International Journal of Artificial Intelligence and Machine Learning
Volume 10 • Issue 2 • July-December 2020

Assessing Hyper Parameter Optimization and Speedup for Convolutional Neural Networks

Sajid Nazir, Glasgow Caledonian University, UK
Shushma Patel, London South Bank University, UK
Dilip Patel, London South Bank University, UK

ABSTRACT

The increased processing power of graphical processing units (GPUs) and the availability of large image datasets has fostered a renewed interest in extracting semantic information from images. Promising results for complex image categorization problems have been achieved using deep learning, with neural networks comprised of many layers. Convolutional neural networks (CNN) are one such architecture which provides more opportunities for image classification. Advances in CNN enable the development of training models using large labelled image datasets, but the hyper parameters need to be specified, which is challenging and complex due to the large number of parameters. A substantial amount of computational power and processing time is required to determine the optimal hyper parameters to define a model yielding good results. This article provides a survey of the hyper parameter search and optimization methods for CNN architectures.

KEYWORDS

Artificial Intelligence, Cognitive Image Processing, Convolution, Deep Learning, Hidden Layers, Machine Learning, Object Recognition, Semantics

INTRODUCTION

The growth in Internet of Things (IoT) (Bubley, 2016), and emergence of social, web and mobile applications have provided access to large image datasets as a result of a move away from text based to visual communications. This coupled with the advances in storage and processing technologies has made it possible to progress from image processing to interpreting images for extracting contextual information. Artificial Intelligence (AI) aims to endow machines with similar capabilities of learning, perception and reasoning as that of a human. The question, 'Can machines think?' was posed in 1950 (Turing, 1950) through an 'imitation game.' Challenges of AI remain, despite substantial progress in learning algorithms (Bengio, 2009). Machine learning is a sub-field of AI that makes it possible for computers to learn without explicitly being programmed (Neetesh, 2017). Machine learning for vision problems comprises techniques that can provide intelligent solutions to complex problems of interpreting and describing a scene, given sufficient data. Much progress has been made in this area, but improvements are needed. One technique that has risen to predominance recently is Artificial Neural Network (ANN) that was inspired by biological neuron interconnections and activations of human brain (Deep Learning tutorial, 2015).

DOI: 10.4018/IJAIML.2020070101

Deep learning, a branch of machine learning (Bhandare & Kaur, 2018) that derives its name from neural networks that comprise of many layers. Multiple layers are used to model high-level features from complex data, with each successive layer using the outputs from the preceding layer as an input (Benuwa, 2016). An overview of deep learning techniques with a focus on convolutional neural networks (CNNs) and deep belief networks (DBNs) is provided together with a discussion on sparsity and dimensionality reduction (Arel, 2010). Benuwa (2016) review deep learning techniques along with algorithm principles and architectures for deep learning. A review of recent advances in deep learning is provided in (Minar, 2018) as well as taxonomy of deep learning techniques and applications. A review of deep supervised learning, unsupervised learning, and reinforcement learning is provided in (Schmidhuber, 2015) covering developments since 1940.

The aim of training neural networks is to find weightings that achieve better classification accuracy (Nguyen, 2018). These networks require a lot of time, processing power, and data in order to be trained. After training, a neural network can be used to make better predictions on test data (Neetesh, 2017). Deep learning algorithms are complex to develop, train and evaluate. A neural net (Krizhevsky, Sutskever, & Hinton, 2012) with 60 million parameters and 650,000 neurons took a long time to train on ImageNet (Deng et al., 2009), in order to classify 1.2 million images. The increased research interest in neural networks is due to the promising results obtained for ImageNet competitions (Krizhevsky et al., 2012). CNN, the leading type of neural networks have been used for classifying large image datasets (Krizhevsky et al., 2012; Szegedy et al., 2014). The application of deep learning for different medical image modalities is provided in (Shen, Wu, & Suk 2017).

CNNs have also been applied for combining image information over a long duration video of up to two minutes (120 frames) to solve classification problem (Ng et al., 2015). A dynamically trained CNN was proposed for object classification in video streams (Yaseen, Anjum, Rana, & Antonopoulos 2019). The image features from hidden layers of deep neural networks were extracted for image recognition in (Hayakawa, Oonuma, & Kobayashi 2017).

Although the fields of artificial intelligence and deep learning are very promising, the techniques are deeply rooted in probabilistic foundations. An important aspect of the neural networks performance is the hyper parameters or the model parameters, and their impact on results. This aspect is critical to designing and developing efficient models. CNN architectures are dependent on hyper parameters and an incorrect choice can have a huge effect on performance (Albelwi & Mahmood, 2016).

Before a neural network can be trained, hyper parameter values must be determined. The number of hyper parameters increases with complex deep neural networks (Ozaki, Yano, & Onishi, 2017). These need to be carefully fine-tuned for a particular application to yield good results (Soon, 2018). Deep neural networks are very sensitive to hyper parameter values (Domhan, Springenberg, & Hutter, 2015) and may fail to train for slightly non-optimal values (Ozaki et al., 2017). Therefore, the success of a neural network, to a large extent, is governed by the correct values of its hyper parameters (Soon, 2018). Hyper parameter optimization is the process of optimizing a loss function over a configuration space (Bergstra, Bardenet, Bengio, & Kégl, 2011). To optimise hyper parameters for a suitable CNN architecture is an iterative and lengthy process (Hinz, Navarro-Guerrero, Magg, & Wermter 2018).

This paper provides a survey of the techniques for determining the optimal CNN hyper parameters which would be helpful to a researcher and implementer in choosing the appropriate strategy depending on the availability of time, expertise, and processing power.

CNN ARCHITECTURES FOR DEEP LEARNING

Computer vision technologies for object recognition have undergone rapid advances, and better techniques with improved results have been proposed (Krizhevsky et al., 2012; Deng et al., 2009). The emergence of neural networks for computer vision applications can be attributed to Deng et al., (2009) and Lecun et al., (1989). Although biological vision systems and processes are not fully understood, the current method of ANNs yields promising results. The ImageNet Large Scale Recognition Challenge

(ILSVRC) has been running since 2009 and provides a common platform for comparing computer vision algorithms for object detection and classification (Russakovsky et al., 2015).

ANNs are modelled on the human nervous system (Kienzler, 2017) and need computational power and large volumes of data to be trained before they can be successfully used. ANNs can learn from any mathematical relation between the input and output (Kienzler, 2017). A large number of labelled true and false examples are required in supervised learning to train the ANN before good results can be obtained. Deep learning, a sub-field of machine learning has provided winning results in pattern recognition (Benuwa, 2016).

Deep learning is driver for many applications in AI (Tibbetts, 2018). Deep learning has replaced the use of handcrafted features through use of feature learning algorithms (Benuwa, 2016). A deep neural network is comprised of many layers. The layers between the input and output layers are termed hidden layers (Figure 1). The depth increases interconnections and complexity of nodes. The initial layers work on low-level features, lines, circles etc., whereas the deeper layers work on higher or complex features, until the whole image is recognized (Kienzler, 2017). Such systems can perform at the same or better levels than humans (Kienzler, 2017). Deep learning models are able to recognize more complex features accurately and in less time compared to a human (Tibbetts, 2018).

Of particular interest are CNNs that can process spatial data and take a fixed size input and generate fixed size outputs. Thus CNNs, due to their inherent nature, are more applicable for object recognition problems exploiting the spatial dimensions of height and width. Generally Deep Neural Networks (DNNs) are considered difficult to train but CNNs fare better and better generalization is possible by CNN architecture to vision tasks (Bengio, 2009) as CNNs are designed to work on two-dimensional data (Arel, 2010). CNNs generally perform better at extracting important features from images making them well-suited for image understanding (Arel, 2010).

IMPORTANT CNN HYPER PARAMETERS

It is important to understand that there is an interplay and interdependence of hyper parameters. Saari (2018) found that the two chosen hyper parameters, depth of CNN and a regularization technique (Dataset Augmentation) affected the results such that it was concluded that instead of applying both, only one could be used for optimal results. In addition, the selection of the hyper parameters for tuning also affects the results as all hyper parameters do not have the same significance for the training or test accuracy of the model. A brief summary of the important hyper parameters is provided below:

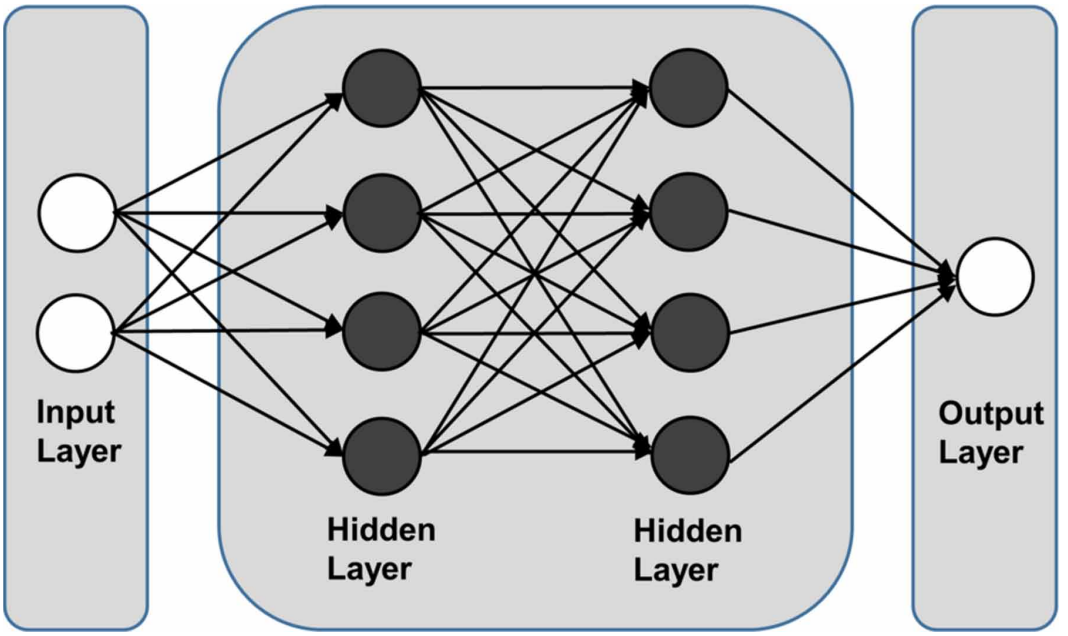
1. Architecture Type and Number of Hidden Layers

The number of hidden layers defines the depth of the network. The depth of the proposed architectures has been consistently increasing and in general was shown to yield better results. However, alternate architectures with less depth have also been proposed (Hasanpour, Rouhani, Fayyaz, & Sabokrou, 2016) that are useful for embedded systems with less processing power and demonstrated that a reasonably deep network can perform competitively to more deeper and therefore complex networks.

2. Optimizers

There are many optimizers reported in the literature, but significant ones include RMSProp, Stochastic Gradient Descent (SGD) and Adam. These provide good results with a batch size of 32 to 512 (Keskar, Mudigere, Nocedal, Smelyanskiy, & Tang, 2017). Keskar et al. (2017) have studied SGD with larger batch sizes for deep learning applications. An important and common parameter in all these optimizers is the learning rate. The value of the learning rate is chosen to be between 0 and 1.

Figure 1. Artificial Neural Network (ANN) with the number of hidden layers defining the depth of network. Each layer transforms its inputs through trainable parameters, that is, weights. A shallow network has fewer hidden layers compared to a deep network.



3. Activation Function

Koutsoukas, Monaghan, Li and Huan (2017) compared the activation functions and found that ReLU provided the best results overall. More complex variants of ReLU have been proposed recently, that is, LeakyReLU signifying improvement in results with rectifier non-linearities as compared to sigmoidal ones (Maas, Hannam, & Ng, 2013) and PReLU (He, Zhang, Ren, & Sun, 2015). The implementations are available in Keras (Keras) as "Advanced Activations".

4. Dropout Regularization

A trained model should perform well on unseen data during testing (Deep Learning tutorial, 2015). However, a complex model can learn the training data perfectly and then fail to generalize to unseen examples, a phenomenon termed as overfitting. Overfitting can be avoided by regularization techniques, such as Dropout (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). Data Augmentation is another mechanism to augment the existing data by generating new images through simple operations such as flip and rotation on existing data.

5. Convolution Layer

A convolution layer comprises of many parameters, the important ones are the number of kernels applied to each layer, the height and width of each convolutional kernel, zero padding and stride. Without zero padding the size of the convolved image will reduce. Stride defines the amount of movement of the kernel after calculating a value. If it is more than one then the convolved image will again reduce in size.

6. Dimensions of Pooling Matrices in Pooling Layers

Generally, a 2 × 2 size for the pooling is used for downsampling the image into half. A larger pooling matrix size would reduce the image size even further than half.

7. Number of Epochs and Batch Size

An epoch consists of one pass of the entire data through the network. The data is passed through the network by dividing it into batches or sets. Thus, many iterations would be required to process all the data through the network. In general, a higher value for epoch will provide better results.

HYPER PARAMETER SEARCH METHODS

Hyper parameter optimization or tuning is a process applied to tune the model by tweaking the parameters for the best results. The model may be susceptible to degradation by small changes to its parameters. For example, the removal of one layer from the five layer convolutional model degraded the performance (Krizhevsky et al., 2012). Thus, a lot of mutual interdependencies might exist amongst the identified optimum hyper parameters. The number of training parameters to be considered for a deep network is large and the required time and computational resources make it infeasible to sweep through the entire parameter space (Benuwa, 2016).

Many different hyper parameters method have been reported in the literature. The choice of a particular method depends on the chosen architecture, number of selected hyper parameters to be tuned, availability of time and processing power. After considering the various strategies, we categorized the different strategies into 3 types, namely conventional, framework based, and optimization speedup. This categorization helps to show an evolution from earlier conventional to recent methods that are focussed on automated optimizations, reduction of hyper parameters, and speedup. We survey and provide state-of-the art techniques for hyper parameter optimizations from the research literature.

Conventional and Exhaustive Search Methods

Conventional methods either try out all the selected hyper parameters exhaustively or restrict the search to a chosen subset based on its significance or selection. These methods were good for simple networks but have limited performance for complex networks having a large number of hyper parameters. Various search methods are outlined below and summarised in Table 1.

Manual Search

The manual search method can have promising results in terms of time and selected hyper parameters because unlike the grid search, a human can rule out sub-optimum hyper parameters easily.

A manual search for DNNs is described in (Koutsoukas et al., 2017) by considering hyper parameters such as activation functions, learning rate, number of neurons per layer, number of hidden layers, and dropout regularization. The performance of DNNs were compared with some other machine learning techniques, such as, Naïve Bayes, k-nearest neighbour, random forests, and support vector machines. DNN were found to outperform the other selected algorithms.

In earlier work (Nazir, et al., 2018), a simple architecture was used to easily investigate the effect of parameter change on improving image classification results. One parameter at a time was investigated to obtain better results. However, the focus was on investigating a large selection of important hyper parameters for learning rates activation layer, momentum, and batch size while at the same time making use of regularization (dropout) (Krizhevsky et al., 2012; Srivastava et al., 2015) and batch normalization (Ioffe, & Szegedy 2015). Keras was used with Tensorflow (Abadi et al., 2015) (as a backend). CIFAR-10 (Krizhevsky et al., 2012; He et al., 2015) was also used.

A manual process for investigating all but one fixed hyper parameter to obtain a set of hyper parameters is reported by Nguyen (2018), aimed at achieving hyper parameters with high classification accuracies and to shorten the training time. A modified CNN model based on VGG was used. The results were provided for CIFAR-10, CIFAR-100, GTSRB, and DSDL-DB. The hyper parameters investigated were learning rate, batch size, and initial weights. The weights were initialized based on similar network weight factors.

Characteristics of manual search:

- The person has insight and understanding of the relative importance of the hyper parameters for the given model.
- The expert could detect failures and terminate training at an early stage (Ozaki et al., 2017).
- Manual optimization is not hindered by any technical overhead (Bergstra & Bengio, 2012).

Grid Search

Grid search is a common method for hyper parameter optimization (Bergstra & Bengio, 2012). It is an exhaustive search for all the selected values of the hyper parameters. This is available in many software packages and can be easily specified by listing the values for the selected hyper parameters to be investigated. Under software control, it will step through all the possible combinations to determine the combination that yields the best results.

Grid search based methods worked well in earlier machine learning models with limited parameters. It is argued in (Bergstra & Bengio, 2012) that grid search may be a poor choice as it also considers hyper parameters which might not be important for a given dataset.

Characteristics of Grid Search

- Most popular method for hyper parameter optimization (Albelwi & Mahmood, 2016).
- Grid search implementation is simple and can be parallelized (Bergstra & Bengio, 2012).
- It can find better values than a purely manual search given that sufficient computing resources are available (Bergstra & Bengio, 2012).
- It can perform reliably for one and two-dimensional hyper parameter spaces (Bergstra & Bengio, 2012).
- It tries all possible combinations thus having an exponential growth with an increase in the number of hyper parameters (Hinz et al., 2018).

Random Search

Bergstra and Bengio (2012) showed that random search can be used to search for hyper parameter values yielding better results compared to Grid search in higher hyper parameter spaces. Random search also requires less computational power. It was revealed that many datasets have only a few hyper parameters that are really important and that for different datasets, different hyper parameter configurations may be required. Random search can act as a baseline against which other optimization methods can be evaluated. Random search and Grid search are similar in that both can be simply implemented using the same tools. Random search did not perform as well as the combination of manual search followed by grid search, compared to an expert (Bergstra and Bengio, 2012).

Characteristics of random search:

- Works by drawing a random value from each parameter of interest based on given distribution (Hinz et al., 2018).
- These can also be used to investigate the effect of one hyper parameter, similar to a manual search (Bergstra & Bengio, 2012).

Table 1. Conventional methods for hyper parameters search

Techniques considered	Hyper parameters	Dataset	Accuracy/ Benefits	Reference
Manual	Activation function, learning rate, number of neurons per layer, dropout regularization, number of hidden layers	CHEMBL	ReLU activation function performed better than Sigm or Tanh	(Koutsoukas et al., 2017)
Manual	Optimizer, learning rate, number of epochs and batch size, activation function	CIFAR-10	Empirical search of hyper parameters	(Nazir et al., 2018)
Manual, Merged datasets	Learning rate, batch size,	CIFAR-10, CIFAR-100, GTSRB, DSDL-DB	Data pre-processing increased classification accuracies, and training CNN accelerated by momentum optimizer	(Nguyen, 2018)
Grid, Manual, Random	8 global hyper parameters and 8 hyper parameters for each layer resulting in 32 hyper parameters for a 3 layer model	MNIST	Random search is efficient compared to grid search, random search appropriate as baseline for performance comparisons	(Bergstra & Bengio, 2012)
Bayesian Optimizations	Standard Bayesian optimization with standard hyper parameters, Optimization function and learning rate	CIFAR-10 and Caltech-101	10% better results than the baseline in transfer learning compared to manual methods	Borgli (2018)

- Can be parallelized (Ozaki et al., 2017).
- Can handle integer and categorical hyper parameters (Ozaki et al., 2017).

Bayesian Optimizations

Bayesian Optimization uses probabilistic Gaussian processes for approximating and minimizing the error function for hyper parameter values. However, this requires estimates of many statistics of the error function that can make these methods inefficient for evaluating deep neural network hyper parameters (Ilievski, Akhtar, Feng, & Shoemaker, 2017). Therefore, many other methods have been proposed, such as Gaussian Process (GP) and Tree-structured Parzen Estimator (TPE) method.

A tutorial on Bayesian optimizations is provided by Brochu et al. (2010). Bayesian optimizations use Bayesian techniques to get a posterior function. Two techniques, active user modelling and hierarchical reinforcement learning are also described therein. The limitations of Bayesian Optimizations such as feature selection and time-varying models are also described.

Bergstra et al. (2011) have considered the Bayesian optimization with a Gaussian Process based method, Sequential Model-Based Optimization (SMBO), and Tree-structured Parzen Estimator method (TPE).

Bayesian optimizations with standard parameters were used by Borgli (2018) to optimize the hyper parameters optimization of CNN for transfer learning for two publicly available image datasets for gastroenterology. It was shown that automatic hyper parameter optimization provided 10% better results than the baseline in transfer learning compared to manual methods.

Framework Based Methods

The limitation of conventional manual or exhaustive search methods is that they require lot of computation and time, and may require expert insights for optimal hyper parameter selection. On the other hand, automated hyper parameter optimizations can be used by non-experts. For a deep neural network, it could still require significant computing resources and time thus hindering its adoption (Domhan et al., 2015). This section describes various framework-based methods with a summary provided in Table 2.

Optimization Framework

An optimization framework is proposed that can automatically determine the architecture of a CNN for a given application (Albelwi & Mahmood, 2016). They used visualization for deconvolution networks and accuracy to produce better results. The computational cost was overcome using the Nelder-Mead algorithm. They concluded that CNN optimized hyper parameters favoured small strides and pooling windows, and deep networks.

Nelder-Mead is proposed for hyper parameter optimization (Ozaki et al., 2017) for character recognition and age/gender (CNN) classification. The authors contend that this is easier for non-experts who may find it difficult to implement Bayesian optimization and covariance matrix adaptation evolution techniques that also require large computing resources. The results were better than other selected techniques and the Nelder-Mead method was found to perform best for hyper parameter optimization as it quickly converged to local optimum.

A metaheuristic optimization method, parameter-setting-free harmony search (PSF-HS) is proposed (Lee, 2018) to adjust the hyper parameters. The hyper parameter tuning was proposed for CNN in the feature extraction step. The hyper parameter to be adjusted was set as a harmony; harmony memory was updated based on CNN loss by generating the harmony memory after the harmony. Simulations were performed using CNN architectures for LeNet-5, MNIST, CifarNet and Cifar-10 datasets. The simulation results show improved performance compared to other techniques through hyper parameter tuning.

Deterministic RBF Surrogates

A deterministic algorithm based on Radial Basis Function (RBF) is proposed that requires lesser function evaluations compared to Bayesian Optimization (Ilievski et al., 2017). The evaluations on MNIST and CIFAR datasets were shown to be better, that is about 6 times faster for obtaining best set of 19 hyper parameters, compared to Bayesian Optimizations such as GP, SMAC and TPE.

Evolutionary Based Algorithms

Genetic algorithms were used to automatically learn the CNN architecture. The network structures are represented using a fixed-length binary string and each generation used standard methods of selection, mutation, and crossover (Xie & Yuille, 2017). The genetic algorithms were used for MNIST and CIFAR-10, and it was shown that the automatically generated structures performed better than the manual ones. The structures were then used for a larger dataset, ILSVRC2012.

A genetic algorithm was proposed in (Bhandare & Kaur, 2018) for hyper parameter optimization on MNIST dataset. A number of hyper parameters were selected for optimizations. It was reported

that the accuracy was over 90% but the best run had an accuracy of 99.2%. The simulation results for the Genetic Algorithm based method were better than manual search methods (Loussaief & Abdelkrim, 2018).

An Enhanced Elite CNN Model Propagation method is proposed in (Loussaief & Abdelkrim, 2018) that can automatically learn an optimized structure of CNN using genetic algorithm. The classification accuracy was found better than public CNNs using transfer learning.

An evolutionary algorithm-based framework is proposed for automatic optimizations of CNN hyper parameters (Bochinski, Senst, & Sikora, 2017). This framework was then extended for joint optimization of CNNs to provide significant improvement over state-of-the-art algorithms on MNIST dataset. Other techniques using committees of multiple CNNs are outlined by Bochinski, Senst, & Sikora (2017).

Particle swarm optimization was used to automatically select the architecture and CNN hyper parameters with an aim to reduce the user variability in training (Soon, 2018). With optimised hyper parameters, CNN architecture was trained for better convergence and classification. The proposed methods were applied to vehicle log images. The proposed method produced better results compared to other state-of-the art methods obtaining 99.1% accuracy.

Evolutionary algorithms were proposed for automatic discovery of image classifier networks (Real, et al., 2017). Simple evolutionary algorithms were used to discover models for CIFAR-10 and CIFAR-100 datasets achieving an accuracy of 94.6% although the computation costs were significant.

Reinforcement Learning

A reinforcement learning based meta-modelling algorithm that can generate better CNN architectures automatically is proposed in (Baker, Gupta, Naik, & Raskar 2017). CNN layers are chosen using Q-learning with greedy exploration strategy to train the learning agent. The agent selects higher performing CNN models through random exploration.

Q-learning with a greedy exploration strategy was used in (Zhong, Yan, Wei, Shao, & Liu, 2018) with a learning agent to choose component layers. They used a block-wise generation that provided better results compared to hand-crafted networks and decreased the search space. An early stopping strategy was also used for fast block search.

An algorithm is proposed in (Mortazi, 2018) for an automatic search of optimal hyper parameters for neural architecture design for medical image segmentation. The proposed method was based on policy gradient reinforcement learning and was computationally efficient compared to other medical image segmentation methods. The proposed hyper parameter search algorithm was applied on a proposed architecture with dense connected encoder-decoder CNN. The results with better accuracy were obtained for cine cardiac MR images from Automated Cardiac Diagnosis Challenge (ACDC) MICCAI 2017 without any trial-and error or close supervision of hyper parameter changes as in other methods.

In general, the reinforcement learning techniques limit optimization to architectural hyper parameters and manually choose other hyper parameters like learning rate and regularization parameters (Hinz et al., 2018).

Hyper Parameters Optimization Speedup

The methods in this category are aimed at reducing the time taken for example, by using techniques to exploit the CNN architecture. This section describes various speedup approaches with a summary provided in Table 3.

Early Termination

A probabilistic model was used for early termination of bad runs and it was shown that the method provided a twofold increase compared to human experts for selected optimization methods. This follows the strategy used by human experts for early termination of a bad run to save time. Learning

Table 2. Framework-based methods

Techniques considered	Hyper parameters	Dataset	Accuracy/ Benefits	Reference
Optimization framework (Nelder-Mead algorithm)	Depth, number of layers, kernel size, number of pooling layers	CIFAR-10 and Caltech-101	Improvement in overall results, framework contributing to network depth, stride and pooling size	(Albelwi & Mahmood, 2016)
Co-ordinate search and Nelder-Mead for age/ gender classification	LeNet hyper parameters for MNIST, iterations 20000, batch size 50, learning rate decay	MNIST, age/gender	Nelder-Mead outperformed random search, Bayesian Optimization, CMA-ES, and coordinate search	(Ozaki et al., 2017)
Parameter-setting-free harmony search (PSF-HS)	Optimal hyper parameters through harmonies, kernel size, stride, zero padding, number of channels, kernel size and stride	LeNet-5, MNIST, CifarNet, Cifar-10	Improved performance(reduced number of weights and bias to be trained) compared to earlier CNN architectures	(Lee, 2018)
Deterministic RBF Surrogates	6, 8, 15, 19 hyper parameters	CIFAR-10, MNIST	6 times faster than Bayesian Optimizations such as GP, SMAC and TPE.	(Ilievski et al., 2017)
Genetic Algorithm	LeNet structure used, learning rate, epochs	MNIST, CIFAR10, ILSVRC2012	Performance of the generated structures was better than the manually designed structures.	(Xie & Yuille, 2017)
Genetic Algorithm	Twelve selected hyper parameters including epochs, hidden layers and neurons, activation function, optimizers	MNIST	No human intervention required, accuracy above 90% and 99.2% (best of 10 runs) for MNIST	(Bhandare & Kaur, 2018)
Genetic Algorithm (Enhanced Elite CNN Model propagation)	Chosen from AlexNet, normalization, pooling layers, ReLU optimizer	Caltech-256	Pre trained CNN accuracy of 98.94%	(Loussaief & Abdelkrim, 2018)
Evolutionary Algorithm based framework	Hyper parameters describing CNN structure, such as layer and kernel size were considered	MNIST	Generic framework, improvements over state-of-the-art methods	(Bochinski et al., 2017)
Particle swarm optimization	Training epoch, 3 convolution layers each followed by ReLU, 2 pooling layers	XMU and XMUPlus	obtaining 99.1% accuracy compared to other state-of-the art methods	(Soon, 2018)
Evolutionary Algorithms	SGD with momentum of 0.9, batch size of 50, and weight decay of 0.0001.	CIFAR-10 and CIFAR-100	Accuracies of 94.6%, no human participation required	Real et al. (2017)
Reinforcement learning	13 selected hyper parameters including learning rate, epochs, number of layers	Street View House numbers (SVHN), CIFAR-10, MNIST	MetaQNN obtained an error of 6.92% compared to 21.2% by Bergstra et al. (2011) on CIFAR-10, MetaQNN performs better than other meta modelling networks	(Baker, 2017)
Reinforcement learning	7-layer network with learning rates, number of epochs, batch size, optimizers	CIFAR-10	BlockQNN had 3.54% top-1 error rate on CIFAR-10 which was better than all auto-generate Networks.	(Zhong, 2018)
policy gradient reinforcement learning	Number of filters, filter height, filter width for each layer, type of pooling, total 76 parameters	ACDC- MICCAI 2017	Low computation cost, state of the art accuracy	(Mortazi)

curve extrapolation was used to devise an early termination criterion. The method that was used to investigate small and larger neural networks was independent of hyper parameter optimizer (Domhan et al., 2015).

Optimization of Selected Hyper Parameter

Hyper parameter optimizations have generally low effective dimensionality. Although there are a large number of hyper parameters, only few have a significant impact on performance (Hinz et al., 2018). The selected hyper parameter optimization algorithms were applied to CNN hyper parameters for images with increasing resolutions. It was found that the same hyper parameters were relevant independent of the image resolution. This was used to speed up the hyper parameter optimization. The result was that it took less time to find significant hyper parameters and the method can also be applied to data other than images, if its dimensions can be reduced.

Massively Parallel Hyper Parameter Tuning

A large scale parallel hyper parameter tuning is proposed in (Li, Jamieson, Rostamizadeh, Gonina, & Talwalkar, 2018) to evaluate many hyper parameter configurations in parallel to reduce the training time significantly. They also used early stopping in conjunction with parallelism to further reduce time. The proposed algorithm can find optimal hyper parameters much faster than random search. Separate GPUs were used to train each model but obviously the speedup did not increase linearly due to communications cost.

POP Scheduling

A scheduling algorithm is proposed in (Rasley, He, & Yan, 2017) called POP that quickly identifies the promising, opportunistic, and poor hyper parameter configurations. An infrastructure was also proposed that could work across different scheduling algorithms. A speedup of about 6.7 times was reported compared to random/grid search and 2.1 compared to state-of-the-art methods.

Parameter Reduction

Hyper parameter reduction for CNN for field devices with low resources is proposed in (Atanbori, 2018) for segmentation of plant phenotyping. The results showing trade-off between number of hyper parameters and obtained accuracy were obtained using four baseline neural networks by increasing the network depth and reducing the number of hyper parameters, termed “Lite” CNNs.

Complexity Reduction

Reduction in computation by factorization of the convolutional layer was proposed in (Wang, 2017). The operations in convolution layer were treated separately as spatial convolution in each channel while maintaining the accuracy and reducing the computations. The model’s performance was evaluated on ImageNet LSVRC 2012 dataset. The proposed model achieved many times less computations with similar performance for VGG-16, ResNet-34, ResNet-50 and ResNet-101.

Flattened convolutional neural networks are presented in (Jin, 2015) that were trained to obtain similar performance to conventional CNNs. For similar performance in accuracy as 3D filters, speed-up of about two times compared to baseline was obtained during the feedforward pass. After the model has been trained there is no requirement for manual tuning or post processing.

A small CNN architecture, termed SqueezeNet, requiring 50 times less parameters but achieving AlexNet level of accuracy was proposed in (Iandola, 2017).

Table 3. Optimization speedup methods

Techniques considered	Hyper parameters	Dataset	Accuracy/ Benefits	Reference
Learning curves Extrapolation-optimizer agnostic	17 selected hyper parameter including epochs, learning rate, batch size and optimizers	CIFAR-10, CIFAR-100, MNIST	Twofold increase in state-of-the-art optimization methods	(Domhan et al., 2015)
Early lower dimensional representation to identify promising-TPE, SMAC, GA	9 hyper parameters including learning rate, number of layers, number and size of filters, batch size and regularization parameters	CIFAR, MNIST	Use of lower dimensional data to speed up the optimization process	(Hinz et al., 2018)
Massively Parallel Hyper parameter Tuning	LeNet hyper parameters, 8 hyper parameters for 3-layer CNN including number of layers, batch size and number of filters	CIFAR-10, MNIST	Order of magnitude faster compared to random search	(Li et al., 2018)
HyperDrive, POP (Promising, opportunistic, and poor) Scheduling	learning rate and decay, momentum, number of layers,	CIFAR-10	6.7 times speedup compared to random/grid search, 2.1 times speedup compared to other state-of-the-art methods	(Rasley, He, & Yan, 2017)
Reduced parameter CNN for limited resource devices- Lite CNN	14.7 million parameters reduced to 5.3 and 0.3 million	Oxford Flower dataset	Lite CNN models with comparable accuracy to baseline model	(Atanbori, 2018)
Complexity Reduction (factorized CNN)	Batch normalization after convolutional and before ReLU layer, dropout as 0.2, batch as 256, learning rate of 0.1 and changed by dividing by 10	ImageNet LSVRC 2012, VGG-16, ResNet-34, ResNet-50, ResNet-101	Same accuracy but with significantly reduced computations, similar performance to VGG-16, ResNet-34, ResNet-50, ResNet-101 with 42x, 7.32X, 4.38X, and 5.85 less computation respectively	(Wang, 2017)
Flattened CNN with reduced CNN parameters	3 convolutional layers, 5x5 filters, double stage multilayer perceptron, ReLU, max pooling of 2, strides of 3, 5 dropout layers, initial learning rate of 0.1 (reduced by 1/10 after 8 epochs) and momentum of 0.9	CIFAR-10, CIFAR-100, MNIST	Two times speedup during feedforward pass compared to baseline model with reduction of learning parameters, with similar accuracies	(Jin, 2015)
SqueezeNet-smaller CNN with same accuracy	3x3 filters replaced by 1x1 filters, ReLU, Dropout, learning rate of 0.04, batch size	ImageNet	AlexNet accuracy with 50X fewer parameters, and 510X smaller size	(Iandola, 2017)

CONCLUSION

There have been many advances in the application of convolutional neural networks to image classification with promising results, similar to a human. The tuning of model architectures could be driven by intuition and experimentation resulting in the optimal values of the hyper parameters. This method of experimental tuning to determine the values does not scale well with the number of hyper parameters, which increase exponentially with the number of network layers.

The search for an optimal configuration of hyper parameters for CNNs requires computational power, time, and associated cost. With large numbers of hyper parameters it is critical to quickly converge to an optimal set from the search space.

The initial models were manually optimized by the researchers starting an era of image classification enabling results, comparable to humans for the first time. With the rise in the number of hyper parameters, research efforts were focussed on using traditional methods such as grid search

wherein different sets or configurations are tried out sequentially. The need for early termination was initially met by a manual process and some automated processes could also terminate non-performing hyper parameter selection.

The choice of a particular hyper parameter optimization strategy depends on the chosen architecture, number of selected hyper parameters to be tuned, availability of time and processing power. Current research is focussed on automated optimizations, for which we provide state-of-the art techniques from the research literature. We have provided an overview of the hyper parameter optimizations techniques signifying the contribution of each method. A natural progression in the research methods is from simple methods to state-of-the-art techniques. Better hyper parameter optimizations would be aided by shallow neural networks (with results comparable to deep networks), parallel execution of hyper parameter configurations, and optimization frameworks that might take care of all the intricate optimization details for the researchers in the future. The state-of-the-art methods have automated the process for hyper parameter optimizations, as well as employed parallel processing to exploit the CNN architecture in order to save time. The future is promising for automated hyper parameter optimization, where the operation would not only be fully automated but would save time and computations in the process.

ACKNOWLEDGMENT

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., & Dean, J., …Zheng, X. (2016). TensorFlow: a system for large-scale machine learning. In *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation.* ACM.

Albelwi, S., & Mahmood, A. (2016). Automated Optimal Architecture of Deep Convolutional Neural Networks for Image Recognition. In *Proceedings of 15th IEEE International Conference on Machine Learning and Applications.* IEEE. doi:10.1109/ICMLA.2016.0018

Arel, I., Rose, D. C., & Karnowski, T. P. (2010, November). Deep Machine Learning - A New Frontier in Artificial Intelligence Research. *IEEE Computational Intelligence Magazine*, *5*(4), 13–18. doi:10.1109/MCI.2010.938364

Atanbori, J., Chen, F., French, A. P., & Pridmore, T. (2018). Towards low-cost image-based plant phenotyping using reduced-parameter CNN. In *CVPPP 2018: Workshop on Computer Vision Problems in Plant Phenotyping*. Academic Press.

Baker, B., Gupta, O., Naik, N., & Raskar, R. (2017). Designing neural network architectures using reinforcement learning. In *Proceedings of Int. Conf. Learning Representations*. Academic Press.

Bengio, Y. (2009). Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, *2*(1), 1–127. doi:10.1561/2200000006

Benuwa, B.-B., Zhan, Y. Z., Ghansah, B., Wornyo, D. K., & Banaseka Kataka, F. (2016). A Review of Deep Machine Learning. *International Journal of Engineering Research in Africa*, *24*, 124–136. doi:10.4028/www.scientific.net/JERA.24.124

Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, *13*, 281–305.

Bergstra, J. S., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. In Advances in neural information processing systems (pp. 2546-2554). Academic Press.

Bhandare, A., & Kaur, D. (2018). Designing Convolutional Neural Network Architecture Using Genetic Algorithms. In *Proceedings of Int'l Conf. Artificial Intelligence*. Academic Press.

Bochinski, E., Senst, T., & Sikora, T. (2017). *Hyper-parameter Optimization for Convolutional Neural Network Committees based on Evolutionary Algorithms, 2017*. IEEE.

Borgli, R. J. (2018). *Hyperparameter optimization using Bayesian optimization on transfer learning for medical image classification* [Master thesis]. University of Oslo.

Brochu, E., Cora, V. M., & De Freitas, N. (2010). A tutorial on Bayesian optimization of expensive cost functions, with application to active user modelling and hierarchical reinforcement learning. Retrieved from https://arxiv.org/abs/1012.2599

Bubley, D. (2016). IoT & Realtime Communications. *IEEE newsletter*, (March). Retrieved from https://iot.ieee.org/newsletter/march-2016/iot-realtime-communications.html

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. IEEE.

Domhan, T., Springenberg, J. T., & Hutter, F. (2015). Speeding up Automatic Hyperparameter Optimization of Deep Neural Networks by Extrapolation of Learning Curves. In *Proceedings of the 24th International Conference on Artificial Intelligence.* ACM.

Hasanpour, S. H., Rouhani, M. H., Fayyaz, M., & Sabokrou, M. (2016). Let's keep it simple, using simple architectures to outperform deeper and more complex architectures.

Hayakawa, Y., Oonuma, T., Kobayashi, H., Takahashi, A., Chiba, S., & Fujiki, N. M. (2017). Feature Extraction of Video Using Artificial Neural Network. *International Journal of Cognitive Informatics and Natural Intelligence*, *11*(2), 25–40. doi:10.4018/IJCINI.2017040102

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE.

Hinz, T., Navarro-Guerrero, N., Magg, S., & Wermter, S. (2018). Speeding up the Hyperparameter Optimization of Deep Convolutional Neural Networks. *International Journal of Computational Intelligence and Applications*, *17*(2), 1850008. doi:10.1142/S1469026818500086

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2017) Squeezenet: Alexnet-level accuracy with 50x fewer parameters and < 0.5 mb model size.

Ilievski, I., Akhtar, T., Feng, J., & Shoemaker, C. A. (2017). Efficient Hyperparameter Optimization of Deep Learning Algorithms Using Deterministic RBF Surrogates. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*. Academic Press.

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning*. ACM.

Jin, J., Dundar, A., & Culurciello, E. (2015). Flattened convolutional neural networks for feedforward acceleration.

Keras: The Python Deep Learning library. (n.d.). Retrieved from https://keras.io/

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., & Tang, P. T. P. (2017). On Large-batch Training for Deep Learning: Generalization Gap and Sharp Minima. In *Proceedings of ICLR 2017*. Academic Press.

Kienzler, R. (2017). Developing cognitive IoT solutions for anomaly detection by using deep learning, Part 1: Introducing deep learning and long-short term memory networks: Detecting anomalies in IoT time-series data by using deep learning. IBM.

Koutsoukas, A., Monaghan, K. J., Li, X., & Huan, J. (2017). Deep learning: Investigating deep neural networks hyper parameters and comparison of performance to shallow methods for modeling bioactivity data. *Journal of Cheminformatics*, *9*(1), 42. doi:10.1186/s13321-017-0226-y PMID:29086090

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems. ACM.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, *1*(4), 541–551. doi:10.1162/neco.1989.1.4.541

Lee, W. Y., Park, S. M., & Sim, K. B. (2018). Optimal hyperparameter tuning of convolutional neural networks based on the parameter-setting-free harmony search algorithm. *Optik (Stuttgart)*, *172*, 359–367.

Li, L., Jamieson, K., Rostamizadeh, A., Gonina, E., & Talwalkar, A. (2018). Massively Parallel Hyperparameter Tuning.

Loussaief, S., & Abdelkrim, A. (2018). Convolutional Neural Network Hyper-Parameters Optimization based on Genetic Algorithms. *International Journal of Advanced Computer Science and Applications*, *9*(10). doi:10.14569/IJACSA.2018.091031

Maas, A. L., Hannam, A. Y., & Ng, A. Y. (2013). Rectifier Nonlinearities Improve Neural Network Acoustic Models. In *Proceedings of the 30th International Conference on Machine Learning*. Academic Press.

Minar, M. R. & Naher, J. (2018). Recent Advances in Deep Learning: An Overview.

Mortazi, A. & Ulas Bagci, U. (2018). Automatically Designing CNN Architectures for Medical Image Segmentation.

Nazir, S., Patel, S., & Patel, D. (2018). Hyper Parameters Selection for Image Classification in Convolutional Neural Networks. In *Proceedings of 17th IEEE International Conference on Cognitive Informatics and Cognitive Computing*. IEEE. doi:10.1109/ICCI-CC.2018.8482081

Neetesh, M. (2017, December 1). The Connect between Deep Learning and AI. *Open Source for You*.

Ng, J., Yue-Hei, Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., & Toderici, G. (2015). Beyond Short Snippets: Deep Networks for Video Classification. In *Proceedings of Conference on Computer Vision and Pattern Recognition*. IEEE.

Nguyen, H. N., & Lee, C. (2018). Effects of Hyper-parameters and Dataset on CNN Training. J. Inst. Korean Electr. Electron. Eng., 22(1), 14-20.

Ozaki, Y., Yano, M., & Onishi, M. (2017). Effective hyperparameter optimization using Nelder-Mead method in deep learning. *IPSJ Transactions on Computer Vision and Applications*, *9*(1), 20. doi:10.1186/s41074-017-0030-7

Rasley, J., He, Y., & Yan, F. (2017). HyperDrive: Exploring Hyper parameters with POP Scheduling. In *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference*. ACM. doi:10.1145/3135974.3135994

Real, E., Moore, S., Selle, A., Saxena, S., Suematsu, Y. L., & Tan, J. … Kurakin A. (2017). Large-scale evolution of image classifiers. In *Proceedings of Int. Conf. Machine Learning*. ACM.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., & Fei-Fei, L. et al. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, *115*(3), 211–252. doi:10.1007/s11263-015-0816-y

Saari, M. (2018). *The effect of two hyper parameters in the learning performance of the convolutional neural networks* [Bachelor thesis]. Tampere University of Technology, Finland.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85–117. doi:10.1016/j.neunet.2014.09.003 PMID:25462637

Shen, D., Wu, G., & Suk, H.-I. (2017). Deep Learning in Medical Image Analysis. *Annual Review of Biomedical Engineering*, *19*(1), 221–248. doi:10.1146/annurev-bioeng-071516-044442 PMID:28301734

Soon, F. C., Khaw, H. Y., Chuah, J. H., & Kanesan, J. (2018). Hyper-parameters optimisation of deep CNN architecture for vehicle logo recognition. *IET Intelligent Transport Systems*, *12*(8), 939–946. doi:10.1049/iet-its.2018.5127

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, *15*, 1929–1958.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., & Anguelov, D., …Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of Conference on Computer Vision and Pattern Recognition*. IEEE.

Tibbetts, J. H. (2018). The Frontiers of Artificial Intelligence. *Bioscience*, *68*(1), 5–10. doi:10.1093/biosci/bix136

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, *59*(236), 433–460. doi:10.1093/mind/LIX.236.433

Tutorial, D. L. (2015). *Release 0.1, LISA Lab*. University of Montreal. Retrieved from http://deeplearning.net/tutorial/deeplearning.pdf

Wang, M., Liu, B., & Foroosh, H. (2017). Factorized Convolutional Neural Networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW)* (pp. 545-553). IEEE Press. doi:10.1109/ICCVW.2017.71

Xie, L., & Yuille, A. (2017). Genetic CNN. In *Proc. of the IEEE Conf. Computer Vision and Pattern Recognition*. IEEE.

Yaseen, M. U., Anjum, A., Rana, O., & Antonopoulos, N. (2019, January). Deep Learning Hyper-Parameter Optimization for Video Analytics in Clouds. *IEEE Transactions on Systems, Man, and Cybernetics. Systems*, *49*(1), 253–264. doi:10.1109/TSMC.2018.2840341

Zhong, Z., Yan, J., Wei, W., Shao, J., & Liu, C.-L. (2018). Practical block-wise neural network architecture generation, In *Proceedings of Conf. Computer Vision and Pattern Recognition*. IEEE.

Sajid Nazir is a lecturer at Glasgow Caledonian University, Glasgow, UK. He received a PhD degree in Electrical Engineering from Strathclyde University, Glasgow, UK, in 2012. He has worked on remote video monitoring projects as a Research Fellow at the University of Aberdeen from 2012 to 2015. He worked on virtualization, CCTV, and SCADA projects as a KTP Associate with the School of Engineering, London South Bank University, London, and Firstco Ltd., UK. Dr. Sajid is a Fellow of HEA, and Member of IET. He has authored one book and over 40 research publications. His research interests include video communications, machine learning, industrial systems, and networking.

Shushma Patel (BSc (H), PhD., FBCS, CITP, PFHEA) is a Professor of Information Systems and the Director of Education and Student Experience in the School of Engineering. She studied life sciences as an undergraduate, before completing a PhD from the Faculty of Medicine, University of London. She has more than 25 years of teaching and research experience in Cognitive Informatics, Information Systems and Qualitative Research. Before joining London South Bank University, Professor Patel worked on many clinical research projects, in collaboration with, and funded by leading pharmaceutical and medical research councils. She has worked on many EU and commercially funded projects, exploiting innovative technologies for business solutions. Her current interest in cyber security is informed by her considerable experience working with industry and in particular her interest in user behaviours.

Dilip Patel is a Professor Emeritus at the London South Bank University. His academic career in computer science spans over 30 years. He has published extensively in the areas of object technology, cognitive informatics, and database technologies. He was an editorial advisory board member for the book "Model-Driven Domain Analysis and Software Development: Architectures and Functions." Dilip is also on the editorial board for the International Journal of Software Science and Computational Intelligence.