EAI Endorsed Transactions
on Cloud Systems
Research Article EAI.EU

Hyperparameter optimisation for Capsule Networks

Gagana B[1,*], S Natarajan [1]

[1]PES University, Bangalore

Convolutional Neural Networks and its contemporary variants have proven to be ruling benchmarks for most image processing tasks but resort to pooling techniques and routing mechanisms that affect classification accuracy and lose spatial relationship information between involved data points. Hence, Hinton et al, proposed a layered architecture called Capsule Networks (Capsnets) which outperform traditional systems by replacing pooling techniques with dynamic routing abilities. Capsnets are, thus, en-route to proving themselves as prospective future benchmarks in visual imagery tasks by surpassing existing state-of-the-art results on the MNIST dataset. The two novel aspects inspected in this paper are: the enhancement of this performance on CIFAR-10 through regularization and hyperparameter optimization which, henceforth, augment applicability to stochastic numeric healthcare data helping uncover newer challenges of predictive neural networks.

Received on 14 February 2019; accepted on 11 March 2019; published on 15 April 2019

Keywords: hyperparameter optimisation, Stochastic numeric healthcare data, Capsule Networks, ReLU, performance benchmarks

doi:10.4108/eai.13-7-2018.158416

1. Introduction

Machine Intelligence, involving visual imagery tasks like segmentation, detection and reconstruction, has primarily been propelled by Convolutional Neural Networks (Convnets) but the major limitations of such systems include: loss of information due to max and average pooling techniques along with the inability to encode orientation and positional variations into predictions[1]. Although, the brain's mechanisms of information processing are significantly different from how traditional systems are wired, latest advancements like Capsule networks[2] apply the Hebbian Learning principles which are closer to emulating human capabilities by applying active vectorization techniques and by filtering features on layered dimensionality planes to reduce disparity over disagreement.

Hinton et.al. first proposed shape representation models in parallel systems[3] to synthesize assignment of coordinate intrinsic object frames of reference which deals with individualised hypotheses(on interpretation of local fragments of visual input and unit interactions encoding knowledge on local interaction and spatial disposition detection constraints) representing units further enabling interaction organization between parallelized network entities so that a single pattern activity representation is obtained as the frames undergo simultaneous convergence but continue to remain remain dimensionally viewer centric affected by attributes such as bilaterally symmetric plane, gross elongation and gravitational or contextual verticals. Since, this was implemented using coordinate hardware units, the architecture implicitly couples surrounding elements as coordinate frames affecting the relative environment where it's based. Activities, in terms of corresponding channels, can profoundly influence each other stimulating a rough object segmentation while the computational overheads can be reduced by optimizing mappings of distributed encoding by clustering associated regions.

The idea further evolved into 'Transforming auto-encoders'[4] modeling scalar features into vector activity representations and instantiation parameters

*Gagana B. gaganab20496@gmail.com, S Natarajan. natarajan@pes.edu

that effortlessly blended into the domains of the given visual entity. The resulting probability is multiplied element-wise to the capsule output as implicit routing learns to detect visual features over time. The obtained probability is expected to be stable even when the entity varies over the space of appearance transformations and the value determines weight in the overall autoencoder prediction and thus, is a reassurance of Capsnets abilities to be potential benchmarks in the future .

The nested set of neural layers resort to dynamic routing mechanisms of selected features by implementing denoising algorithms at the lower levels of capsule predictions before hierarchically routing activities of the local pool to higher level capsules which unearth convoluted data patterns resulting in highly concise informative outputs.

For all capsule i in layer l and for all capsule j in layer (l+1):
$b_{ij} \leftarrow 0$
For r iterations do:
For all capsule i in layer l: $c_{ij} \leftarrow Softmax(b_i)$
For all capsule j in layer (l+1): $s_j \leftarrow \Sigma c_{ij} u_{ji}$
For all capsule j in layer (l+1): $v_j \leftarrow Squash(s_j)$
For all capsule i in layer l and all capsule j in layer (l+1): $b_{ij} \leftarrow b_{ij} + u_{ji} v_j$
return $\mathbf{v_j}$

Hinton et al proposed an expectation maximization routing algorithm logistic[5], which deals with encoding of the relationship between entity and pose. greatly improving the efficiency of capsule routing by recursive updates to the weighted assignment coefficient matrix and clustering probabilities(that are relatively closer). This ConvCaps structure extends dynamic routing into convolutional filters as the requisite feature maps are tiled into kernel wise batched data. The underlying transformation matrix is trained discriminatively by back-propagating through unrolled iterations between adjacent pairs of capsule layers, enabling effective representation of part-whole relationships and demonstrates Capsule's increased robustness towards white box adversarial attacks significantly lesser vulnerable than baseline ConvNets while also enhancing the generalisability of learned pose based matrices and corresponding learning capacities.

Capsules have also been applied to health-care applications primarily for lung tumor and brain fMRI data. (a)Lung disease detection dataset[6] where the CapsNets architecture was trained on a 400 image dataset to address two levels of the problem statement: a. Is the patient ill? and b. if yes, what type of lung related disease is it? The model was first scanned for fundamental attributes such as gender, age, noise filtering. The secondary processing involved testing the convolutional neural units ability to accelerate the convergence and optimize them using spatial transformation techniques. The experiments proved that Capsnets have an inherent ability to thrive in scenarios involving limited data.

(b)Brain fMRI images proposes "Capsnet architecture based visual reconstruction"[7] to reconstruct image stimuli by decoding position, orientation, and categories the object could potentially belong to, from activities in visual cortex. The approach is said to have 10% more accuracy than previosu state-of-the-art approaches. This implementation encompasses experimentation across four hypothesis:
a. Implement and ensure successful application to design an improved architecture maximizing performance accuracy;
b. Investigate the over-fitting on a real set of fMRI data;
c. Explore if the solution can be extrapolated to fit the entire brain or the limitations are confined to the segmented tumor;
d. Development of visualization paradigm to better convey learned features.
The network was trained on nonlinear mappings between the image stimuli and high-level capsule features in an end-to-end manner. After estimating the serviceability of voxels by encoding performance to accomplish optimal selection, the system is re-trained on these nonlinear mappings.

Other systems such as CapsuleGAN[8] have been explored where the discriminator of a generative adversarial network(GAN) is replaced by capsules to model different objective functions evaluated qualitatively and quantitatively on the Generative Adversarial Metric (GAM) whose objective can be mathematically summarized as:

$$\min_G \max_D V(D,G) = E_{x\, P_{data(x)}}[logD(x)] + E_{z\, P_{z(z)}}[log(1 - D(G(z)))] \quad (1)$$

These models are typically used to represent highly complex distributions which are known to be unstable, suffer from vanishing gradients, mode collapse and inadequate mode coverage issues which CapsuleGAN deals with, by introducing better objective functions, sophisticated training strategies, empirical tricks and structured hyperparameters.

The aforementioned papers have laid the basis for the work carried out and the rest of this paper is structured as follows: Section 2 articulates the architectural framework of Capsule networks, Section

3 deals with the experimentation framework with proposed changes followed by results obtained and discussions in Section 4, conclusive remarks in Section 5 and bibliography in Section 6.

2. Architectural Framework

The broad architectural framework of Capsule networks is composed of an encoder and a decoder, former of which comprises of a two dimensional convolutional ReLU layer for detecting the basic features, a PrimaryCaps layer for producing combinations of the above feature outputs and a DigitCaps layer for the generation of the loss function and transformational weight matrix; the Decoder of Capsnets constitutes three fully connected layers, FC1 (With ReLU activation unit), FC2 (With ReLU activation unit), and FC3 (With sigmoid activation unit); Both the components effectively work together towards reconstruction of the input image while dealing with the accuracy and loss performance parameters. The loss parameter, in turn, entails margin loss as computed for each capsule and reconstruction loss which is scaled down by 0.0005 to prevent domination.

The detailed technical functionality of each of the Capsnet layers is as follows:

(i) The ReLU convolutional layer: The layer has 256 kernels each with a bias term, stride of 1, size of 9x9x1 followed by the ReLU activation. The layer handles 20992 parameters and outputs 20x20x256 tensor.

(ii) The supporting PrimaryCaps layers: The 32 capsule layer applies 9x9x256 convolutional kernels to the 20x20x256 input volume while handling 5308672 parameters and outputs 6x6x8x32 tensor.

(iii) The DigitCaps layers: This 10 node digit capsule layer ingests the 6x6x8x32 tensor and as per inner workings of each capsule, a weight matrix is computed and 8 dimensional input space is mapped to the 16 dimensional capsule output space. The layer outputs a 16x10 matrix associated with 1497600 parameters.

The loss function is a weighted sum calculated for correct DigitCaps and incorrect DigitCaps, primarily defined as 1 for a matching training label and 0 otherwise. A zero loss event is initiated either when a correct prediction occurs with probability greater than 0.9 in case of matching training labels or when an incorrect prediction occurs with probability less than 0.1 in case of mismatched training labels.

$$L_c = T_c max(0, m^+ - ||v_c||)^2 + \gamma(1 - T_c) max(0, ||v_c|| - m^-)^2 \quad (2)$$

The transformation matrix W_{ij} is maps the 8-D capsule to a 16-D capsule output space for each class j in relation to u_i, the capsule output of the previous layer. The probability magnitude could be mathematically summarised by v_j.

$$_{j,i} = W_{ij} u_i \quad (3)$$

The final output v_j for class j is computed using the novel squashing function as:

$$v_j = \frac{||S_j||^2}{1 + ||S_j||^2} \frac{S_j}{||S_j||} \quad (4)$$

where

$$S_j = \sum_i C_{ijj,i} \quad (5)$$

with c_{ij} coupling coefficients measuring the likelihood of primary capsule i probabilistically triggering capsule j with s_j representing the weighted sum further shrinked by the squashing function.

The decoder is a regularizer network which recreates the original 28x28 image while forcing capsules to learn the features of the data. The first and the penultimate layer of the capsnet decoder have the ReLU activation function while the last layer retains the sigmoid activation unit.

The first fully connected layer calculates the number of parameters based on bias which outputs a 512 vector, processing 82432 trainable parameters.

The second fully connected layer calculates the number of parameters based on bias which outputs a 1024 vector, processing 525312 trainable parameters.

The final fully connected layer calculates the number of parameters based on bias which outputs a 784 vector, processing 803600 trainable parameters.This is also the final 28 X 28 output. Thus, the total number of parameters in the capsule network are: 8238608. The Capsnet architectural framework is as visualised[9] in Figure 1.

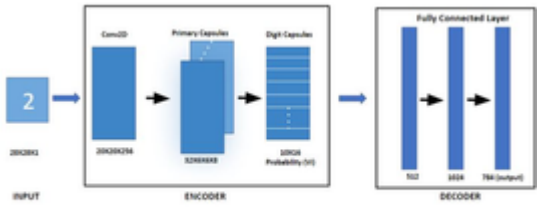

Figure 1. Block Diagram of Capsule Network Architecture

3. Experimental Design

This section entails the proposed enhancements and optimizations to Capsule networks which primarily fall under the following categories:
(i)Activation function optimisations involving variants of ReLU[10] and Swish[11] variants[12].
(ii)Data augmentation with Neural Style Transfer[13]
(iii) Optimisation with additional Softmax[14] layers
(iv) code shuffling[15] within the dataset to model stochasticity
(v) hyperparameter tuning using grid search and random search[16]
(vi)parallel implementation of hyperparameter regularisation[17] with early stopping[18]

3.1. Object Recognition Tasks

A tensorflow backed Keras implementation in a Jupyter notebook environment run on a tesla k40c GPU configuration was the key framework while executing Capsnets on the CIFAR-10 dataset[19] for object recognition and reconstruction tasks. However, each of the following optimisations and regularizations were executed within the same framework environment as opposed to original capsule network architecture with default settings as the baseline benchmark. Each of the implementations have been executed for 20 epochs as results are known to fairly stabilize thereafter.

Activation Functions. Activation functions or Transfer functions are defined as non-linear transformations or complex functional mappings between response variables and incoming data. The generic form of the equation is:

$$Y = \sum(Weight * Input) + Bias \tag{6}$$

where 'Y' can bound from negative to positive infinity. The activation function applied over an input signal which decides whether a neuron fires or activates, depending upon the weight over input which when paired with back-propagation iterates over the bias aggregate and update gradients resulting in a loss metric.

The mathematical definitions of the various activation function units are as follows: The non-linear ReLU activation is defined as:

$$A(x) = max(0, x) \tag{7}$$

ReLU is a simple, efficient, and most widely used monotonic function which ensures convergence six times faster than the tanh function. It rectifies the vanishing gradient problem but the major limitation of ReLU is that the neurons are unlikely to recover if they fall into the negative slope thus, outputting zero independent of the input scenarios.
The leaky ReLU function (LReLU)[20] defined as

$$f(x) = 1(x < 0)(\alpha x) + 1(x >= 0)(x) \tag{8}$$

is an improvised version of ReLU implementation which deals with the above mentioned limitation by implementing small negative slopes.
Latest activation functions like the Swish is defined in terms of the sigmoid as:

$$f(x) = x.sigmoid(\beta x) \tag{9}$$

proposed by Ramachandran et al. The e-Swish activation with a learnable beta component is mathematically defined by:

$$E - Swish = \beta x * sigmoid(x) \tag{10}$$

Data Augmentation Using Neural Style Transfer. Data Augmentation refers to the process of augmenting the dataset with relevant synthetically modified data to enhance performance. We used neural image style transfer mechanisms on representative examples of the dataset which are further used to train the network resulting in improved performance. The method however introduces two types of loss namely style loss and content loss. The weighted sum of the same is as represented below.

$$L_{\text{total}}(S, C, G) = \alpha L_{\text{content}}(C, G) + \beta L_{\text{style}}(S, G) \tag{11}$$

where the content loss of layer I is defined as:

$$L_{\text{content}}(S, C, I) = 0.5 \sum_{ij}(a[I]C_{\text{ij}} - a[I]G_{\text{ij}})^2 \tag{12}$$

and the loss of the style associated gram matrix is defined as:

$$L_{\text{GM}}(S, C, I) = 1/4{N_{\text{I}}}^2{M_{\text{I}}}^2 \sum_{ij}(GM[I]S_{\text{ij}} - GM[I]G_{\text{ij}})^2 \tag{13}$$

Optimisation with additional softmax layers. During multiclass classification, softmax layers are used with the same number of nodes prior to the output layer to obtain the probability distributions for all involved classes and the same is mathematically defined as:

$$P(y = j|x) = e_{{w_j}^{\text{T}}x + b_j} / \sum_{k \in K} e^{{w_k}^{\text{T}}x + b_k} \tag{14}$$

These layers are augmented into the decoder of the Capsnet architecture before the output layer to enhance object recognition performance.

Code shuffling. We introduced stochasticity into the data to enable the model to minimize the training loss to adapt to the dynamic characteristics of healthcare data. If we assume the process of training the network with a minimum loss value function to be defined by L_w over the training set with w representing the weight matrix. So, the minimisation of loss value functions occurs with say, c elements of the training set, L is then a surface on the c+1 dimensional space. To geometrically generalize, loss function can be evaluated over any training set and associated weight matrix, but it's plausible that the resulting value remains unchanged over training iterations which would make the model susceptible to the local minima problem. Code shuffling with mini batch diversification ensures changes over training iterations. Assuming the local minimum of the loss value function is L_{w_i} at training iteration i, the loss surface geometrically changes over the next iteration on the stochastic training set assumed to be defined as: $L_{w_{i+1}}$. L_{w_i} is different from $L_{w_{i+1}}$ which likely won't be the local minimum and we hence can compute the gradient update and continue training.

Hyperparameter tuning using grid search and random search. Grid search and random search explore the same parameter space by simultaneously searching for parameters that potentially influence learning. Grid search refers to the process of building models for combinations of hyperparameters, evaluating the model for each of the combinations and finally, choosing the set of parameters with the highest classification accuracy. Random search refers to the process of randomly choosing hyperparameters which in general converge in lesser time than the grid search tuning techniques.

parallel implementation of hyperparameter regularisation with early stopping. Hyperparameter regularisation of learning rate is implemented on Capsnets using the Sherpa library which works well on problems with computationally expensive iterative function evaluations. Results surpassed the ReLU benchmark much before the 20th epoch and early stopping on the 9th epoch was found to avoid overfitting issues.

3.2. Healthcare

As the world appraises the billion dollar healthcare market, which is poised to grow, technological advancements are substantial for outreach to meet the large scale demand for quality. In order to ensure affordability, exploring this transformational space in terms of the latest machine learning systems like predictive neural net frameworks could be an influential progress in this direction. This subsection conceptualizes the modelling of Capsnets for the analysis of hyperglycemia data spanning clinical databases involving 74 million unique cases which correspond to 17 million unique patients with 70,000 inpatient diabetes encounters. The original linear regression statistic model suggests that relationship between the HbA1c levels and readmission probability depends primarily on the diagnosis[21]. The results of the study are significant and critical due to their influence on morbidity and mortality rates which in-turn depends on the treatment modality. HbA1c levels of greater than or equal to 7% were associated with increased morbidity where as both high and low levels of the component was associated with increased mortality[22].

The stochastic numeric healthcare diabetes dataset was mapped to a time series forecast along with appropriate channel labels and fed into the Capsule architecture to predict readmission rate of patients based on their HbA1c levels and other associated values. The conv2D layer detects basic features forming a feature map in the form of a 20 x 20 x 256 tensor. The PrimaryCaps layers produces combinations of the feature map and outputs a 6 X 6 X 256 tensor. The DigitCaps layers generates the transformation weight matrix W_{ij} and the entailing loss function. The three fully connected layers of the decoder calculate the number of parameters based on bias. These layers resort to ReLU while fundamental analysis proves that variants of Swish and ReLU better enhance accuracy. The same is as elaborated in the subsection below. With performance attributes being key indicators in healthcare sectors, the application and implementation here is chosen to draw attention to the fundamental aspects within the organic framework and to demonstrate the intuitional insights of the architecture.

4. Results and Discussion

The survey with healthcare data involving hbA1c levels has proved to be phenomenal with 19.5% increased relative correlation as compared to previous benchmarks of linear regression when experimented with the ReLU activation function on Capsnets.

The results of activation function optimisations of Capsnets on object segmentation tasks of the CIFAR-10 dataset in terms of accuracy are as follows.

While the leaky ReLU variant performs best, e-Swish activation functions consistently outperform the ReLU benchmark.
The results of other optimisations and regularisations of Capsnets on object segmentation tasks of the CIFAR-10 dataset in terms of accuracy are as shown (Table 2)

Table 1. Activation function optimisation of Capsnets on CIFAR-10 dataset in terms of Accuracy

	Maximum Value	Minimum Value	Average Value
ReLU	56.9	26.95	48.8
LReLU	59.25	28.35	50.76
e-Swish (=0.2)	58.2	27.2	49.2125
e-Swish (=0.625)	58.55	26.7	49.535
e-Swish (=0.75)	58.15	25.45	49.4075
e-Swish (=0.875)	58.2	24.45	48.8125

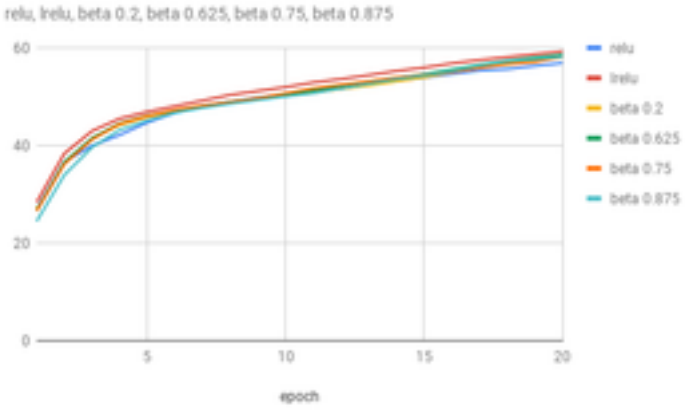

Figure 2. Visualisation of activation function optimisation of Capsnets on CIFAR-10 dataset in terms of Accuracy

Table 2. Optimisations and Regularisations of Capsnets on CIFAR-10 dataset

Methodology	Average Value
ReLU	48.796%
Data Augmentation	50.9725%
Softmax Optimisation	51.4375%
Stochastic Data modeling	50.2325%
Sherpa Optimisation and early stopping	50.65%
Grid Search	48.1025%
Random Search	49.4345%

The results of activation function optimisations of Capsnets on object segmentation tasks of the CIFAR-10 dataset in terms of the loss Parameter are as shown (Table 3).

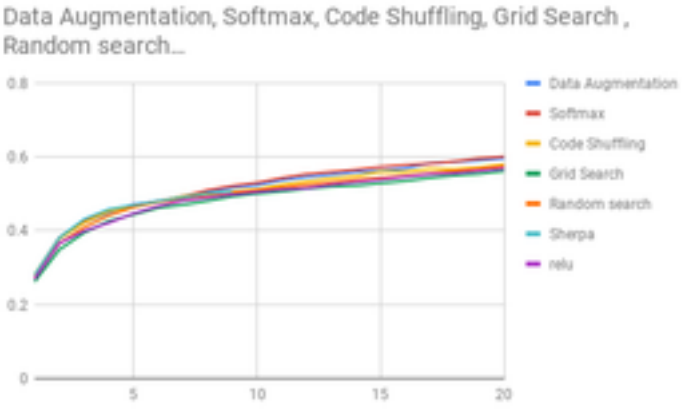

Figure 3. Visualisation of optimisation and regularisation of Capsnets on CIFAR-10 dataset

Table 3. Comparison between ReLU, LReLU and e-Swish with reference to loss parameter

	Maximum Loss	Minimum Loss	Average Loss
ReLU	1.16376	0.33637	0.4224748
LReLU	0.82282	0.31885	0.3840183
e-Swish (=0.2)	0.77288	0.33611	0.387684
e-Swish (=0.625)	0.79539	0.33427	0.390891
e-Swish (=0.75)	0.79478	0.33656	0.3904483
e-Swish (=0.875)	0.8042	0.33553	0.3930248

LReLU outperforms both e-Swish and ReLU in terms of the loss metric. A graphical representation of the same is as follows where the y axis represents the loss and the horizontal x axis represents the epoch value.

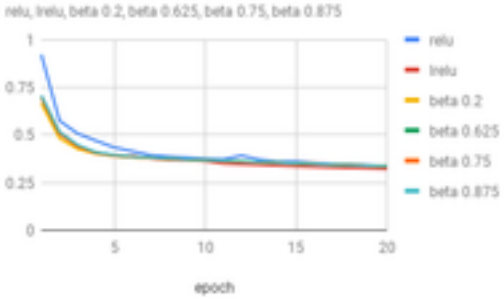

Figure 4. Graphical Representation of LReLU and e-swish with reference to loss

The results of other optimisations and regularisations of Capsnets on object segmentation tasks of the CIFAR-10 dataset in terms of loss are as shown (Table 4)

Table 4. Optimisations and Regularisations of Capsnets on CIFAR-10 dataset with respect to loss parameter

Methodology	Average Value
ReLU	0.42247425
Data Augmentation	0.3820635
Softmax Optimisation	0.380025
Stochastic Data modeling	0.426145
Sherpa Optimisation and early stopping	0.383187
Grid Search	0.4244465
Random Search	0.42041665

A visual representation of the above data is as follows:

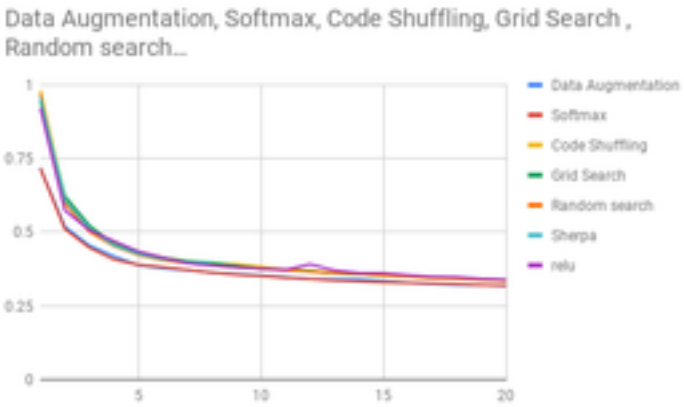

Figure 5. Graphical Representation of Capsnet optimisation and regularisation with reference to loss

These results are currently being extrapolated to cancer research models which are expected to surpass the ConvNet accuracy benchmarks leading to implications and inferences on demographic constitutions. These activation functions could be experimented on different datasets in terms of application, volume and complexity. Newer functions, optimizers and regularisation techniques or tweaks to the existing implementations would be worthwhile research avenues to explore.

5. Conclusion

From the experiments, it is evident that, in terms of activation functions, the e-Swish, and PReLU not only better optimize and outperform the currently used ReLU but also, ensure faster convergence and lesser training time. In terms of other optimisations and regularizations, additional softmax layers, data augmentation, sherpa optimisation and code shuffling outperform the ReLU benchmark but the grid, random search are on the borderlines of ReLU. Future work in this promising direction could entail newer and novel functions applied to more complex models.

The non-normalized, distributed data with changing behavioral attributes and complex curves often pose new challenges. This ambitious research venture could redefine modern processing with respect to time series analysis and forecasting where age old contemporary systems seem to have failed miserably with techniques that are possibly profoundly flawed.

With the consideration of the aforementioned ideology, these upcoming architectures are expected to rule and drive systems of the future where technologies are rapidly advancing and landscapes are fast changing. While current technological revelations delineates the aforementioned scenarios, the future could spur out formidable and impressive inroads to advancements to not only effectively tackle current challenges but to create and solve newer problems in this space that we don't even know exist yet.

References

[1] Third International Symposium on Computer Vision and The Internet - VisionNet'16 Vinay. A. (2016) RISA: Rotation Illumination Scale and Affine Invariant Face Recognition. In ACM Digital Library ISBN: 978-1-4503-4301-5, Jaipur, India. September 2016. (New York, NY, USA: ACM)

[2] Thirty First Conference on Neural Information Processing Systems - NIPS Sabour S. (2017) Dynamic Routing between Capsules. In Computer Vision and Pattern Recognition - arXiv:1710.09829, Long Beach, California, United States. November 2017.

[3] GE Hinton et. al. (1981) *Shape Representation in parallel systems* (International Joint Conference on Artificial Intelligence)

[4] Hinton G.E. (2011) *Transforming Auto-Encoders* (Artificial Neural Networks and Machine Learning âĂŞ ICANN , Lecture Notes in Computer Science), vol 6791. Springer, Berlin, Heidelberg.

[5] Sara Sabour, Nicholas Frosst and Geoffrey E Hinton (2018) *Matrix Capsules with EM Routing* (6th International Conference on Learning Representations âĂŞ ICLR)

[6] Joanna Krajewska (2017) *How Capsule Network works on lung cancer classification problem?* (Enigma Pattern)

[7] Kai Qiao et. al. (2018) *Accurate reconstruction of image stimuli from human fMRI based on the decoding model with capsule network architecture* (Computer Vision and Pattern Recognition; Artificial Intelligence (cs.AI); Neurons and Cognition (q-bio.NC), arXiv: 1801.00602)

[8] Ayush Jaiswal et. al. (2018) *CapsuleGAN: Generative Adversarial Capsule Network* (Machine Learning (stat.ML),

Learning (cs.LG); arXiv: 1802.06167)
[9] Gagana B et. al. (2018) *Activation Function Optimizations for Capsule Networks* (International Conference on Advances in Computing, Communications and Informatics, ISBN: 978-1-5386-5314-2)
[10] A. L. Mass et.al. (2013) *Rectifier nonlinearities improve neural network acoustic models* (International Conference on Machine Learning - ICML), vol. 30.
[11] Quoc V. Le et. al. (2017) *Swish: a Self-Gated activation function* (Neural and Evolutionary Computing (cs.NE); Computer Vision and Pattern Recognition (cs.CV); Learning (cs.LG), arXiv: 1710.05941).
[12] Eric Alcaide (2018) *E Swish: Adjusting Activations to Different Network Depths* (Computer Vision and Pattern Recognition (cs.CV); Learning (cs.LG); Machine Learning (stat.ML), arXiv: 1801.07145).
[13] Leon A. Gatys et. al. (2016) *Image Style Transfer Using Convolutional Neural Networks* (Conference on Computer Vision and Pattern Recognition).
[14] Ming Liang et. al. (2015) *Recurrent Convolutional Neural Network for Object Recognition* (Conference on Computer Vision and Pattern Recognition).
[15] Wei Chu et. al. (2011) *Unbiased Online Active Learning in Data Streams* (Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining).
[16] James Bergstra et. al. (2012) *Random Search for Hyper-Parameter Optimization* (Journal of Machine Learning Research).
[17] Peter Sadowski (2018) *Sherpa: Hyperparameter Optimization for Machine Learning Models* (Thirty-second Conference on Neural Information Processing Systems).
[18] Lutz Prechelt (1998) *Early Stopping - But When?* (Orr G.B., MÃijller KR. (eds) Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, vol 1524. Springer, Berlin, Heidelberg).
[19] Alex Krizhevsky (2009) *Learning Multiple Layers of Features from Tiny Images* (Technical Article).
[20] Andrew L Maas et. al. (2013) *Rectifier Nonlinearities Improve Neural Network Acoustic Models* (Proceedings of the 30th International Conference on Ma-chine Learning, Atlanta,Georgia, USA).
[21] Beata Strack et. al. (2014) *Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records* (BioMed Research International Journal), Article ID 781670, 11 pages.
[22] Skriver MV et. al. (2012) *Short-term impact of HbA1c on morbidity and all-cause mortality in people with type 2 diabetes: A Danish population-based observational study* (Diabetologia, Springer-Verlag), Volume 55, Issue 9, pp 2361âĂŞ2370, 1432-0428