network, colour conversion algorithms, and (in some cases) feature extraction.

In this report, a convolutional neural network (CNN) architecture, capable of producing believable and vibrant colour from otherwise colourless images, is presented. In some cases, where classification is possible, the proposed architecture uses condensed features obtained from a pre-existing classifier and merges them with network results halfway through processing. Where otherwise the network would struggle to detect certain characteristics and generalise to higher level features correctly, a small indicator linking relevant inputs (such as say; a cat having a closer resemblance to a dog than an aeroplane) provides the network with an amalgam of possible colourisation, in order to test whether or not adding additional image features can provide the network with a wider range of colour choice.

Two datasets have been used to achieve this end and demonstrate final results: the Cifar-10 dataset, and the Flickr faces dataset. Only 1,100 images from the Flickr faces dataset has been used, and those in use have been scaled down to a 128 * 128 resolution (not including colour channels). This is mostly in part due to the high overhead of using the whole dataset in relation to the device the network is trained on, but it also demonstrates the overall effectiveness of the architecture in cases where training data is either diminished or of a low volume.

2. Background and in-depth Literature Review

In this chapter, three sections of specific and important relevance have been produced to orchestrate a better designed final artefact through understanding of the problem space and background in general. These three sections cover large and generalized areas of relevant background, being (in order of appearance): a summary of similar work, a background on neural networks and the types that are often used for image processing, and a synopsis on digitalised colour.

2.1. Similar Work

Examples of similar work within both this subject field and problem space indicate that there is no one well-established solution or architecture for greyscale colourisation (as of yet). The discussion below acts as both detailed evidence of this fact, and also serves to show the multitude of ways in which this problem can be approached.

Colorful Image Colorization

One recent proposal of automatic grayscale image colourisation that encouraged renewed vigour into the subject field was this network design and project methodology, released in 2016 (Zhang, Isola, and Efros, 2016). It used a convolutional neural network which included a final probability matrix. which identified a per-pixel region analysis of the image and idealised the most likely colour from the information available to it.

Infrared Colorization Using Deep Convolutional Neural Networks

Similar to the colour translation of RGB to CIELAB explored within this project, the methodology used in this network successfully colourises images with relatively high accuracy by converting RGB values to near infrared (Limmer and Lensch, 2016). To improve performance, this model also integrates the mean filtered input image to the final output, and joins it bilaterally using the actual input as a guide. Whilst the overall accuracy of this method is rather high, the limitations if infrared become evident when encountering objects with specific vibrancy in an image, such as the green in a traffic light.

Fully Automatic Image Colorization Based on Convolutional Neural Network

This method places strong emphasis onto the semantic information contained within an image (Varga and Szirányi, 2016). An example used within the accompanying report is as follows: ‘the colour of leaves on a tree may be some kind of green in spring, but they could be brown in autumn'. As a result, high accuracy is obtained from this method, although if the semantic information is left relatively unknown or badly understood, resulting images are overlaid with a general sepia tone.

Deep Colorization

An unusual turn from other architecture developed for this problem space, this solution, unlike others, uses a standard deep neural network, rather than a convolutional neural network (Cheng and Yang, 2015). Integrated into this methodology is included chrominance values and feature descriptors of input features. Still achieving results with verifiable accuracy, this architecture is also subject to sepia transformation when applied to badly understood images.

Colourization of Greyscale Images Using Deep Learning

Proposed as another deep learning approach for image colourisation, this project combines global and local information within an image, and also incorporates user hints to give further merit to colourisation results (Bhushan, Khumar, and Reshi, 2018). It uses a convolutional neural network alongside human intervention to produce colourisations that are quantifiably accurate.

Colourizing Monochrome Images

This method utilises different minimization formulas in an attempt to find the most accurate colourisation technique within their proposed architecture, which is done on high-resolution images to a high level of success (Al-Jaberi, Jassim, and Al-Jawad, 2018). The two main minimization formulas used for this academic paper are the Poisson formula and the Euler Larange operation. In addition, exploration was undertaken with colourisation of images that had been enhanced with colour scribbles, and spatial and frequency domains.

Perceptual Conditional Generative Adversarial Networks for End-To-End Image Colourizations

A conditional generative adversarial network architecture has been used to great effect within this research, as it explores the split-functionaltity between each component of the generative adversarial network framework (Halder, De, and Roy, 2018). Using this architecture alongside adversarial loss and pixel based restraint, the final architecture is reportedly able to predict colour at high levels of accuracy, whilst continuing to be adaptable.

LUCCS: Language-based User-customized Colorization of Scene Sketches

Focusing far more on instance based image segmentation to use with generative language models, the proposed system therein uses literal language to identify objects within an image and colourises them using a generative adversarial network (Zou et Al, 2018). Instead of real-life imagery, this proposal uses pencil sketches, segmenting each element within an image (such as a house or the moon) using language-based inference.

Evaluation of Image Colourization Approaches

In this research paper, there are detailed multiple approaches to the colourization of monochrome 64 * 64 images, taken from the ImageNet dataset in order to evaluate the overall performance of each approach (Appelgren, Berggren, Båvenstrand, and Hahr, N.d). In short, they conclude that: Adam is a more effective optimization function than stochastic gradient descent, and that regression methods tend to wash out an image, whereas classification approaches produces vibrant, yet incorrect colourization results.

2.1.1. Overall

To conclude this segment, image colourisation is a difficult problem, close to a regression-based machine learning task. Solutions therein are based upon extrapolation of image features, which is nearly always done using a neural network style architecture. Other machine learning algorithms, such as random forests, support vector machines, or decision trees are seemingly not often used to solve image colourisation, likely because results are so often based upon semantic detail. Algorithms and solutions that can extrapolate semantics from inputs especially well are shown to have great application in this problem field, with convolutional neural networks consequently being a cornerstone for colourisation development.

2.2. Neural Networks

2.2.1. Artificial Neural Networks

The pre-modifier of 'artificial' must be applied to neural networks of this type, so as to displace them from biological neural networks (BNNs), which are the inspiration for artificial neural networks (ANNs) (Hassoun, 1995). BNNs are built upon a groundwork of neurons connected locally by synapses, typically sending biological signals from dendrites, and sending information in turn from an axon (Dongare, Kharde, and Kachare, 2012), exhibiting collective behaviour (Valencia et Al, 2009). Neurons are touted as the critical building block for biological computation: they are cells specialised to repeat the process of signal receiving, calculation, and delivery (Ananthanarayanan, Esser, Simon, and Modha, 2009).

Connected via white matter (specifically, long-range axonal-fibre pathways) (Menon, 2011), these biological neural networks are the structure for large-scaled brain regions, such as the cerebellum and fronto-parietal regions (Yeo et Al, 2011), and are thus vital for practically any learning scenario imaginable.

Artificial neural networks aim to emulate and replicate the lower level functions of these brain regions, or the simpler biological neural networks (Maind and Wankar, 2014). Instead of neurons and synapses, artificial neural networks use neurons, weights, and (sometimes) biases (Zupan, 1994) (Figure 1). Neurons take a weighted sum from any number of connected inputs and performs further computation (Jain, Mao, and Mohiuddin, 1996), sending results through the network and propagating further calculation. In a feed-forward ANN, neurons are unconnected to neurons in the same layer, but this is not the case for every network topology, such as in a recurrent ANN (Krenker, Bester, and Kos, 2011). Connectivity, or how many connections a single neuron has, is dependent on the type of network in question.

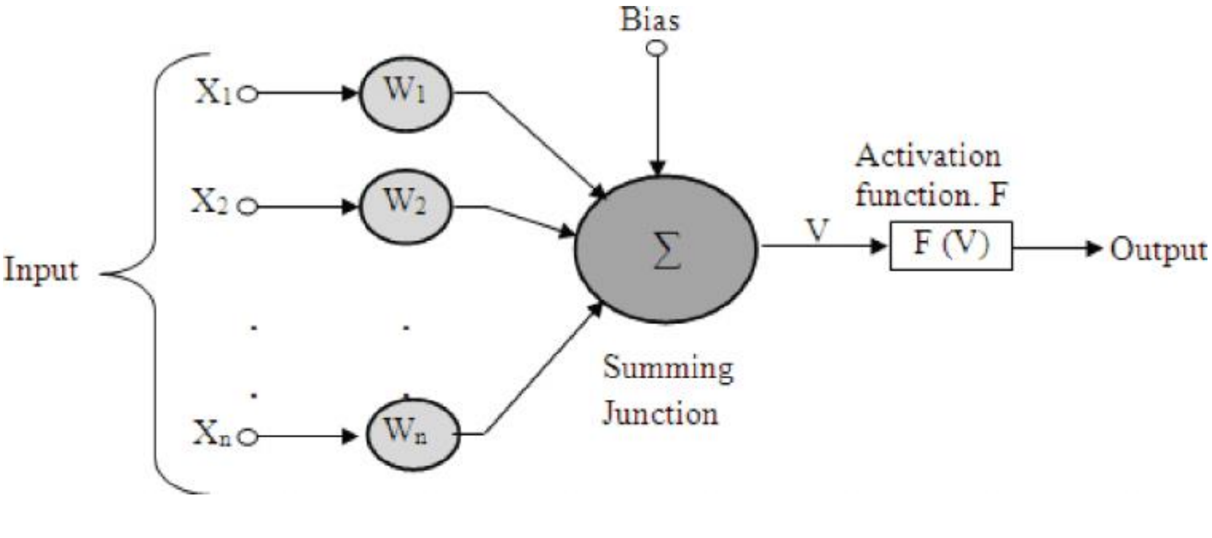


Figure 1: A simplified neuron in an ANN (Kabir and Hasin, 2013)

Additionally, there can exist a number of bias nodes in an ANN, sometimes one for each layer, and sometimes one for each neuron. The role of the bias node is similar to that of the weight; it manipulates an output value of its connected neuron (James and Tucker, 2004). However, whereas the weight value applies a dot procedure to the input value, the bias node is instead added to that consecutive total (Wong and Hamounda, 2003), ensuring that every neuron will be correctly activated or deactivated, shifting the output to a more applicable value. Essentially, weights are multiplicative, and bias is additive.

Note that output values and transformations from layer functions are almost always subject to an activation function, most often a logistic one (e.g sigmoid), or the current popular choice, ReLU (Rectified linear unit) (Ramachandran, Zoph, and Le, 2017),. These activation values quantise the values given by the output into a particular given range, adding non-linearity to the network and reducing the probability that incredulously high numbers are generated, reducing the overall computational need of the network. Figure 1 shows a simplified representation of a neuron in an ANN.

Typically, the standard ANN has an input layer and an output layer (some network structures can contain more than one input and/or output layer, as dependent on problem space), with zero to any number of hidden layers in between (Matrik et Al, 2014) (Figure 2). These hidden layers are named as so simply because they are layers that the end user does not see or interact with. The user only enters an input and receives an output; the intricacies of linear transformation within hidden layers is typically unbeknownst to them.

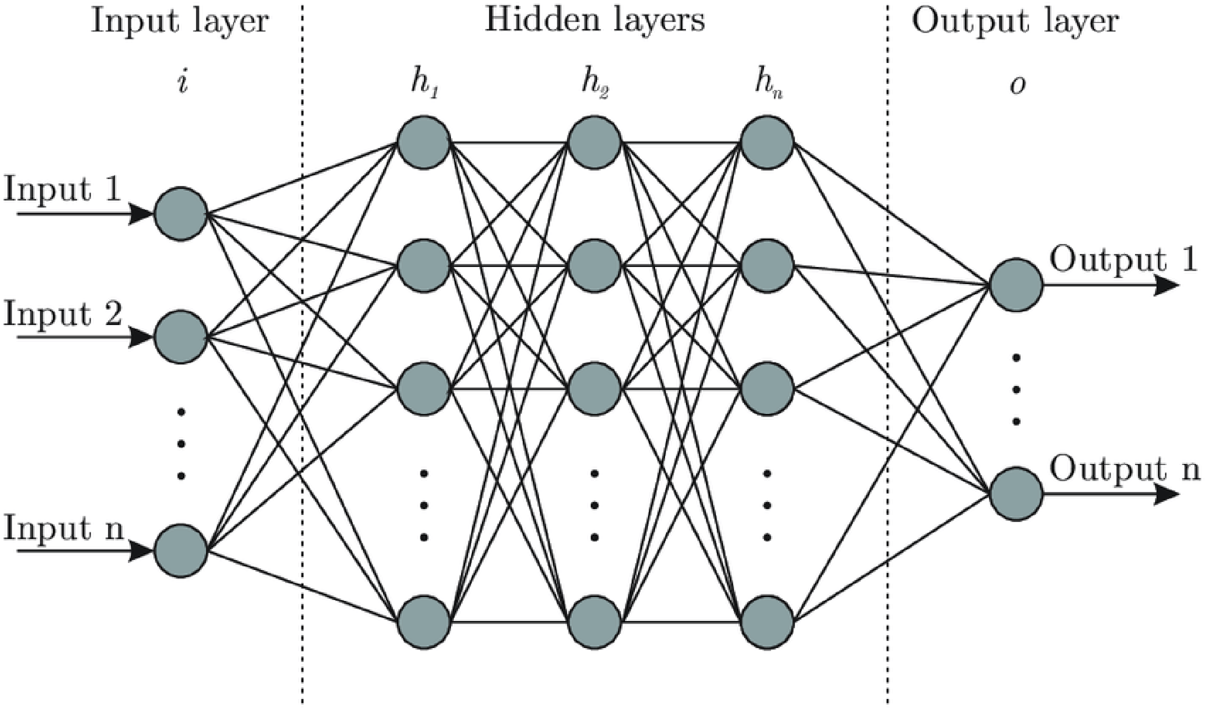


Figure 2: A simple feed-forward ANN (Bre, Gimenez, Fachinotti, 2018). Note the direction of the arrows indicate the trajectory of data from the input layer to the output layer.

The network itself will take an input through the input layer and parse it through hidden layers at least once (based on how recursive the network is) (Socher, Lin, Ng, and Manning, 2011), until it reaches the output layer whereupon the results are released. Each hidden layer can apply a different transformation based upon it’s general, preconceived purpose, or even through what it learns throughout training – an indicator of an adaptive feature map (Hu, Lin, and Hsiu, 2018).

After parsing data through the network the weights and bias values are adjusted based upon the specific training algorithms in use. Commonly, backpropagation – gradient descent - is used to recalculate these metrics (Örkcü and Bal, 2011, which does so by discerning the derivatives of every metric and the overall error, parsing backwards through the network and modifying the weights and biases of the network (Rumelhart, Durbin, Golden, and Chauvin, 1995).

At the closing stage of each epoch, the overall error and cost for each layer, and their respective weights and biases, is calculated (Da Silva et Al, 2017). This is a vital stage for training the network, as through these calculations the neural network adapts and learns what it should be outputting based on a set input.

Typically, each epoch only updates the weights and biases by a minute factor (based upon the ‘learning rate’ of the network), nudging them slightly closer to a preferred value that reduces the cost and increases the amount of correct predictions (Thimm, Moerland, and Fiesler, 1996). If these values were changed by a large value the neural network would become more accurate, but the cost would rapidly skyrocket, increasing computational overhead by an unbeknownst factor: thus smaller increments are preferred.

The network is typically exposed to two types of data during this process, and one afterwards. It is trained on ‘training’ data and glimpses at ‘validation’ data during training (Foody, 2017). The training data supplies the general parameters and guidelines for the network to be trained, whereas validation data provides the network with better recourse where mistakes are made, allowing for more fine-tuned hyper-parameters.

After training, the network is tested on ‘testing’ data: data that is unknown to the network: it is used to verify the effectiveness of an ANN. Generally, there is a large amount of training data, and smaller amounts of validation and testing data, although they all share the same categories of features.

The measurement of success for an ANN is often boiled down to a measure of metrics; specifically accuracy and loss. Accuracy is the measure of the total sum of correct prediction divided by the total number of all items. Loss is not statically defined, as it is often obtained through specific calculation of different loss functions, such as mean-squared error (MSE) or binary cross entropy. These metrics area often used to evaluate the overall success of the network, with each score being differently justifiable as dependent on project context.

2.2.2. Data Normalization

Data normalization is the process of condensing a series of input data down into a smaller, regularized format. It is crucial to obtaining good results (Sola and Sevilla, 1997), and the effectiveness of any neural network architecture is heavily dependent on the success and efficiency on the normalization of input data (Banja and Das, 2018).

In a great deal of learning-based algorithms, especially those dealing with large amounts of raw data, it is expected that data is to be somewhat erroneous, susceptible to noise, or otherwise distorted (Nayak, Misra, and Behera, 2014), simply due to these being natural properties of real-world data. Thus, it is often quantized into a specific range (however necessary) to guarantee the quality of the data before it is used to train any algorithm or network, through a combination of reduction, transformation, and integration (Ogasawara et Al, 2019).

In images themselves, imprecise data can often take form through irregular lighting or other visual artifacts. Though digital images already come in a set range, normalizing them regardless reduces or otherwise removes the image dependencies on said image noise (Finlayson, Schiele, and Crowley, 1998).

Normalization, when applied to neural networks, has the great benefit of accelerating training speed dramatically, and increases overall training, validation, and test accuracy alike (Huang and Qin, 2018). Intrinsically, this is because of a reduction to computational overhead (as the complexity of the data is reduced), and also because the margin between important features is reduced.

2.2.3. Convolutional Neural Networks

Convolutional Neural Networks are so aptly named due to their innate design; they are neural networks with a convolution layer. The namesake of the CNN, a convolution is an image processing technique that takes an input image feature and manipulates it using a (typically small) matrix – or 'kernel' - to create a filtered output (Ludwig, 2013). Examples of image convolutions can include contrast adjustment, edge detection, and image enhancement. These kernels are defined by width and height, which results in the creation of three-dimensional outputs.

In a convolutional layer, these filters are learnt throughout training to attempt to identify which filters work best for the problem space, based on a three-dimensional transformation of an input (height, width, and depth) (Albawi, Mohammed, and Al-Zawi, 2017). As a filter is applied continuously to small regions of the input, a feature map is produced, which acts as the output of the layer. This feature map is the scalar summation of weights, biases, and the input, built up steadily over time.

These feature maps are typically built sequentially, in that each layer of the network uses the features devised from prior layers as it's basis. Starting at the shallowest layer of the network, low-level features such as edges and lines are identified, leading to higher level feature identification at deeper layers (Ghosh, Roy, and Ghosh, 2014), such as the wheels of a car, the wings of a bird, or the eyes of a dog.

Different layers can be integrated into a CNN to encroach upon different results – the most basic of which is a fully connected layer (Figure 4). A fully connected layer is a series of interconnected neurons that translates an input to an output using standard forward propagation. For all intents and purposes, a fully connected layer is identical to a typical ANN layer. Depending on the problem space, these layers can improve performance and accuracy dramatically (Sainath, Vinyals, Senior, and Sak, 2015), when used in tandem with a convolutional layer.

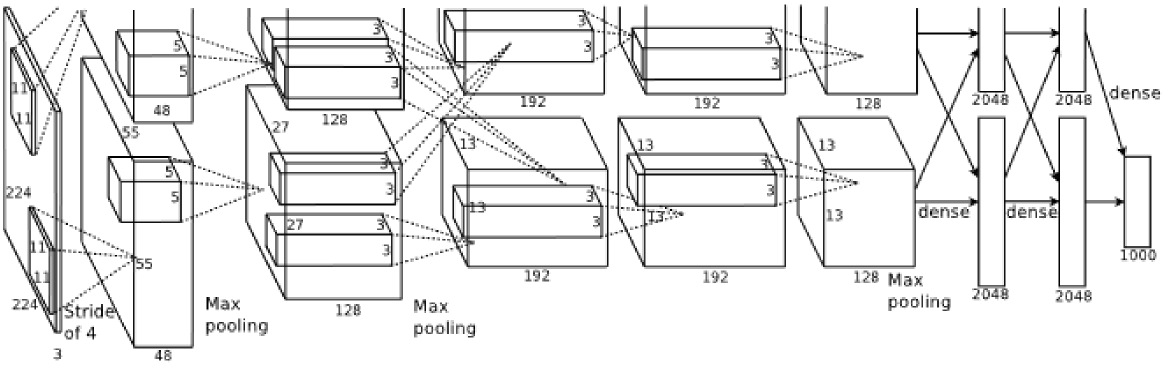


Figure 3: The well-known ImageNet submission by Krizhevsky et Al utilised a mixture of CNN layers, and a variety of kernel sizes (Krizhevsky et Al, 2011)

Oftentimes, ANNs are unfit to use when processing images. Consider an image with a resolution of 32 * 32 * 3: in a fully connected ANN layer that equates to 3072 weights, a fairly manageable amount. However, the number of weights increases exponentially with the size of the image. Consider now an image with a resolution of 128 * 128 * 3: in the same fully connected ANN layer, now 49,152 weights are needed.

This massively increases the demand for a larger ANN, and thus increases the computational requirements in turn, resulting in unfeasible use-case scenarios. In a CNN however, neurons are connected to local regions of an input through the construction of said feature maps, allowing for computation of images at a relatively low expense in comparison.

2.2.4. Pooling, Upsampling, and Kernels

In convolutional neural networks, pooling – layers that reduce the dimensionality of data at a certain point – has been often used to great effect within neural network. Through the selection of invariant features, they massively reduce the algorithmic time (convergence rate) it takes to send an input through the network (Nagi et Al, 2011).

As an image processing tool, generalization tends to be effective as pattern recognition is built upon neighbourhood dependency - or the relationship between pixels (Chen et Al, 2005). This means that colour information of pixels in an area can be assumed based on a small selection, assuring the effectiveness of pooling layers.

Similarly, the kernel of a convolution layer is also built around the logic of pixel-neighbourhood dependency. The standard kernel size is 3 × 3, as that provides each pixel with a neighbour from every angle. Even (not odd) kernel sizes, such as 2 × 2, 4 × 4, and so on, are not commonly used: they focus on asymmetrical neighbourhood regions, often producing asymmetrical results without symmetrical padding (Wu et AL, 2019).

Much like smaller kernels, larger kernels (such as a 7 × 7 or a 9 × 9 kernel) find less commonplace use. Instead, using large kernels quickly accentuates training time, due to a quickly expanding number of resources required to convolve an image pixel by pixel with such large regions being accounted for. Because of this, larger kernels see greater application where large neighbourhood regions must be considered, such as salt-and-pepper noise removal (Ramadan, 2014).

2.2.5. Autoencoders

Autoencoders work by encoding the information input into layers of the neural network, parsing it through the network, and then decoding it at whatever stage deemed necessary (Chen et Al, 2017), changing the size of the input at particular stages. The framework of autoencoding and decoding is designed to reduce the dimensionality of a network (Wang et Al, 2014),which in turn increases processing speed and decreases training time (thanks to the reduction of data integrity).

Though there may be some initial concern over the loss and subsequent reconstruction of data, the goal of an autoencoder is not just to efficiently conform to the primary requirement of the project, whether it be classification, regression, or any other goal, but it also aims to find parts of the input that are most important (Bank, Koenigstein, and Girves, 2020).

By selectively choosing portions of an input, the encoder is able to reduce the cardinality of data whilst keeping its integrity as close to normal as it can. In succession, the decoder works in the exact same way, except it increases the cardinality of the data to reconstruct it, and thus attempts to construe parts of the image using the given regions.

Generally, the computational requirements of a neural network are dependent on the network being efficient in both memory and computation time during interaction (Badrinarayanan et al, 2017). The framework of an encoder and decoder, working in unison, can provide both of these benefits, severely reducing the amount of computational overhead.

However, in some cases this combination may not be desired, as autoencoders particularly are prone to the destruction of data interpretability (Zhou et Al, 2020), and an overfocus on unimportant elements of data, due to autoencoders proclivity to use all data – a boon in some cases, and a detriment in others.

2.2.6. Generative Adversarial Networks

Developed in 2014 in an attempt to reduce the linearity of neural networks, a generative adversarial network creates, designs, and trains two neural networks in tandem: one aims to generate data that matches the training set, the other attempts to distinguish between real and fake data (Goodfellow et Al, 2014). This transforms into, in essence, a zero-sum gain, where through competition between the two, the gain of one network is the loss of another. Behaviour such as is beneficial to the generation of data, as this unsupervised learning forces each network to adapt, thereby becoming better at their integral task.

To extrapolate the semantics behind a GAN in the form of analogy: it can be considered somewhat equivalent to a specialist variant of evolution, where predator and prey take the presence of discriminator and generator. The goal of the discriminator (predator) is to detect false data (hunt its prey), whereas the goal of the generator (prey) is to generate data that goes undetected by the discriminator (avoid its predator). Wherever one fails in its goal, it learns (survival of the fittest), propagating a continuous loop of constant improvement, only ending once a clause has been satisfied, or to the limitations of hardware/software.

As discussed within the similar work section of this report, GANs have been used to colourise monochrome images to levels of high success, successfully colourising images without falling to beige overtones.

2.2.7. Dropout

When a machine learning algorithm is trained using data with an overabundance of unnecessary features or with data that is too simple for a complex machine learning algorithm, the algorithm will have a high chance of overfitting to that data (Hawkins, 2004)

Dropout is a concept introduced within neural networks in an attempt to reduce overfitting (Stristvana et Al, 2014). With large networks containing potentially massive amounts of parameters, it becomes likely that overfitting becomes evident. By acting as the name suggests, dropout integrated into a network causes certain neurons to deactivate during training, reducing the probability that neurons (or their respective weights and bias) do not over-adapt to training input (Srivastava, 2013). This causes a veritable cost to train accuracy for the boon of heightened testing accuracy, although increasing dropout too much can cause the network to underfit instead (Bayer et Al, 2013). Regardless, dropout often reduces overfitting (where the network is overexposed to training data) and increases regularization, thinning the network and stopping nodes from over adapting to the training data (Molchanov, Ashukha, and Vetrov, 2017).

In the case of image colourization, the rate of dropout must be fine-tuned so as to not overfit the network on training data, but also to not invalidate the learning of different characteristics of image classes – such as grass being green, metal often having a sheen but sometimes rusted and muddy brown.

2.3. On Colour

Whilst it may be otherwise arbitrarily defined, colour as defined by electronic devices exists on a wholly numeric scale; where a human might see the colour red as literally as it is, a computer reads that information as an array of integers or floats, values as dependant on the colour space in which it exists. This makes colour easy to manipulate on the digital level, as simple changes can be made to alter one colour and transform it into another.

However, the scope of such manipulations are dependent on the actual colour space being used, of which there are many. Through the following transformation as detailed throughout this section, it is possible to extrapolate one channel of a greyscale image within a colour space, from colour channel conversion across multiple types.

2.3.1. Human Perception of Colour

Before detailing the ways in which artificial colour spaces replicate colour digitally, it is prudent to give a preliminary brief on how the human visual system detects and perceives colour.

Technically, the human eye does not detect *colour*. Instead, it is specifically the rods and cones intrinsic to the eye's physiology that transmit wavelengths of light reflected from the surface of an object to the brain (Labin, Safuri, Ribak, and Perlman, 2014), which then imparts cognitive processing to establish better colour identity (Bartels and Zeki, 2000). Certain cones detect different semblances of colour, such as wavelengths of red from a strawberry, whereas rods do not detect colour. Cones actively partition colour from bright areas and objects, whereas rods only detect shades of grey in darker environments (Reitner, Sharpe, and Zrenner, 1991).

Processing a single visual colour is not purely reliant upon general perception. Much of actual cognition takes high importance in the role of colour perception, with effects such as contrast enhancement, gamut expansion, and colour constancy being evidence of this fact (see figures: 3, 4, and 5a through 5c). It is theorised that the human brain potentially rationalizes the perceived colour information of an object not just through the object itself, but also by decrypting the accompanying colour detail of neighbourhood objects and preconceived bias (Zeki and Marini, 1998). When viewing coloured objects under controlled environments, it has been observed that similar brain areas are activated, giving evidence to multiple regions being important in colour perception (Banneret and Bartels, 2013; Vandenbroucke et Al, 2016).

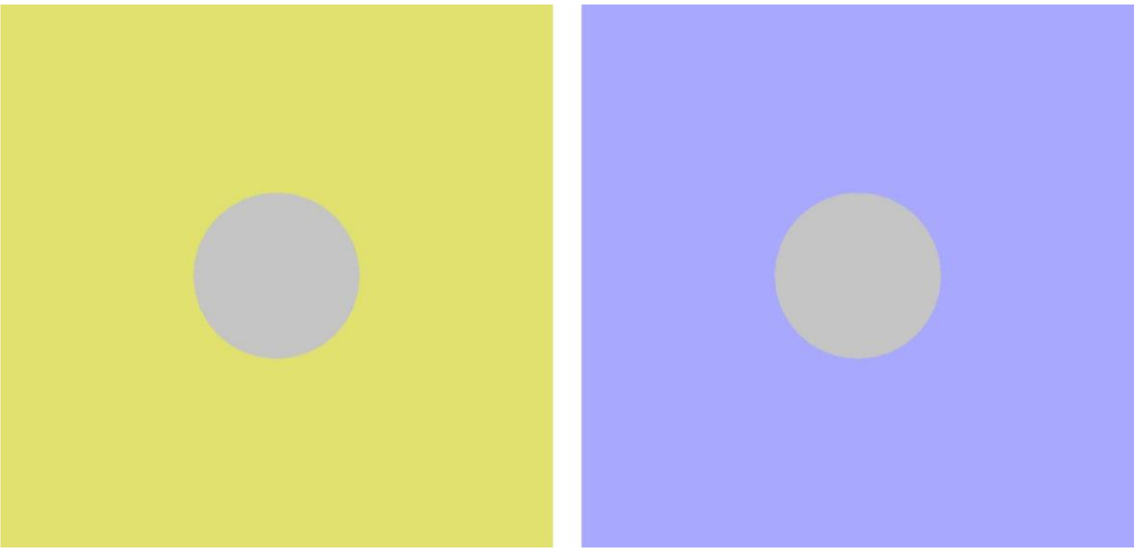


Figure 4: Shows contrast enhancement where each circle is slightly tinged by their respective background colour (Ekroll, Faul, and Wendt, 2011)

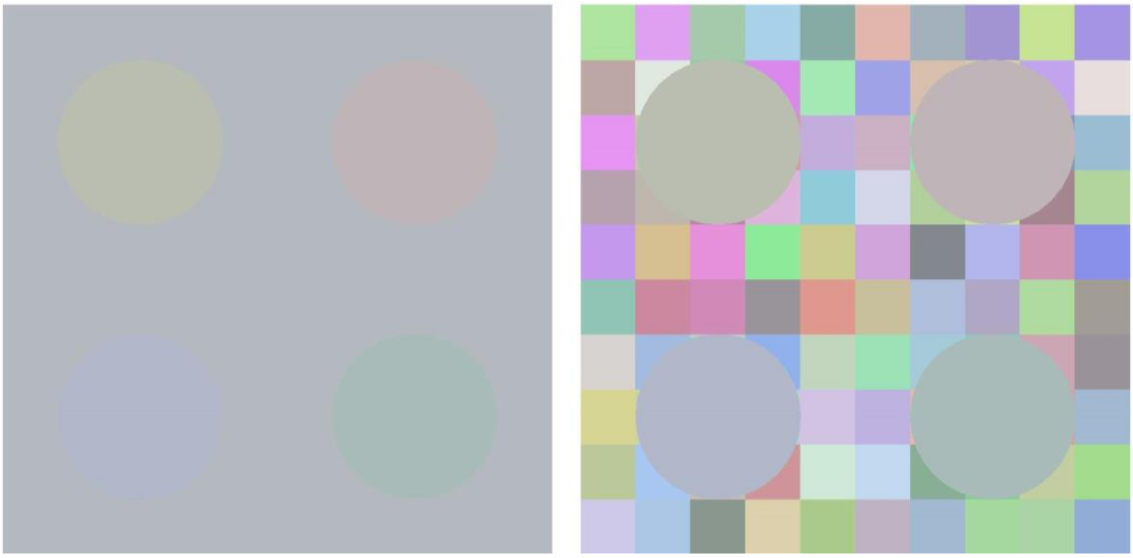


Figure 5: Both images in this figure share the same four discs, but Ekroll, Faul, and Wendt evidence that those embedded in uniform grey (left) appear far more saturated and difficult to divine than those in the variegated background (right), evidencing gamut expansion.

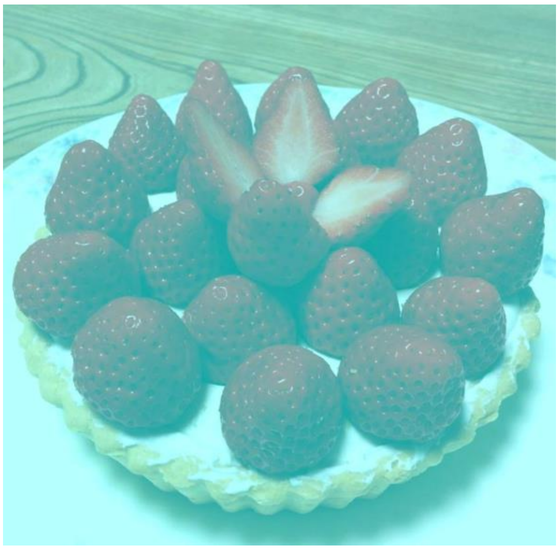


Figure 6a: This image (Akiyoshi Kitaoka, 2018) shows what appears to be a bowl of red strawberries. But in reality, the image contains no red pixels whatsoever. Colour constancy indicates that likely past experiences lead the brain to expect strawberries to be red, thus is it colours it so

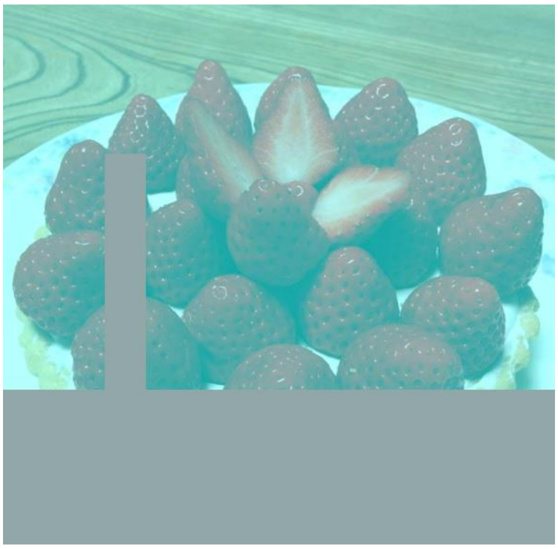


Figure 6b: Demonstrates the ground truth of the grey strawberries

.

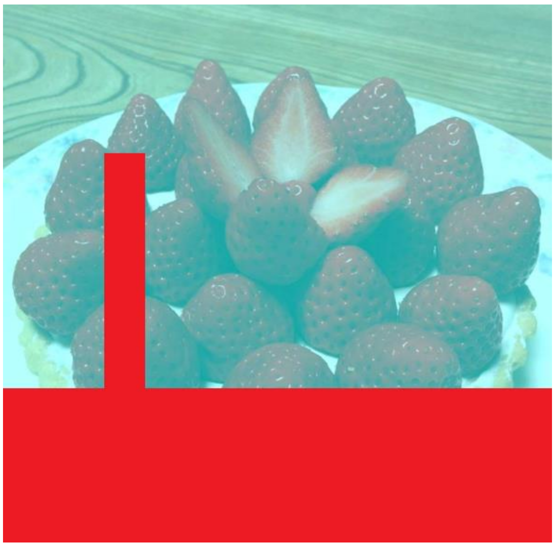


Figure 6c: Focussing on a red bounding box can often trick the brain into discerning the strawberries as a more vibrant red.

Essentially, human perception of colour is not perfect in how it perceives colour. It can be considered close to an unbelievably complicated case of biological jury-rigging, so as to make general vision easier to comprehend. What the human visual system actually ‘sees’ is based upon wavelengths of light detected by the cones and rods within the eye (or retina, to be precise), cross-referenced with contextual bias from the brain in an attempt to give otherwise confusing scenes sufficient and believable visual confirmation (Bannert and Ba 2013). In colourisation attempts, certain colours will be expected of an image given sufficient past knowledge and evidence, and in their absence the image will otherwise seem fake and unrealistic.

2.3.2. RGB and sRGB

Commonly, the red-green-blue colour space (RGB) is oft misinterpreted as one digital manifestation of colour, though in truth this is somewhat of a misnomer. RGB represents colour using three colour channels; red, green, and blue (Figure 6). Combined, they form a number of different colour combinations, although they do not account for the effect of digitised gamma values, or the relation between pixel value and luminance (Poynton, 1998).

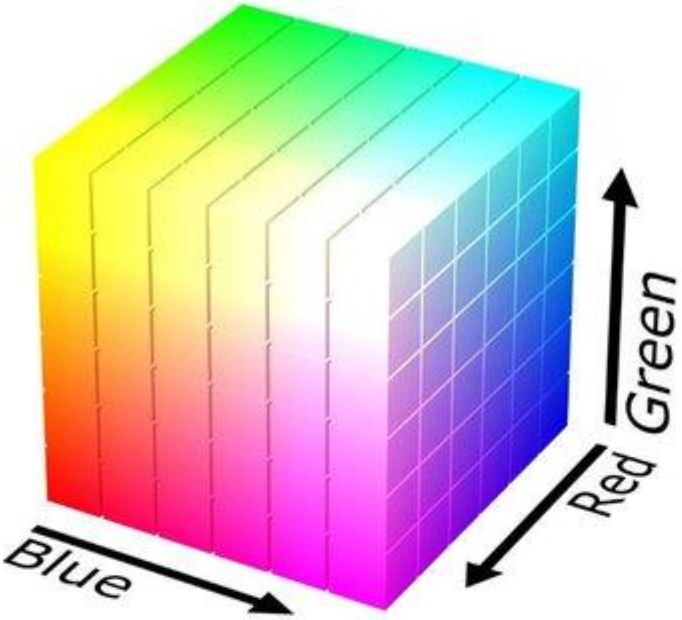


Figure 7: The RGB colour space visualised within a three-dimensional space (Popov, Ostarek, and Tenison, 2018).

Surprisngly, this means that – using standard RGB – one colour can appear variant based on the viewing device, with levels of disparity dependent on the colour in question, the hardware in use, and any settings or software that could also interfere with luminance values (such as the brightness of a computer monitor) (Bilissi, Jacobson, and Attridge, 2008). The output of any hardware or software that outputs a colour based on digital request, such as a colour printer, is also included in the broad scope of 'viewing devices'. In addition, it is perceptually un-uniform, meaning that the difference between two colours in Euclidean distance can often have a low correlation (Tkalcic and Tasic, 2003).

To counteract this effect, a markedly similar colour space known as sRGB was developed in 1996 to serve as an accepted digitized variant of RGB (Anderson, Motta, Chandrasekar, and Stokes, 1996), being now the standard colour space which most digital services use. Quite simply, sRGB is a standardised variant of RGB, where colour results – despite being conformed to a smaller range – are uniform in their saturation and accuracy regardless of device (Figure 7). Converting RGB channels to sRGB (Figure 8) is a necessary step in the conversion of RGB to XYZ channels, as they must be made linear with respect to energy (Lindbloom, 2017) -or gamma.

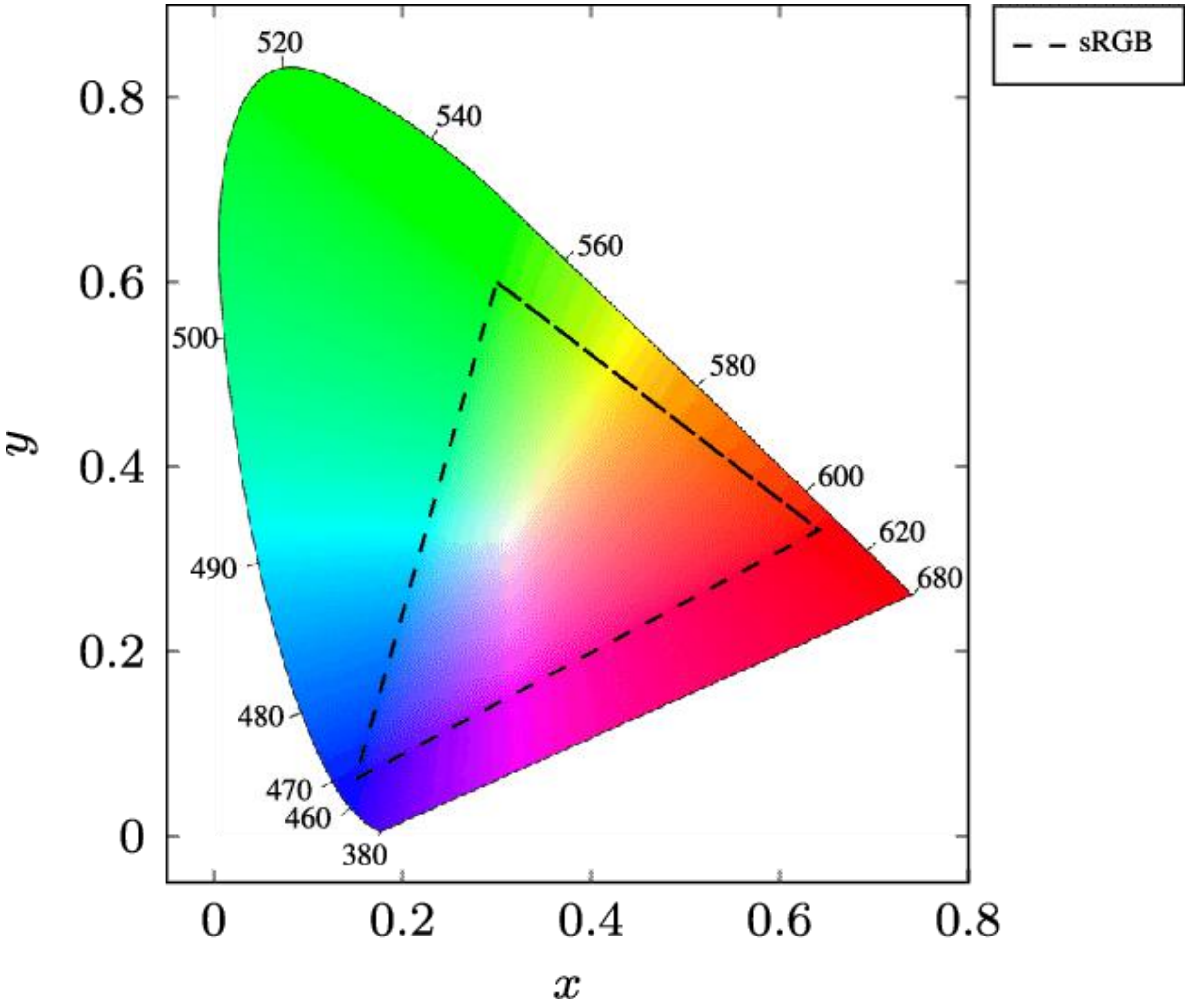


Figure 8: sRGB values as described within CIExy chromaticity coordinates (Amara et Al, 2018)

$$v \in \{ \mathrm{r, g, b} \}$$

$$V \in \{ R, G, B \}$$

$$v = \begin{cases} \dfrac{V}{12.92} & if\ V \leq\ 0.0405 \\ \dfrac{(V + 0.055)^{2.4}}{1.055} & else \end{cases}$$

Where:

- r, g, b denotes sRGB channels
- R, G, B denotes RGB channels

Figure 9: Conversion of RGB to sRGB (Lindbloom, 2017).

2.3.3. XYZ and L*a*b*

Developed to replicate the human physiological perception of colour, the XYZ and CIELAB (L*a*b*) colour spaces were developed in 1931 by the International Commission on Illumination (abbreviated to CIE for its French name; *Commission Internationale de L'éclairage*). The colour spaces, of which there are many, aim to mathematically define perceptible colour using a trichromatic system, emulating the function of cone photoreceptors in the human eye (Smith and Guild, 1931).

Given sRGB values, it is viable to convert colour values into the XYZ colour space. Instead of channels indicative of colour alone, XYZ makes use of luminance, declaring Y to indicate the lightness of an image (normally ranging from 1 to 100, or 0 to 1), Z to represent blue – or more accurately, the S cone – and X being a mix of non-negative values (Figure 10). The conversion of sRGB to XYZ channels (Figure 11) requires the use of a transformation matrix (Afifi et Al, 2020), as required based upon the workspace and reference white in use. The reference white is an indicator as to which white point is in use, and attempts to define 'white' within a set of tristimulus values.

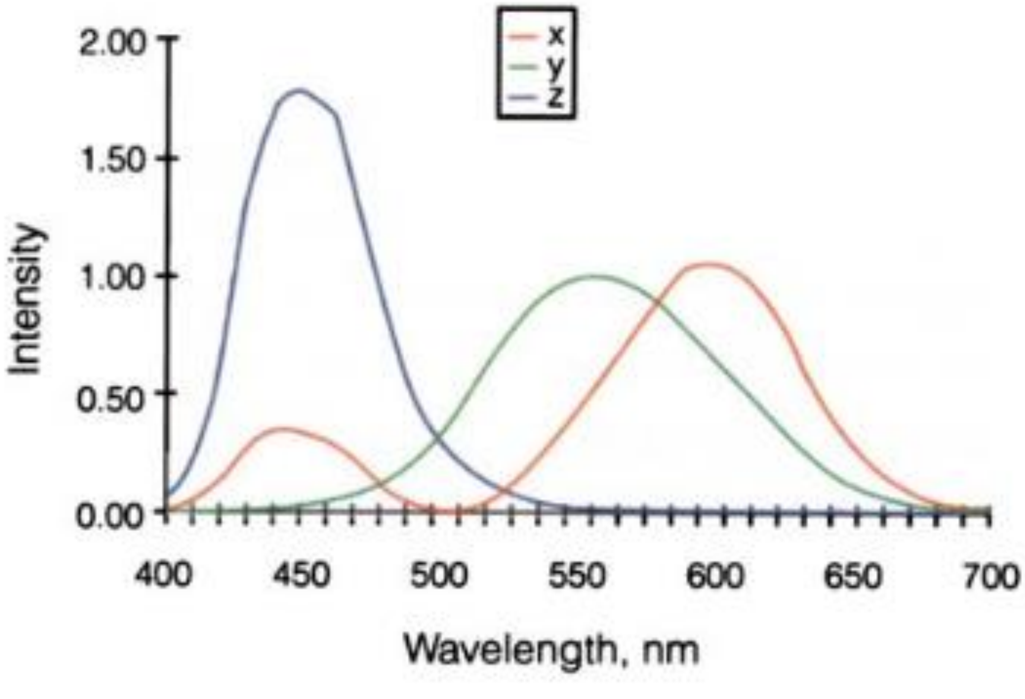


Figure 10: Standard CIE XYZ colour matching primaries (Westland, 2003)

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = [M] \begin{bmatrix} r \\ g \\ b \end{bmatrix}$$

Where:

- X, Y, Z denotes XYZ channels
- r, g, b denotes sRGB channel
- $[M]$ denotes a transformation matrix

Figure 11: Conversion of sRGB to XYZ (Rowlands, 2020).

XYZ tristimulus values only predict colour identity; only referring to how two colours will appear the same, not how they appear different. A rudimentary solution was made to subtract these XYZ values from CIExy chromaticity coordinates to predict colour difference between two colours, yet this solution was ineffective (MacEvoy, n.d.). Two colours subtracted from an equal XYZ could result in colours that were either nearly identical or completely different. As such, there was need for a solution that was more uniform, leading to the creation of the L*a*b colour space.

From this point onwards, it is finally possible to transform these XYZ colour values into L*a*b* format. L* indicates lightness, or the wavelength of light, a* represents green-red colours, and b* represents blue-yellow colours (Figure 12); giving the full range of standard human colour perception. Given suitable values, the description of XYZ to L*a*b* conversion is given throughout multiple stages of conversion (Figure 13).

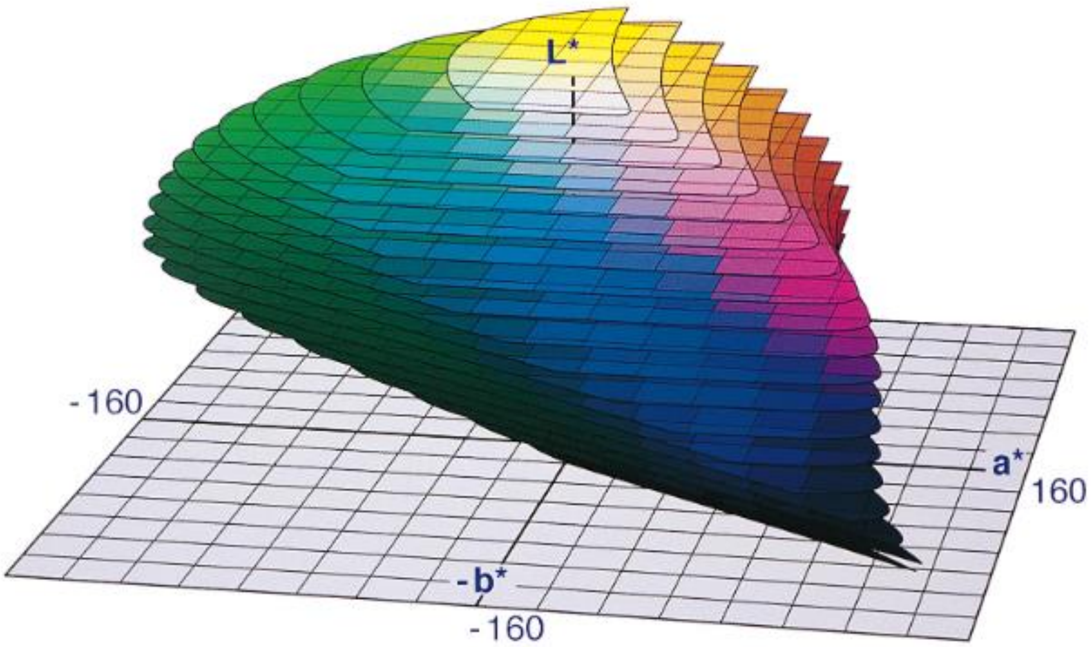


Figure 12: The CIEL*a*b* colour space composed of L* spaced 5 units, with a* and b* being spaced 20 units (Hill, Roger, Vorhagen, 1997)

$$L = 116(f_y - 16\ )$$
$$a = 500(f_x - f_y)$$
$$b = 200(f_y - f_{z)}$$

Where:

$$f_x = \begin{cases} \sqrt[3]{x_r} & if\ x_r > \epsilon \\ \dfrac{kx_r + 16}{116} & else \end{cases}$$

0

$$f_y = \begin{cases} \sqrt[3]{y_r} & if\ y_r > \epsilon \\ \dfrac{ky_r + 16}{116} & else \end{cases}$$

0

$$f_z = \begin{cases} \sqrt[3]{z_r} & if\ z_r > \epsilon \\ \dfrac{kz_r + 16}{116} & else \end{cases}$$

And:

$$x_r = \frac{X}{X_r}$$

0

$$x_r = \frac{Y}{Y_r}$$

0

$$x_r = \frac{Z}{Z_r}$$

0

$$\in = 0.008856$$

$$k = 903.3$$

Figure 13: Conversion of XYZ to L*a*b* (Lindbloom, 2017).

In this case, $(X_r, Y_r\ Z_r)$ indicate a reference white.

Converting XYZ, and by extension RGB, to this format produces a varying colour channel L despite the lack of any distinguishable colour. Of course, the image is still in a greyscale format, but by extrapolating this previously non-existent value the dimensionality of a CNN will have been reduced by 33%, as only two channels must be predicted instead of three. In addition, chromaticity can be predicted without any interference from luminance – in RGB, each channel of red, green, and blue has an equal measure on the luminance of a pixel. In L*a*b* format, this responsibility lies solely upon the channel of L*, which is always known, and is thus static.

Those two channels can finally be merged with the L channel, meaning that (disregarding extreme miscolouring) any change to the image is likely to be an improvement. Afterwards, conversion back to RGB values is undertaken by inversing the formulas given above.

3. Methodology

3.1. Project Management

As stated in the proposal for this project, no specific work methodology has been used. Modularity, efficiency, and speed are vital requirements for the construction of the final product, so as to make sure that the relatively limited resources available are all used as efficiently as possible. Realistically, this project has only one imperative goal: successful colourisation. The problem space is currently too limited to reasonably produce any core or enhancement features outside of this primary goal.

However, focus has been placed upon establishing the core benefits of using learned image features alongside the CNN. As such, a burgeoning enhancement feature for a task outside of the direct project is good rate of success for obtaining these image features. This is done through simple classification metrics where possible.

In addition to the complexity of the primary goal (colourisation), it is also one that is time-consuming by nature, given to the nature of training a neural network. In order to make best use of the time and resources available, the Gantt chart produced for the proposal for this project has been strictly followed. Where present goals and results occasionally outperformed the requirements set in place by this chart, expansions and changes were made to still present an idealisation of progress.

3.2. Toolsets and Machine Environments

3.2.1. Python

Developed in 1991 by Guido van Rossum, Python is an open source language designed to be easy to read, accessible, and modular (van Rossum, 2007); containing a vast array of custom made, similarly open source libraries that can be used (Millman and Aivazis, 2011). Many of these libraries are applicable to this project, especially those that include the framework for machine learning models, such as Tensorflow, and Keras.

The ease of use regarding Python is one of its strongest features – being described as near English in verbatim (Saabith, Fareez, and Vinothraj, 2019). It has shown to be one of the most growing popular computer science languages within the past five years (Srinath, 2017), especially in areas related to data science and machine learning (Stančin and Jović, 2019). From the get-go, Python was designed to be readable, and executes line-by-line. Though this comes at some deficit to execution time, it also results in pin-point error detection, where errors can be identified at their origin line. Additionally, supporting detail and documentation is provided that notifies a user as to what specifically is causing the error, and what can be done to fix it.

Specifically, convolutional Neural Networks designed in Python have a long standing history of success, most predominantly starting with the well-known ImageNet classification submission by Krizhevsky et Al in 2012. Their submission used a deep convolutional neural network and achieved top error rates of 37.5% and 17%, which – at the time - far exceeded prior state-of-the-line results within this competition (Krizhevsky, Sutskever, and Hinton, 2012).

Primarily, the most important modules that have been used within this project include NumPy, Keras, SciPy, and OpenCV. NumPy is one of the most popular libraries to be used within Python, being the most prominent module for any program requiring scientific or numeric computation (Harris et Al, 2020). NumPy has received attention for standardising numerical arrays within Python: an attribute that will be of use within this project (Van der Walt et Al, 2011).

To assist in the computer vision operations required for this project, OpenCV is capable of multiple practices with Python, such as video capture, image manipulation, and object tracking (Howse, 2013). Primarily, the main advantage of using OpenCV within this project is image manipulation, specifically in its regard to colour. OpenCV already contains the structure and logic for colour conversion, which makes for optimized colour conversion (Baggio, 2012). Despite this fact, colour conversion algorithms have been designed for this program to demonstrate the algorithm behind them, although they do not match the speed at which OpenCV operates in this regard.

In this project, code is produced using Jupyter Notebook, and accessed via Anaconda Navigator. In order to allow for parallel GPU processing alongside machine learning tools, the NVidia toolset of CUDA has been used. Compared to using only the CPU, enabling CUDA support and training models on the GPU resulted in an approximate 20 times increase in training speed on a model with roughly 2 million parameters, using the Cifar-10 dataset and no down sampling or dimensionality reduction.

With the CPU alone, training on that network with the same parameters took nearly 42 minutes per epoch, whereas with the GPU enabled, each epoch took roughly 2 minutes instead.

3.2.2. Keras

Keras is a high-end API that allows for the design, construction, and training of neural network frameworks . Based upon Tensforflow 2.0 – an API that provides low-level and high-level APIs for a multitude of machine learning tasks – Keras can be used to generate efficient CNNs, providing a variety of tools and options for training such a network. For example, sample datasets – such as Cifar10 -, different layers, optimization functions, and other tools can be downloaded and used with ease.

Keras has been shown in multiple sources – such as those discussed within the similar work section of this report – to have great accuracy with training/test results in machine learning. Amongst the top-5 winning teams on machine learning challenges presented by Kaggle, Keras is the most used deep learning framework (Chollet et Al, 2015).

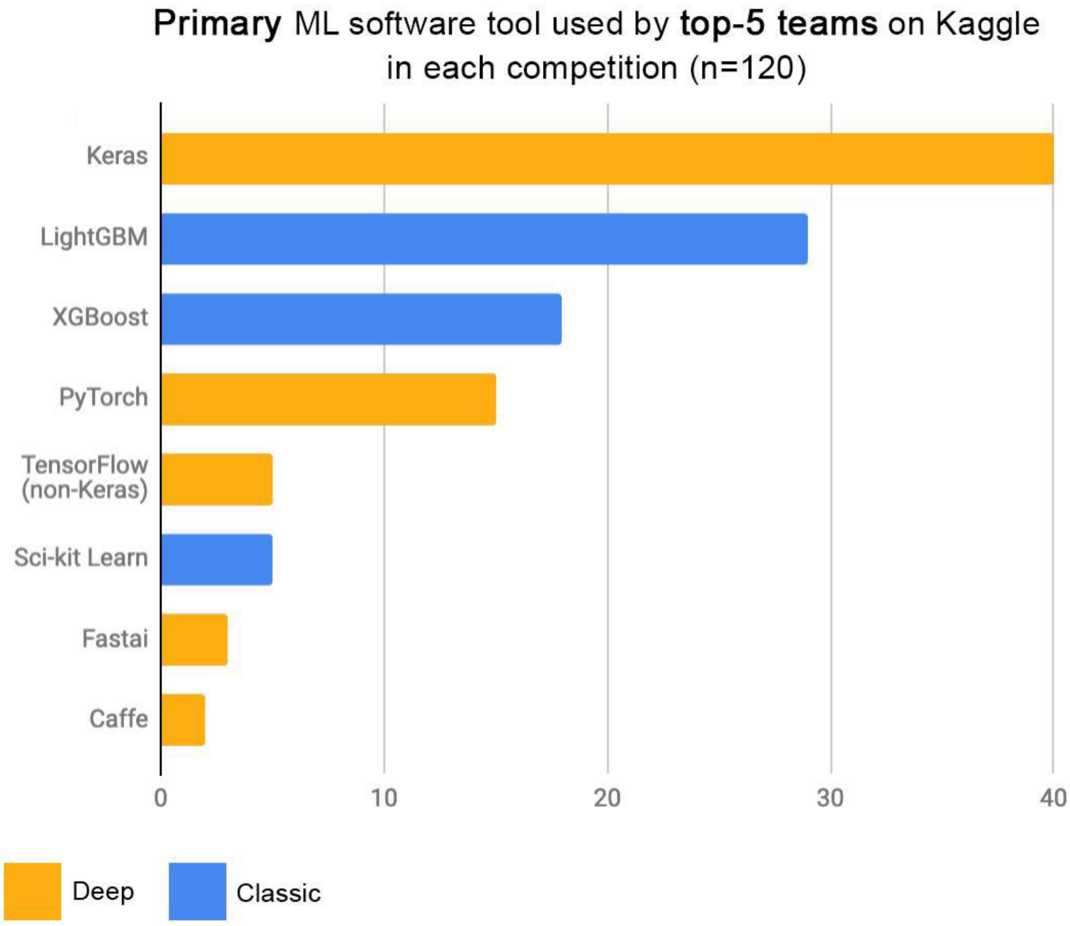


Figure 14: Popularity of primary machine learning tools used in Kaggle competitions (Keras Team, N.d)

One of the strongest advantages to using Keras is the readability metrics it offers (Lee and Song, 2019). One can easily train a neural network and get detailed updates on training process by epoch and loss/accuracy results by each epoch. This is an invaluable asset, as it allows one to keep track of process and find a global minima for the network.

In addition, due to Keras being a modular toolset, it takes substantially less time for a network to be developed and theorised (Atienza, 2018). Much like the reasoning behind using the Cifar-10 dataset, using Keras provides the means to theorise, design, and then analyse multiple network architectures much faster than manually designing each network.

Whilst there can be some argument towards merit derived from a neural network created entirely from scratch, doing so would only be truly justifiable if the project aim were to either: identify a method of improving general CNN performance, or if the end goal of said project was incredibly particular. So much so that it is not covered by conventional mean. As this project goal does not meet either of these standards it is unfeasible to design a bespoke solution to accommodate on grounds of both time and efficiency.

3.2.3. The Cifar-10 Dataset

The Cifar-10 dataset is a publicly available dataset provided by the University of Toronto, containing 60000 total colour images of 10 classes (Krizhevsky and Hinton, 2009). There are 50000 training images and 10000 test images already pre-arranged for use. Each image in this dataset is a labelled subset of the 80 million tiny images dataset, meaning that they have a resolution of only 32*32*3 file: 32 pixels each by width and height and 3 colour channels, or 3072 elements in total. In addition, images are labelled and classified into 1 of 10 classes, ranging from automobile and aeroplane to frogs and dogs (Figure 15).

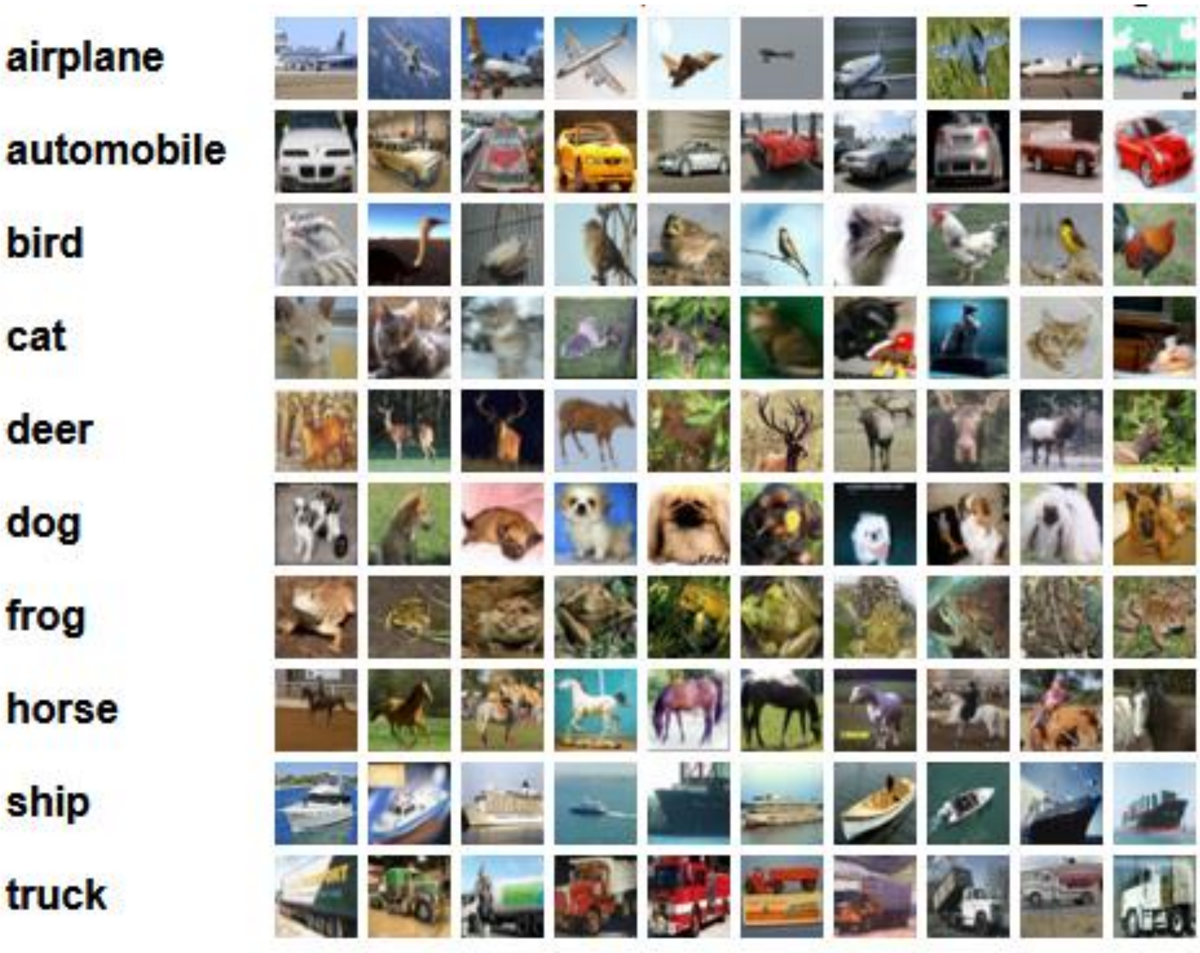


Figure 15: 10 random images from every class in the Cifar-10 dataset (Krizhevsky, n.d.).

Despite the fact that less detail is stored within the image compared to one usually used to train a network, the Cifar-10 dataset is still capable of producing results just as accurate as those from other datasets (Lin, Chen, and Yan, 2013). Their unusual size also provides a more evident benefit: a smaller size means less storage space required, and the smaller pixel arrays reduce the computational overhead, thus reducing the time needed to train and update the network exponentially.

One may assume that in return for the advantages gained from using the Cifar-10 dataset, there is a sacrifice of the quality and efficacy of any neural network trained using it. Surprisingly, this is not the case. It has been shown in numerous work that the Cifar-10 dataset is capable of receiving similar results as other – higher resolution – image datasets. Instead, the Cifar-10 dataset finds prominent use for the case study of generalization (Recht, Roelofs, Schmidt, and Shankar, 2018). Higher resolution datasets provide CNNs with the ability to detect fine-grained detail (Tan and Le, 2019). The fewer pixels, the less data there is to extrapolate information from.

Regardless, pixel-by-pixel information within an image is highly dependent on related neighbourhood regions (Dastane, Rao, Shenoy, and Vyavaharkar, 2018) – a sudden colour shift may indicate two different objects, whereas a slow and steady shift in luminosity points towards shading of a single object (Fei-Fei and Li, 2010). There is a vast amount of information that can be predicted in this way, and less data available does not necessarily reduce this by a quantifiable amount: it only generalises what is available.

Of course, the Cifar-10 dataset has less practical use comparative to other datasets; after training the network, any realistic real-world image is not likely to be composed of such a small resolution. However, where this dataset shines is in its practicality of testing and configurating a network from the beginning.

Becaues of this, many researchers use the Cifar-10 dataset to construct, train, and test the architecture of a neural network, then transition to a more computationally heavy dataset (Devries and Taylor, 2017; Recht, Roelofs, Schmidt, and Shankar, 2019; Cubuk et Al, 2018) . This process massively reduces the amount of time required to design a neural network, as where the Cifar-10 dataset has an aforementioned 3072 elements, and as evidenced earlier: large images increase the need for more neurons exponentially.

Computational and hardware limitations are the main reason for development to be undertaken using the Cifar-10 dataset. Due to the time constraints put in place by the related project plan and in general, it is improbable that a convolutional neural network could be designed and trained within this time period without sacrificing quality, efficiency, or reliability.

If more computationally powerful hardware were more readily available, other datasets would become preferable due to an increase in semantic detail and a larger number of classes available. Both of which would likely lead to a network that could better recognize patterns through semantic detail, and thus become more effective at colourization. Despite this, for the reasons laid out in this section and others, the use case of the Cifar-10 dataset within this project has been justified for essential reasons of efficiency and speed.

3.2.4. The Flickr Faces Dataset

Although a success in using the Cifar-10 dataset is a success regardless, it would be unwise to not test the artefact on different – and higher resolution – datasets. To fully test colourisation efficiency, the network architecture has been left unmodified (aside from the input layer and feature concatenation) and trained to colourise images from the Flickr-Faces-HQ dataset. Doing so validates the networks ability to generalize across different resolutions and provides a greater visual representation of colourisation in action – some images from the Cifar-10 dataset are difficult to interpret simply due to the lack of detail. Using this dataset gives a glance into the network as a real world application and provides a clearer view as to evaluate success.

The Flickr Faces dataset is an image dataset composed entirely of high-definition human faces (Figure 16). It contains no labelling data, originally prepared for use with human face generation with GANs (Karas, Laine, and Aila, 2019). It provides an interesting parallel to evaluate different characteristics of images and how they impact colourisation results. The higher resolution provides more semantic, context-based information, and the lack of object classes reduces overall complexity. Although as said, these images have been scaled down so they can be used on the limited resources available, and only a minute amount have actually been used. The dataset itself contains faces with a wide range of characteristics, including different ages, ethnicities, lighting conditions, and positions, providing a wide berth with which to train the network with.

Figure 16: Examples of unmodified faces from the Flickr Faces dataset (Karras, 2019).

3.3. Research Methods

3.5.1. The Data in Question

In spite of this though, this project being visual in nature, human perception and bias plays a considerable role in its success. In nearly every circumstance outside of the extreme, some colourisation to a good level of accuracy is an improvement over pure, monochrome grey. Take, for example, a flowerbed of tulips in greyscale. It actuality, it is currently genetically impossible (Noda, 2018) for tulips to be entirely blue, yet if the flowerbed were colourised as such, an argument could be made for the validity of the colourisation. Ire will only be drawn to such results if the user is especially knowledgeable about tulips or the actual image. Just as such, a red car may be misinterpreted as a blue car, yet the success of the output can still be justified.

Of course, this argument has it’s limit: it is incredibly unlikely that a tree would be configured of a rainbow bark with orange leaves, or that the moon would be purple. Such unlikely cases would only have merit under the guise of artistic value, which leads to a multitude of different opinions that become near impossible to define, not at least debate. If this artefact mistakenly produces such colourisation, then it’s success becomes limited to the generation of images that can be labelled as ‘art’ which is of course entirely subjective, and fails to meet the standards and goals set out in the project proposal.

In this way, the required data is a mixture of objective and subjective, as depending on the overall success of the project. Obviously, the primary aim of iterating over network designs should be to reduce loss and increase accuracy because these metrics almost always indicate overall measures of success. The lower these values, the greater the network can be evaluated on an objective level. If though, these values are low, and still the output looks realistic – not because it belies reality or expectations, but because it exceeds them in a way not measurable by exactness to the expected then it will have to be measured subjectively. In effect, it is not just these two metrics that the network is trying to defeat, but also any person looking at the output.

To expand upon this; primarily, results from the network can be considered as interval data (so as to compare ground truth versus predicted results) on a purely numerical basis. When analysed by a human however, the data can be considered as both nominal and ordinal. Nominal analysis of the network output would be as simple as asking test users ‘which of these images is coloured artificially: A or B?”, whereas ordinally, the question could be phrased instead as “looking at this image, how would you rate the colourisation attempt out of 10?”.

For classification results, measuring the data is a much simpler task as it is quite literally a measure of closeness. In a machine learning algorithm (such as a CNN), classification results are often quantised as a measure of closeness from one category to another (in binary classification, where there can only be one of two outputs, this is not needed). The same quantisation has been applied to classification results in this project, and as such it is considered nominal data when rounded to its closest category, and interval as a measure of general accuracy and loss.

3.3.2. Qualitive and Quantitative Research

Being technically arrays of numerical data, an evaluation of predicted outputs versus ground truth images is a measure of numerical closeness, making it subject to quantitative research.

To this end, careful consideration must be made between the balance between qualitive and quantitative ways of evaluating this artefact, and consequently a line must be drawn between the two. Obviously, the higher the accuracy; the better. Similarly, the lower the loss; the better. Objective data such as this can be considered empirical in this regard, though to the limits as those discussed earlier.

On the other hand, bias plays a large role in the subjective success of this artefact, which must be accounted for in so far that it is equally as important as objective numerical closeness. To assure this, a measure of qualitive and quantitative research methods have been used to create a balanced justification of success. Prior methods of loss and accuracy will be examined, whilst partiality shall be reviewed over human trials, whereupon they shall appraise an output versus the ground truth image to determine overall success and validity of the output. These methods are not required for classification data, as it is not vulnerable to human bias or empirical judgement.

4. Design, Development, and Evaluation

4.1. Requirements

The primary requirement of this project is simple in description: to realistically and agreeably construct colour predicted channels based upon input monochrome images. Accuracy is important in this regard, but the accuracy of image colourisation is often based upon individual opinion.

Moreover, it is important that the network does not take a significant amount of time to process input images: with one image, or a select few, a long processing time is not very significant. With a large dataset of images however, expending an increasing amount of time on image processing would become quickly impactful. But, despite this, the actual success of this project reckons primarily upon colourisation success, and also in classification success.

As a measure of the networks capability in classification, it shall also be tested using the Cifar-10 dataset and their linked labels. If the network can correctly classify images at a high level of accuracy and distinction then it is able to extrapolate common features that distinguish different image classes, which likely extends to a greater probability of colourisation success. With a high success rate, these features can be sent to an expanded colouriser to give basis to colourisation predictions.

4.1.1. Accuracy and Loss

For the analysis of accuracy in neural networks, there exist a number of different metrics for analysis of accuracy within different requirements

Within this problem space, any accuracy over 50% indicates some level of overall image enhancement (Zhang et Al, 2016). Any value under this shows a likelihood that the opposite is true, and the output image has been badly impacted by the network's attempt at unsupervised colourisation. To that end, accuracy is a good indicator of success within this project.

As stated before, however, the Cifar-10 data is of a smaller resolution, and only a minute number of images from the Flickr Faces dataset are used. Accordingly less semantic information is available, which can make high levels of accuracy difficult to obtain. What may be a more important measure of success is loss. Where accuracy is a measure of exactness, loss is a measure of closeness. It is more important for the network to produce results that are justifiably correct rather than ones which are exact to the letter.

4.1.2. Human Bias

With some extrapolation, 'accuracy' can be considered not just a performance variable, but the rate at which the network successfully fools a target audience into thinking that its output colourisation is the original image, or that the ground truth image is a falsehood. Accuracy in this regard is then dependent on specific colourisation in the image, which may not be wilfully compared with accuracy. For example, an image with heavy brown/sepia tones seeping into the image has been shown to have relatively high levels of accuracy (above 64%); similar to an image that colourises some areas with suitably vivid and vague sections at varying rates of authenticity.

In this case, human perception and bias will lead them to believe that the brown/sepia images are false, as intuition and a known relationship between dated photographs and a lack of colour (Gómez and Meyer, 2012) impacts the viewers opinion of the image. If that particular style of output image were to be produced and shown alongside a much. Any colourisations produced such as this can be discounted as largely unsuccessful. To that end, emphasis will be placed upon vivid imagery and colourisation over blanket scheme interpretation, even if it comes at the cost to some measure of true accuracy and loss.

Similarly, there is to be much consideration placed upon success if pixel colourisation is partly regionally correct in an image, placing far greater emphasis on some image regions over others. In some cases, such as face colourisation, this is an insignificant issue as long as those required regions are properly colourised. The faces are more important that the background. When trained on a multitude of different possible image classes however, it is more likely that other regions will be improperly colourised, so the efficacy of that solution must be evaluated, or at least considered, on a case-by-case basis.

4.2 Design

Given to the software development structure detailed in the section prior, the final network designed has been designed based upon the successes and failings of different designs before it. Recurrently, every network has slowly been made more complex and deeper to find a suitable balance between training time and accuracy. Rather than simply adding extra layers to networks, making them deeper, experimentation with increasing other dimensions of networks has also been tested.

After consecutive changes and improvement to network designs, a U-net style autoencoder has been constructed that operates identically with both the Cifar-10 dataset and the Flickr Faces dataset, both without features (Figure 17) and including features from the Cifar-10 dataset (Figure 18). A similar, yet slightly older version of the same network has been used to generate classification results, although it has not been updated to the most recent architecture. The network has been trained using the Adam optimizer. Nearly every layer uses the LeakyReLU activaiton function, as it is shown to have great results in many neural networks, and with the Cifar-10 dataset (Xu, Wang, Wang, and Li, 2015). The only few layers that do not use this activation function are the two final layers: the penultimate layer uses the ‘softmax’ function – to derive probability, and the final layer uses the ‘tanh’ function – to derive a*b* values in a normalized range. Dropout and batch normalization are used after every layer except the last. Networks using Cifar-10 data use batches of 64, whereas the Flickr Faces dataset uses batches of 4.

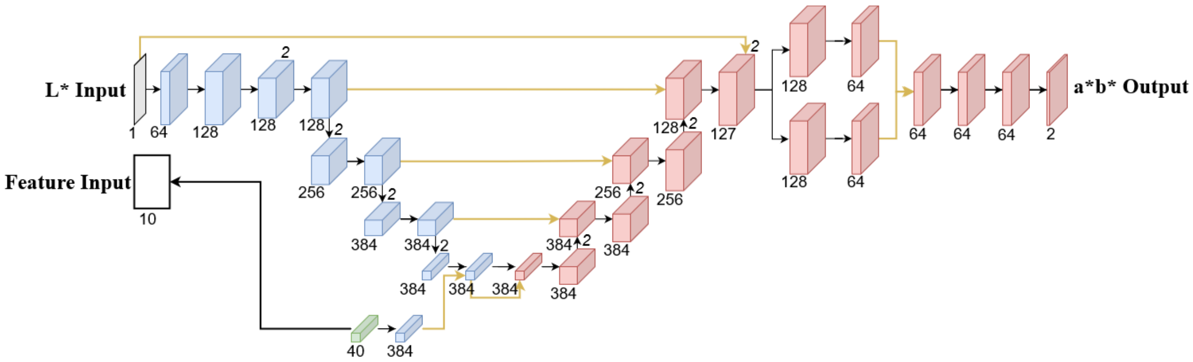


Figure 17: The proposed network with feature inclusion (see Figure 20 for the key, and Figure 19 for layer example)

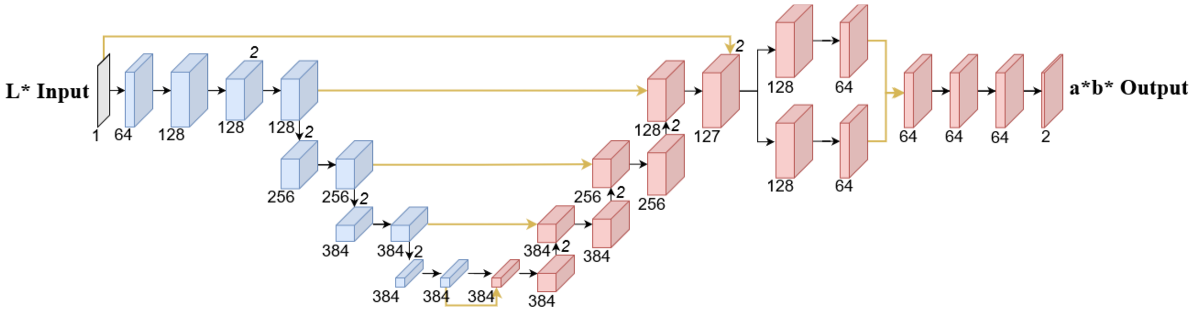


Figure 18: The proposed network without feature inclusion (see Figure 20 for the key, and Figure 19 for layer example)

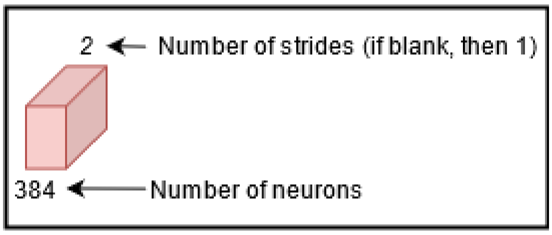
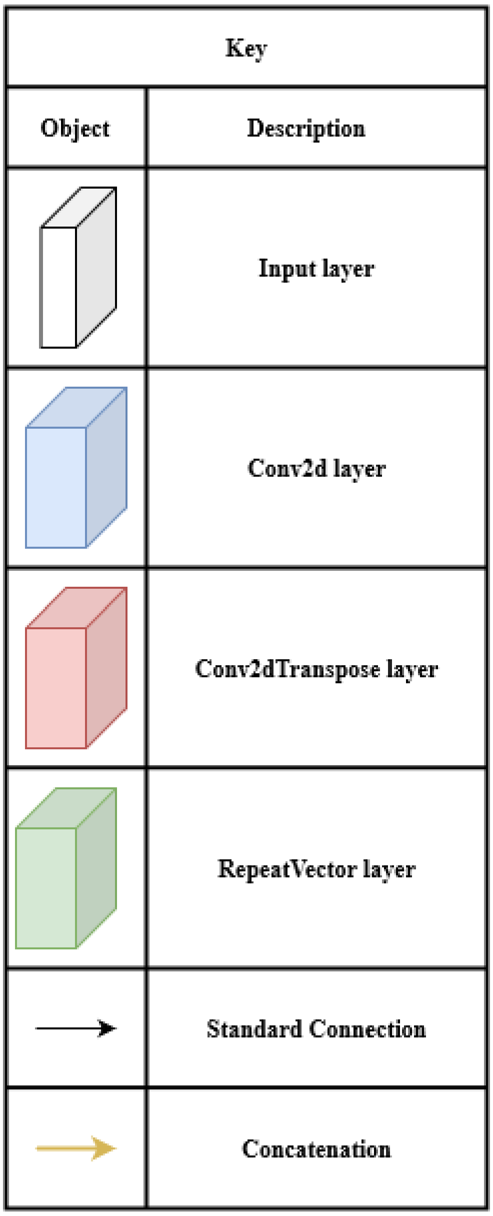


Figure 19: Descriptor of a layer and linked attributes within the network. Any layer denoted with ‘2’ has a stride of (1, 1), whereas every other layer has a stride of (3, 3).



Key	
Object	**Description**
□	**Input layer**
🟦	**Conv2d layer**
🟥	**Conv2dTranspose layer**
🟩	**RepeatVector layer**
→	**Standard Connection**
→	**Concatenation**

Figure 20: The key for visualisation of the proposed network. Note that layers are merged by concatenation after both layers have finished computation, but before the second layer has sent an output to the next layer in the network.

4.3 Methods

Firstly, as the images for both datasets are stored as RGB NumPy arrays, the procedure for colour conversion (as detailed in section 2.3) was implemented to not only convert RGB values to L*A*B* values, but also to split the converted array into two: one for the L* channel, and one for A*B* channels. This creates the new training/test data for colourisation as a singular task. Existing training/testing data for labelled information was kept in wholly different arrays, so as to test the effectiveness of classification for outside feature collection on colourisation.

Arranging images channels in this manner conforms the arrays to the requirements needed for Keras training; those being stored in arrays in the image format of [data index, height, width, colour]. The dimension of ‘data index’ is needed for Keras training to work, and it can be used to reference any particular image and retrieve it from an array with ease.

Specifications of image resolution is, as expected, dependent on the dataset in use. Stored in an array, the values for said resolution make no difference to image appearance other than size: all visual detail is denoted by the colour channels, of which there are three by default (RGB). Any outside greyscale images processed for colourisation also have three colour channels, except that each channel is exactly the same (e.g 100, 100, 100 in RGB format is a shade of grey). When converted to L*A*B* format however, outside images go through the same procedure as default dataset images, except the output for the process results in only the L* array, as there is no ground truth identifier of A*B* channels.

As detailed in section 4.1, metrics of accuracy and loss are accounted for on a basis of pure mathematical closeness between predicted and ground truth values. There is no need for outside measures to be developed to measure this, as Keras includes ways to easily evaluate these metrics during and after training. Measuring human bias is a different matter entirely, with, ironically, innate human bias towards results evaluating human bias. Therefore, it is important that bias is as far removed from results screening as possible.

To get an approximation of human opinion on colourisation results, a series of questionnaires and interviews have been developed for the results of the network. Individually, results are shown to people on an online form, which reduces the chances of cross-contamination of opinions (group mentality indicates that if one person holds a certain opinion, a different person is more likely to develop the same opinion). Overall, 8 people of different background have been questioned for these purposes, ensuring that human perception of the success of this project remains as far prejudiced as is possible.

4.4 Results

Examples of colourisation results from both datasets are displayed below to showcase the final results of colourisation. These images are a combination of the first twenty of each dataset, so as to remove any bias from selection of best-case scenario images. In addition, accuracy and loss metrics are provided for each dataset; for colourisation and classification (Figures 21, 22, and 27

4.4.1 Cifar-10 Results

Classification

Data	**Accuracy**	**Loss**
Training	85.08%	0.0214
Validation	85.84%	0.0209
Test	84.55%	0.0229

Figure 21: Classification metrics for training, validation, and test sets of the Cifar-10 dataset.

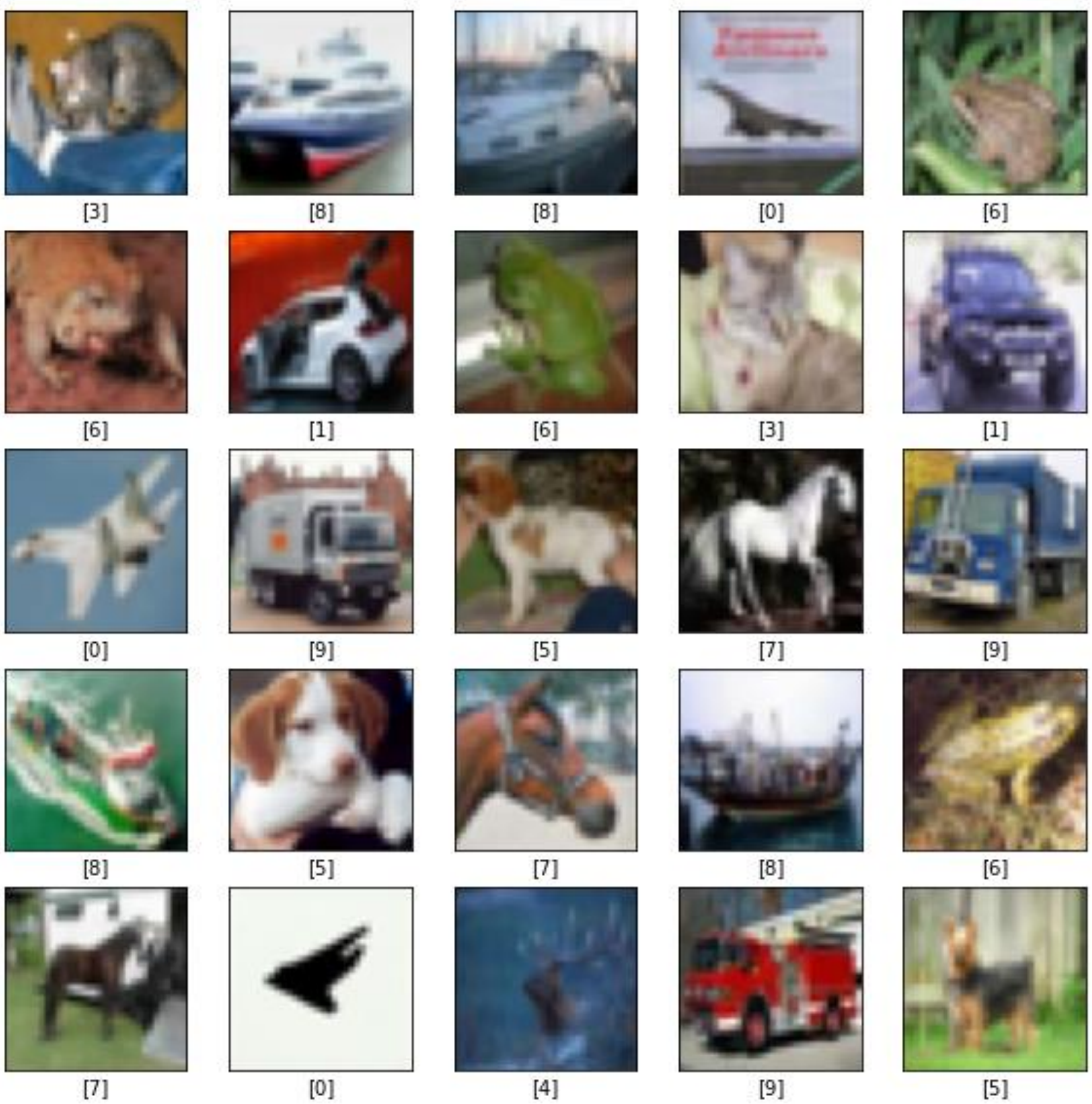


Figure 22: The first 25 Cifar-10 images in the testing set alongside ground truth classes.

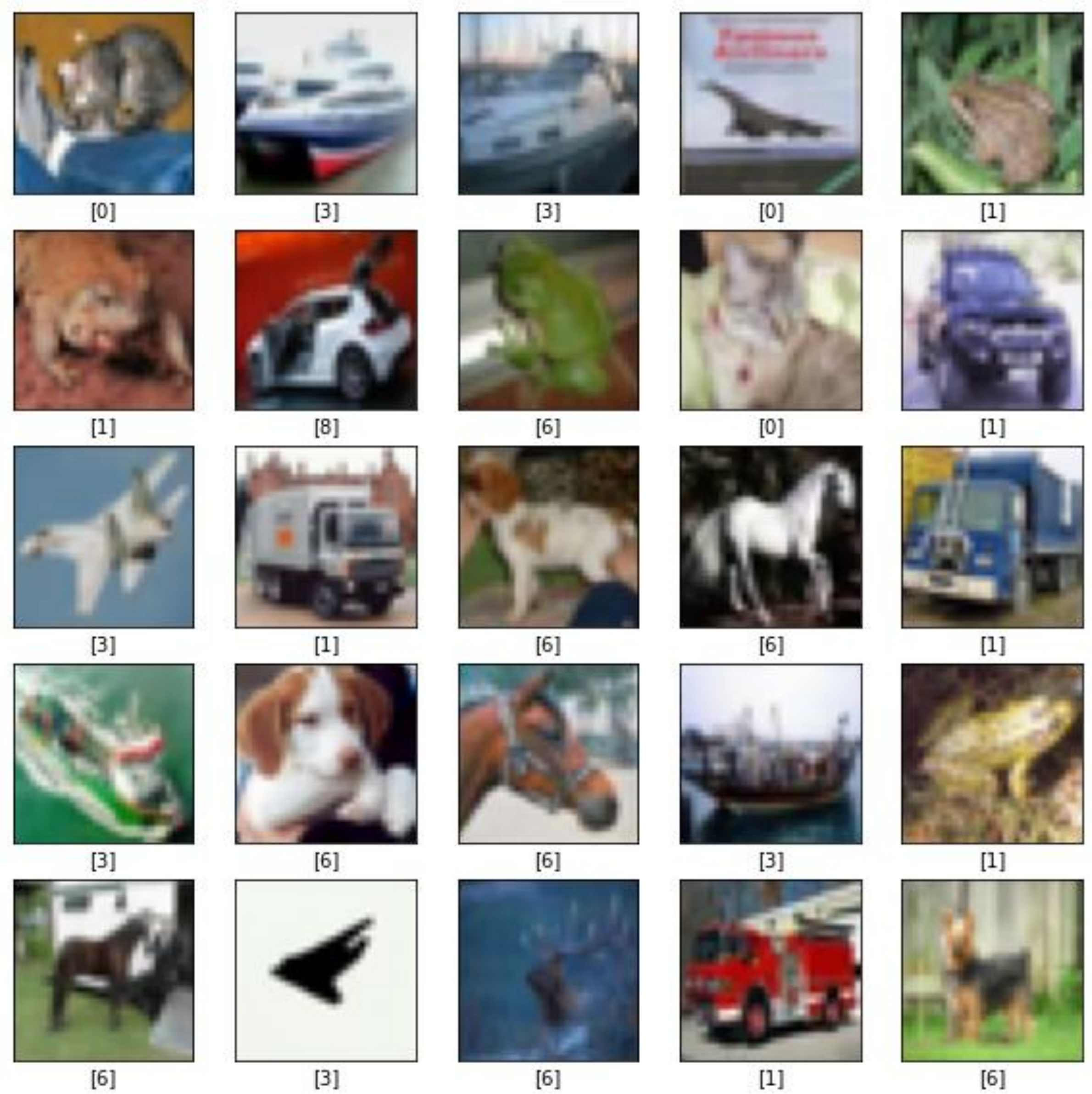


Figure 23: The first 25 Cifar-10 images in the testing set alongside predicted classes.

Colourisation

Test Set	**Accuracy**	**Loss**
With features	65.33%	0.084
Without features	69.03%	0.082

Figure 24: Details colourisation metrics for networks with and without features.

Displayed below is - in order of appearance – input greyscale images for the Cifar-10 dataset (Figure 25) and Flickr Faces dataset (Figure 30), the ground truth for both datesets (Figure 26 and Figure 31), and predicted colourisations (Figure 27, Figure 28, and Figure 32).

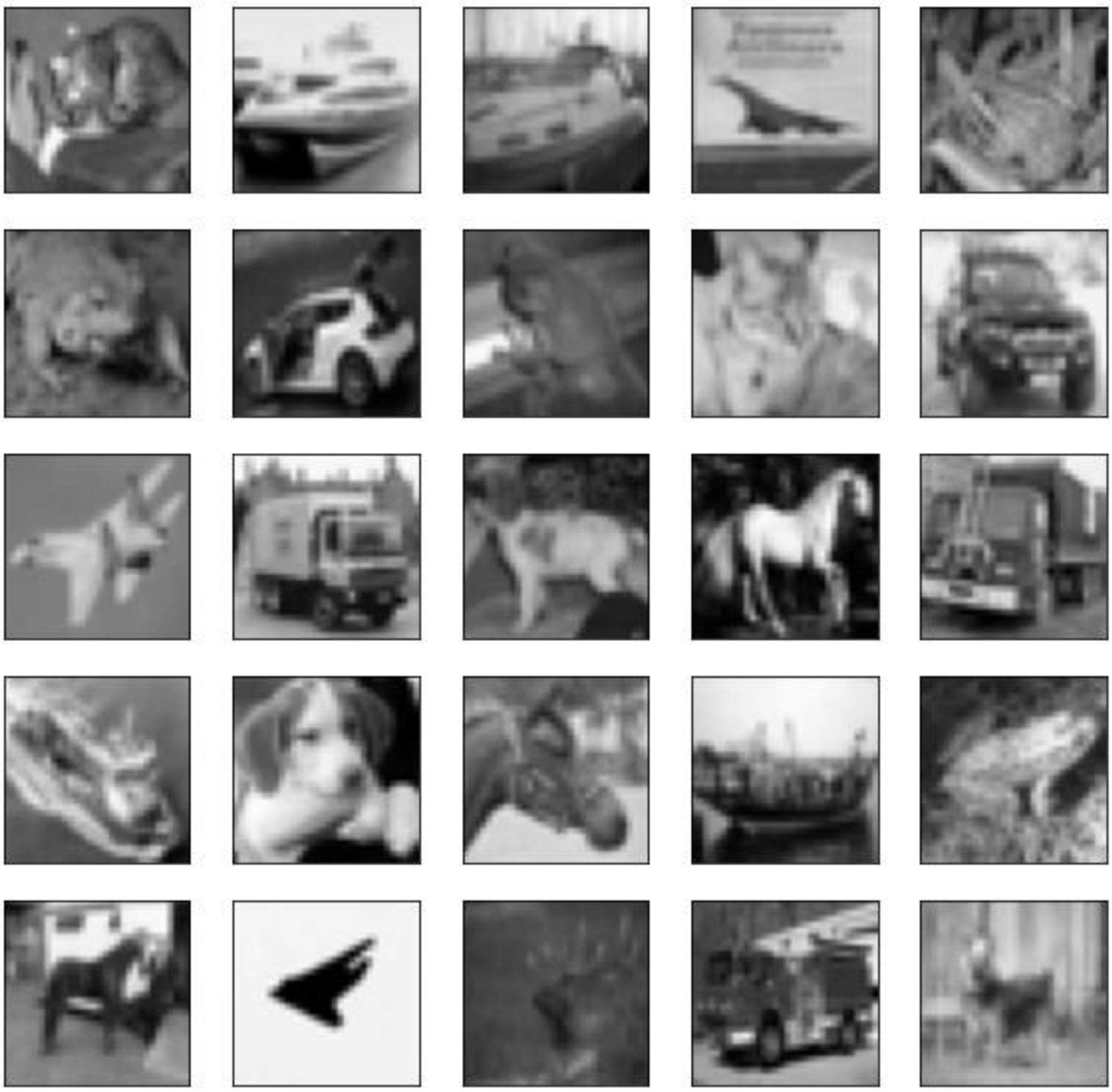


Figure 25: Greyscale Cifar-10 images.

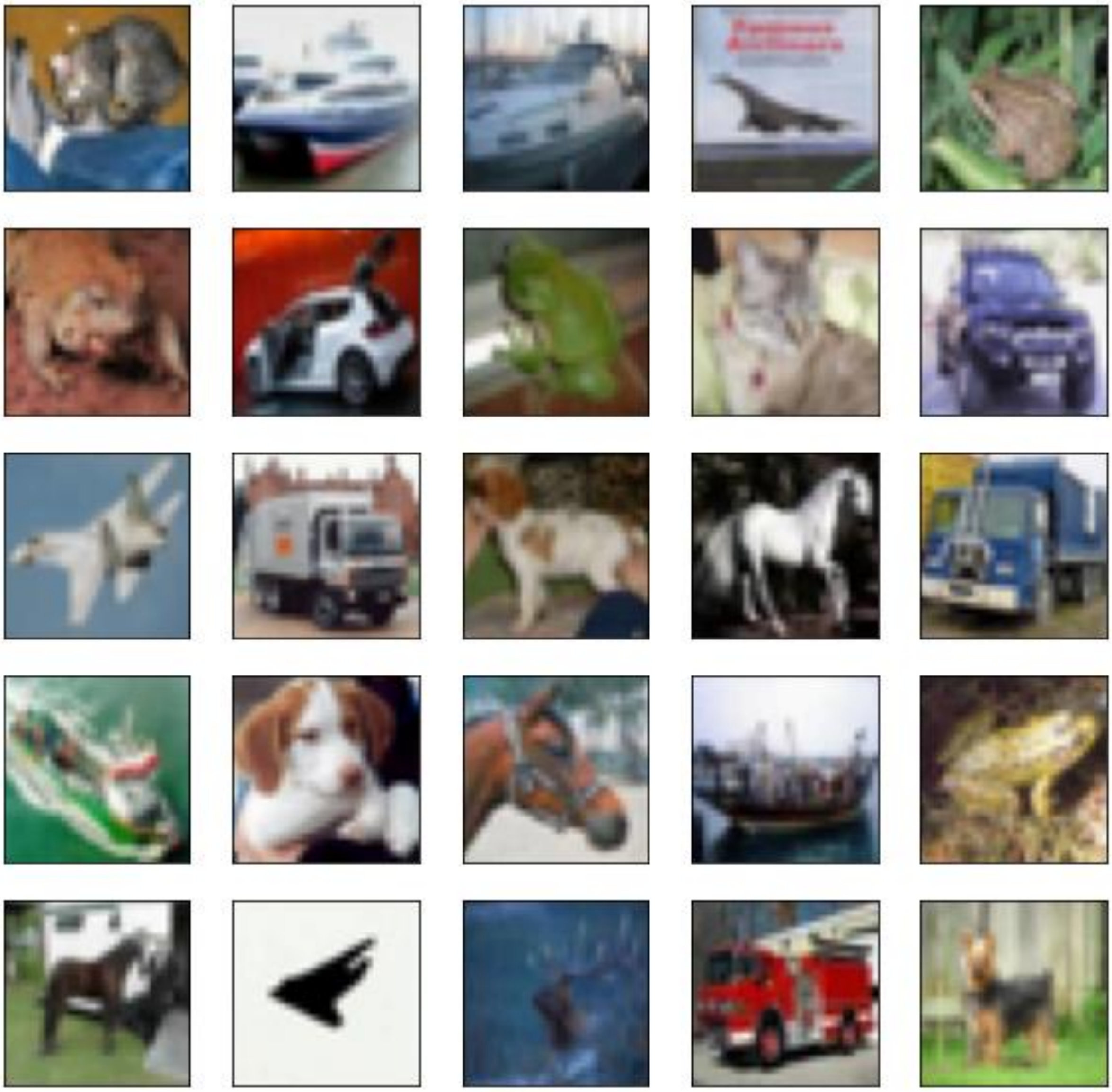


Figure 26: Ground truth Cifar-10 images.

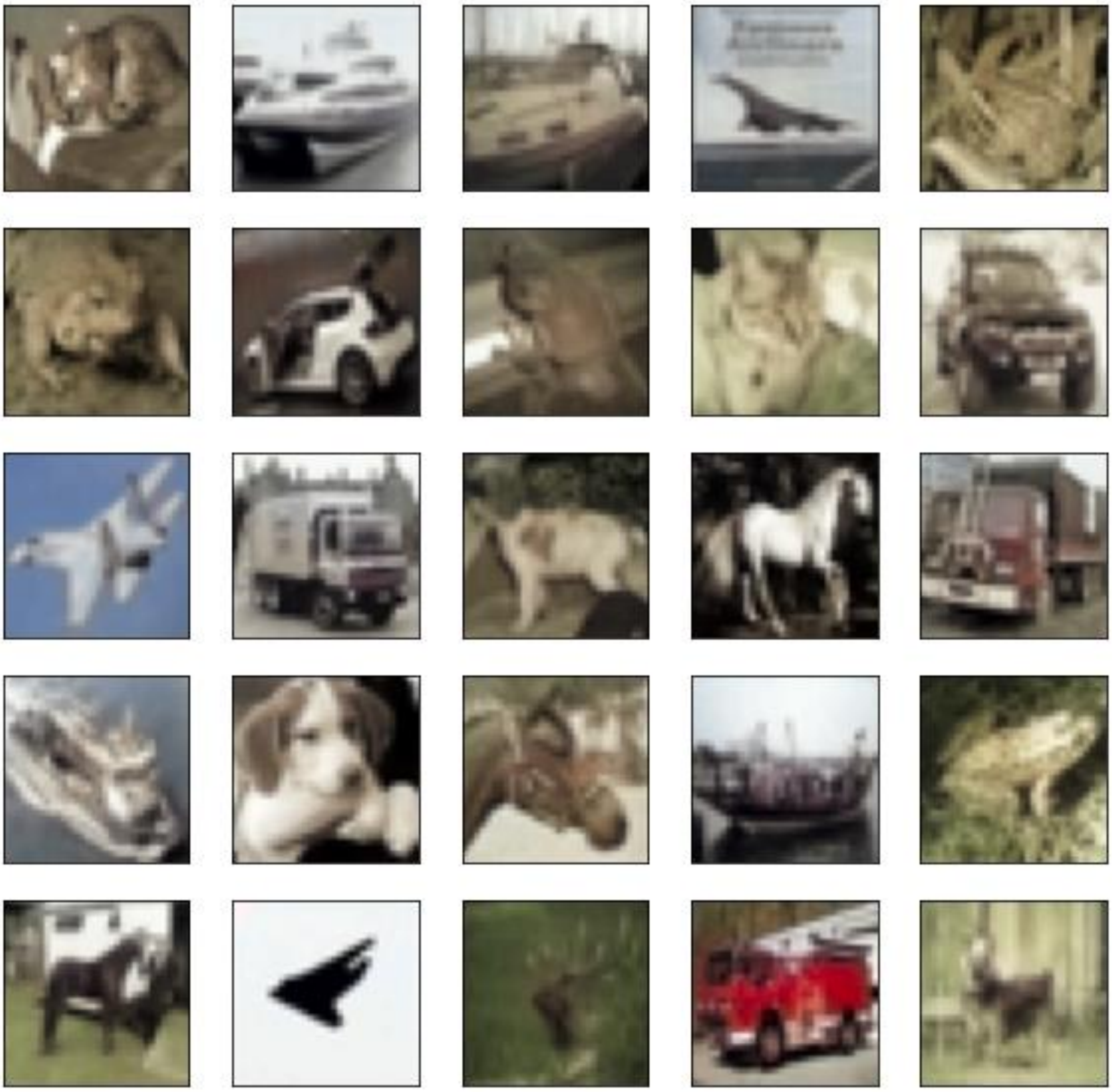


Figure 27: Cifar-10 images colourised by the network (with features).

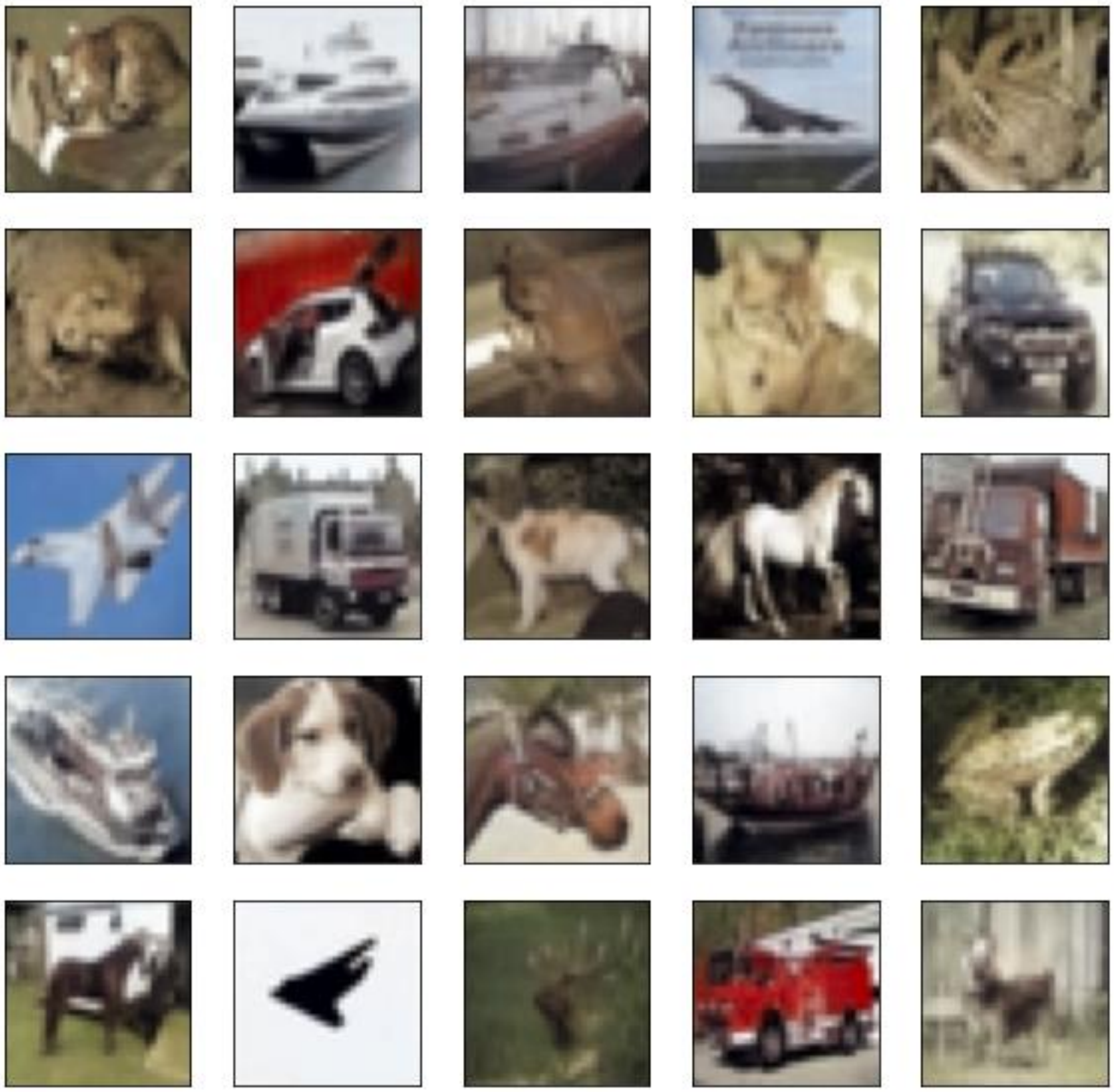


Figure 28: Cifar-10 images colourised by the network (without features).

These images demonstrate the visual success of the network in actuality. Note that some images contain brown overtones, and most in these examples have had different areas colourised successfully. For example, colourisations with animals differentiate between background and foreground elements by applying colours such as green to grass and brown to the animal itself. Interesting parallels between colourisations with and without features are obvious in some cases and oblivious in others. Note that network without features has been trained on more epochs, which likely imparts some detriment to colourisation quality.

4.4.2 Flickr Faces Results

	Accuracy	Loss
Test Set	60.06%	0.087

Figure 29: Details colourisation metrics for the Flickr Faces dataset.

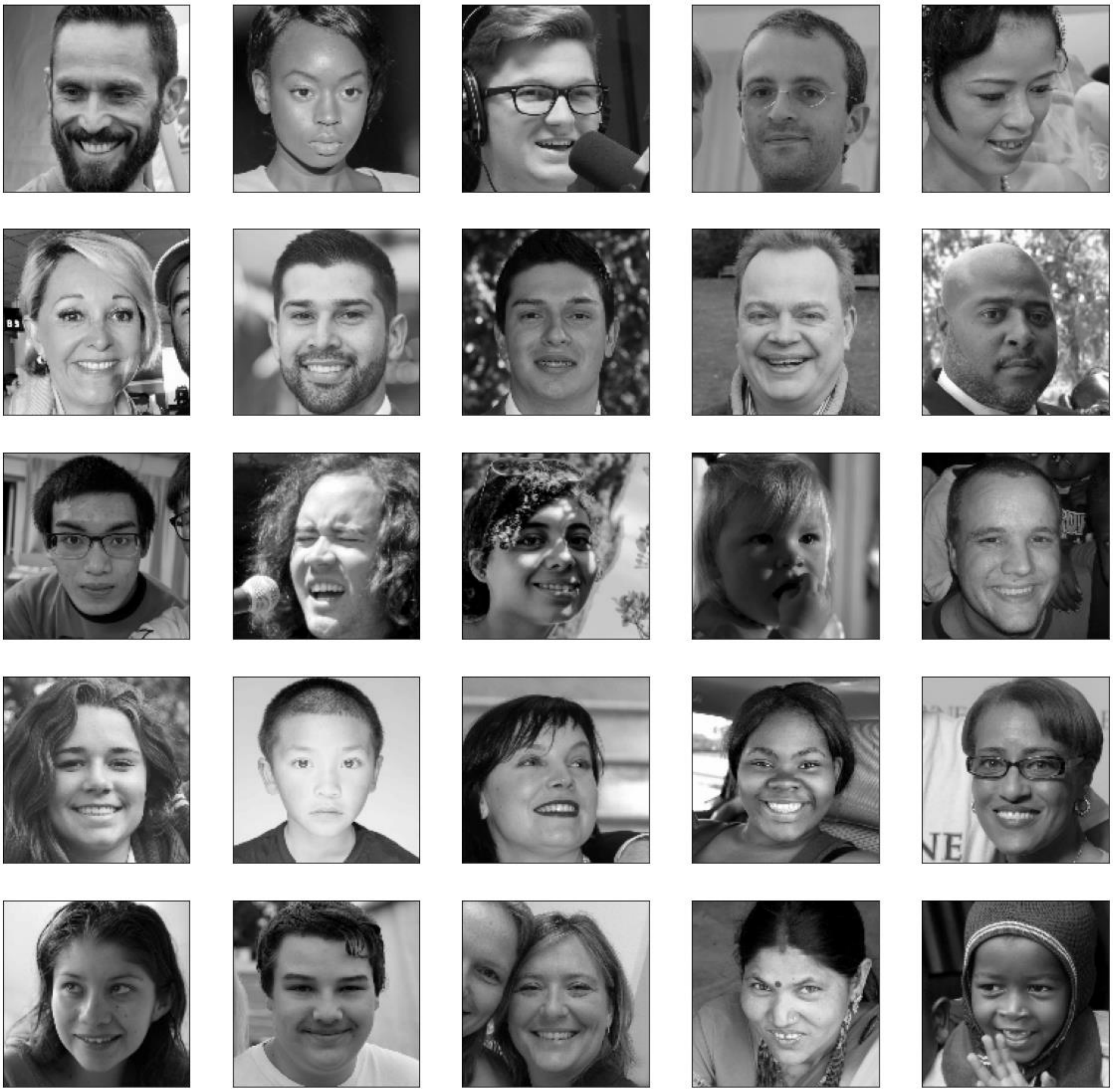


Figure 30: Greyscale Flickr Faces images.

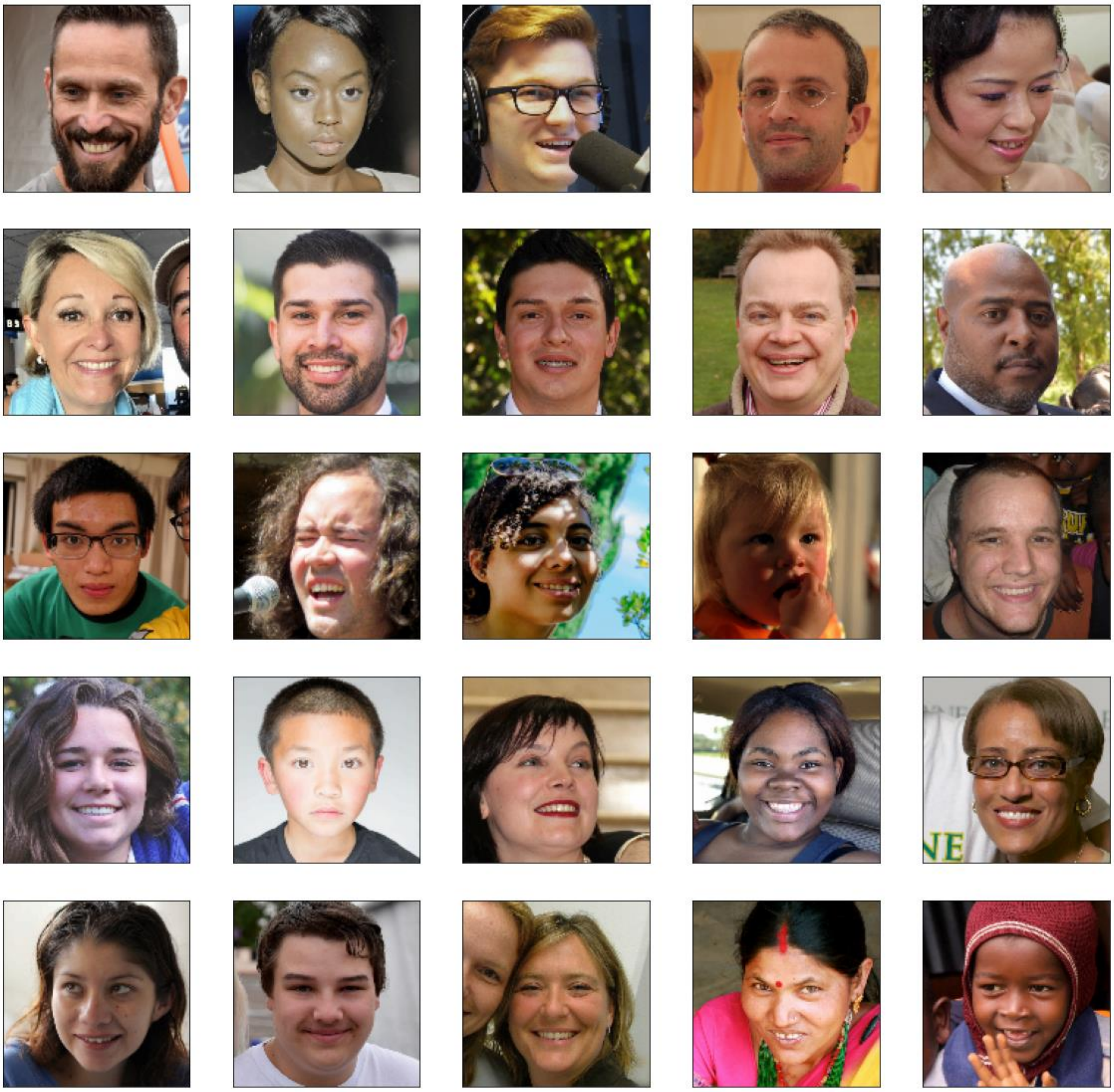


Figure 31: Ground truth Flickr Faces images.

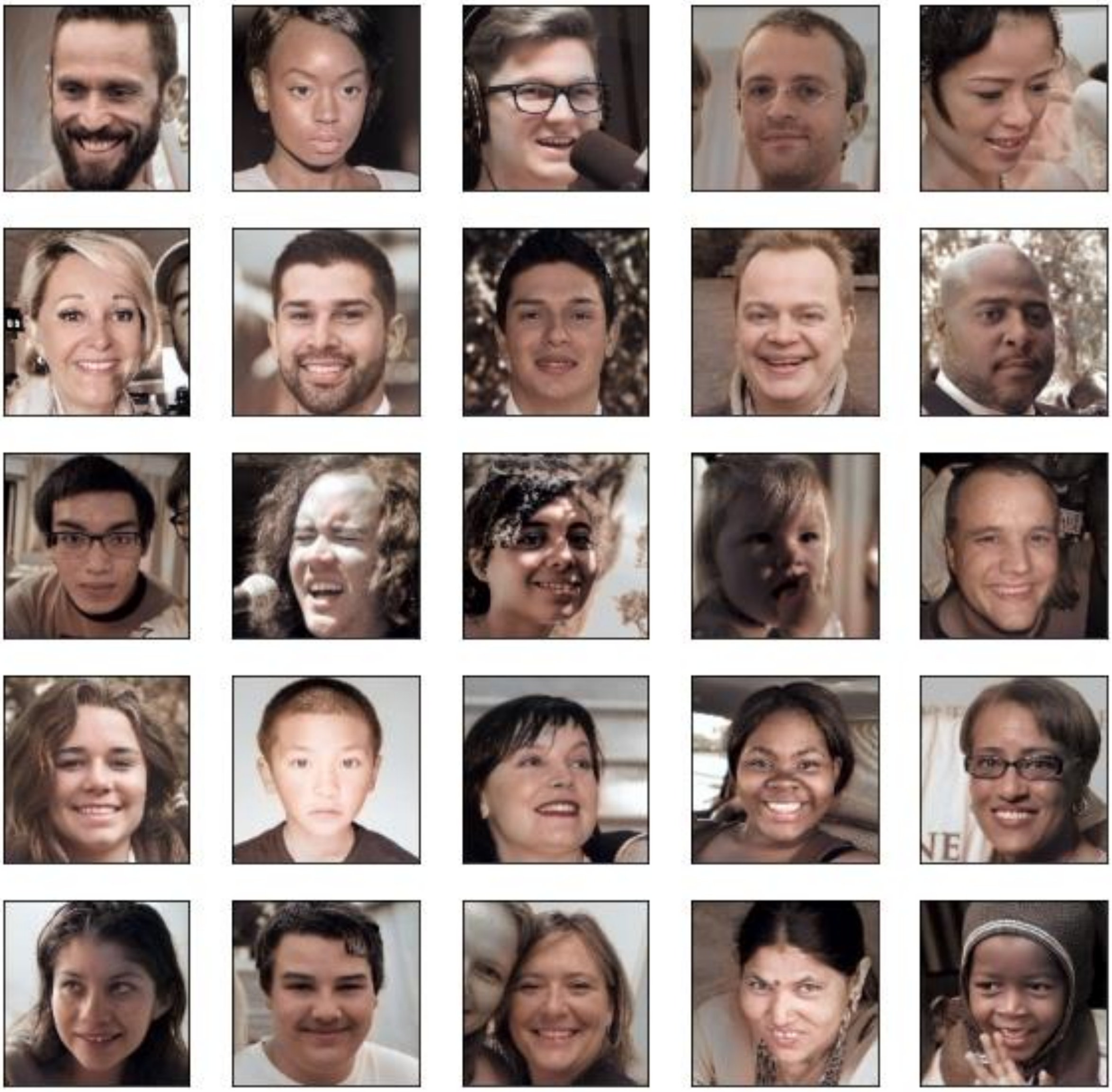


Figure 32: Flickr Faces images colourised by the network.

Results of Flickr Faces colourisations lack background colour, but effectively colourise and shade human faces nearly perfectly.

4.4.3 Unrelated Results

In addition to results from actual datasets, two other images have also been colourised using the network, and the weights learnt from training on the Flickr Faces dataset. Both are greyscale images, dated considerably, and thus contain image noise as expected of camera technology at the time, adding additional challenge to successful colourisation. All images were scaled down to fit the required input size of the network.

Greyscale image	Image colourised with network

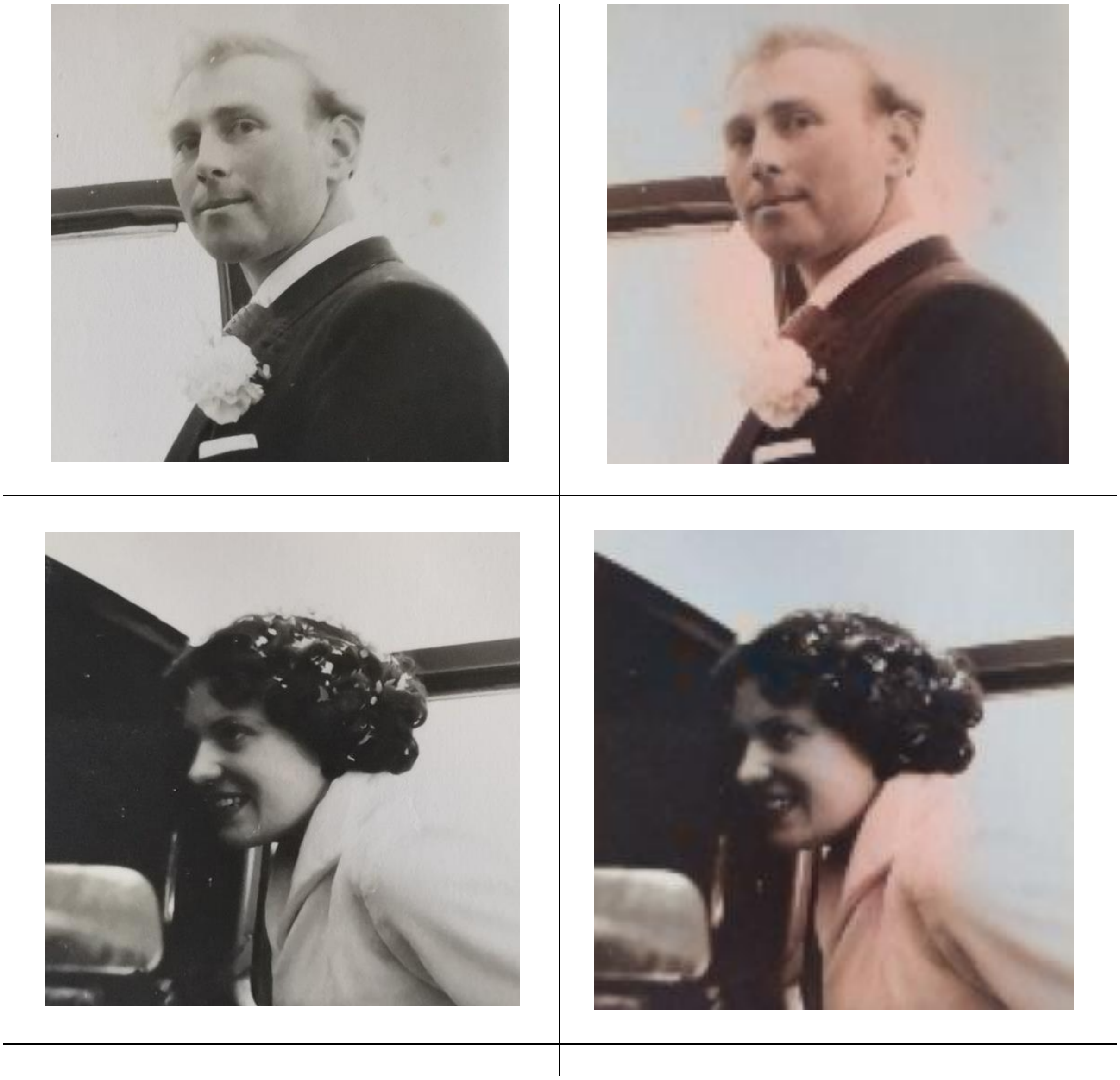


Figure 33: Unrelated images colourised by the network.

4.5 Discussion

4.5.1 Classification

Though the primary goal of this project was to produce hypothetically agreeable colourisations of greyscale image, there is some discussion on the networks results in the field of classification initially.

Overall, the network has been proved to be effective in both classification and colourisation attempts. Modern benchmarks for Cifar-10 classification accuracy often encroach upon near perfect (100%) accuracy (Foret, Kleiner, Mobahi, and Neyshabur, 2020; Kwom, Kim, Park, and Joi, 2021; Harris et Al, 2020), but these networks often use a staggering number of neurons in comparison, or use incredibly specialised and advanced methods to reach such lofty heights. Notably, one attempt from 2020 by Dosovitskiy et Al received an incredible 99.5% accuracy, but used over 600 million neurons (Dosovtskiy et Al, 2020). The difficulty therein is mostly dependent on an exponential challenge with increasing accuracy.

Potentially, the classification results for this network architecture could continue to increase, as the training, validation, and test accuracy are all roughly around the same levels accuracy and loss. This is possibly evidence of underfitting (which would be impressive with a high level of accuracy regardless), which means that results could improve further with more training or with a greater number of hidden layers (Chauhan, Ghanshala, and Joshi, 2018) .

4.5.2 Colourisation

Cifar-10

Cifar-10 results, many predicted AB channels are evocative and of bright colour compositions. Though they are not always of the correct tone or shade, the network is able to effectively colourise many of these images to a high degree of success. It appears that the network generalises to common areas and image regions (either through image composition or luminosity values in pixel neighbourhood). In the Cifar-10 dataset specifically, the network appears to use red as a solution to confusing objects, and brown as a solution to confusing scenes.

Perhaps it could be said that the network is not exactly tailored for colourising the Cifar-10 dataset. The necessity of using said dataset has already been stated, but the network uses an incredibly potent autoencoder framework which has a greater chance of damaging low-resolution image composition. With the strides in place, the network halves the dimensions of an input four times. This reduces the size of any Cifar-10 image from 32 * 32 to 2 * 2. No matter the power of the decoder, it is probable that reducing the image to such a small resolution is likely to have some negative effects. However, such image degradation is not evident in classification results, although colourisation is a much tricker task to finesse, especially when completely unsupervised (Chakraborty, 2019).

Of course, the network is shown to be a far stride from perfect. Incorrect colourisations are not common, but they are apparent. The factor of slight brown/sepia undertones can likely be attributed to the aforementioned negative with using loss metrics in a task such as this. In addition, the network generalises somewhat too much: it believes that nearly every land vehicle should be a shade of red, for example. This is much more apparent in the network that incorporated learnt features, but said network was trained on substantially fewer epochs. More training is required to fully explore the potentials of this incorporation, although test metrics did seem to reach the same block as with the other network. Perhaps feature incorporation via class labels is more befitting to a dataset with a greater abundance of classes.

Once again, the cause of this could be boiled down to simple representation of the abundance of specific or popular characteristics of classes – red is one of the most common colours for vehicles (reference). However, to reiterate a point made earlier: incorrect yet plausible colourisation is better than suspicious colourisation. The fact that a car is red is not suspicious or implausible, and if used in a real-world scenario as-is it would take prior knowledge for a user to discern that the colourisation was incorrect.

Flickr Faces

Trained on the Flickr Faces dataset, the network is able to highlight and summarily colourise faces from the dataset, though it has difficulty colourising surrounding regions, such as the environment. In addition, it does fail to tell apart warm and cool tones of skin colour; compartmentalizing every face as warm toned. This is likely due to small number of training samples giving the network a poor representation of background elements, and thus information in general (Bruni and Bianchi, 2015).

In addition, faces colourised are roughly of the same colour, though in varying shades. Specific details, such as eye colour, are typically generalized to the highest common denominator. In this case, nearly every image is colourised with brown eyes, as those are statistically the most common eye colour for humans (Moyer, M.D, 2019). Random sampling notwithstanding, this statistic is likely represented within the Flickr Faces dataset.

In a vein similar to this, the Flickr Faces dataset is colourised to quite a low accuracy, yet this is mostly because of incorrect colourisation to background elements. The fact that the actual faces are nearly perfectly colourised every time highlights the importance of bias yet again: if someone is colourising a photograph of a person, the element they likely want colourised most effectively is that person; everything else is secondary. This is marginally true in the unrelated images colourised using the same weights: it appears that the visual noise and odd angles result in lacklustre colourisations.

In General
Throughout colourisation the network is shown to effectively locate specific regions within an image and colourise it to an often high degree of satisfaction. Areas are commonly vibrant, or at the very least appropriate for the subject matter. In addition, the network is able to colourise ‘super classes’ (vehicles comprised of ‘trucks’, ‘planes’, ‘ships’, and ‘automobiles, for example) very effectively, and with respect and differentiation to other super classes. Colourisation with the Cifar-10 dataset is especially successful, given for the reasons outlined prior.

Regarding testing methods of evaluating human perception: the test users were shown the colourisations as resulting from above. In short, they were deceived 3 to 5 times by Cifar-10 results, thinking that predicted colourisations were the ground truth, and rated said colourisations at an average of 6.5/10, with 1 being no improvement, and 10 being excellent improvement. Results were less impressive for the Flickr Faces dataset, with no test users being deceived by colourisation attempts, and with an average score of 4/10.

However, despite the high accuracy emitted by the artefact, there are some cases where image colourisations are somewhat saturated or of an overall brown/sepia tone. This is especially evident in the Flickr Faces dataset, but, once again, this can be justified with the small amount of training data. Other attempts at colourisation highlight the fact that the Euclidean distance brought about by loss metrics can cause networks to adopt such an overbearing saturated brown/sepia tone (Blanch, Mrak, Smeaton, and O’Connor, 2019), to serve as a ‘one size fits all’ solution.

This is where the network is at its weakest: the colourisation of elements or image regions less important than the subject matter or class. Simple background elements – such as the sky – are colourised effectively in nearly every picture, but finer detail such as buildings just over the horizon or the yellow sheen brought about by sunlight are often left out. This may simply be due to a lack of representation in both datasets. Using an image from the Cifar-10 dataset, the network will probably decompose specific elements such as those stated prior in lieu of improving the efficacy in general. In a larger, higher resolution image, this problem may see less prominence, as those areas will not be shrunken to completely miniscule levels.

4.5.3 Future Development

In order to improve the network, there are a number of modifications that could be made that would potentially factor into enhanced colourisation. Firstly, more training examples could be provided to give the network a greater depth of field when applying different colourisations (Zhu, Vondrick, Fowlkes, Ramanan). More is almost always better in this case (Howard, 2013). As this argument is general to nearly every ANN and ANN subtype, little needs to be said on the specifics of this matter when applied to this project.

Also, the network could possibly be expanded if the framework for feature extraction and application was also enhanced. As is, labels from the Cifar-10 dataset provide the minimum required for identification. If, for example, these labels were integrated alongside ground truth segmentation then there is a possibility that the network would learn to segment and summarily colourise different elements of an image, and also widen the consistency of possible outputs.

As the network is able to produce rich colours the number of neurons is likely sufficient for each layer. Doubtless, if more layers were added then more neurons should be added, but increasing the number of neurons further exponentially made no general improvement that was noticeable given the number of epochs each network was tested on.

Primarily, the focal point of this improvement should be on improving the efficiency of the network, which is not necessarily done by arbitrarily adding more elements that can be used in computation. Simply increasing the number of neurons and layers made little improvement to colourisation attempts (up to approximately 30 million neurons). Many recent advances use GANs to bridge the gap between plausible and implausible colourisations in difficult scenarios, such as those discussed within the similar work section.

Finally, as this artefact essentially tries to emulate part of the visual human system, perhaps there is a need for either a much more complicated network, or for a series of interlinked networks working together. Colourisations exist in a range that is unprecedented and could be unknown to a machine learning algorithm trained to colourise things in a certain way. Consider data where the training data only included yellow bikes; every bike the network encountered would be colourised yellow. What is needed to expand upon this network is not just a larger number of weights and layers, mimicking rods and cones within the human visual system; but perhaps a semblance of creativity or fine extrapolation. In future, this network could be used for the framework of a GAN, which would allow the network to emulate the required creative essence (Mazzone and Elgammal, 2019).

Earlier in this report, a brief summary on how ANNs were inspired by neural networks in the brain was explained. As said, these neural networks form larger brain groups; one of them being the human visual system. Although ANNs began to shift away from their biological inspiration after some time (Eluyode and Akomolafe, 2013), the goal of this problem space is, as said, not truly empirical. As such, it is a problem space that requires a solution much more inspired by the human visual system, because it is trying to achieve something astonishingly similar. More research is required to explore this avenue of expansion, such that cannot be sufficiently done within this report as it exists in a scientific field deserving of its own specialist care.

5 Project Conclusion

In conclusion, this report provides a summary on how colourisation attempts should be made using a machine learning and, by extension, a CNN style framework. The artefact developed as an example of this is able to effectively colourise greyscale input images, using image features in cases where classes of objects exist. In addition, sufficient background detail into similar attempts made, neural networks, and colour itself has been given for future projects to use as a basis for further improvement. Given the resulting metrics discussed within sections prior, and the exemplar colourisations, ample evidence and detail into the success of these characteristics has been given.

Though some colourisations are incorrect or neutered, segmentation based colourisation is achieved to a high standard nearly every time. In addition, it is demonstrated throughout that the inclusion of pre-ordained image features provides a network with an additional frame of dimensionality with which it can calculate and define colourisations. This improvement of segmentation and colourisation however does bring about a more limited gamut of possible colourisations, as the network standardises inputs from the Cifar-10 dataset to their most common denominator.

6 Reflective Analysis

Overall, the developed network is able to meet the requirements set out by the project proposal, and in turn exceeds expectations through the hypothesis and successive generation of classification predictions, and thus research into feature-based colourisation. Colourisation attempts are close to, match, and sometimes exceed examples given from similar work in this area, and classification results are close to other top results without having specialised a network for such an effort. Together, this makes the network a great success. Whilst it is not to become the grand standard set out for all future CNNs, it still works markedly well given circumstances.

However, a great many difficulties were encountered along the way; the majority of my own design. Mismanagement of networks and saved weights was common, leading to needed reiteration of designs a number of times throughout project completion. In addition, the most egregious error made was the lack of GPU support until a few months into design. This severely cut down the efficiency of training, and the error stems from a simple ignorance that it was already enabled. It was not.

This is not to say that the project was mismanaged entirely, with me spinning plates and waiting for my errors to catch up to me. In fact, training the network was the most time-consuming task, so I spent a great deal of effort into training the network, developing weights, then evaluating network designs given the amount of time trained. There is likely a bias of my own to networks that learn quickly, but this transitions from a simple bias to a near necessity when time and computational constraints are considered. Any difficulties that I encountered were dealt with as best I could, with network faults being the greatest cause of ire. Thanks to preparation and learning throughout, any time a network irreparably failed or glitched, losing progress, there was always a past basis and many frameworks ready to be used. This allowed me to not only repair any errors, but also gave me time to explore into the dimensionality of CNNs, resulting in a greater final project.

In addition, whilst development is expected to be challenging and stressful, these issues were only exacerbated by quarantine practically eliminating many possible avenues of recreation, making it difficult to reduce stress and stay mentally at my best at times where it was most needed. However, whilst everyone’s experience is different, everyone else has nonetheless encountered the same hurdles this year, so little else will be said on the subject.

There is a great deal that can be said about research methods and development of sufficient architecture within this section. Of course, with hindsight comes a greater reference with which to plan, prepare, and execute one’s work. Nonetheless, I should have focused more on researching into common architectures and modern developments – my curiosity got the better of me, and I spent too much time in the beginning experimenting with architectures and designs, just to see how it all worked. Perhaps this was a hidden boon though, as it did give me both confidence into my abilities going forwards, and insight into the intricacies of neural networks.

To conclude this segment, the project delivered is (in my opinion) a great success. Any issues encountered have and were effectively dealt with. In addition, proposal for future enhancement and theorisation into different areas of colourisation has been given and justified, using this research as a basis for such exploration. Future development can thus be done off the back of this work, taking steps to more complicated colourisation methods that could have a greater chance of success, giving evidence to an altogether successful framework and project.

7. References

1. Afifi, M., Abdelhamed, A., Abuolaim, A., Punnappurath, A. and Brown, M.S., 2020. CIE XYZ Net: Unprocessing images for low-level computer vision tasks. *arXiv preprint arXiv:2006.12709*.
2. Akiyoshi Kitaoka, 2018, 30th May. Available at https://twitter.com/AkiyoshiKitaoka/status/1001691875882012672 (Accessed: 27th March 2021)
3. Albawi, S., Mohammed, T.A. and Al-Zawi, S., 2017, August. Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)* (pp. 1-6). Ieee.
4. Al-Jaberi, A.K., Jassim, S.A. and Al-Jawad, N., 2018, May. Colourizing monochrome images. In Mobile Multimedia/Image Processing, Security, and Applications 2018 (Vol. 10668, p. 1066806). International Society for Optics and Photonics.

5. Amara, M., Mandorlo, F., Couderc, R., Gerenton, F. and Lemiti, M., 2018. Temperature and color management of silicon solar cells for building integrated photovoltaic. *EPJ Photovoltaics*, *9*, p.1.
6. Ananthanarayanan, R., Esser, S.K., Simon, H.D. and Modha, D.S., 2009, November. The cat is out of the bag: cortical simulations with 109 neurons, 1013 synapses. In *Proceedings of the conference on high performance computing networking, storage and analysis* (pp. 1-12).
7. Anderson, M., Motta, R., Chandrasekar, S. and Stokes, M., 1996, January. Proposal for a standard default color space for the internet—srgb. In *Color and imaging conference* (Vol. 1996, No. 1, pp. 238-245). Society for Imaging Science and Technology.
8. Appelgren, F., Berggren, J., Båvenstrand, E. and Hahr, O., Evaluation of Image Colourization Approaches.
9. Badrinarayanan, V., Kendall, A. and Cipolla, R., 2017. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, *39*(12), pp.2481-2495.
10. Baggio, D.L., 2012. *Mastering OpenCV with practical computer vision projects*. Packt Publishing Ltd.
11. Bank, D., Koenigstein, N. and Giryes, R., 2020. Autoencoders. *arXiv preprint arXiv:2003.05991*.
12. Bannert, M.M. and Bartels, A., 2013. Decoding the yellow of a gray banana. *Current Biology*, *23*(22), pp.2268-2272.
13. Bhanja, S. and Das, A., 2018. Impact of data normalization on deep neural network for time series forecasting. *arXiv preprint arXiv:1812.05519*.
14. Bhushan, A., Kumar, J.M. and Reshi, V., 2018. Colourization of grayscale images using Deep Learning. Mahesh and Reshi, Vivek, Colourization of Grayscale Images Using Deep Learning (May 6, 2018).
15. Bilissi, E., Jacobson, R.E. and Attridge, G.G., 2008. Just noticeable gamma differences and acceptability of sRGB images displayed on a CRT monitor. *The Imaging Science Journal*, *56*(4), pp.189-200.
16. Blanch, M.G., Mrak, M., Smeaton, A.F. and O'Connor, N.E., 2019, September. End-to-end conditional gan-based architectures for image colourisation. In *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)* (pp. 1-6). IEEE.
17. Bre, F., Gimenez, J.M. and Fachinotti, V.D., 2018. Prediction of wind pressure coefficients on building surfaces using artificial neural networks. *Energy and Buildings*, *158*, pp.1429-1441.
18. Bruni, R. and Bianchi, G., 2015. Effective classification using a small training set based on discretization and statistical analysis. *IEEE Transactions On knowledge and data engineering*, *27*(9), pp.2349-2361.
19. Chakraborty, S., 2019. Image colourisation using deep feature-guided image retrieval. *IET Image Processing*, *13*(7), pp.1130-1137.
20. Chauhan, R., Ghanshala, K.K. and Joshi, R.C., 2018, December. Convolutional neural network (CNN) for image detection and recognition. In *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)* (pp. 278-282). IEEE.
21. Chen, G.Y., Bui, T.D. and Krzyżak, A., 2005. Image denoising with neighbour dependency and customized wavelet and threshold. *Pattern recognition*, *38*(1), pp.115-124.
22. Chen, M., Shi, X., Zhang, Y., Wu, D. and Guizani, M., 2017. Deep features learning for medical image analysis with convolutional autoencoder neural network. *IEEE Transactions on Big Data*.
23. Cheng, Z., Yang, Q. and Sheng, B., 2015. Deep colorization. In Proceedings of the IEEE International Conference on Computer Vision (pp. 415-423).
24. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V. and Le, Q.V., 2018. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*.
25. Da Silva, I.N., Spatti, D.H., Flauzino, R.A., Liboni, L.H.B. and dos Reis Alves, S.F., 2017. Artificial neural network architectures and training processes. In *Artificial neural networks* (pp. 21-28). Springer, Cham.

26. Dastane, T., Rao, V., Shenoy, K. and Vyavaharkar, D., 2018. An Effective Pixel-Wise Approach for Skin Colour Segmentation-Using Pixel Neighbourhood Technique. *International Journal on Recent and Innovation Trends in Computing and Communication*, *6*(3), pp.182-186.
27. DeVries, T. and Taylor, G.W., 2017. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*.
28. Dongare, A.D., Kharde, R.R. and Kachare, A.D., 2012. Introduction to artificial neural network. *International Journal of Engineering and Innovative Technology (IJEIT)*, *2*(1), pp.189-194.
29. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S. and Uszkoreit, J., 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.
30. Ekroll, V., Faul, F. and Wendt, G., 2011. The strengths of simultaneous colour contrast and the gamut expansion effect correlate across observers: Evidence for a common mechanism. *Vision Research*, *51*(3), pp.311-322.
31. Eluyode, O.S. and Akomolafe, D.T., 2013. Comparative study of biological and artificial neural networks. *European Journal of Applied Engineering and Scientific Research*, *2*(1), pp.36-46.
32. Fei-Fei, L. and Li, L.J., 2010. What, where and who? telling the story of an image by activity classification, scene recognition and object categorization. In *Computer vision* (pp. 157-171). Springer, Berlin, Heidelberg.
33. Finlayson, G.D., Schiele, B. and Crowley, J.L., 1998, June. Comprehensive colour image normalization. In *European conference on computer vision* (pp. 475-490). Springer, Berlin, Heidelberg.
34. Foret, P., Kleiner, A., Mobahi, H. and Neyshabur, B., 2020. Sharpness-Aware Minimization for Efficiently Improving Generalization. *arXiv preprint arXiv:2010.01412*.
35. Ghosh, S., Roy, M. and Ghosh, A., 2014. Semi-supervised change detection using modified self-organizing feature map neural network. *Applied Soft Computing*, *15*, pp.1-20.
36. Gómez Cruz, E. and Meyer, E.T., 2012. Creation and control in the photographic process: iPhones and the emerging fifth moment of photography. *Photographies*, *5*(2), pp.203-221.
37. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*.
38. Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J. and Kern, R., 2020. Array programming with NumPy. *Nature*, *585*(7825), pp.357-362.
39. Harris, E., Marcu, A., Painter, M., Niranjan, M. and Hare, A.P.B.J., 2020. Fmix: Enhancing mixed sample data augmentation. *arXiv preprint arXiv:2002.12047*, *2*(3), p.4.
40. Hassoun, M.H., 1995. *Fundamentals of artificial neural networks*. MIT press.
41. Hawkins, D.M., 2004. The problem of overfitting. *Journal of chemical information and computer sciences*, *44*(1), pp.1-12.
42. Hill, B., Roger, T. and Vorhagen, F.W., 1997. Comparative analysis of the quantization of color spaces on the basis of the CIELAB color-difference formula. *ACM Transactions on Graphics (TOG)*, *16*(2), pp.109-154.
43. Howard, A.G., 2013. Some improvements on deep convolutional neural network based image classification. *arXiv preprint arXiv:1312.5402*.
44. Howse, J., 2013. *OpenCV computer vision with python*. Packt Publishing Ltd.
45. Hu, T.K., Lin, Y.Y. and Hsiu, P.C., 2018, April. Learning adaptive hidden layers for mobile gesture recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 32, No. 1).
46. Jain, A.K., Mao, J. and Mohiuddin, K.M., 1996. Artificial neural networks: A tutorial. *Computer*, *29*(3), pp.31-44.

47. James, D. and Tucker, P., 2004, July. A comparative analysis of simplification and complexification in the evolution of neural network topologies. In *Proc. of Genetic and Evolutionary Computation Conference*.
48. Kabir, G. and Hasin, M.A.A., 2013. Comparative analysis of artificial neural networks and neuro-fuzzy models for multicriteria demand forecasting. *International Journal of Fuzzy System Applications (IJFSA)*, *3*(1), pp.1-24.
49. Karras, T. 2019. *NVlabs/ffhq-dataset*. [online] Available at: <https://github.com/NVlabs/ffhq-dataset> [Accessed 10 March 2021].
50. Karras, T., Laine, S. and Aila, T., 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 4401-4410).
51. Kong, W., Lei, Y. and Ni, X., 2011. Fusion technique for grey-scale visible light and infrared images based on non-subsampled contourlet transform and intensity–hue–saturation transform. *IET signal processing*, *5*(1), pp.75-80.
52. Krenker, A., Bešter, J. and Kos, A., 2011. Introduction to the artificial neural networks. *Artificial Neural Networks: Methodological Advances and Biomedical Applications. InTech*, pp.1-18.
53. Krizhevsky, A. and Hinton, G., 2009. Learning multiple layers of features from tiny images.
54. Krizhevsky, A., n.d. CIFAR-10 and CIFAR-100 datasets [online] https://www.cs.toronto.edu/~kriz/cifar.html. Available at<https://www.cs.toronto.edu [Accessed 19 February 2021].
55. Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*, pp.1097-1105.
56. Kwon, J., Kim, J., Park, H. and Choi, I.K., 2021. ASAM: Adaptive Sharpness-Aware Minimization for Scale-Invariant Learning of Deep Neural Networks. *arXiv preprint arXiv:2102.11600*.
57. Labin, A.M., Safuri, S.K., Ribak, E.N. and Perlman, I., 2014. Müller cells separate between wavelengths to improve day vision with minimal effect upon night vision. *Nature communications*, *5*(1), pp.1-9.
58. Lee, H. and Song, J., 2019. Introduction to convolutional neural network using Keras; an understanding from a statistician. *Communications for Statistical Applications and Methods*, *26*(6), pp.591-610.
59. Limmer, M. and Lensch, H.P., 2016, December. Infrared colorization using deep convolutional neural networks. In 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA) (pp. 61-68). IEEE.
60. Lin, M., Chen, Q. and Yan, S., 2013. Network in network. *arXiv preprint arXiv:1312.4400*.
61. Lindbloom, B., 2017. *Useful Color Equations*. [online] Brucelindbloom.com. Available at: <http://www.brucelindbloom.com/> [Accessed 12 January 2021].
62. Ludwig, J., 2013. Image convolution. *Portland State University*.
63. MacEvoy, B., n.d. *handprint : modern color models*. [online] Handprint.com. Available at: <http://www.handprint.com/HP/WCL/color7.html#CIELUV> [Accessed 6 December 2020].
64. Maind, S.B. and Wankar, P., 2014. Research paper on basic of artificial neural network. *International Journal on Recent and Innovation Trends in Computing and Communication*, *2*(1), pp.96-100.
65. Matrik, K., Alasha'ary, H., Al-Hasanat, A., Al-Qadi, Z. and Al-Shalabi, H., 2014. Investigation and Analysis of ANN Parameters. *European Journal of Scientific Research*, *121*(2), pp.217-225.
66. Mazzone, M. and Elgammal, A., 2019, March. Art, creativity, and the potential of artificial intelligence. In *Arts* (Vol. 8, No. 1, p. 26). Multidisciplinary Digital Publishing Institute.
67. Menon, V., 2011. Large-scale brain networks and psychopathology: a unifying triple network model. *Trends in cognitive sciences*, *15*(10), pp.483-506.

68. Millman, K.J. and Aivazis, M., 2011. Python for scientists and engineers. *Computing in Science & Engineering*, *13*(2), pp.9-12.
69. Moyer, M.D, N., 2019. *Eye Color Percentage for Across the Globe*. [online] Healthline. Available at: <https://www.healthline.com/health/eye-health/eye-color-percentages> [Accessed 7 April 2021].
70. Noda, N., 2018. Recent advances in the research and development of blue flowers. *Breeding science*, p.17132.
71. O'connor, Z., 2011. Colour psychology and colour therapy: Caveat emptor. *Color Research & Application*, *36*(3), pp.229-234.
72. Ogasawara, E., Martinez, L.C., De Oliveira, D., Zimbrão, G., Pappa, G.L. and Mattoso, M., 2010, July. Adaptive normalization: A novel data normalization approach for non-stationary time series. In *The 2010 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). IEEE.
73. Örkcü, H.H. and Bal, H., 2011. Comparing performances of backpropagation and genetic algorithms in the data classification. *Expert systems with applications*, *38*(4), pp.3703-3709.
74. Popov, V., Ostarek, M. and Tenison, C., 2018. Practices and pitfalls in inferring neural representations. *NeuroImage*, *174*, pp.340-351.
75. Poynton, C., 1998. Frequently asked questions about gamma. *Rapport Technique, janvier*, *152*.
76. Ramachandran, P., Zoph, B. and Le, Q.V., 2017. Searching for activation functions. *arXiv preprint arXiv:1710.05941*.
77. Ramadan, Z.M., 2014. Salt-and-pepper noise removal and detail preservation using convolution kernels and pixel neighborhood. *American Journal of Signal Processing*, *4*(1), pp.16-23.
78. Recht, B., Roelofs, R., Schmidt, L. and Shankar, V., 2018. Do cifar-10 classifiers generalize to cifar-10?. *arXiv preprint arXiv:1806.00451*.
79. Reitner, A., Sharpe, L.T. and Zrenner, E., 1991. Is colour vision possible with only rods and blue-sensitive cones?. *Nature*, *352*(6338), pp.798-800.
80. Rowlands, D.A., 2020. Color conversion matrices in digital cameras: a tutorial. *Optical Engineering*, *59*(11), p.110801.
81. Rumelhart, D.E., Durbin, R., Golden, R. and Chauvin, Y., 1995. Backpropagation: The basic theory. *Backpropagation: Theory, architectures and applications*, pp.1-34.
82. Saabith, AL Sayeth, M. M. M. Fareez, and T. Vinothraj. "Python current trend applications-an overview." *International Journal of Advance Engineering and Research Development* 6, no. 10 (2019).
83. Sainath, T.N., Vinyals, O., Senior, A. and Sak, H., 2015, April. Convolutional, long short-term memory, fully connected deep neural networks. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 4580-4584). IEEE.
84. Schmidhuber, J. and Gambardella, L.M., 2011, November. Max-pooling convolutional neural networks for vision-based hand gesture recognition. In *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)* (pp. 342-347). IEEE.
85. Smith, T. and Guild, J., 1931. The CIE colorimetric standards and their use. *Transactions of the optical society*, *33*(3), p.73.
86. Socher, R., Lin, C.C.Y., Ng, A.Y. and Manning, C.D., 2011, January. Parsing natural scenes and natural language with recursive neural networks. In *ICML*.
87. Sola, J. and Sevilla, J., 1997. Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on nuclear science*, *44*(3), pp.1464-1468.
88. Srinath, K.R., 2017. Python–the fastest growing programming language. *International Research Journal of Engineering and Technology*, *4*(12), pp.354-357.
89. Srivastava, N., 2013. Improving neural networks with dropout. *University of Toronto*, *182*(566), p.7.

90. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), pp.1929-1958.
91. Stančin, I. and Jović, A., 2019, May. An overview and comparison of free Python libraries for data mining and big data analysis. In *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* (pp. 977-982). IEEE.
92. Tan, M. and Le, Q., 2019, May. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning* (pp. 6105-6114). PMLR.
93. Team, K., n.d. *Keras documentation: Why choose Keras?*. [online] Keras.io. Available at: <https://keras.io/why_keras/> [Accessed 04 April, 2021].
94. Thimm, G., Moerland, P. and Fiesler, E., 1996. The interchangeability of learning rate and gain in backpropagation neural networks. *Neural computation*, *8*(2), pp.451-460.
95. Tkalcic, M. and Tasic, J.F., 2003. *Colour spaces: perceptual, historical and applicational background* (Vol. 1, pp. 304-308). IEEE.
96. Valencia, M., Pastor, M.A., Fernández-Seara, M.A., Artieda, J., Martinerie, J. and Chavez, M., 2009. Complex modular structure of large-scale brain networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *19*(2), p.023119.
97. Van Der Walt, S., Colbert, S.C. and Varoquaux, G., 2011. The NumPy array: a structure for efficient numerical computation. *Computing in science & engineering*, *13*(2), pp.22-30.
98. Van Rossum, G., 1991. Python.
99. Van Rossum, G., 2007, June. Python Programming Language. In *USENIX annual technical conference* (Vol. 41, p. 36).
100. Vandenbroucke, A.R., Fahrenfort, J.J., Meuwese, J.D.I., Scholte, H.S. and Lamme, V.A.F., 2016. Prior knowledge about objects determines neural color representation in human visual cortex. *Cerebral cortex*, *26*(4), pp.1401-1408.
101. Varga, D. and Szirányi, T., 2016, December. Fully automatic image colorization based on Convolutional Neural Network. In 2016 23rd International Conference on Pattern Recognition (ICPR) (pp. 3691-3696). IEEE.
102. Wang, W., Huang, Y., Wang, Y. and Wang, L., 2014. Generalized autoencoder: A neural network framework for dimensionality reduction. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 490-497).
103. Westland, S., 2003. Review of the CIE system of colorimetry and its use in dentistry. *Journal of Esthetic and Restorative Dentistry*, *15*, pp.S5-S12.
104. Wong, S.V. and Hamouda, A.M.S., 2003. Machinability data representation with artificial neural network. *Journal of Materials Processing Technology*, *138*(1-3), pp.538-544.
105. Wu, S., Wang, G., Tang, P., Chen, F. and Shi, L., 2019. Convolution with even-sized kernels and symmetric padding. *arXiv preprint arXiv:1903.08385*.
106. Xu, B., Wang, N., Chen, T. and Li, M., 2015. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*.
107. Yeo, B.T., Krienen, F.M., Sepulcre, J., Sabuncu, M.R., Lashkari, D., Hollinshead, M., Roffman, J.L., Smoller, J.W., Zöllei, L., Polimeni, J.R. and Fischl, B., 2011. The organization of the human cerebral cortex estimated by intrinsic functional connectivity. *Journal of neurophysiology*.
108. Zeki, S. and Marini, L., 1998. Three cortical stages of colour processing in the human brain. *Brain: a journal of neurology*, *121*(9), pp.1669-1685.
109. Zhang, R., Isola, P. and Efros, A.A., 2016, October. Colorful image colorization. In *European conference on computer vision* (pp. 649-666). Springer, Cham.
110. Zhou, F., Zhang, S. and Yang, Y., 2020, July. Interpretable Operational Risk Classification with Semi-Supervised Variational Autoencoder. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 846-852).
111. Zhu, X., Vondrick, C., Fowlkes, C.C. and Ramanan, D., 2016. Do we need more training data?. *International Journal of Computer Vision*, *119*(1), pp.76-92.

112. Zou, C., Mo, H., Du, R., Wu, X., Gao, C. and Fu, H., 2018. Lucss: Language-based user-customized colourization of scene sketches. arXiv preprint arXiv:1808.10544.

113. Zupan, J., 1994. Introduction to artificial neural network (ANN) methods: what they are and how to use them. *Acta Chimica Slovenica*, *41*, pp.327-327.

8. Word Count

- **Total: 16,230**
- Abstract: 151

- Introduction: 453
- Background and In-Depth Literature Review: 5,377
- Methodology: 2,816
- Design, Development, and Evaluation: 3,792
- Conclusion: 184
- Reflective Analysis: 609
- References: 2,708
- Other: 130