0.1 Motivation

With the continued revolution of AI technologies a desire to utilise the power of neural networks beyond the traditional datacentre bound systems is becoming ever more prevalent. Using neural networks locally as opposed to in a datacentre (in other words at the *edge*) is already prevalent in modern life, to name a few; mobile phones, voice assistants and even our televisions are making use of this technology. Training neural networks is a costly, often slow, process, usually completely invisible to users of a network, so it makes sense to keep training in the datacentre where resources are abundant, however applying the knowledge from a trained neural network (referred to as *inference*) on the other hand has a myriad of advantages when it is performed at the edge.

Keeping data used for inference local to the device that produces this data means we can rely on a consistent inference latency, in particular computer vision applications such as self-driving cars or autonomous manufacturing machines tend to require a consistent low latency and as such cannot rely on cloud services to perform inference. Additional benefits include privacy and security; reducing data sent to datacentres helps mitigate the potential for a major data leak since no data needs to leave the device.

Computing at the edge can be limiting in terms of available memory space or performance capabilities, networks relying on real time data tend to require lower response times, but there is often a locality vs performance trade-off unless expensive custom solutions are used. GPUs are the most widely used piece of hardware to operate a neural network, however they are often physically large and can consume a considerable amount of power, an alternative to the GPU is an AI accelerator such as the Intel Neural Compute Stick, or the Nvidia Jetson, these accelerators are purpose built for neural networks, usually have a much smaller form factor and consume considerably less power.

Pruning is a compression technique that involves removing weights from a network with the goal of making it smaller, intuitively we might think that a network with a smaller memory footprint would naturally have lower inference latency, but it is often not this simple. Utilising neural network compression requires expert level knowledge of not only the network structure but the consequences of compression because compression techniques such as pruning can have cascading effects throughout a neural network. This alone can make compression a daunting task, even for experienced machine learning practitioners, it gets worse however, these compression algorithms

often feature complex parameters with implications that may not be revealed until a substantial amount of time has been invested in retraining a compressed model.

The most apparent reasons to prune a neural network is either to reduce the size of the neural network or its latency, this dissertation will focus on reducing the latency aspect, starting from a readily available off-the-shelf configuration for a neural network we will attempt to automate the process of optimising compression parameters with the goal of reducing latency, ideally with a minimal loss of accuracy.

0.2 Hypothesis

Using optimisation algorithm we can automate compression parameter selection and improve inference latency in a typical edge computing environment.

0.3 Research Aims

Aim 1 — This dissertation will research a methodology for reducing inference latency using off-the-shelf compression techniques as a starting point.
Aim 2 — We will investigate to what degree different parts of the network structure impact inference latency.

0.4 Objectives

- **O0:** Verify the functionality of distiller, check that the pruning methods available function as described in the literature.
- **O1:** Develop a pipeline to compress and benchmark a neural network model.
- **O2:** Select at least 1 neural network model to use for testing.
- **O3:** Select an appropriate compression algorithm for the goal of reducing latency.
- **O4:** Identify an appropriate off-the-shelf set of hyperparameters for the selected model.

- **O5:** Establish baseline measurements in terms of accuracy and latency for both the unpruned and off-the-shelf (pruned) models.
- **O6:** Automate compression parameter optimisation by leveraging the pipeline from O1 to test combinations of parameters and empirically evaluate their impact on a target metric (such as latency or accuracy).
- **O7:** Evaluate how well the automated system optimizes the compression parameters, by comparing with the baseline and off the shelf measurements.