City, University of London

MSc in Software Engineering with Cloud Computing

Project Report

2022

Effectiveness of different dimensionality reduction techniques on pruned deep neural network

Sultangazy Yergaliyev

Supervisor: Dr. Tillman Weyde

Date of submission: 21/12/2022

Declaration

Keywords: Dimensional reduction, Pruning, PCA, ICA, Isomap, matrix

Abstract:

In machine learning, throughout the years the architecture of neural networks has been under rigorous observation and has been developed constantly. However, when it comes to the complexity of the data and the structure of layers within the network our best strategy is to come up with the best architecture so that when we use the network for any reoccurring problem, heavy computational consumption can be avoided. Hence pruning is proposed by many papers that explore the field of pruning on how to find an effective subnetwork that has just as much efficiency as the original network. Furthermore, there are many optimization techniques besides pruning, one of which is dimensional reduction, which is a projection of multivariate data into a lower dimension, for effectivity purposes. In this paper, we propose dimensional reduction methods like Principal component analysis(PCA), Independent Component Analysis(ICA), and Isometric mapping(Isomap), and utilize these methods for weight matrices decomposition within the pre-trained network that were trained using datasets like CIFAR10, and MNIST. We use those methods in two scenarios: one is where the network is unpruned, and two where we prune the network and then apply dimensionality reduction afterwards. Moreover, we develop a technique that combines dimensional reduction and matrix transformation that helps us to preserve as much information as possible. We believe that this experiment provides interesting results with some potential.

Content:

I INTRODUCTION

In recent years, Deep Neural Networks have improved tremendously, and so has the demand for computational power to train a model for efficient inference. To make efficient use of hardware resources (CPU & GPU), it is promising to develop techniques that reduce the size of neural networks. As the model is trained it consumes a relatively large portion of resources due to its computation and dataset size. It can be assumed that the constructed network that is being utilised for a simple problem like a classification problem does not demand a complex architecture to solve a respectfully simple classification task, hence can be compressed to a more optimal subnetwork. Although it is established that the pruning techniques can effectively reduce the number of parameters by approximately 90% without any loss of accuracy (J. Frankle et al, 2019), it can be extremely computationally demanding.

The objective is to perform dimensional reduction without the necessity of retraining the model, as some questions may occur like Why not just construct a smaller network in the first place? However, the goal is to evade any unnecessary procedures like model retraining that is resource-demanding while maintaining as much information of the data within the weight matrix as possible. While, pruning does its effective job of reducing the network architecture, by parameter reduction, parameters remain in their dimension, hence this is where dimensionality is applied.

This serves as a good starting point for effective subnetwork utilisation and potential neural network optimization. Furthermore, it can be used for any pre-trained network and reduced to an optimal size. For this paper, the experiments were conducted mainly on two datasets MNIST and CIFAR10 and custom trained model for simplicity purposes. Moreover, the experiment was conducted using numerous non-linear and linear dimensional reduction techniques like PCA ICA, Isomap, NMF, and MDS As Isomap poorly performs on any relatively complex dataset, we used MNIST for such purpose.

Based on personal research, it can be stated no experiments have been conducted way. While many dimensional reduction research focal objective was to focus on the reduction of the multivariate input data for data analysis, data mining image classification, and effective model training, We approach a problem from a different perspective where we apply dimensional reduction on an already trained model and use it to rescale the parameters of the network, precisely on weight matrices of linear layers.

In this research, we showcase the potential solution and the effectiveness of dimensional reduction techniques on the pruned network by comparing and analysing differently constructed models deriving from the original base network architecture. Furthermore, bringing the pruned network to the next level indicates a better-optimized network and lesser resource consumption. As far as the study goes we have set several objectives which we were working on throughout the research. These objectives are to apply different pruning techniques to the network, Apply different methods of dimensionality reduction on the network's weight matrices, Explore methods for reducing the neural layer, and Implement and test the methods on different networks with different datasets.

II Related Work

II.1 PRUNING

There were several studies done on the pruning of the neural network. One of the most recognized studies is the *Lottery Ticket Hypothesis* which entertains the idea of a dense, randomly initialized, feed-forward network that contains subnetworks that are trained independently, eventually reaching nearly the same accuracy level as the original network (Frankle J, et al, 2019). It states that by performing an *iterative pruning* it achieves the expected result, or to be more precise it breaks down the operation into four main steps:

- Random network initialization
- Training network with *n* number of iterations
- Afterwards, pruning *p%* of the network and creating a mask.
- Reset the remaining parameters and create a winning ticket.

It repeats these steps iteratively, to reach the corresponding accuracy of the base network. It is stated that such an algorithmic procedure gives us the desired winning ticket rather than one-shot pruning (Frankle, J, et al, 2019). This research uses information retrieval by using the same two structurally different models: feed-forward network and CNN. This study uses feed-forward to train on the MNIST dataset and CNN to train on CIFAR10. In out study instead, we approach our problem by using one network architecture, that way we want to establish a problem we might face while experimenting on a relatively simple structure so that when resolved we would be expected to apply such technique to other well-known pre-trained networks like ResNet-50 or VGG16 for instance.

The paper does thorough research by covering every aspect of *winning a ticket* (Frankle, J, et al, 2019). This means it goes through two main methodologies *Iterative pruning* and *one-shot pruning*. It tries to prove how iterative leads to better performance and better results, hence getting a better winning ticket.

II.1.2 FULLY-CONNECTED NETWORKS:

The lottery ticket Hypothesis paper carries out its experiment by pruning weighs of each layer in heuristic, relying on finding the weights with the lowest magnitude possible (Frankle, J, et al, 2019). In this scenario, they used the MNIST dataset and trained on *Leenet-300-100* architecture.

Iterative pruning. According to the paper they claim that iterative pruning the winning ticket has the capability of higher performance in contrast to the original network, they prove their statement by visualising the training process:

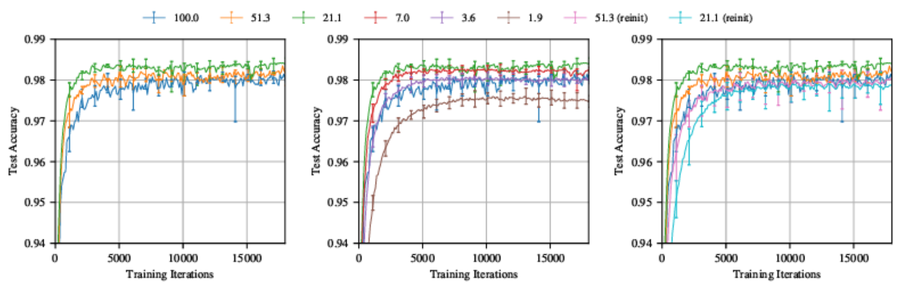


Figure 1 - Test accuracy of the Leenet-300-100 to train the winning ticket that was repeatedly pruned (Frankle, J, et al, 2019)

Essentially stating how pruning approximately half ($P_m = 51.3\%$) of the weights gives a better training performance and reaches a better accuracy compared to the original network, while being inferior to the network if 78.9% ($P_m = 21.1\%$) was pruned. As the pruning amount gets bigger the learning performance starts to converge with the original network. This means that if $P_m \leq 21.1\%$ the learning process starts to decline.

One-shot pruning. If comparing iterative pruning to one-shot pruning it can be noted that doing such computation iteratively may cause a decrease in demand in computational power. Just as iterative pruning running the operation once might lead to finding the winning ticket. Frankle J., et al (2019) state that when $67.5\%. \geq P_m \geq 17.6\%$ of the winning tickets on average have a better validation accuracy when compared to the original network. Furthermore, it is stated that when $95\%. \geq P_m \geq 5.17\%$ This is followed by a higher test accuracy when compared to the original network. Nevertheless, winning tickets pruned in iteratively remains the victor as it provides a faster learning rate and better accuracy with the smaller network.

II.1.2 CONVOLUTIONAL NETWORKS:

The research also focused on a more complex aspect of the lottery ticket hypothesis by applying such theory to three main convolutional networks Conv-4, Conv-5, and Conv-6 which are essentially a duplicate of the VGG network (Frankle, J, et al, 2019). The winning ticket of the convolutional neural networks provides more outlined results when comparing it to a fully-connected network like *Leenet-300-100.* The winning ticket of these convolutional networks tends to reach validation loss at least $x \geq 2x$ times faster with the Conv-2 having $P_m = 8.8\%$, Conv-4 $P_m = 9.2\%$, and Conv-6 $P_m = 15.1\%$. Test accuracies for these three networks have improved by 3% on average when P_m remains at least above 2%. With the given number of iterations 20,000 for Conv-2, 25,000 for Con-4, and 30,000 for Conv-6 the accuracy during the training tends to remain on the same level as test accuracy as long as $P_m \geq 2\%$ it has the capability of reaching 100% training accuracy, indicating the insignificancy of differences between training accuracy and testing accuracy which gives us a better generalization. In summary, the paper states how a convolutional network gives better results and insights when performing an identical operation just like it was done with the FC network.

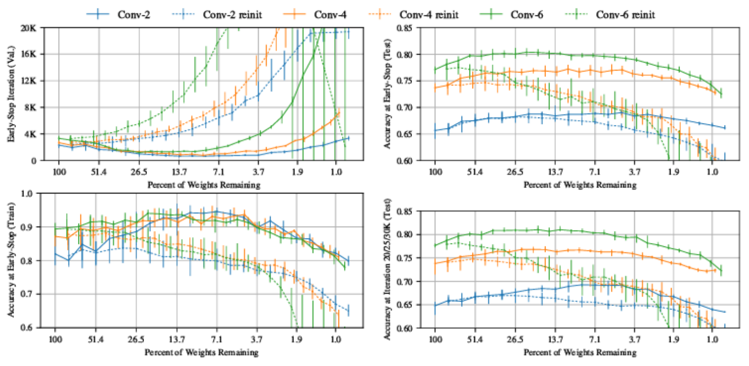


Figure 2 - a visualisation of test and train accuracy of three networks(Conv 2, Conv 4, Conv 6) with the pruned iteration and reinitialization

The hypothesis that this paper attempts to prove is that the iterative pruning procedure can only be implemented on highly parameterized and dense network structures and might only be pruned to a certain degree of sparsity. Otherwise, if going beyond this point, accuracies can only be obtained via random initialization of the network. The paper confirms the potential theory that the same result can be reached with the network being randomly initialized and unpruned, with fewer parameters. This corresponding contradiction can be applied to dimensional reduction, where a network with a smaller number of parameters can be reconstructed and achieve the same result as the original, larger network. Therefore, to disprove such a theory, we aim to demonstrate that a smaller network emerging from a trained architecture can be reused in a real-life scenario.

Furthermore, another study entertains the theory of the lottery ticket hypothesis even further to strengthen the hypothesis. It proposes a new study where that takes over a parameterized network and splits it into two categories, *weight-subnetwork* which eliminates weights from the network and *neuron-subnetwork* which removes neurons instead of the weights. This paper performs a proof, by applying the ReLu am activation function onto the network with the random depth of $2l$ to retrieve a *weight-subnetwork* with the depth of l. Further, it attempts to present a network with one hidden layer that contains *neuron-to-subnetwork* essentially stating how such a network performs just as well as a good random-feature classifier (Malach, E., et al, 2020). They state that pruned networks containing a *weight-subnetwork* tend to perform in the same manner as the original bigger network due to the approximate number of parameters (Malach, E., et al, 2020). Although, it presented fundamental results regarding the pruning of random networks. Essentially Malach, E., et al (2020) concluded their research by discussing the computational complexity of the pruning technique and discussed further how so far there are no more efficient pruning algorithms for randomly initialized neural networks and stating that this is similar to the weight-optimization technique up to some degree. This paper heavily relies on the theoretical part and uses mathematical notations to present the relevant result to prove the hypothesis. By confirming the approximation capabilities of a randomly initialized pruned network, it can target the ReLu network regardless of its layer complexity. However brings the incoming issues with such operation, where weight-pruning of the ReLU network tends to have a big weight on computing power, where stating that algorithm:

$$sup_{x \epsilon X} |\, F(x) - \tilde{G}(x)| \leq \epsilon$$

Figure 3 - Theorem for approximation two depth network to three-depth network by pruning(Malach, E., et al, 2020)

Contrary to popular belief, however, computing the optimum depth-two network approximation for any distribution is computationally challenging, therefore it's highly unlikely to find a more efficient algorithm that can optimize the weight-subnetwork for any significant data prediction(Malach, E., et al, 2020). The paper that studies another pruning technique is heavily inspired by work done by M. ´A. Carreira-Perpi˜n´an. (2017) and M. ´A. Carreira-Perpi˜n´an, et al (2017) and take it and develop it even further. These studies work on neural network compression that demonstrates significant weight optimization capabilities. This study creates an extension to these studies by identifying another potential problem involving the pruning of the deep neural network. Furthermore, it introduces potentially a new algorithm that identifies values in the weights that in actuality should be converted to zero, instead of random initialisation due to the increased chances of uncertainty when working with deep neural networks. Using a custom-made LC("learning-compression") algorithm, it works as an optimization algorithm where instead of calculating loss between predicted and actual data it uses compression techniques like pruning to convert weights w into θ without affecting the loss during the training process (Malach, E., et al, 2020). This study focuses its research on LesNet and ResNet pre-trained models and uses MNIST and CIFAR10 as their datasets. They use these datasets to split insignificant from big value weights so that pruning can be applied only to the former weights and then retrain the network. The performed optimization experiments use Nesterov's optimization technique with the momentum of *0.95* and *100k* of mini-batches while having a learning rate of $\eta \times 0.99^{i}$ where $0.02 < \eta < 0.1$ and having 2k iterations for evey minibatch. The algorithm uses $\mu_j = \mu_0 a^j$ where $\mu_0 = 9.76 \times 10^{-5}$ and $\alpha = 1.1$ (Malach, E., et al, 2020). They state that the methodology that this paper has implemented has good results. It supports the arguments by claiming that the pruning technique has a high success rate of maintaining a good loss rate without any decline. Although such a technique is relatively fundamental, it lacks in finding the right weight set that can be pruned without backtracking. The paper claims the algorithm's legitimacy by stating, that the LC algorithm stands out because it can explore all possibilities of pruning different sets of weight while keeping loss sustainability. Its ability to mark values as already pruned makes it different from hard pruning (Malach, E., et al, 2020).

II.2 DIMENSIONAL REDUCTION TECHNIQUES

There are multiple well-recognized techniques for dimensional transformation that are hugely used for data analysis, hence this could be used to represent the high dimensional data via linear transformation with the techniques like Principal Component Analysis, Factor Analysis, and Independent Component Analysis (Hyvärinen, A. et al, 2000). On the other hand, when it comes to dimensional reduction several risks might need to be considered before implementing the methods. As Carreira-Perpinán, et al (1997) state in the scenario where the system processing data does not require highly dimensional data, in such circumstances dimensional reduction can be the most appropriate optimization method, however, the problem occurs when the dimensionality of the data tensors are extremely high, therefore might lead to significant data loss(Carreira-Perpinán, et al 1997). For instance, a system with greyscale images with $m \times n$ dimension to be used for image classification problems might lead to a large weight matrix generation, consequently, it needs to be rescaled dimensionally (Carreira-Perpinán, et al 1997).

2.1 PCA:

Principal Component Analysis is a widely used linear decomposition method for data analysis and optimization of the input data of the network. This mathematical algorithm uses indicates the directions which are called principal components, so that the variance of the information is maximised. By focusing on a few primary components it can represent the data sufficiently enough while avoiding highly dimensional variables(Ringnér, M., 2008).

The study conducted by Gonzales et al. (2017) examines the idea of combining multiple dimensional reduction algorithms with the angular differential image that is used by the vortex imaging processing(VIP) package presented by this paper to utilize it as an observing technique. This study claims that by using such an observatory technique it can potentially find exoplanets around its host. It uses PCA as a post-processing algorithm to decompose into low-rank space.

Another study done by Kambhatla, N., et al (1997) aims to get a concise description of multivariate data is the main goal of dimension reduction methods. The objective is to produce a concise, accurate representation of the data that minimises or completely removes statistically redundant elements. The study presents experimental results that involve comparing global PCA, five-layer auto associators and several variants of VQPCA that were applied to image and speech to reduce its dimensionality. They use a training set of 120 images to compute several metrics like E norm and training time in seconds.

Another study compares PCA with LDA (Linear Discriminant Analysis) method to emphasize its effectivity as a methodology to utilize for dimensional reduction for Big Data analysis(Reddy, G.T, et al, 2020). It uses Diabetic Retinopathy (DR) and an Intrusion Detection System(IDA) for the experiments. The research is done with the following steps: by applying feature engineering on the CTG(Cardiotocography) dataset to improve the overall quality. The next, step is applying each dimensional reduction technique mentioned above separately and retrieving those important components from the multivariate dataset. This preprocessing step is done so that it is used as input data for ML training. For comparison, the research performs identical operations, however without dimensional reduction and retrieves data for data analysis using particular metrics: Accuracy, F1-score, precision, Recall, Sensitivity, and Specificity. These steps are performed to prove the legitimacy and to point out on insignificance of information loss. Using a confusion matrix it evaluates the the correct percentage of the classifier. It uses PCA to convert from 36 non-target dimensions to 26 non-target dimensions. They combine PCA with other algorithms like Decision Tree, Naïve Bayes, Random Forest, and SVM, the experiment concluded that PCA performs best with Decision

Tree, SVM, and Random Forest, in contrast to Naïve Bayes based on the metrics like F1-score, Recall, and Precision(Reddy, G.T, et al, 2020). In summary, the results that were obtained are:

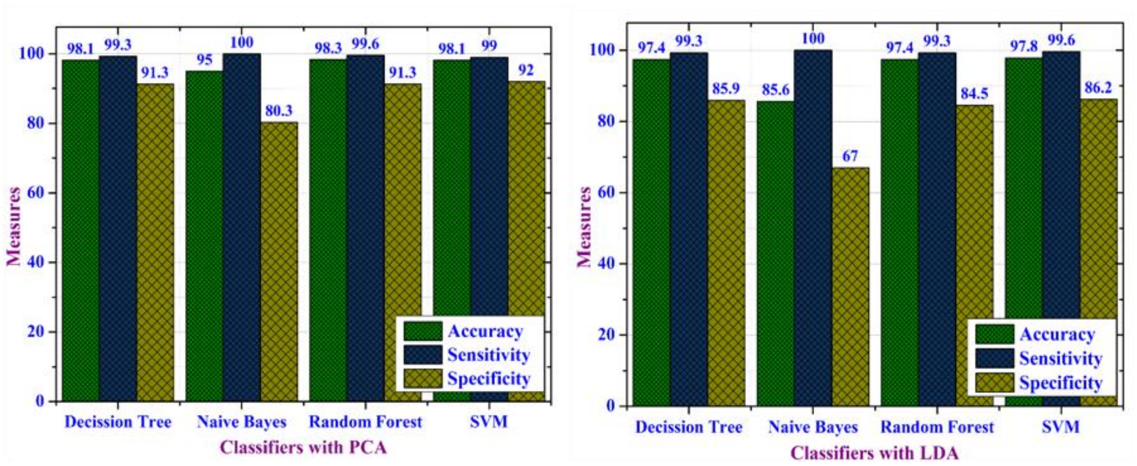


Figure 4 - Measurement of PCA and LDA combined with diferent algorithms (Reddy, G.T, et al, 2020).

The paper concluded its research on quite a positive note stating that PCA has preserved up to 95% of the overall information by reducing 36 components to 26 components as mentioned above. Furthermore, proving the effectiveness of PCA when compared to LDA. However, when performing the procedure on the DR dataset it is stated that it gives a negative outcome due to the small size respectfully, whereas the IDS dataset outperforms other datasets when PCA is being applied.

Another study performs the comparison between two dimensional reduction methods PCA and ICA(Independent Component Analysis). It uses the EEG dataset to examine the effectiveness of these decomposition methods by applying ICA before PCA, applying PCA separately, and without any decomposition methods at all. When applying PCA, such methods affect the stability of the Independent Components(IC) negatively. It is stated, that when transforming into a lower dimension via PCA affects the median of IC from 90% to 76%(Artoni, F, et al, 2018). Furthermore, the paper suggests avoiding PCA when applying the ICA technique as it leads to ambiguity of the IC within the data (Artoni, F, et al, 2018). They prove the effect of PCA on the polarity by contrasting the difference between the quantity of the dipolar $(dip(IC_n) > 85\%)$ and quasi-dipolar $(dip(IC_n) > 95\%)$ Independent components across many experiments through Kruskal-Wallis and Tuckey’s HSD as an evaluation tool (Artoni, F, et al, 2018). They also propose a proof of how PCA affects the stability of IC they compare IC after performing PCA decomposition and before. Using PCA onto each subject's data after RELICA (Artoniet al., 2014) utilization, and then selecting top principal components, that cover at least 85%, 95%, and 99% of the variance of original data. These PCs were ranked in order of the amount of variance they explained. The paper concluded by enforcing the idea that PCA transformation into the smaller dimension by 1% can hugely affect the polarity and sustainability of IC in the EEG data, while simultaneously causing a decline in differentiation between brain and non-brain source activity

2.2 ICA:

There is a study that attempts to do the discovery on the Independent Component Analysis(ICA-DR) to evaluate the independence of the data contrary to its competitors like PCA-DR, and MNF-DR(Maximum noise fraction). The paper claims that ICA-DR is capable of retrieving data values, that cannot be retained by its DR competitors(Wang, J., et al, 2006). Using ICA-DR, MNF-DR, and PCA-DR study enforces the relevant experiment for two applications, endmember extraction and data compression, where stated that ICA-DR performs best in the latter application. This paper introduces the concept of virtual dimensionality (VD) an algorithm that dictates the number of independent components to be rescaled. Due to the random nature of ICA, the research uses ICA together with VD. It demonstrated such performances for the hyperspectral image analysis with the real-life datasets Airborne Visible Infrared Imaging Spectrometer (AVIRIS), and Cuprite data and hyperspectral Digital Image CollectionExperiment (HYDICE)(Wang, J., et al, 2006). Using the two applications mentioned above the experiment concluded that, when using DR for endmember extraction ICA-DR in general outperforms PCA-DR and MNF-DR comparing the ability to extract a more accurate mineral signature. However, when considering the overall performance of each DRPCA-DR, ICA-DR1 and ICA-DR3 are considered top winners due to the successful extraction of five minerals, while MNF-DR and ICA-DR2 fail to extract at least one(Wang, J., et al, 2006). When it comes to Data compression, the study claims that utilization of traditional DR techniques like PCA, and MNF are inclined to sacrifice valuable portions of the data when compression is done. Wang, J., 2006, support the ICA legitimacy by stating that such decomposition has the capability of resolving data loss issues. For the data compression experiment, the paper claims that all three(ICA-DR, ICA-DR1, ICA-DR3) produced impressive results, but different projection vectors can produce different results caused by different vector projecting algorithms like ATGP, and random vectors (Wang, J., et al, 2006).

Furthermore, according to L.J. Cao, et al 2003, their research focuses mainly on establishing a piece of solid evidence for the dimensional reduction of PCA, KPCA, and ICA. It evaluates the promising results of these dimension reduction techniques, claiming that SVM(Support Vector Machine) performs feature extraction substantially better with these methods as opposed to without them. It uses three datasets: *Sunspot data, Santa Fe data set A,* and *Financial data set* to test the efficiency of these optimization algorithms. By using three feature extraction methods on SVM it retrieves necessary results from the machine. For the *Sunspot dataset,* the study illustrates the normalised mean square error(NMSE) for four scenarios(original+SVM, PCA+SVM, KPCA+SVM, and ICA+SVM) suggesting that the best NMSE results have been shown by KPCA, and ICA+SVM where KPCA has NMSE of 0.1544, ICA has NMSE of 0.1651, PCA has NMSE of 0.1853, and SVM without any feature extraction has NMSE of 0.1933. For the second dataset *Santa Fe data set A*, the results were practically the same where KPCA remained with the best results having 0.0111 of NMSE and ICA with the value of 0.0121 making both of these decompositions almost identical in terms of its effectiveness. For the third experiment, the paper (L. J. Cao, et al, 2003) used a set of multiple data sets of Financial data (CME-SP, CBOT-US, CBOT-BO, EUREX-BUND, MATIF-CAC40). They conclude the final experiment with approximately identical results as presented with the previous dataset, that is KPCA with the NMSE values of 0.6988 for CME-SP, 0.8642 for CBOT-US, 0.8855 for CBOT-BO, 0.7089 for EUREX-BUND, and 0.7698 for MATIF-CAC40 (L. J. Cao, et al, 2003). The second best result was achieved by ICA decomposition, having approximately higher NMSE by at least 0.1000. This paper concluded its experiments by proposing that ICA and KPCA are optimization methods that work best with multivariate datasets. They support their claim by saying that these algorithms take a bigger range of information to find these components when compared to PCA (L. J. Cao, et al, 2003).

Another paper raises the issue of the curse of dimensionality when working with multivariate datasets (Anowar, F., et al, 2021). Using multiple optimization techniques(linear and non-linear, supervised and unsupervised, random projection-based and manifold-based), one of which is ICA, the study attempts to prove the incompetence of these methods empirically (Anowar, F., et al, 2021). By comparing three main datasets (ECG200, Arcene, HAR), the paper uses a classification problem to weight the difference in F-score among all dimensional reduction techniques. For the ECG200 datasets, best optimization methods that presented the best F-score were KPCA(0.9303), ICA(0.9271), PCA(0.9205), and SVD(0.9205) (Anowar, F., et al, 2021). This paper summarised the performance by stating that out of all feature extraction techniques, non-linear ones performed better compared to linear ones when applied to high-dimensional data. While based on the result, it can be said that ICA does not show the best results it still outperformed other dimensional reduction methods, which confirms the reliability of the ICA. In this research, we propose a confirmation in the case of ICA to analyse if ICA is capable of demonstrating impressive results when working with complex dataset and relatively simple datasets.

2.3 ISOMAP:

It is believed that out of all non-linear decomposition methods, ISOMAP tends to remain one of the hugely researched and promising optimization methods. As its approach is non-linear it is believed it is capable of picking perspectives that other methods are unable to do like different face angles for instance, as stated by Tenenbaum JB, et al (2000). Tenenbaum JB also believed that PCA and MDS are simple algorithms and relatively easily executable while maintaining their effectiveness. However, when it comes to non-linear structures of the dataset, these methods fail to be competent. This is one of the many papers that introduces the Isometric mapping technique and then tries to prove how its effectiveness can be combined with PCA or MDS to improve computational efficiency, global optimality, etc. For this scenario, the article(Tenenbaum J.B., et al, 2000) uses $N=698$ number of images with the dimensions of 64x64 which represents greyscale faces of different angles, and handwritten digits datasets MNIST. Based on the visualised results provided by the article(Tenenbaum J.B., et al, 2000), the asymptotic convergence confirms the improvement of the approximation of the internal geodesic distance due to Isomap's ability to generate globally optimally rescaled the Euclidean geometry that describes/analyse data. As for the face images, the article believes that Isomap has success in representing the dimensionality of such datasets accurately, by correctly categorizing different face angles. The article(Tenenbaum J.B., et al, 2000) believes that Isomap can correctly identify low-dimensional subspace of the images of hand variations(different angles, and rotations) despite the underlying noise of the real images. Furthermore, Isomap succeeded in retrieving solid coordinates from MNIST datasets, despite lacking manifold geometry, while on the other hand, PCA and MDS failed to perform poorly on it. In This paper(Tenenbaum J.B., et al, 2000), the work confirmed the legitimacy of the Isomap reduction method and how it can capture data variables that other methods like PCA, and MDS are unsuccessful in.

A different study (Geng, X., et al, 2005) for ISOMAP explores the topic of vitalisation and classification, and how dimensional reduction is an important pre-processing part of these applications. As stated in previous papers mostly for such scenarios people tend to use more of a traditional way of resolving the dimensionality through PCA MDS, or ICA. This paper mentions the problem of Isometric Mapping by claiming its weakness towards noise sensitivity and states a new solution by proposing an S-Isomap. They support its evidence by comparing S-Isomap to LLE, Isomap, and WightedISO. S-Isomap's effectiveness lies in breaking down data values into neighbourhoods based on the value difference to integrate the classes more effectively. This functionality makes S-Isomap stand out and makes it especially impactful for visualization and classification (Geng, X., et al, 2005). The experiment that this study performs uses visualization and classification as a form of proof to showcase the effectiveness of S-Isomap. For the visualization of the study (Geng, X., et al, 2005). uses a correlation coefficient to compare the distance between a pair of data points, this is done due to observing the relationship between data points before and after linear transformation. Furthermore, to evaluate the results, the experiment was performed on a 2-D structure with about 50 classes, and real-life data sets containing 505 greyscale images of faces of different angles with a size of 60x50. By sampling 1000 random data values from the dataset, the study uses it to embed the points into "S-curve" and "Swiss-roll". For the artificial dataset as the study illustrated, and by measuring correlation it can be concluded that out of all manifold methodologies, S-Isomap reconstructs visually best in contrast to its competitors with a global correlation of 0.9880 and class correlation of 0.9945. For the Isomap the values are 0.9277(glob. correlation) and 0.9366(local correlation) which makes ISOMAP the second worst dimensional reduction. As a justification for poor performance, it can be stated that the problem lies within the noise and non-linearity as mentioned before. However, according to Geng, X., et al,(2005) when eliminating non-linearity and noise, the ISOMAP can almost perfectly reconstruct the original structure of the data. Evnautal results for the visualization stated that S-Isomap has produced better results in standard deviation and its average correlation than Isomap. For the classification problem, the article(Geng, X., et al, 2005) uses several classification algorithms(SVM, K-NN, BP neural network, J4.8 decision tree) together with S-Isomap, and Isomap, and 10-fold cross-validation to evaluate and compare each algorithm individually. After the calculation, the results confirm that once again the productivity of the S-Isomap dimensional reduction technique, by declaring that the average of the Mean Precision (MP) of the S-Isomap is 0.8305, and making K-NN second best algorithm with the value of 0.8130 MP on average. when it comes to the average MP of the Isomap it has resulted in a 0.7081 value, which unfortunately makes it the worst algorithm for the classification problem(Geng, X., et al, 2005). In conclusion, the paper states and once again confirms its prior objective by proving the effectiveness of S-Isomap, by asserting its dominance among other DR techniques, especially Isomap. Furthermore, declaring that a common issue with generalization error occurs when transforming original data to a lower dimension does not come explicitly from S-Isomap as feature extraction is used by another non-linear interpolation method. If a better method can be found for mapping the data points from the original space to the feature space using S-Isomap, the algorithm's ability to generalize and make accurate predictions on new data will be improved (Geng, X., et al, 2005).

Next, the article studies(Lee, J.A., et al,2004) the differences between two non-linear decomposition methods: Isomap, and Curvilinear Distance Analysis(CDA) and shows their advantages and disadvantages. These two algorithms are somewhat similar to some degree, where Isomap uses MDS as a base of its algorithm, while CDA relies on a nonlinear variant, which is Curvilinear Component Analysis(CCA). First, the paper compares these two methods visually, as the paper (Lee, J.A., et al,2004) compares both of these manifold methods by visualization. When comparing these two methods visually, as the paper states(Lee, J.A., et al,2004), it can be seen that when the projection of the Swiss-roll is done by Isomap, it presents some inconsistencies when "unrolling" data. According to Lee J. A., et al (2004), they claim that due to randomized retrieval of the data variables done by Isomap, this may lead to potential problems, hence the inconsistent data distribution, in contrast to CDA. CDA has shown more stable results due to vector quantization preprocessing, meaning when used with Isomap this potentially can stabilize the spread of data values. When dealing with image processing of artificially generated faces consisting of 698 images, it can be stated that CDA and Isomap produced somewhat similar output, where CDA was performed without vector quantization preprocessing. This is because a dataset of different face angles requires geodesic distance or curvilinear distance to resolve the problem of the orientations. However, the article believes that such results cannot be achieved when working with a clock reading problem that consists of 720 images. The subtle issues with such datasets lie in the fact that no PCA preprocessing was performed like it was done on the face orientation problem. This indicates the unsuccessful one-dimensional manifold detection done by Isomap. The one-dimensional representation is advised to be the projection of the circle into a line, however, Isomap is incapable of differentiating it, is due to 1D's combination of two halves of the circle(Lee, J.A., et al,2004). Whereas, a different result is achieved with CDA when combining it with gradient descent. The paper(Lee, J.A., et al,2004) compares two algebraic procedures Isomap and CDA, stating that a global approach like Isomap has a strong advantage in its "flexibility", meaning when presenting the problem the algorithms can come up with the relevant solution regardless of the quality of it. However, with the local method like CDA, there are some precautions are need to be considered when performing the computation even for a well-fit model, in which this approach heavily relies on the parameters, that is when well-parameterized it can give a good solution. The paper concluded by justifying the effectiveness of the Isomap, declaring that the Isomap possesses speed and is strong fundamentally in terms of theory, which CDA is lacking.

2.4 SPARSE PCA:

Throughout the years these traditional decomposition algorithms have been under constant development for effective information retrieval and data analysis, where Sparse PCA is the product of such experiments. Zou, H. et al (2006), presents these experiments in the paper, by observing the limitation of the PCA, saying that the principal component is just a product of a linear combination of the raw data. For this, first PCA was implemented on the data, afterwards applied the *elastic net* (Zou, H. et al, 2006) to retrieve sparse elements of the values produced by PCA. This was all produced by applying SPCA to simulated and real datasets. Using Pitprops data presented by Jeffers (1967), it uses SCoTLASS, which is a variant of modified PCA. A study (Zou, H. et al, 2006) uses this and SPCA to compare its modified output. It states that PCs produced by SCoTLASS give a lower variance output(69.3%) when comparing the variance(75.8%) of PCs produced by SPCA. One of the alternative thresholding methods that can be used is soft thresholding which is compared to the methods mentioned above. Based on the comparison results generated by the study, it can be evaluated that the simple-thresholding method is more efficient than SCoTLASS but SPCA overtakes with its performance.

Furthermore, SPCA has the advantage of identifying a correct number of values, which means that it can present sparse principal component structures most perfectly, just like SCoTLASS. As most of these thresholding methods apply the lasso penalty algorithm, they are successfully able to retain the perfect structure of sparse PCs. For the second experiment, the study applied PCA-modified techniques to Ramaswamy Data that were needed to accurately predict the outcome of the analysis package of genes. SPCA succeeded in achieving 46% of the overall variance, whereas SCoTLASS appeared to be unfunctional for such data as it could not find the sparse. As the level of the sparsity increased($\lambda = \infty$) the coefficient of variance gradually started to decrease from 46% to 40% approximately taking 2% of the total dataset to create the principal components. In the case of a simple threshold, this method covers more variance by a small margin compared to SPCA. Based on the results stated by Zou, H. et al (2006), we can surely say that SCoTLASS gained its confidence in retrieving sparse from PCs, however, this is compensated by computational inefficiency and requires rigorous fine-tuning, whereas SPCA on the other hand, have the versatility of control in the sparse matrix and dominates SCoTLASS in computational efficiency.

Further development of such a relatively undiscovered concept of SPCA has been brought by Lai, Z, et al (2012). They propose the technique called multilinear sparse principal component(MSPA) analysis to extract the feature from the complex tensors. Using different varieties of datasets like COIL-20, Weizmann action, and Face Recognition Technology datasets they support the statement by claiming that MSPCA indeed outperforms PCA. What makes MSPCA multilinear is the projection into multilinear regression which makes it eligible for sparse regression by losing particular coefficients. Throughout the experiment, the study stated that for MSPCA to produce accurate prediction in face recognition, it has to remain within the *2~8* range (Lai, Z. et al 2012). Furthermore, when assigning L_2 norm penalty to 0 it degrades the output, however when combining L_1 and L_2 norms penalty it can boost the performance of the algorithm. The experiment performed used several combinations of datasets and decomposition techniques to evaluate the results and compare them accordingly. Through repetitive experiments(10 times), the paper analyses these methods through average accuracy, standard deviation, and optimal dimensionality. For the Yale Face Database MSPCA+LDA has an accuracy of 93.92% while SPCA+LDA has an accuracy of 90.14. For the FERET dataset, MSPCA+LDA has an accuracy of 78.95, and SPCA+LDA has an accuracy of 69.93%. For the COIL-20 dataset, MSPCA+LDA has achieved an accuracy of 86%, while SPCA+LDA has the rate of SPCA+LDA. For the Weizmann dataset, MSPCA+LDA has an accuracy of 78.63, and SPCA+LDA has an accuracy of 75.11%. Based on the collected results (Lai, Z. et al 2012) it can be confirmed that SPCA provides a good outcome but MSPCA gives an even better one. Combined with Linear Discriminant Analysis(LDA), this combination gives an even better result in general. This confirms the efficiency of MSPCA when compared to PCA, 2-DPCA, MPCA, and especially SPCA(Lai, Z. et al 2012).

III Methods:

In this section, we elaborate in detail on the methods we used and approached to solve relevant problems. Here we describe in short every DR and pruning technique used in the experiment and the coding tools used to implement such methods. Furthermore, The design approach and the idea behind were to implement a small library that operates as dimensional reduction with the pruning that can be reused for any possible pre-trained neural network without any unnecessary retraining. In this section, we describe the methods and approaches we used for identifying the problem and finding the solution to it. This research mainly focused on using these two main libraries: PyTorch, Scikit-Learn, *CIFAR10* and *MNIST* for simplicity purposes.

III.1 MNIST & CIFAR10:

For experiment purposes, we used two basic datasets to perform several experiments and compare relative results. Using these datasets, we study the custom-built network that was constructed for these datasets specifically. We used two base CNN networks that are somewhat similar to each other *cifar_original_network* and *mnist_original_network*. These structures use similar *MaxPoold2d* layers and two Convolutional layers *Conv1, and Conv2* to lessen the load of the input data. Using pooling or *MaxPool2D* we reduce the computational complexity of the model by reducing the number of parameters the layer would contain, thus reducing the load during the training process. Using the *MAX* function, it rescales the dimensionality of a given input on each activation layer (O'Shea, K., et al, 2015). We utilize fully connected layers of a custom neural network to retrieve relative weights matrices, which in the current case three main layers were reconstructed:

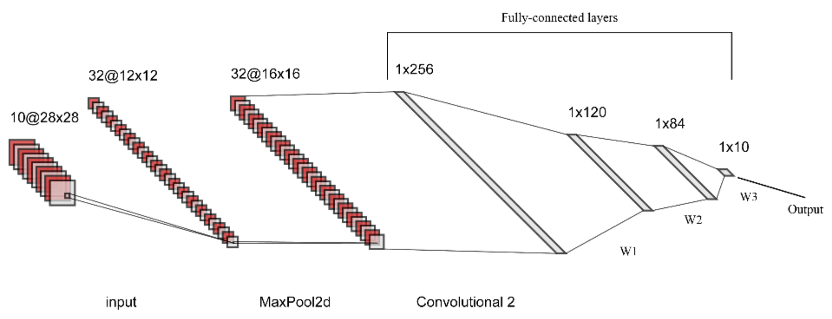


Figure 5 Rough representation of network architecture for MNIST dataset before DR (dimensional reduction)

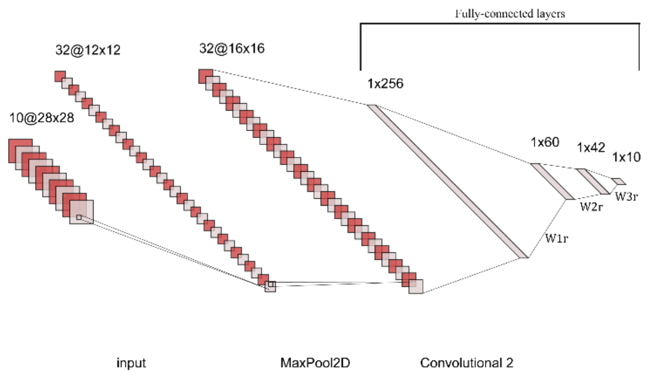


Figure 6 Rough representation of network architecture for MNIST dataset after DR

The network utilises mainly SGD (Stochastic Gradient Descent) as the main strategy for the optimization and we use the Cross-Entropy Loss method to calculate and monitor the performance of the training process by outputting the loss. Furthermore, to monitor the stability of the learning process we use validation data and training data together to evaluate the potential overfitting or underfitting. To get validation data we split the training data of MNIST and CIFAR10 accordingly.

III.2 THEORETICAL APPROACH:

As it can be shown in Figure 3, it can be noted that fully connected layers have received a drastic change, hence such an occurrence is unavoidable the objective is to retain as much information as possible stored in the weight matrices of these layers.

Identifying the problem. We can analyse the problem, based on the result using dimensional reduction only, which has a significant effect on the neural network and brings the neural network to the nearly initial state with accuracy at most *10%* or below, thereby confirming, when it comes to the weight matrix, dimensional reduction on its own are extremely ineffective unless retrained, however, thus we achieve no effectiveness since re-building a smaller network from scratch would be as effective as retraining dimensionally reduced net. Therefore, to bypass such an obstacle we multiplied the product of weight matrices of FL layers by multiplying the original weight matrix W_n and W_{nr} where W_{nr} is a new dimensionally reduced matrix, and W_n weight matrix. Afterwards, when producing a new instance, we use it to transform the weight matrix of the following layer. Such operation was broken down into two main steps where after each weight dimension rescaling, such weight matrix multiplication is performed. It denotes that if W_1 with the dimensions of *256x120* was rescaled to W_{1r} matrix with the dimensionality of *256x60*, such a set of vectors then applied for matrix multiplication $W_1 \times W_{1r}^T$ to extract newly created value. Alternatively, such an equation can also be written as:

$$\sum\nolimits_{k=1}^{n} a_{i,k} b_{k,j} = c_{i,j}$$

Where: $C_n = \{ \text{c} \in \mathbb{R}^{i \times j} \}$.

Then, we use such a matrix to transform the weight matrix(W_2) of the following layer. Let's say C_1 is a newly created matrix with the dimensions of *120x60*, using this matrix we can reutilize the formula illustrated above to multiply with the matrix of the following layer. Hence, the following procedure for layer 2 can be elaborated in the way described above. We perform the multiplication of two matrices $C_1^T \times W_2$, hence we get C_2 and apply DR onto new value and obtain C_{r2}. For the final output layer, no dimensional reduction is needed as we need to contain the number of columns of the matrix to output a correct prediction. However, we reduce the number of rows for the weight matrix of the final layer W_3 where its dimensions are: *84x10* while we aim to change it to *42x10,* essentially the way to do this is by multiplying two matrices C_{r2}^T with the dimensions of *42x60* and $C_2$with the dimension of *60x84,* thus it produces a new weight matrix W_{r3} with the dimension of (*42x10).* The way dimensional reduction techniques were utilized, we retrieved weight matrices of the trained model and reduced its matrix by a factor of 2, hence means if:

$$w1_n = \begin{bmatrix} a_{1,0} & \cdots & a_{i,0} \\ \vdots & \ddots & \vdots \\ a_{i,0} & \cdots & a_{ij} \end{bmatrix}$$

Where *n* is the number of weight matrices and *I* and *j* represent its dimensions, the calculation performed by the dimensional reduction method would convert matrix *w1* to:

$$w2_n = \begin{bmatrix} a_{1,1} & \cdots & a_{k,1} \\ \vdots & \ddots & \vdots \\ a_{k,1} & \cdots & a_{kl} \end{bmatrix}$$

Where *k* and *l* are presented as dimensionally reduced dimensions of the matrix.

III.3 PRACTICAL APPROACH:

To confirm the similarity between the two outputs of two different networks, we eliminated non-linearity and normalised output of DR network and non-DR networks via PyTorch. Through several attempts, we used two different methods to analyse the similarity and accuracy of both networks. Using two types of metrics we can examine the similarities among several models that have been constructed throughout the experiment. Using pairwise distance we compute the distance of two columns within the matrices, that way we find its similarity and use cosine similarity to measure its similarity based on the angle, which is undependable from its magnitude.

COSINE SIMILARITY: Such a metric is an extensively used technique to measure relevant similarities in certain studies (Rahutomo, F., et al, 2012). As mentioned above such metric heavily relies on the angle the vectors are positioned on, and measures based on the [-1,1] interval, So the lesser the angle between two angles, the higher cosine similarity is. Bypassing two values x^1, x^2 the calculation of two vectors is represented as:

$$sim(x^1, x^2) = \frac{\sum_{n=1}^{d} x_i^1 \times x_i^2}{\sqrt{\sum_{i=1}^{d} x_i^1} \times \sqrt{\sum_{i=1}^{d} x_i^2}}$$

Figure 7 - Cosine Similarity (Xia P, et al, 2015)

The equation calculates the cosine similarity between two variables and then returns the angle between them. Cosine similarity limits itself by focusing only on the degree of the vectors, while Euclidean distance, for instance, calculates its similarity by using line segment to measure the distance, hence meaning close distance leads to higher similarity, however, it is heavily dependent on the magnitudes (Xia P, et al, 2015). To use the metric, we generated the number of networks that are derived from two original networks as mentioned above. Consider four general networks: *original_network, dimensionally_reduced_network, dimensionally_reduced retrained_network,* and *small_network* with the reinitialized FC layers to be similar to the DR network, which was also trained. We create a newly initialized network with the FC layers resembling the *dimensionally_reduced_network* to answer the question of the dimensional reduction effectiveness and conclude if there is a value in a performed operation when a smaller network can be created and just trained instead. These networks are generated for every DR technique used(PCA, Isomap, ICA, SparsePCA), using *original_network* as a base example we use cosine similarity to contrast the models and analyse the similarity. In the next stage, we use the PyTorch library to perform global unstructured randomized pruning, to prune half of the weight matrix of each layer, this gives an additional *pruned_original_network.* Next, depending on the DR technique, we use it to apply it to build a new network that is both pruned and rescaled dimensionally. The same procedure as mentioned above, where we compare those networks to the original and then compare the pruned network with the DR network and pruned network to obtain cosine similarity.

III.4 ACCURACY:

For accuracy, we want to visualise the learning curve of several networks: *original_network, dr_network_retrained,* and *small_network.* We aim to observe the stability and performance during the training process of these three networks and compare the accuracy achieved from those networks per each dataset. We use 5 iterations for MNIST and 10 epochs for CIFAR10. Keeping track of the accuracies of three main networks *original_net, dr_network_retrained* and *small_network*, the experiments involve the analysis of the loss or the error during the training and the gradual change of the accuracy per each training, we use matplotlib to visualise the training process. For the remaining network that training was not implemented on, we use a testing set to calculate an average accuracy to summarize any potential change in the accuracy.

III.4 PRUNING:

In the experiment, we apply one of the PyTorch functions that perform global pruning called *randomized_untstructured_pruning* for simplicity purposes. This is a state-of-the-art method that heavily depends on over-parameterized that is unnecessarily complex. Furthermore, we use tuning to dictate the *p% of* weight matrices to prune. Networks trained on MNIST data were pruned by the coefficient of 0.5, whereas networks trained on CIFAR10 were pruned by the coefficient of 0.25, this was done due to the unpredictive nature of *randomized_unstructured_pruning* and the prediction rate of cifar network. The network trained on cifar, if pruned would at least lose *20%* of overall accuracy, when the average is 65%.

Just like *Lottery Ticket Hypothesis* (Frankle, J, et al, 2019), our approach is somewhat similar when comes to pruning. In the paper, it is stated that the approach they have used involves randomly initializing the network and then pruning small *p%* of weights after training the model (Frankle, J, et al, 2019). However, our approach differs yet is somewhat similar to a certain degree. The technique involves using randomised unstructured pruning. We prune about 50% of the amount. Approximately speaking, this test causes the model to lose at least $20\% \leq x$ of the accuracy on average which indicates that using dimensional reduction on the pruned network will lead to tremendous information loss which the experiment confirms. In the experiment, we conducted pruning and dimensional reduction on two datasets mentioned above. We perform training on cifar_original_net and mnsit_original_net with 5 epochs for every model. It can be noted that mnist_original_net gives higher accuracy than MNIST having been a simpler dataset, surprisingly according to the results, it can be stated that pruning and dimensionally reducing the network trained on CIFAR10 dataset preserves at most $36\% \geq x$, inclines that the model has 36% of accuracy. On the other hand, the model trained on the MNIST dataset has $29\% \geq x$ of prediction chance.

To adapt a newly transformed weight matrix into the network, we use deep copy to duplicate an *original_network* and use *nn. Linear* to reconstruct the FC layer accordingly to match the dimension of the recently developed weight matrix. To eliminate the randomness of the CNN we pass all the significant information from the original network and disable biases for FC layers to stabilize the accuracy. Arbitrarily generated biases generated from new networks tend to have little to no effect and tend to fluctuate the accuracy of the prediction, sometimes for the better and sometimes for the worse, to avoid such unpredictability we disable biases for FC layers to prioritize stable results.

III.6 DIMENSION REDUCTION TECHNIQUES:

In this section, we discuss different dimensional reduction techniques and compare their effectiveness between them. The approach was to utilise non-linear and linear methods to observe the performance and analyse its distinctions. The experiments were conducted to gain some expertise on what technique works best when applying the weight matrix rather than the raw dataset. Even though some of the techniques failed to showcase any performance, Due to the nature of the weight matrix, several implementations succeeded in their operations. NMF application was an unsuccessful attempt as weight matrices tend to have negative elements, which we tried to exclude from matrices by using the ReLu function. Although this resulted in NMF utilization, it gave a tremendous decrease in network accuracy which made the model unpractical for use.

One of the issues encountered within the experiment is that when just applying the dimensional reduction we face a big loss of information. One of the first attempts was to implement the DR onto the weight and assign those data into new architecture, this leads to a poorer accuracy which is x<=10% and fails to go above such threshold. When performing different dimensional reduction techniques, weight matrices had to reshape, or be adapted accordingly for the method to function. This implies when working with Isomap for instance all relevant matrices are reshaped according to the function requirement where $x \leq \min(n\ features, n\ components)$. For example, when performing dimensional reduction it was essential to reconstruct the columns of the weight matrix. Isomap is one of the major reasons reshaping had to be implemented, this was due to the significance of negative eigenvalues. Furthermore, for linear transformation to work effectively, we initialize *random_state* to zero and perform preprocessing called *Whitening.* Whitening is the process of eliminating a portion of the information from a given input, hence this potentially can improve or worsen the accuracy where in our case it significantly improves overall results. For the case of linear transformations like PCA and FastICA, we use the “unit-variance” whitening method, which is a whitening of the matrix where we guarantee that the source has unit variance(sklearn.decomposition.FastICA, 2022).

III.6.I PCA method:

Principal component analysis or PCA in short is a widely recognized dimensional reduction that uses the linear transformation of the data and converts to a low-rank matrix that has the maximized variance to describe multivariate data, by rotating the axes of the vectors (Hatipoğlu, G. et al 2022). When formulating the covariance matrix of the given dataset, the purpose is that we want to observe the difference between the variables, and we use this formula:

$$cov_{xy} = \sigma_{xi}\sigma_{yi} = \sum_{i=1}^{n}(x_i - \mu_x) \times (y_i - \mu_y) \qquad \begin{bmatrix} \sigma_{1,1} & \sigma_{1,2} & \cdots & \sigma_{1,j} \\ \sigma_{2,1} & \sigma_{2,2} & \cdots & \sigma_{2,j} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{i,1} & \sigma_{i,2} & \cdots & \sigma_{i,j} \end{bmatrix}$$

This mathematical formula(Fig. 12) gives us the matrix of covariances(Fig. 13) that describes pair-wise values, that describe the correlation between two values, meaning if the covariance is high this means that data points vary together (Hatipoğlu, G. et al 2022). Using the covariance matrix we calculate the eigenvectors and eigenvalues, then we retrieve principal components by finding the biggest eigenvectors based on the eigenvalues. We use these principal components as weight matrix W to project data into a lower dimension:

$$T = W \times X$$

Where alternatively can written as:

$$t_{i,j} = \sum_{k=1}^{n} (w_{i,j\times} x_j)$$

In this research this calculation was mostly performed by the Scikit-Learn library, we initialize the number of components based on the dimensions of the matrix as within the context of our research number of components is represented as the number of columns in the weight matrix.

III.6.II ICA method:

Both ICA and PCA are linear decomposition methods, whereas, unlike PCA, ICA components are independent, which makes them independent through additive subcomponents. In our scenario, we use the FastICA function called from the Scikit-Learn library. ICA has two different types of applications which are: blind source separation, and feature extraction(Hyvärinen, A. and Oja, E. 1997).

III.6.III ISOMAP method:

For the algorithm explanation of the Isomap, we will be referencing Tennenbaum J. B. et al (2000) as a primary example for the algorithm. There are three main steps that best describe Isomap. First, the data is used to construct a neighbourhood graph. When constructing the graph, it is built based on all data points and connects them by relevance (relevance is determined by distance on the manifold where $d(i,j)_x$ the function returns the distance between two data values i and j) and categorizes them using K-Nearest Neighbour, hence creating a neighbourhood graph. The distance of the points can be calculated through Euclidean distance which then produces the corresponding edge of the nodes that can also be represented as weighs. Secondly, the algorithm finds the shortest path of geodesic distance between each vertex of the manifold using methods like Dijkstra's algorithm. Then, since Isomap is a form of extended MDS(Multidimensional scaling) decomposition method, it uses MDS as its final step of computation. Multidimensional scaling is another dimensional reduction technique that finds a decreased number of data points and then finds the similarity based on the distance between the value points. By obtaining the information from the neighbourhood representation it, maps the groups based on similarity and dissimilarity, to organise and simplify the data (Hout, M.C. et al, 2013).

III.6.IV SPARSE PCA method:

Sparse PCA is an alternative to PCA that searches for sparse components to perform a transformation. A method called sparse PCA (SPCA) has been introduced as a way to identify modified components with no loadings. This is possible because PCA can be expressed as an optimization problem similar to regression. This enables the use of LASSO, a technique that involves penalization using the l_1 norm, to be applied(d'Aspremont, A. et al, 2004). Essentially SparsePCA is a combination of two algorithms sparse dictionary learning and PCA, sparse dictionary learning, where it finds a linear combination of the input data with zero loadings and minimizes the non-zero coefficient in the combination. It uses this formula(Mairal, J. et al, 2009) to perform such minimalization for the sparse coding problem(Fig. 8):

$$min_x f(x) \equiv ||y - Ax||^2 + \gamma||x||_1$$

Figure 8 -Algorithm for sparse coding problem (Mairal, J. et al, 2009)

This sparse dictionary learning method is computed by two functions *SparsePCA*(), *MiniBatchSparsePCA.*Both are different in computational time and accuracy, where *SparsepPCA(),* is more precise, however requires more computational time than *MiniBatchSparsePCA.* In this experiment, we attempted to perform the first function, however, due to time longevity we had to switch to *MiniBatchSparsePCA* as it is more time efficient, despite the compromises. and To implement *MiniBatchSparsePCA* we used the sci-kit-learn library to call the function. This technique linearly finds the sparse components to reconstruct relevant data when calling the function. During this stage, an issue was encountered where after the calculation the output leads to a complete zero matrix for every weight-applied dimensional reduction.

IV Results:

In this section, we evaluate and compare the results that were achieved with dimensional reduction on CNN. We explore the possible effectivity of every decomposition method and evaluate the value of the experiment. As mentioned above our primary datasets are CIFAR10 and MNIST, we split our result evaluation into main sections for every dataset. For every dataset, we apply the dimensional reduction method: PCA, ICA, and ISOMAP individually and then use cosine similarity to weigh the similarities with the original network. First, we explore the training processes of three main network types: *original_network, small_network, and dr_network(*PCA, ICA, Isomap). Then, as mentioned above we explore the cosine similarity of every generated network, which involves dimensional reduced network, pruned network, dimensionally reduced and pruned network, and compare those similarities with the *original_network* of every dataset. As mentioned above we implemented a small function that produces the dimensional reduction with relevant matrix transformation. We use it to apply it to the relevant networks to retrieve weight matrices and return them as newly transformed weight matrix lists. So far we have found that when performing DR onto the set of vectors without additional matrix transformation, we have a risk of losing data and resulting in an unstable network which eventually would require retraining. As mentioned above, such a method would result in an accuracy of at most 10% and wouldn't go any further

IV.1 CIFAR10 RESULTS:

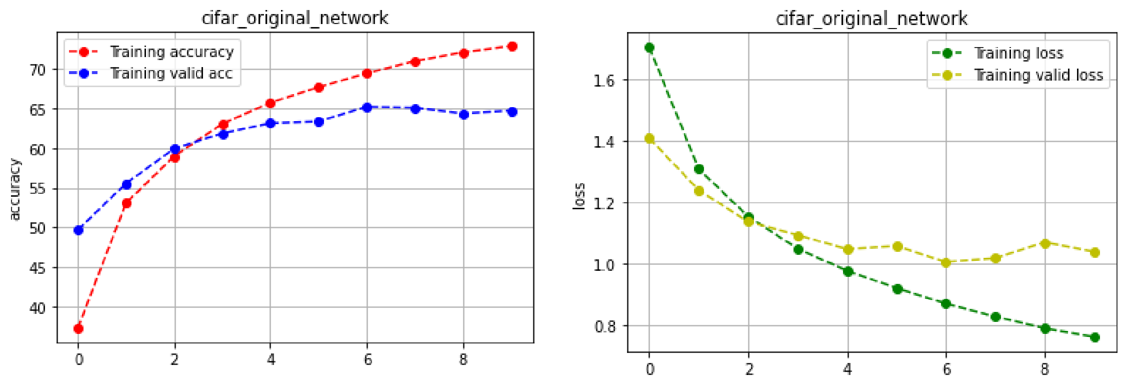


Figure 9 - Training accuracy and loss of cifar_original_network

Here we briefly study the training process of the small convolutional network that was trained using CIFAR10. As mentioned before this network consists of 5 main layers, where the first two layers are *conv1* and *conv2*, where we applied *MaxPool2D* for every convolutional layer. The next three layers are fully connected layers *fc1, fc2, and fc3*, where we perform the non-linear activation function ReLU onto the first two layers. Our dimensional reduction will mainly work on these three layers. For the CIFAR10 dataset, we use 10 epochs for the training process. As it can be stated by observing the learning curve of the accuracy of the *cifar_original_net,* we can evaluate that the training rate of the network increases dramatically from 37.3% off accuracy and dramatically grows to 33.725 and starts gradually increasing up to the highest points which is 72.8%. Based on the visualised results it can be stated that the accuracy rate during training provides a stabilized performance. Furthermore, as it can be observed by the learning curvature it can be seen that both validation data and training data remain somewhat on the same level, hence means we avoided unferfitting and overfitting. Now as we confirmed that our trained network performed relatively well during the training process, it confirms that no potential issues may be caused by the *original_cifar_network*.

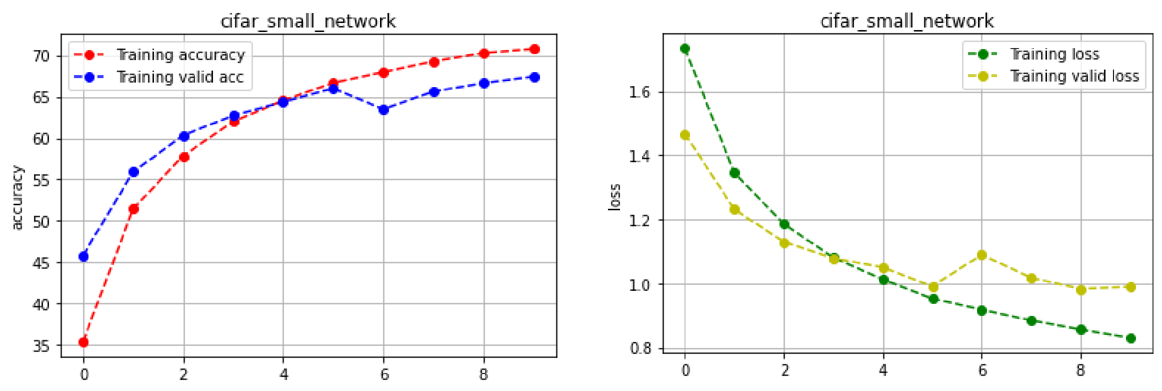


Figure 10 - Training accuracy and loss of small_network_cifar

Here we describe the linear curvature of the training process of *cifar_small_network*. By evaluating Fig. 10, we can see that when comparing it to the original network, the performance of the training is similar to some degree, this insinuates the possibility of initializing a smaller network that has the capability of the performance *original_network*. The *cifar_small_network* has reached an accuracy of 70.0% from 35.4%. When making a comparison with the *original_network* we can analyse the insignificance in the maximum accuracy, where *original_network* has the maximum accuracy of 72.8%. Moreover, the graph(Fig. 10) shows the linear curvature of validation and training loss, which also indicates the stability and performance. The value loss that was reached for the training curvature peaked at 0.83, whereas for validation loss the value has been dropped down to 0.98, which means for the *small_network* no overfitting/underfitting has occurred.

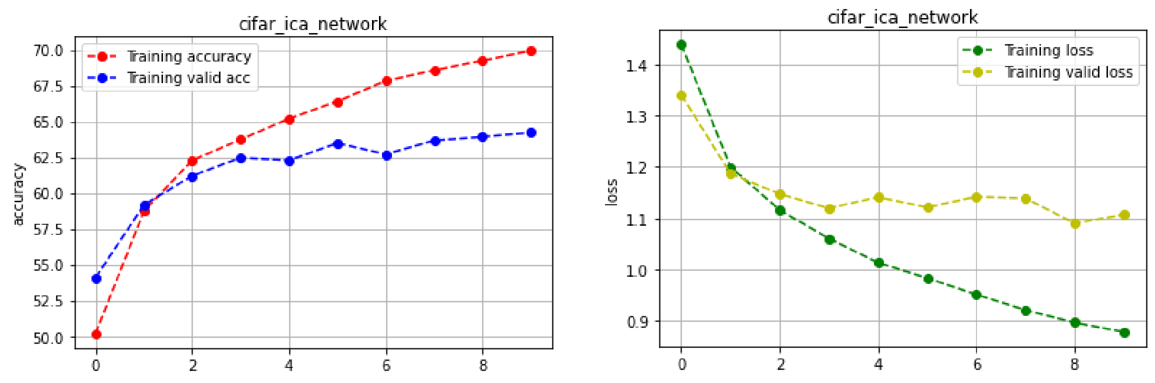


Figure 11 - Training accuracy and loss of cifar_ica_retrained_net

However, when performing the training process after the ICA was applied, we can observe slight instability in the linear curvature(Fig. 11). As the process progresses we can detect the divergence between validation accuracy and training accuracy, such occurrence can potentially lead to underfitting, we might assume due to FC layer reconstruction within the network architecture, this possible lead to a minor underfitting, if when compared *original_network* (Fig. 10), it can be assumed that occurrence unlikely to happen.

For cosine similarity, we take a batch (which in our case batch is equal to 4) sample data of the MNIST dataset and use it to approximate the similarity between two networks by feeding the model the corresponding sample. We then calculate the average of the sample batch, once we get the output of cosine similarity, this way we get an average value per batch.

ica_cifar_network	*Ica_cifar_retrained_network*	*small_network*	*pruned_network*	*pruned_ica_net*
54.399%	86.62%	71.713%	95.29%	45.2%

Figure 12 - Table representing the cosine similarity of each network compared to original_network

The value shown in the table(Fig. 12) represents the score of similarity between vectors based on the angle. As cosine similarity is magnitude-independent it only relies on the angle of the vector. Based on the % coefficient we can approximately determine if vectors are directed somewhat in the same direction, where if it is equal to 0%, this means both vectors are orthogonal hence no similarity can be seen, which is not desired in our case. For the *pruned_network* we used global randomized unstructured pruning. We can analyse the resemblance of the output when comparing *pruned_network(* 95.29%) and *original_network* and conclude that *pruned_network* maintained its prediction creation. When comparing two architectures *ica_cifar_network* and *ica_cifar_retrained_network*, it can be seen that both of these maintained a relatively high score($x \geq 50\%$), where *ica_cifar_network* is 54.4% and ica_cifar_retrained_network 86.62%, thereby make a gap of 30% difference between these two network architectures.

Networks	Average accuracy
cifar_original_network	64 %
ica_cifar_network	32 %
ica_cifar_retrained_network	63 %
small_network	64 %
pruned_network	54 %
pruned_ica_net	36 %

Figure 13 - Table representing the average score for average accuracy for every network

For the evaluation, we calculated the average accuracy of each network after every operation. This way we identify potential losses in the accuracy and conclude the matrix transformation effectiveness. Table(Fig. 13) represents the calculated score for average accuracy using the testing set of the CIFAR10 dataset. The table shows that for *ica_cifar_network* the average accuracy significantly drops originally from 64%(Fig. 13) to 32%, thereby leading to an undesirable score rate. We perform the same ICA decomposition onto the *pruned_network* with an average accuracy of 54%, in result we get 18% lower rate accuracy, however, when measuring the accuracy difference, we can state that *prune_ica_net* has achieved an accuracy that better than *ica_cifar_network by* 3%.

When dealing with we considered another additional method for decomposition. In this instance, we examine the result of dimensional reduction when applying the SparsePCA function to the pruned network only. We can observe(Fig.14) the drastic change in cosine similarity as opposed to other methods. As expected DR method PCA is inclined to negatively affect the overall performance of the network. By that, we insinuate the low coefficient rate in *pca_cifar_network* with a value of 27.16% and *pca_cifar_retrained_net* with a value of 27.16%. We can observe the shift of similarity of *pca_cifar_retrained_net,* despite the high average accuracy of approximately 50%, this indicates the directional difference between the two vectors.

pca_cifar_network	*pca_cifar_retrained_net*	*small_network*	*pruned_network*	*pruned_pca_net*	*Sparsepca_net*
27.16%	27.16%	83.13%	89.31%	31.05%	53.7%

Figure 14 - Table representing the cosine similarity of each network compared to original_network

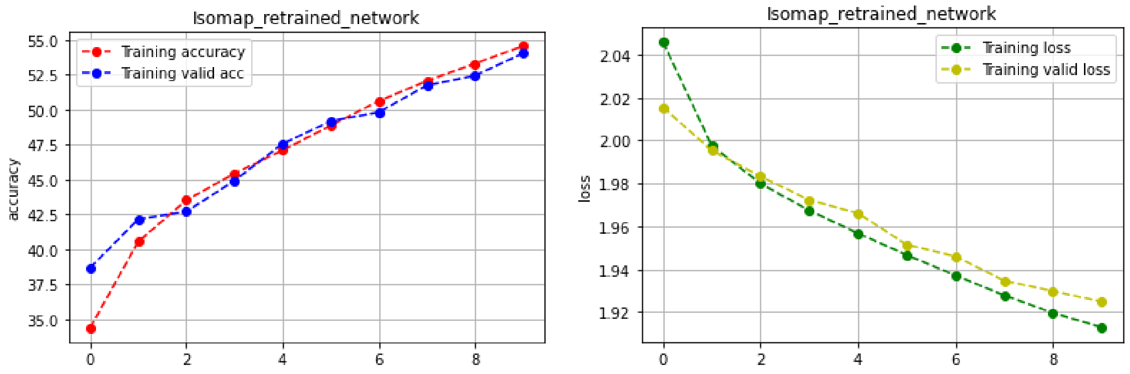


Figure 15 - Training accuracy and loss of ica_retrained_network

The linear curve of the Isomap-based network gives us some insight into the learning process of such a network. We can see the gradual growth of accuracy, and a slow decrease in the training loss/validation loss, which almost gives a straight diagonal line. The diagram displays us a perfect training growth with moderate generalization performance, where the highest accuracy achieved is 54.56% and the lowest point for the loss is 1.92. Despite the gradual deterioration shown by training loss, the error score has slowed down at the 4th epoch, potentially we assume that this is caused by the additional normalization function.

For the Isomap decomposition, we apply the same technique. We use Isomap to transform weight matrices into the lower dimensionality, and if succeed we then use this variable to perform basic matrix transformation for the following layer(see Sec. 3.3). When comparing results for the Isomap and ICA, we can state that ICA performance provided positive output, whereas Isomap results tend to less reliable. After changing the weight matrix of the *origin_network* the average accuracy tends to drop down to at most 30%. Many factors could be considered into the account. One of them might be the learning process of the network or fine-tuning of the hyperparameters. For this scenario, our approach was to run dimension reduction numerous times, especially for the Isomap case, as it has the inclination of shifting the matric and deteriorate the overall accuracy. However when comparing cosine similarity with Isomap, we can examine(Fig.14), the angular similarity of both networks(*Isomap_cifar_net,original_cifar_net).* The *Isomap_cifar_network* displays 50.80% of the similarity.

Isomap_cifar_network	*Isomap_cifar_retrained_network*	*small_network*	*pruned_network*	*pruned_isomap_net*
50.80%	73.00%	71.64%	85.79%	37.30%

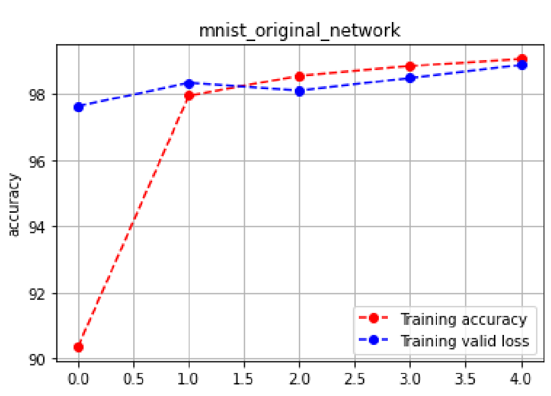
Figure 16 -Table representing the average score for average accuracy for every network

As presented in the table below, we can analyse the average accuracy of the CIFAR10-based network. Based on the data we produced, ICA succeeded in achieving the best result for every scenario. For example, when performing the optimization operation on the cedar-based network with an average accuracy of 66%, we inspect the changes that were made after optimization was done. We reduced the weight matrices of the FC layer by factor 2, hence the given accuracy provided below 32%. Furthermore, when such an operation is performed without corresponding matrix projection(See Sec. 3) it can be confidently stated that the accuracy coefficient dramatically drops to 20%.

Networks	Average accuracy
cifar_original_network	66 %
ica_cifar_network	32 %
ica_cifar_retrained_network	53 %
small_network	64 %
pruned_network	60 %
pruned_ica_net	21 %

Figure 17 -Table representing average score for average accuracy for every network

IV.2 MNIST RESULTS:



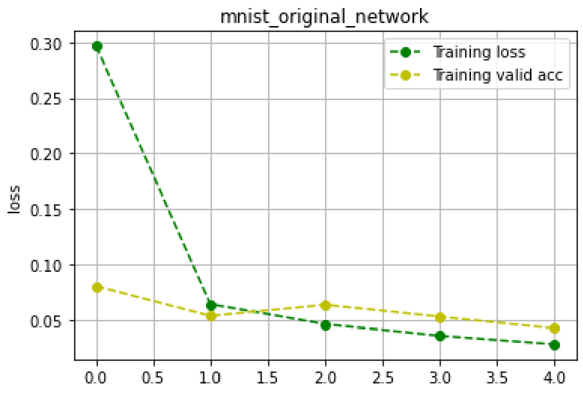
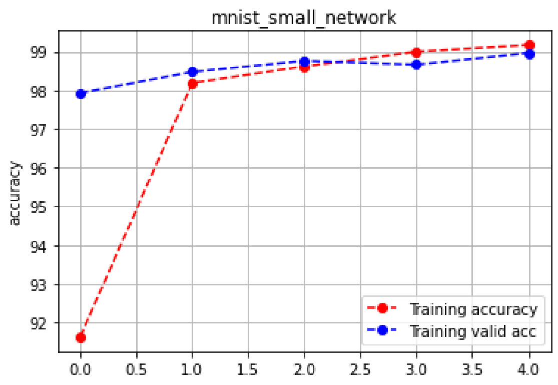


Figure 18 - mnist_original_network learning process visualization

In this section, we ensure the stability and briefly examine the state of the learning process for the neural network that was trained using MNIST.In the case of CNN trained on MNIST, we can observe that the average accuracy achieved per epoch is roughly 90%, while it shows a good learning curve it is believed that the structure of the network is slightly complex for such a relatively simple dataset, if compared to CIFAR10. The visualisation learning curve shown in Fig. 12 it can be seen that for *mnist_original_network* validation curve and training progressively increase while remaining at the adequate level which indicates a good training performance with no overfitting/underfitting, likewise can be said about training loss and validation loss where both of the curves are moderately decreases as desired. The simplicity of the dataset like MNIST should give us a more promising result, as in contrast with CIFAR10.



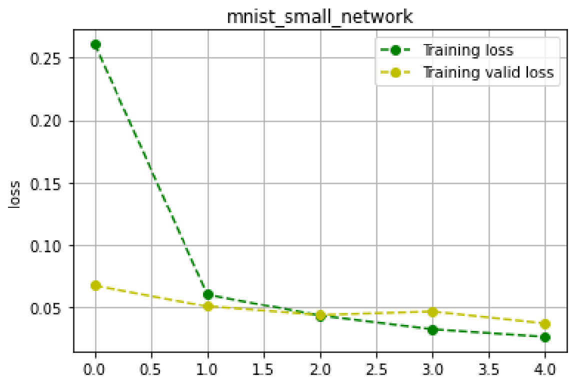


Figure 19 - small_network learning process visualization

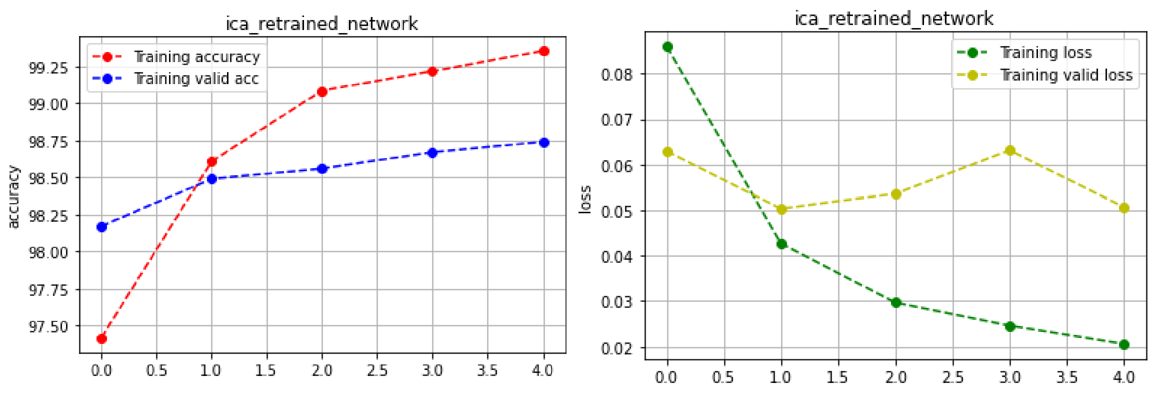


Figure 20 - ica_retrained_network learning process visualization

For cosine similarity, we take sample data from the MNIST dataset and use it to approximate the similarity between two networks by feeding the model the corresponding sample. As we can see the given result for the *ica_mnist_retrained_network* provides a lot more promising results. It can be seen that the highest accuracy value for cosine similarity is the *ica_mnist_retrained_network* with a value of 94.42%, whereas the smallest value is *pruned_ica_net* with a value of 66.15%. This corresponding output results from two-step optimization(pruning, dimensional reduction). We achieved such accuracy without any retraining and saved the model as it is.

ica_mnist_network	*ica_mnist_retrained_network*	*small_network*	*pruned_network*	*pruned_ica_net*
82.79%	94.42%	92.26%	84.4%	66.15%

Figure 21- ica_retrained_network learning process visualization

For the PCA and Isomap, we used *nn. functional.normalize()* to normalize the output, as when performing the dimensional reduction and matrix transformation, this causes instability in the values of the output vector. When analysing the table, it can be evaluated that PCA, does not provide a promising result. As the table(Fig. 22) states, the output of two vectors produced by *original_mnist_network* and networks shown below have little to no resemblance, meaning that the angle between two vectors is close to being orthogonal. The highest value that the table(Fig. 22) shows is *small_network* (24.52%) which we reconstructed and retrained.

pca_mnist_network	*pca_mnist_retrained_net*	*small_network*	*pruned_network*	*pruned_pca_net*	*Sparsepca_net*
12.20%	10.02%	24.52%	23.25%	7.95%	18.95%

Figure 22 - pca_mnist_network learning process visualization

In the instance where we applied Isomap onto corresponding networks, it was expected for the non-linear optimization method to perform relatively poorly, however, when testing average accuracy for the *Isomap_cifar_network*, the accuracy coefficient drops significantly, whereas for the CIFAR10 dataset *isomap_network* has reached the accuracy of approximately 30% while for the MNIST it was decreased down to 18%. There are many factors that may contribute to such a drastic decrease, some might assume it is due to poor coding issues, or fine-tuning of the network through hyperparameters. As it can be expected highest cosine similarity value was *small_network* (87.17%)¸ *followed by pruned_mnist_network* with a value of 77.19%.

Isomap_mnistr_network	*Isomap_mnist_retrained_network*	*small_network*	*pruned_network*	*pruned_mnist_net*
54.02%	64.75%	87.17%	54.07%	77.19%

Figure 23 - pca_mnist_network learning process visualization

V Discussion:

In this section, we examine the results and analyse them with the objectives that were established for the project. Furthermore, we evaluate the wider perspective of the theoretical and the relevant work that has been applied to the project itself. When experimenting, numerous dimensional reduction techniques could be considered for application, In this instance, our approach was to consider every type of dimensionality reduction to observe the effect on the stability of the network, whether it's positive or negative, however, due to time limitation not all of them were considered. Furthermore, using a combination of decomposition methods and the matrix transformation method mentioned before, we achieve information preservation, thus we avoid any unnecessary retraining of the network and bring potential compactness into the network.

V.1 Apply different pruning techniques to the network:

The basis of this objection lies in the utilization of possible pruning techniques to sparsify the corresponding neural network. Unlike, dimensional reduction which is the umbrella term for different mathematical techniques to optimize multivariate data for data analysis and data mining, pruning is an individual technique with different approaches. In this instance, we used global pruning using the PyTorch library to create a mask to combine it with the parameter and store it as a weight attribute. Global pruning prunes assign layers separately at a constant rate, where we prune *cifar_original_network* and *mnist_original_network* globally. When there is a difference in the number of parameters between the layers, that might lead to the "bottleneck" caused by simultaneous pruning of all layers, hence global pruning solves the issue. Such operation was widely researched by Frankle, J. et al (2009), who examined the idea of different approaches of pruning like iterative pruning, and one-shot pruning, to find a *'winning ticket'*, which is a contextual terminology for the subnetwork. The purpose of the paper was to achieve the closest accuracy of the original network with the *winning ticket*. By utilizing the state-of-the-art technique using PyTorch, we implemented the combination of global pruning in a random unstructured way.

V.2 Apply different methods of dimensionality reduction

The principle of the objective is to examine the optimization methods from a different perspective. Most studies focus on investigating the dimensionality reduction methods primarily as a pre-processing step for the raw. Our approach lies in the post-processing approach where we take a pre-trained model to optimize the parameters of the network without the possibility of retraining. Due to time limitations, not every proposed dimensional reduction technique has been implemented. Also, as every method mathematical approach of this technique differs from another, we use the one that has been the most widely researched. First, we used a principal component analysis. Principal component analysis or PCA, in short, is a widely recognized dimensional reduction that uses the linear transformation of the data and converts it to the low-rank matrix that has the maximized variance to describe multivariate data, by rotating the axes of the vectors (Hatipoğlu, G. et al 2022). Principal Component Analysis generates a covariance matrix, and then we use it to find eigenvectors and eigenvalues so that it finds principal components. These PCs are then used as weights to project to smaller dimensions.

VI Evaluation

In this section, we conclude and evaluate the project progress and the changes that have been made. Furthermore, we examine the main objectives and the aims that were set for this study. Considering the provided objectives for the study, it can be stated that most of the objectives of the study were achieved throughout the experiment.

In the research, we examined the possibility of finding the effectiveness of dimensional reduction using on trained unpruned neural network and on pruned trained neural network. For this experiment, we only considered several decompositions like PCA, ICA, SparsePCA, and Isomap, where Isomap is a nonlinear dimensional reduction, and the remaining are linear techniques. Furthermore, we used simplified custom-built CNN, that is if the experiment provides good results such a technique can be reapplied to much bigger pre-trained CNNs like ResNet-50 or VGG16 for instance. In our case, we believe that we got relatively good results, yet still require further development. However, it is a known fact that the primary function of dimensional reduction is to be applied for multivariate raw data, which is used for data analysis and optimisation, whereas we approached the technique from a different perspective. Through such a technique we emphasize the potential possibility of CNN optimization through dimensional reduction, and pruning. Based on the observation it can be concluded that decomposition techniques incline affecting the network positively by weight matrix modification. Using ICA transformation in the study presented positive results among other dimensional reduction techniques. The weight matrix decomposition of the *original network* trained on MNIST using ICA produced an accuracy of 77% from 98%.

VII Reflection

As a person, I have always been fascinated by the concept of neural networks due to their simplicity yet effective output. I believe this project has helped me to familiarize myself with such concepts. This project has been a great opportunity for me to learn about machine learning and to strengthen my knowledge and skills in the theory of deep neural networks. As a software engineering student, I lacked any background knowledge in the data science-oriented field; however, the project provided me with some growth in my theoretical background, which I am looking forward to using in a professional environment. Furthermore, it helped to improve my practical skills in coding, particularly in Python and PyTorch.

A significant amount of research and effort went into this project, from the initial concept to the final execution. Through my research, I have gained and developed a fundamental understanding of machine learning models and the various theoretical approaches required to support the project, such as PCA, ICA, Isomap, Pruning, etc. These skills will provide me with a skill set that will help me in relevant fields like data engineering or data science. Furthermore, this project helped to develop a range of skills which I believe will aid in my employability.

VIII Conclusion

In this project, we examine the potential effectiveness of dimensional reduction techniques on pruned and unpruned neural networks. For the project, we utilized non-linear and linear decomposition methods such as Principal Component Analysis, Independent Component Analysis, Isomap, and SparsePCA. In this paper, we demonstrate a new potential optimization solution that is open to future development. We showcase a potential step toward a broader solution by providing an analysis of rigorous experiments with optimization methods. We argue by comparing dimensional reduction with a newly initialized neural network, thereby questioning if a smaller network can be built instead. However, our primary goal was to eliminate retraining from the equation, so that we avoid computational time and cost. Such a project has to be approached with an experimental mindset, hence there are always methods or algorithms that can be found and utilized for the same purpose. As we covered only three methods, there is the possibility for future development of the project. The experiment can be extended to multiple dimensional reduction techniques (linear and non-linear) like MDS, NMF, KernelPCA, etc.

The implementation of the code and the network created for the analysis can be found in **Appendix B**, which is the link to the GitHub account.

IX References:

Anowar, F., Sadaoui, S. and Selim, B., 2021. Conceptual and empirical comparison of dimensionality reduction algorithms (pca, kpca, lda, mds, svd, lle, isomap, le, ica, t-sne). Computer Science Review, 40, p.100378.

Artoni, F., Delorme, A. and Makeig, S., 2018. Applying dimension reduction to EEG data by Principal Component Analysis reduces the quality of its subsequent Independent Component decomposition. NeuroImage, 175, pp.176-187.

Blalock, D., Gonzalez Ortiz, J., Frankle, J. and Guttag, J. (6AD). WHAT IS THE STATE OF NEURAL NETWORK PRUNING?.

Cao, L.J., Chua, K.S., Chong, W.K., Lee, H.P. and Gu, Q.M., 2003. A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine. Neurocomputing, 55(1-2), pp.321-336.

Carreira-Perpiñán, M., Eecs and Idelbayev, Y. (n.d.). 'Learning-Compression' Algorithms for Neural Net Pruning. [online] Available at: https://faculty.ucmerced.edu/mcarreira-perpinan/papers/cvpr18.pdf [Accessed 22 May 2022].

Carreira-Perpinán, M.A., 1997. A review of dimension reduction techniques. Department of Computer Science. University of Sheffield. Tech. Rep. CS-96-09, 9, pp.1-69.

d'Aspremont, A., Ghaoui, L., Jordan, M. and Lanckriet, G., 2004. A direct formulation for sparse PCA using semidefinite programming. Advances in neural information processing systems, 17.

Frankle, J. and Carbin, M. (4AD). THE LOTTERY TICKET HYPOTHESIS: FINDING SPARSE, TRAINABLE NEURAL NETWORKS. [online] arxiv: Cornell University. Available at: https://arxiv.org/pdf/1803.03635.pdf [Accessed May 21AD].

Geng, X., Zhan, D.C. and Zhou, Z.H., 2005. Supervised nonlinear dimensionality reduction for visualization and classification. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 35(6), pp.1098-1107.

Gonzalez, C.A.G., Wertz, O., Absil, O., Christiaens, V., Defrère, D., Mawet, D., Milli, J., Absil, P.A., Van Droogenbroeck, M., Cantalloube, F. and Hinz, P.M., 2017. Vip: Vortex image processing package for high-contrast direct imaging. The Astronomical Journal, 154(1), p.7.

Hatipoğlu, G., 2022. Utility of PCA and Other Data Transformation Techniques in Exoplanet Research. arXiv preprint arXiv:2211.14683.

Hout, M.C., Papesh, M.H. and Goldinger, S.D., 2013. Multidimensional scaling. Wiley Interdisciplinary Reviews: Cognitive Science, 4(1), pp.93-103.

Hyvärinen, A. and Oja, E. (2000). Independent component analysis: algorithms and applications. Neural Networks, 13(4-5), pp.411–430. doi:10.1016/s0893-6080(00)00026-5.

Hyvärinen, A. and Oja, E., 1997. A fast fixed-point algorithm for independent component analysis. Neural computation, 9(7), pp.1483-1492.

Kambhatla, N. and Leen, T.K., 1997. Dimension reduction by local principal component analysis. Neural computation, 9(7), pp.1493-1516.

Lai, Z., Xu, Y., Chen, Q., Yang, J. and Zhang, D., 2014. Multilinear sparse principal component analysis. IEEE transactions on neural networks and learning systems, 25(10), pp.1942-1950.

Lee, J.A., Lendasse, A. and Verleysen, M., 2004. Nonlinear projection with curvilinear distances: Isomap versus curvilinear distance analysis. Neurocomputing, 57, pp.49-76.

Mairal, J., Bach, F., Ponce, J. and Sapiro, G., 2009, June. Online dictionary learning for sparse coding. In Proceedings of the 26th annual international conference on machine learning (pp. 689-696).

Malach, E., Yehudai, G., Shalev-Shwartz, S. and Shamir, O. (2020). Proving the Lottery Ticket Hypothesis: Pruning is All You Need. [online] Available at: https://arxiv.org/pdf/2002.00585v1.pdf [Accessed 22 May 2022].

O'Shea, K. and Nash, R., 2015. An introduction to convolutional neural networks. arXiv preprint arXiv:1511.08458.

Rahutomo, F., Kitasuka, T. and Aritsugi, M., 2012, October. Semantic cosine similarity. In The 7th international student conference on advanced science and technology ICAST (Vol. 4, No. 1, p. 1).

Reddy, G.T., Reddy, M.P.K., Lakshmanna, K., Kaluri, R., Rajput, D.S., Srivastava, G. and Baker, T., 2020. Analysis of dimensionality reduction techniques on big data. IEEE Access, 8, pp.54776-54788.

Ringnér, M., 2008. What is principal component analysis?. Nature biotechnology, 26(3), pp.303-304.

sklearn.decomposition.FastICA (2022). Available at: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.FastICA.html.

sklearn.decomposition.PCA (2022). Available at: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html.

Tenenbaum JB, Silva VD, Langford JC. A global geometric framework for nonlinear dimensionality reduction. science. 2000 Dec 22;290(5500):2319-23.

Wang, J. and Chang, C.I., 2006. Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis. IEEE transactions on geoscience and remote sensing, 44(6), pp.1586-1600.

Zou, H., Hastie, T. and Tibshirani, R., 2006. Sparse principal component analysis. Journal of computational and graphical statistics, 15(2), pp.265-286.

Appendix A: Effectiveness of Different Dimensionality Reduction Techniques on Pruned Deep Neural Networks

Name:Sultangazy Yergaliyev

Student ID: 210050508
Email: Sultangazy.Yergaliyev.2@city.ac.uk
Phone: (+44) 7412 303 606
Supervisor: Dr Tillman Weyde

A.1 Introduction:

In recent years, Deep Neural Network has improved tremendously, and so has the demand for computational power to train a model for efficient inference. In order to make efficient use of hardware resources (CPU & GPU), it is promising to develop techniques that reduce the size of neural networks. Pruning techniques, eliminating nodes with insignificant weights, has been shown to reduce the size in some cases up to 90% per cent without significant loss of performance (Carreira-Perpinan et al, 2018).

Pruning algorithms are used to eliminate insignificant nodes as mentioned above, this indicates that we find the weight of nodes, which are relatively close to being zero and equalise them to zero. Pruning is a form of compression used for neural networks, where unpruned weights $w \in \mathbb{R}^n$are compressed into sparse weights $0 \in \mathbb{R}^n$**,** which means results in satisfactory results depending on conditions (Carreira-Perpinan et al, 2018). This results in the sparse neural network, which is a network architecture with mostly zero parameters.

The main problem with sparse networks is that they their efficiency is much worse than dense networks on current standard hardware (CPU and GPU) for similar numbers of non-zero parameters. Therefore we ask the following research question in this project:

Research Question: How can we combine pruning and dimensionality reduction to large neural networks so that we obtain small dense neural networks with little loss of performance?

The Objectives:

As far as the study goes we have identified the objectives that help the research to withstand:

- Find suitable datasets and pre-trained networks
- Apply different pruning techniques to the network
- Apply different methods of dimensionality reduction on the network's weight matrices
- Explore methods for reducing the neural layer if there is time
- Implement and test the methods on different networks with different datasets
- Evaluate and analyse results

There are numerous papers on pruning techniques and dimensionality reduction techniques separately. This study will attempt to combine different algorithms for pruning and dimensional reduction to see the compatibility and the ability to reduce network size without reducing performance, thus reducing in the computational cost. This study is novel in answering the question, if this solution can be applied to the practical scenario,

A.2 Critical Context:

According to the paper (Blalock et al., 2020), the way that neural network can be best described as an aggregation of functions *f(x; ·)*. The way architecture operates, it configures the parameters based of the input and produce relevant output, this involves arrangement of parameters into convolutions, activation functions, and etc (Blalock et al., 2020) The paper describes many techniques, which result in pruned models. These pruning approaches mainly follow the principle of Algorithm shown in the Figure 2. The essence of pruning is the elimination of the unnecessary connection and followed by fine-tuning of the network after the model has been trained with trained data. As was already mentioned pruning might have multiple objectives, however different purpose requires different evaluation techniques for pruning. Thera are multiple metrics that are used for evaluation: Accuracy, Computational time, and size. Accuracy determined by efficiency of the model prediction and computational time as the name suggests a metric to evaluate overall time it takes to perform the training. With regard to dimensional reduction, it is a mathematical methodology responsible for data visualisation and its pre-processing for machine learning. It is a technique that searches for lower dimensions within the matrix while preserving the data information. It is a statistical paradigm that has gained popularity due to its effectiveness, such analysis is usually performed before feeding information into the model (Carreira-Perpinan et al, 2019).

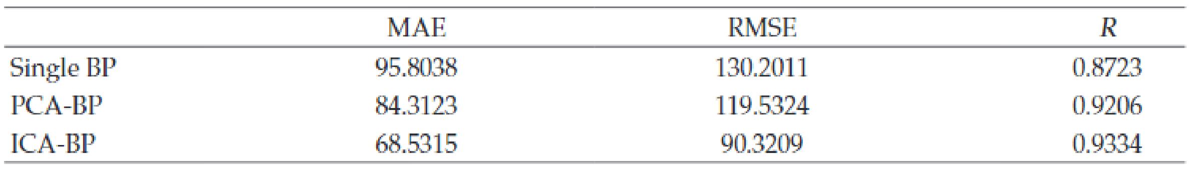
Lottery Ticket Hypothesis:

Melach et al (2020) published a paper called the Lottery Ticket Hypothesis, in which they try to prove the effectiveness of the pruning technique on the whole network. They claim that magnitude-based pruning and second derivatives based pruning, while positively achieving overall compression expectations, greatly fails in computational cost efficiency. This hypothesis supposes that in large dense networks only some parts (subnetworks) are needed for achieving the same accuracy as the whole network (Frankle, Carbin, 2018). Hence, it means that after the training of multi-parameter network is done , it is suggested that having the dense network is unnecessary, as a subnetwork can be used due to its simplicity and infer just as good. The experiment performed in the research failed to utilise big dataset like ImageNet due to computational cost intensity as it requires multiple trials. Being the computational cost the focal points and the concern, it can be assumed that applying both optimization techniques like dimension reduction and pruning can eliminate that inconvenience. Assumingly, this theory can be used in a practical manner in such way that the convoluted network can be dimensionally reduced and pruned while preserving the required accuracy, thus this may potentially avoid the excessive computational cost.

ICA & PCA:

Ramachandran, Ravichandran and Raveendran (2020) compare different dimensional reduction methodologies, to pick the most efficient option for the reduction. Furthermore, the study goes into more depth and provides a detailed elaboration on these methods. According to the paper, PCA which stands for Principal Component Analysis, is utilized for the reduction of the higher dimensional matrix and lowers it from M-matrix to N-matrix. This analysis has several stages of calculations, First, it computes the standard deviation of the square root of the variance in order to retrieve the diffusion of the value. Then, it finds the difference between the dimensionality of the standard deviation and its mean. Afterwards, it does the analysis of the formation of the matrix based on the Eigenvector that is visualised on the graph in a clustering manner. PCA achieves its accuracy via using small eigenvalues, it separates small eigenvalues by creating the threshold, where it calculates the result by dividing the sum of eigenvalues by the sum of eigenvalues that are threshold values. Another technique that the paper has covered in depth is Independent component Analysis (ICA). ICA is the technique that splits unsorted signals. it can be stated that PCA is the type of analysis that does the dimensional reduction in a linear pattern just like ICA.

In addition to the previous paper, Liu and Wang (2011) attempt to use these two-dimensional reduction perspectives on neural networks and compare the final result to analyse the effectiveness of each algorithm. Furthermore, it separates models into three types BP, ICA-BP, and PCA-BP for the observation of the performance based on these three metrics MAE (mean absolute error) and RMSE (the root mean square error) and correlation coefficient (R).



	MAE	RMSE	*R*
Single BP	95.8038	130.2011	0.8723
PCA-BP	84.3123	119.5324	0.9206
ICA-BP	68.5315	90.3209	0.9334

Figure 10 metrics measures (Liu and Wang, 2011)

This table is one of the examples that used SHCI to show the metrics calculated using three different models. This paper mainly uses two datasets to evaluate the results and see how these error values might differ. In that research, it was concluded that according to the values shown in Figure 1 the model that shows the best performance is ICA-BP, as it has the smallest MAE RMSE and highest R unlike its competitors (Liu and Wang, 2011). Furthermore, after conducting another experiment the conclusion of the results stated that the overall performance of the PCA-BP starts to be closer to ICA-BP and the reason behind that s due to the fact that PCA is based on Gaussian assumption, while ICA is not. This paper compares several compression methods and uses one model for the training. This research will be carried out in a somewhat similar fashion. However, theoretically using these dimension reduction techniques used on the non-pruned network might have a chance of having a better coefficient correlation on the pruned network because of the implementation of the “double optimization” using pruning and dimensional reduction together.

Meneghetti, Demo and Rozza (2021) entertains the idea of utilizing Active Subspace (AS) and Proper Orthogonal Decomposition (POD) reduction techniques that are widely used for the convoluted neural network.

<u>Active Subspaces:</u> AS method finds the appropriate route in the parametrical space in order to use the gradient of the function of interest. It stimulates the rotation of the direction to decrease the dimensionality of the network by obtaining the approximation.

<u>Proper Orthogonal Decomposition:</u> this type of reduction operates in the way that it lowers the number of parameters of the networking system.

Using VGG-16 model the research (Meneghetti, Demo and Rozza, 2021) trains the dataset using CIFAR-10 and attempts to apply those reduction techniques. Based on the results provided by the research paper it can be observed that on average accuracy for the CIFAR-10 dataset is 77.98% while for the customed dataset was 95.65%. Alongside, the reduction methods, the library called ATHENA was used to implement the function Frequent Direction method. This method was designed to calculate AS. For the CIFAR-10 it can be noted that AS and FNN have reached the most accurate level it can be seen that training time was 5 hours. Whereas for custom dataset the most accuracy has gained POD and FNN with the time of 12 min. The research paper has concluded that both cases have given similarly reduced CNN and given similar accuracy.

Learning the parts of objects by non-negative matrix factorization:

The dataset that this experimental research used for face image data and text based dataset for semantic analysis, it showed that NMF algorithm can be applied onto different types of datasets. The methods were used for $m = 2149$ of images of faces, where each consisting of $n = 19x19$ pixels each (D. Lee et al, 1999). Within the context of the research $n \times m$ V matrix is represented as image dataset where n is positive pixel values and m is facial images. Furthermore, V gets split into two approximate factors W and H where each matrix has a dimensions of $n \times r$ and $m \times r$ where r is an arbitrary number as long as $(n + m)r < nm$ (D. Lee et al, 1999). Due do non-negative nature of NMF method, it does not access negative variables, hence additive allowed only, this results in sparse matrices of W and H. the linear scale of the grayscale intensities were adjusted in order for a standard deviation would be equal to 0.25 (D. Lee et al, 1999). This paper performs comparative research of three matrix factorization based techniques like: PCA, NMF, VQ, insinuating on NMF's effectiveness. While it generally goes through each technique by applying into the image, and text based data, where it relatively fails, is in providing the numerical comparison among the algorithms.

A Global Geometric Framework for Nonlinear Dimensionality Reduction:

According to this journal article written by J. B. Tenenbaum, its purpose is to provide newly the algorithms for computing quasi-isometric, low-dimensional embedding of set of high-dimensional data points, which goes by the name ISOMAP. Its presents its effectiveness by comparing somewhat similar techniques like PCA and MDS, where MDS is an extension of this method. For the experimental purpose this paper uses two image datasets where first input image is composed of 4096 dimensional vectors and 64x64 pixel size image showing a combination of face angles with diverse lighting angles. The second dataset the paper used for the research is MNIST dataset. While this is a relatively old paper and it covers the algorithmic essence in depth, it fails to go into more details about PCA and MDS methodology, and on why the complexity of nonlinear geometry of the data barriers PCA and MDS from being utilised for dimension reduction. Likewise, paper presented by J. B. Tenenbaum fail in providing the information about the efficiency of such algorithm with other manifold learning techniques which might be more applicable as PCA classifies as linear method. (J. B. Tenenbaum, 2000).

Multidimensional Scaling by Optimizing Goodness of fit to a Non-metric Hypothesis:

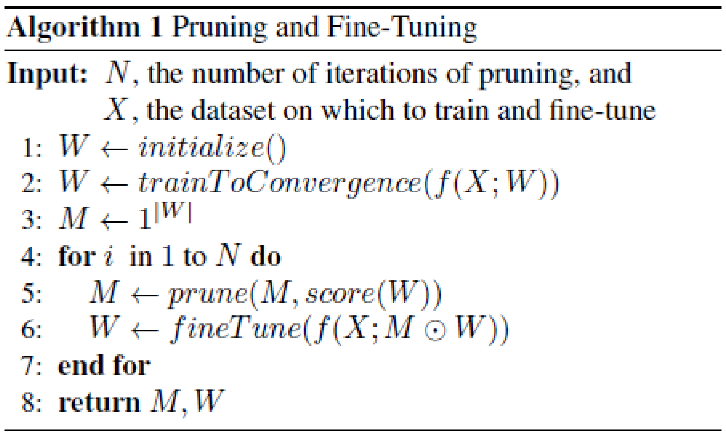
Research made by J. B. Kruskal covers the similarity of several papers, however where it mostly focuses on, is the research developed by R. Shepard. Its purpose dedicated to using R. Shepard MDS technique as a basis for inspiration and potentially improve the methodology. By developing the relative definition of “stress”, the paper tries to achieve best goodness of fit based on the “stress indicator, in other words the solution is to find best-fitting configuration of points, which have the least amount of "stress" (J. B. Kruskal, 1964). This papers concludes by stating that the research resulted in configuration similarities with R. Shephard's techniques, however yield a smoother looking curves for dissimilarity versus distance. Although the papers is a reliable source for mathematical fundamental of MDS, however the due to the recent spark of machine learning it can be assumed the result that this paper provides alongside using "stress" as a measuring pivot, might be irrelevant, as there are more effective way of MDS implementation.

A.3 Approach:

Dataset and Tools:

This research will use CIFAR-100, ImageNet, COCO, and DrivFace Dataset as the main dataset to implement relevant experiments. Furthermore, this research will use PyTorch as the main library for model training, and Other different Python libraries, like MatPlotLib for the data visualization and SciKitLearn for dimensional reduction implementation. The main environment that will be worked on is Jupyter Notebook. Moreover, this research will use VGG-19 and ResNet-56 models. Moreover PyTorch will be used for FeedForward neural network implementation.

Pruning Techniques:



Algorithm 1 Pruning and Fine-Tuning

```
Input: $N$, the number of iterations of pruning, and
       $X$, the dataset on which to train and fine-tune
1: $W \leftarrow initialize()$
2: $W \leftarrow trainToConvergence(f(X;W))$
3: $M \leftarrow 1^{|W|}$
4: for $i$ in 1 to $N$ do
5:     $M \leftarrow prune(M, score(W))$
6:     $W \leftarrow fineTune(f(X;M \odot W))$
7: end for
8: return $M, W$
```

Figure 11 Pruning Algorithm

The algorithm illustrated above (from Blalock et al, 2020) shows the essence of how the logic for the pruning methods works. This algorithm goes through iterations in order to find the similarity and eventually outputs the expected score for each parameter, thus suing those to prune parameters accordingly. The procedure usually has two stages. First, it prunes, losing its accuracy, and then it performs fine-tuning. These operations are iterated numerous times until the desired result is achieved. Blalock et al (2020) provide the instructions on how to implement the open source library for pruning techniques using different methods. ShrinkBench is a PyTorch library that automatically prunes the network and computes all required metrics for the pruning neural network. This metric involves the calculation of all models, datasets, and the level of pruning. Blalock et al (2020) state that ShrinkBench provides implementations for pruning methods in a heuristic manner.

Global Magnitude Pruning: seeks to find for value closest to zero within the network and then does the pruning.

Layerwise Magnitude Pruning: seeks to find the value closest to zero within each layer of the network and prunes it.

Global Gradient Magnitude Pruning: finds the node with the weights based on the value closest to zero

Layerwise Gradient Magnitude Pruning: finds the value closest to zero in each layer of the network and then prunes it.

Random Pruning: randomly finds the weights of the node in the network based on the probability.

Dimensional Reduction Techniques:

For the dimensional reduction, this study will focus on the techniques ICA and PCA, to implement the algorithms for this technique using the SciKit-Learn library. This library provides two main functions that allow to implementation of the algorithm.

FastICA: The way this function works is that it uses the fixed-point iteration scheme that looks for a max of non-Gaussianity(A. Hyvarinen et al, 2000)

PCA: This is a function that performs linear dimensionality reduction that uses Singular Value Decomposition, it uses to decrease the dimensionality. PCA function uses LAPACK library which is the library used for numerical linear algebra. (SciKitLearn, 2022)

Non-Negative Matrix Factorization: The way the function works is that it automatically supposes that the values are non-negative. It takes the matrix *V* for instance and factorizes them into *WxH* matrices under the condition that these values are non-negative. (D. Lee et al, 1999). This condition eases the interpretation of involved quantities, that PCA, and ICA have difficulties with. (P. Hoyer, 2004)

Isometric Mapping: It is a quasi-isometric, non-linear dimensional reduction approach performed in manifold learning manner in which the technique is broken down into several algorithms. It finds closest neighbour of each point in the radius by using KNN, whereafter creates a neighbourhood graphs of the weighted nodes, this helps to approximate the geodesic distance between two points via pathfinding algorithm (Dijkstra’s and etc.). It then performs MDS embedding. (H. Park, 2012).

Multidimensional Scaling: Multidimensional Scaling or MDS is a visualisation technique that maps the data points across the Cartesian space using Proximity matrix. Proximity Matrix is created via calculating pair-wise distance among data points using different methodologies like Euclidian or Manhattan distance. The output result gives the clustering of data points based on the distance. It can be metric and non-metric. (H. Park et al, 2010)

Factor Analysis: The number of variables *(X1, X2,..., Xp)* and the number of underlying factors (m) are denoted in the mathematical model known as "classical factor analysis" *(F1, F2,...,Fm).* The variable *Xj* is what latent factors represent. Because each observed variable is a linear function of each of the m underlying components in this model, together with a residual variate, it is assumed that there are m such factors. The maximal correlations are what this model aims to reproduce. When performing a calculation, Factor Analysis uses algebraic matrix. It finds the correlation coefficient among two variables that indicates its relationship.

Locally Linear Embedding: Assuming the data points are mapped in a patched manner in a manifold space, this function LLE works in such way that it finds the neighbour vectors of *X* and creates the neighbourhood by constructing the weights *W* of the vectors, if *X* is not part of the neighbourhood, thus means *W=0.* Then, it performed eigenvector optimization in order to search for low-dimensional embedding data points. (Roweis, S.T. et al, 2000)

This is an experimental research that will be conducted quantitatively, consequently, the research will obtain the relevant data and perform several methodologies and experiments on the dataset for data analysis. The approach of conducting the research using the library mentioned above is split into two steps, using the dataset like CIFAR-10 for example, and then using two types of dimension reduction(Linear, and Non-linear) techniques. For instance, the algorithm computes the ICA by reconstructing the signals and getting the approximate mixing matrix of the provided dataset. Then, ICA is proof is performed within the algorithm by reverting the unmixing matrix. The algorithm approach for the PCA techniques is similar to ICA, but reconstruction is performed based on ontological components. Like PCA, the NMF method factorises the matrix into two components and reconstructs the matrix with a lower dimension. We perform the training of the model by giving input variables into the model without dimension reduction, then when the fitting is done the model Is pruned and visualized by using matplotlib library. After the procedure completion, we perform another experiment but with dimension reduction and then compare relevant results. As our main objective is neural network optimization, the predictive modelling problem that the network resolve is quite irrelevant, hence simple classification will be performed based on the image input.

Analysis and Evaluation:

After the data collection, the next process is to analyse the data and conclude whether if research has met the expected requirements and if the study has answered the stated question.
The research will attempt to conduct several experiments to compare the results with different models and different techniques. As the main purpose is to reduce the computational cost, the First stage of the analysis is to visualise the result of the pruned network. We use this data to compare networks that are dimensionally reduced and collect the information to use for further network analysis. This study will approach the problem in a quantitative manner, which is the method of measure.
The next stage of the analysis is to perform both of the compression techniques on the network to reduce the matrices after pruning. As the main purpose is generally to improve the computational cost we look for three main components in the result. The study will mainly use accuracy, to evaluate the inference and ensure we avoid overfitting and underfitting. In the study, we also test the overall computational time for each training and use relevant libraries for plotting graphs. Calculation of total time will be performed during the training process of the training. Conducting the accuracy calculation will be through every epoch, this way the training process can be monitored throughout the process.
PyTorch library provides a parameter function that gives the number of parameters of the specified model. However, as the library does not provide the calculation of the total number of parameters, this sort of logic will require manual calculation. This will help the research to conduct a relevant comparison of different models by graph visualisation via MatPlotLib

library. Computational procedures will be conducted in the clusters of the City, University of London.

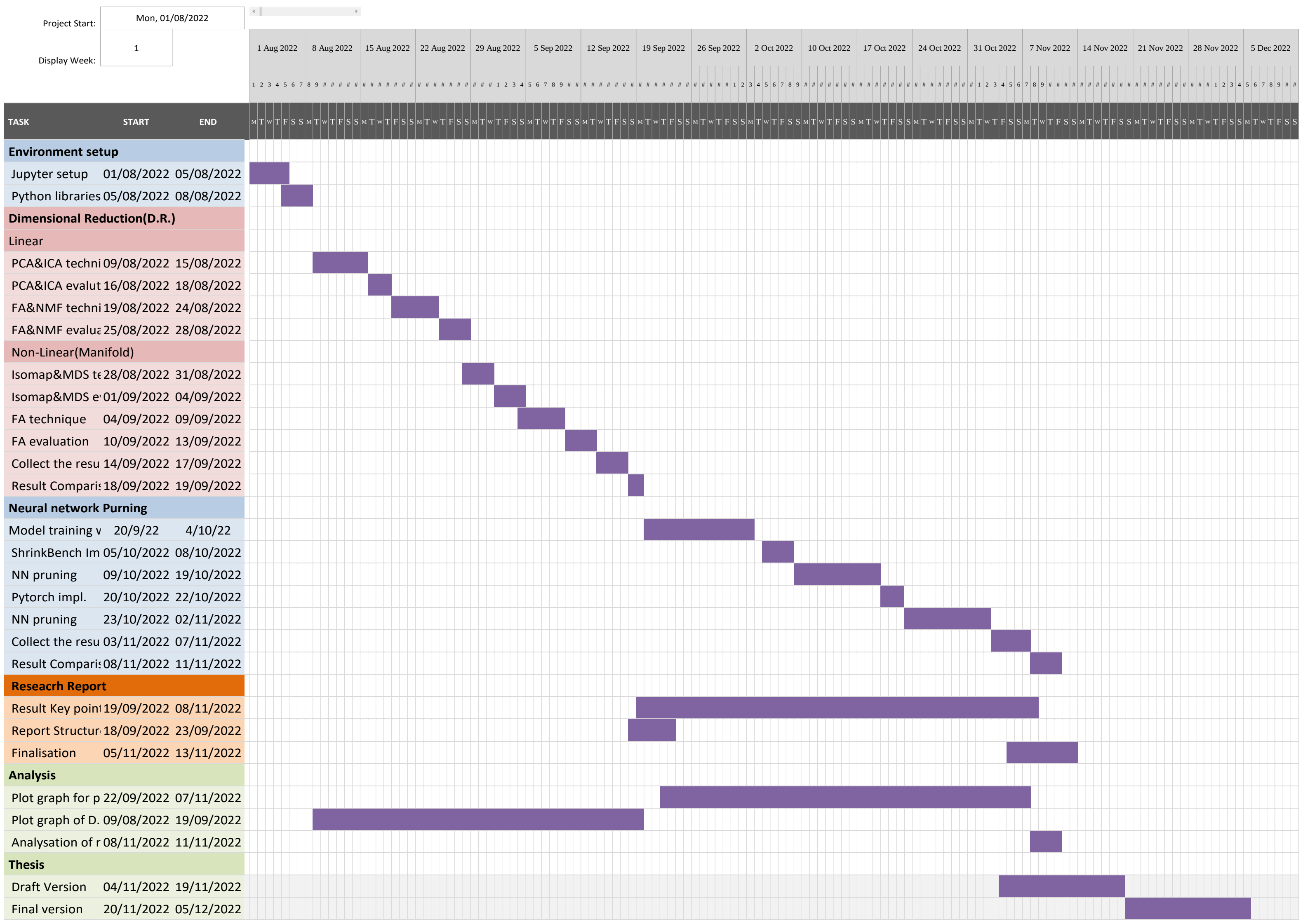
Limitations:

Potential limitations lie beneath the algorithm and how they operate on a certain network. Some pruning algorithms tend to prune unnecessary weights of the node or some have the tendency to prune irrelevant hidden matrices. However, one of the major possible issues is low efficiency. This means that highly accurate pruning methods have to compensate for this with the computational time, this result might greatly affect the overall objectives of the research, as the study aims for improving the computational efficiency.

Ethics & Legal Concerns:

As the dataset that will be used are mainly open source and are available to anyone, hence it indicates no ethical concerns can arise.

A.4 Work Plan:



Project Start:	Mon, 01/08/2022
Display Week:	1

TASK	START	END
Environment setup		
Jupyter setup	01/08/2022	05/08/2022
Python libraries	05/08/2022	08/08/2022
Dimensional Reduction(D.R.)		
Linear		
PCA&ICA techni	09/08/2022	15/08/2022
PCA&ICA evalut	16/08/2022	18/08/2022
FA&NMF techni	19/08/2022	24/08/2022
FA&NMF evalua	25/08/2022	28/08/2022
Non-Linear(Manifold)		
Isomap&MDS te	28/08/2022	31/08/2022
Isomap&MDS e	01/09/2022	04/09/2022
FA technique	04/09/2022	09/09/2022
FA evaluation	10/09/2022	13/09/2022
Collect the resu	14/09/2022	17/09/2022
Result Comparis	18/09/2022	19/09/2022
Neural network Purning		
Model training v	20/9/22	4/10/22
ShrinkBench Im	05/10/2022	08/10/2022
NN pruning	09/10/2022	19/10/2022
Pytorch impl.	20/10/2022	22/10/2022
NN pruning	23/10/2022	02/11/2022
Collect the resu	03/11/2022	07/11/2022
Result Comparis	08/11/2022	11/11/2022
Reseacrh Report		
Result Key point	19/09/2022	08/11/2022
Report Structur	18/09/2022	23/09/2022
Finalisation	05/11/2022	13/11/2022
Analysis		
Plot graph for p	22/09/2022	07/11/2022
Plot graph of D.	09/08/2022	19/09/2022
Analysation of r	08/11/2022	11/11/2022
Thesis		
Draft Version	04/11/2022	19/11/2022
Final version	20/11/2022	05/12/2022

Figure 12 Project Gantt Chart

A.5 Risks:

Risk ID	Risk	Mitigation Response	Likelihood 0/5	Impact 0/5	Score 0/25
1	Crash of the machine	Create the backups and store in external storage	2	5	10
2	Incompatible Dataset for the network architecture	Find another dataset and check its compatibility	2	1	2
3	Inability of completion of the project within the timeframe	Organise the work using the Gantt Chart/Start project early	3	4	12
4	Undesired Analysis results	if results unmet the expectation, use different pruning methods and dimension reduction techniques	4	2	8
5	Loss of the working laptop	Use backups to ensure that project is secured	3	5	15
6	Failure in resolving coding errors	Ask for a support from your supervisors or other students	4	3	12
7	Crash during the training of the model	Fix relevant errors and return the training again	5	2	10
8	Communicational Risk with the supervisor	Setting up the meeting with the supervisor to discuss the project	3	3	9
9	Forgetting the objectives	Refer to Proposal/Ask for help from Supervisor	3	2	6

Figure 13 Risks Table

A.6 References:

- Blalock, D., Gonzalez Ortiz, J., Frankle, J. and Guttag, J. (6AD). *WHAT IS THE STATE OF NEURAL NETWORK PRUNING?*.
- Carreira-Perpiñán, M., Eecs and Idelbayev, Y. (n.d.). *'Learning-Compression' Algorithms for Neural Net Pruning*. [online] Available at: https://faculty.ucmerced.edu/mcarreira-perpinan/papers/cvpr18.pdf [Accessed 22 May 2022].
- Frankle, J. and Carbin, M. (4AD). *THE LOTTERY TICKET HYPOTHESIS: FINDING SPARSE, TRAINABLE NEURAL NETWORKS*. [online] arxiv: Cornell University. Available at: https://arxiv.org/pdf/1803.03635.pdf [Accessed May 21AD].
- Hyvärinen, A. and Oja, E. (2000). Independent component analysis: algorithms and applications. *Neural Networks*, 13(4-5), pp.411–430. doi:10.1016/s0893-6080(00)00026-5.
- Hao, Y., Nakajima, H. and Ying, X. (2019). An Overview of Overfitting and its Solutions Separation and Characterization of Overpotentials in Electrochemical Hydrogen Pump with a Reference Electrode An Overview of Overfitting and its Solutions. *An Overview of Overfitting and its Solutions*, 1168(2), p.22022. doi:10.1088/1742-6596/1168/2/022022.
- Liu, H. and Wang, J. (2011). Integrating Independent Component Analysis and Principal Component Analysis with Neural Network to Predict Chinese Stock Market. *Mathematical Problems in Engineering*, 2011, pp.1–15. doi:10.1155/2011/382659.
- Meneghetti, L., Demo, N. and Rozza, G. (2021). *A Dimensionality Reduction Approach for Convolutional Neural Networks*. [online] Available at: https://arxiv.org/abs/2110.09163 [Accessed 22 May 2022].
- Malach, E., Yehudai, G., Shalev-Shwartz, S. and Shamir, O. (2020). *Proving the Lottery Ticket Hypothesis: Pruning is All You Need*. [online] Available at: https://arxiv.org/pdf/2002.00585v1.pdf [Accessed 22 May 2022].
- Ramachandran, R., Ravichandran, G. and Raveendran, A. (2020). *Evaluation of Dimensionality Reduction Techniques for Big data*. [online] IEEE Xplore. doi:10.1109/ICCMC48092.2020.ICCMC-00043.
- Scikit-learn.org. (2009). *sklearn.decomposition.PCA — scikit-learn 0.20.3 documentation*. [online] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html.
- scikit-learn. (n.d.). *Blind source separation using FastICA*. [online] Available at: https://scikit-learn.org/stable/auto_examples/decomposition/plot_ica_blind_source_separation.html?highlight=ica [Accessed 22 May 2022].
- Kruskal, J.B., 1964. *Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis*. Psychometrika, 29(1), pp.1-27.
- Kruskal, J.B., 1964. *Nonmetric multidimensional scaling: a numerical method*. Psychometrika, 29(2), pp.115-129.
- Park, H. and Seo, J., 2011. *Application of multidimensional scaling to quantify shape in Alzheimer's disease and its correlation with Mini Mental State Examination: a feasibility study*. Journal of neuroscience methods, 194(2), pp.380-385.
- Park, H., 2012. *ISOMAP induced manifold embedding and its application to Alzheimer's disease and mild cognitive impairment*. Neuroscience Letters, 513(2), pp.141-145.
- Lee, D.D. and Seung, H.S., 1999. *Learning the parts of objects by non-negative matrix factorization. Nature*, 401(6755), pp.788-791.
- Hoyer, P.O., 2004. *Non-negative matrix factorization with sparseness constraints*. Journal of machine learning research, 5(9).
- Roweis, S.T. and Saul, L.K., 2000. Nonlinear dimensionality reduction by locally linear embedding. science, 290(5500), pp.2323-2326.
- Yong, A.G. and Pearce, S., 2013. A beginner's guide to factor analysis: Focusing on exploratory factor analysis. Tutorials in quantitative methods for psychology, 9(2), pp.79-94.
- Tenenbaum, J.B., Silva, V.D. and Langford, J.C., 2000. *A global geometric framework for nonlinear dimensionality reduction. science*, 290(5500), pp.2319-2323.
- Carreira-Perpinán, M.A., 1997. A review of dimension reduction techniques. Department of Computer Science. University of Sheffield. Tech. Rep. CS-96-09, 9, pp.1-69.

Research Ethics Review Form: BSc, MSc and MA Projects

A.1 If you answer YES to any of the questions in this block, you must apply to an appropriate external ethics committee for approval and log this approval as an External Application through Research Ethics Online - https://ethics.city.ac.uk/		*Delete as appropriate*
1.1	Does your research require approval from the National Research Ethics Service (NRES)? *e.g. because you are recruiting current NHS patients or staff?* *If you are unsure try - https://www.hra.nhs.uk/approvals-amendments/what-approvals-do-i-need/*	**NO**
1.2	Will you recruit participants who fall under the auspices of the Mental Capacity Act? *Such research needs to be approved by an external ethics committee such as NRES or the Social Care Research Ethics Committee - http://www.scie.org.uk/research/ethics-committee/*	**NO**
1.3	Will you recruit any participants who are currently under the auspices of the Criminal Justice System, for example, but not limited to, people on remand, prisoners and those on probation? *Such research needs to be authorised by the ethics approval system of the National Offender Management Service.*	**NO**
A.2 If you answer YES to any of the questions in this block, then unless you are applying to an external ethics committee, you must apply for approval from the Senate Research Ethics Committee (SREC) through Research Ethics Online - https://ethics.city.ac.uk/		*Delete as appropriate*
2.1	Does your research involve participants who are unable to give informed consent? *For example, but not limited to, people who may have a degree of learning disability or mental health problem, that means they are unable to make an informed decision on their own behalf.*	**NO**
2.2	Is there a risk that your research might lead to disclosures from participants concerning their involvement in illegal activities?	**NO**
2.3	Is there a risk that obscene and or illegal material may need to be accessed for your research study (including online content and other material)?	**NO**
2.4	Does your project involve participants disclosing information about special category or sensitive subjects? *For example, but not limited to: racial or ethnic origin; political opinions; religious beliefs; trade union membership; physical or mental health; sexual life; criminal offences and proceedings*	**NO**
2.5	Does your research involve you travelling to another country outside of the UK, where the Foreign & Commonwealth Office has issued a travel warning that affects the area in which you will study? *Please check the latest guidance from the FCO - http://www.fco.gov.uk/en/*	**NO**
2.6	Does your research involve invasive or intrusive procedures?	**NO**

	These may include, but are not limited to, electrical stimulation, heat, cold or bruising.	
2.7	Does your research involve animals?	**NO**
2.8	Does your research involve the administration of drugs, placebos or other substances to study participants?	**NO**
A.3 If you answer YES to any of the questions in this block, then unless you are applying to an external ethics committee or the SREC, you must apply for approval from the Computer Science Research Ethics Committee (CSREC) through Research Ethics Online - https://ethics.city.ac.uk/ **Depending on the level of risk associated with your application, it may be referred to the Senate Research Ethics Committee.**		*Delete as appropriate*
3.1	Does your research involve participants who are under the age of 18?	**NO**
3.2	Does your research involve adults who are vulnerable because of their social, psychological or medical circumstances (vulnerable adults)? *This includes adults with cognitive and / or learning disabilities, adults with physical disabilities and older people.*	**NO**
3.3	Are participants recruited because they are staff or students of City, University of London? *For example, students studying on a particular course or module.* *If yes, then approval is also required from the Head of Department or Programme Director.*	**NO**
3.4	Does your research involve intentional deception of participants?	**NO**
3.5	Does your research involve participants taking part without their informed consent?	**NO**
3.5	Is the risk posed to participants greater than that in normal working life?	**NO**
3.7	Is the risk posed to you, the researcher(s), greater than that in normal working life?	**NO**
A.4 If you answer YES to the following question and your answers to all other questions in sections A1, A2 and A3 are NO, then your project is deemed to be of MINIMAL RISK. **If this is the case, then you can apply for approval through your supervisor under PROPORTIONATE REVIEW. You do so by completing PART B of this form.** **If you have answered NO to all questions on this form, then your project does not require ethical approval. You should submit and retain this form as evidence of this.**		*Delete as appropriate*
4	Does your project involve human participants or their identifiable personal data? *For example, as interviewees, respondents to a survey or participants in testing.*	**NO**

Appendix B

Github link: https://github.com/mcnugets/Effectiveness-of-different-dimensionality-reduction-techniques-on-pruned-deep-neural-network