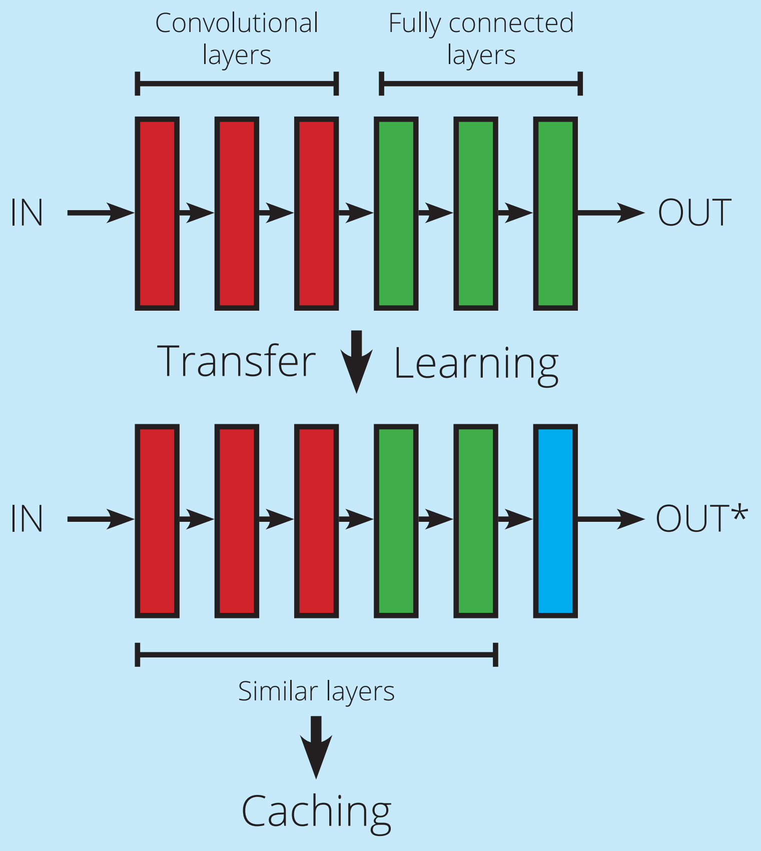
Analysis of the effect of caching convolutional network layers on resource constraint devices

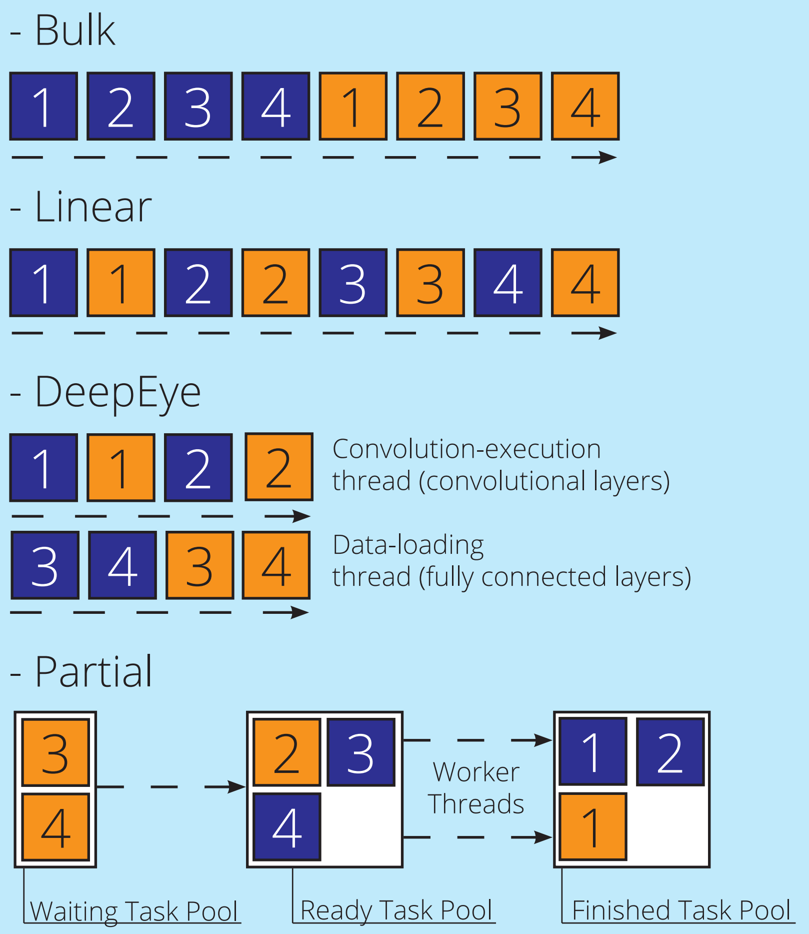
Research Project - CSE3000 Wouter van Lil - Supervisors: Lydia Chen, Bart Cox & Masoud Ghiassi 25/06/2020

Transfer Learning



With the use of transfer learning, a network created for a specific use case can be retrained to fit another field. During this process most of the layers are frozen and will be unchanged. This is a good approach of creating a convolutional neural network whenever there is not much training data available as this prevents overfitting.
A side effect of this is that many layers are similar between the two networks. Whenever we apply caching to these layers, the effect of storing the data is duplicated as the data is used twice.

Modes of Operation



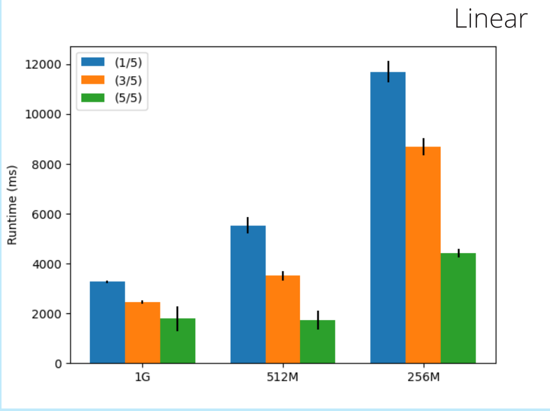
Bulk and Linear make use of a single thread. DeepEye uses one thread for the convolutional layers and one for the fully connected layers. Partial uses a variable number of threads operating on the tasks that are ready to be performed.
In order for a layer to be finished, it needs to be loaded and executed. A layer cannot be executed before it has been loaded and the previous layer has been executed providing the current layer with input. Whenever these prerequisites are met, the task can be placed in the ready pool.

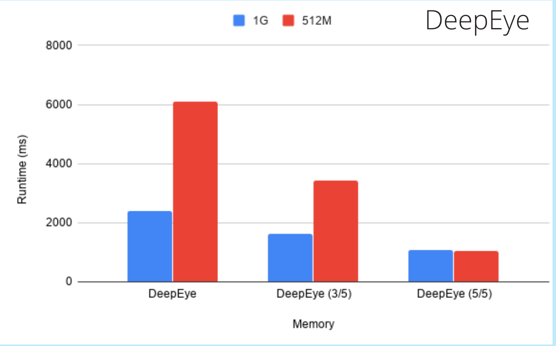
1	= Load Layer 1
2	= Execute Layer 2

Results

In the figures, the definition (X/Y) is used, where X is the amount of similar networks and Y is the total amount of networks.

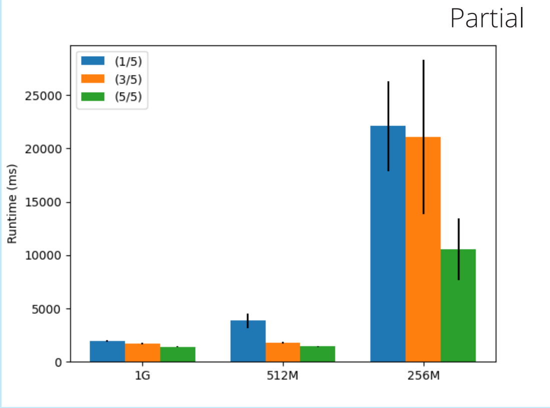
Linear: As more cached layers can be reused, the loading time decreases and so does the overall execution time as it is single threaded. The loading of tasks takes up a significant amount of the total time allowing for visible differences in total runtime between the different tests. At 256M of memory, linear still performs relatively well and the runtime does not grow as fast when providing less memory than the other modes.

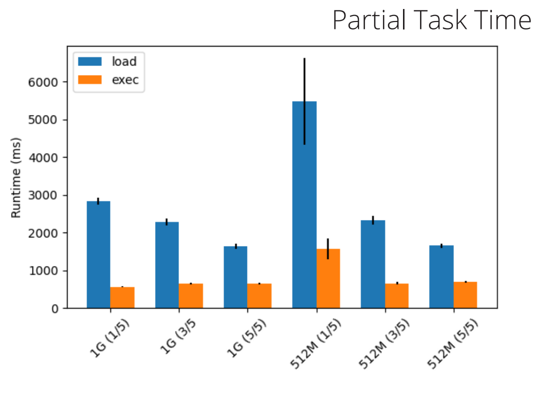

Linear


DeepEye

DeepEye: The proportions of running with 1G or 512M of memory are getting more equal as more of the networks become similar. When all networks are similar the advantage of running with extra memory appears to be gone. A trade-off can be made, with on one side possibly decreasing the accuracy by using networks trained with transfer learning and on the other side investing more in better hardware.

Partial: When running the inferences with partial loading, 512M of memory and not being able to reuse any layers, the runtime has a small spike. This is the brim of where the runtime will start growing rapidly if the memory is slightly decreased or the workload is increased. Partial loading works the fastest of any mode when there is enough memory available, but does not scale as well as linear when this is not the case.


Partial


Partial Task Time

Partial Task Time: The execution times of the layers remains roughly the same, with the only exception being 512M with no reusable layers which has a memory shortage to run. The load time is larger than the execution time, meaning caching to lower this will decrease the overall runtime.