LiSHT: Non-Parametric Linearly Scaled Hyperbolic Tangent Activation Function for Neural Networks

Swalpa Kumar Roy, Suvojit Manna, Shiv Ram Dubey, and Bidyut Baran Chaudhuri

***Abstract*—The activation function in neural network is one of the important aspects which facilitates the deep training by introducing the non-linearity into the learning process. However, because of zero-hard rectification, some of the existing activation functions such as ReLU and Swish miss to utilize the large negative input values and may suffer from the dying gradient problem. Thus, it is important to look for a better activation function which is free from such problems. As a remedy, this paper proposes a new non-parametric function, called Linearly Scaled Hyperbolic Tangent (LiSHT) for Neural Networks (NNs). The proposed LiSHT activation function is an attempt to scale the non-linear Hyperbolic Tangent (Tanh) function by a linear function and tackle the dying gradient problem. The training and classification experiments are performed over benchmark Iris, MNIST, CIFAR10, CIFAR100 and twitter140 datasets to show that the proposed activation achieves faster convergence and higher performance. A very promising performance improvement is observed on three different type of neural networks including Multi-layer Perceptron (MLP), Convolutional Neural Network (CNN) and Recurrent neural network like Long-short term memory (LSTM). The advantages of proposed activation function are also visualized in terms of the feature activation maps, weight distribution and loss landscape. The code is available at https://github.com/swalpa/lisht.**

***Index Terms*—Activation, Convolutional Neural Networks, Non-Linearity, Tanh function, Image Classification.**

I. Introduction

THE deep learning method is one of the breakthroughs which replaced the hand-tuning tasks in many problems including computer vision, speech processing, natural language processing, robotics, and many more [1], [2], [3]. In recent times, the deep Artificial Neural Networks (ANNs) have shown a tremendous performance improvement due to availability of large scale datasets as well as high end computational resources [4]. Various types of ANN have been proposed for different type of problems such as Multilayer Perceptrons (MLP) [5] to deal with the real vector R-dimensional data [6], [7]. Convolutional Neural Networks (CNN) are used to deal with the image and videos [8], [9], [10]. Recurrent Neural Network (RNN) like Long-Short Term Memory (LSTM) are used for the speech and text data classification [11] and the sentiment analysis [12], etc.

S.K. Roy is with Department of Computer Science & Engineering, Jalpaiguri Government of Engineering College, West Bengal, India (email: swalpa@cse.jgec.ac.in).

S. Manna is with the CureSkin as a Data Scientist, Bengaluru, Karnataka, India (email: suvojit@heallo.ai).

S.R. Dubey is with the Computer Vision and Biometrics Lab, Indian Institute of Information Technology (IIIT), Allahabad, Uttar Pradesh, India. Earlier, he was IIIT Sri City, India (e-mail: srdubey@iiita.ac.in).

Techno India University, Kolkata, India and Indian Statistical Institute, Kolkata, India (email: bidyutbaranchaudhuri@gmail.com).

Recently, a multipath ensemble CNN (ME-CNN) model is proposed which directly concatenates the low-level with that of high-level output features [13]. In ME-CNN, the back-propagation has shorter path, suitable for the ultradeep network training. Ensemble Neural Networks (ENN) is introduced as an efficient stochastic gradient-free optimization method that relies on covariance matrices [14]. Wen et al. proposed a Memristive Fully Convolutional Network (MFCN) for accelerating the hardware image-segmentor, where the memory segment is able to do the computation [15]. A light gated recurrent unit (LGRU) based neural network is proposed recently by Ravanelli et al. for Speech Recognition [16]. Recently, the deep learning is used in the different tasks such as deep CNN for audio-visual speech enhancement [17], RCCNet (a CNN) for histological routine colon cancer nuclei classification [18], DFUNet (a CNN) for diabetic foot ulcer classification [19], combining CNN and RNN for video dynamics detection [20], dual CNN models for unsupervised monocular depth estimation [21], deep CNNs for face anti-spoofing [22], and node classification [23], hyperspectral image classification [24], [25], [26], and nonsymmetric deep autoencoder (NDAE) based unsupervised deep learning for network intrusion detection [27], etc. The neural architecture learning is also one of the recent areas where various activation functions are being explored [28], [29].

The main aim of any type of neural network is to transform the input data in some other feature space, where it becomes linearly separable into classes. In order to achieve it, all the neural networks rely on a compulsory unit called the activation function [30]. The job of activation functions is to introduce the non-linearity in the network to facilitate automatic learning of connection weights for a specific problem. Thus, the activation function is the backbone of any neural network.

The $Sigmoid$ activation function was commonly used in the early days of neural networks. It is a special case of the logistic function. The $Sigmoid$ function squashes the real-valued numbers into 0 or 1. In turn, the large negative number becomes 0 and large positive number becomes 1. The hyperbolic tangent function $Tanh$ is the another popular activation function. The output values of $Tanh$ lie between -1 and 1 depending on the input real values. The vanishing gradient in both positive as well as negative directions is one of the major problems with both $Sigmoid$ and $Tanh$ activation functions. The Rectified Linear Unit ($ReLU$) activation function was proposed in recent past for training deep networks [8]. $ReLU$ is a breakthrough against vanishing gradient. It is an identity function for the non-negative inputs and zero function (i.e., the output is zero) for the negative inputs. The $ReLU$ has become very popular due to its simplicity and used in various types of neural networks. The major drawback with $ReLU$ is the diminishing gradient for the negative values of the inputs. Due to the diminishing gradient, the $ReLU$ units can be fragile during training and the gradient may die.

Several researchers have proposed the improvement on $ReLU$ such as Leaky ReLU ($LReLU$) [31], Parametric ReLU ($PReLU$) [32], $Softplus$ [33], Exponential Linear Unit (ELU) [34], Scaled Exponential Linear Unit ($SELU$) [35], Gaussian Error Linear Unit ($GELU$) [36], Average Biased ReLU ($ABReLU$) [37], Linearized sigmoidal activation ($LiSA$) [38] etc. The $LReLU$ is similar to $ReLU$, except it allows a small, non-negative and constant gradient (such as 0.01) for the negative regime of input in oder to reduce the dying neuron input problem [31]. The $PReLU$ is the extension of $LReLU$ by training the coefficient of leakage into a parameter instead of fixing it with a constant value [32]. The leakage parameter in $PReLU$ is learned along with the other neural network parameters. Thus, both $LReLU$ and $PReLU$ are in between the linear function and $ReLU$ which is a drawback in terms of decreased amounts of non-linearity. The $Softplus$ activation function tries to make the transition of $ReLU$ (i.e., at 0) smooth by fitting the log function [33]. Otherwise, the $Softplus$ activation is very similar to the $ReLU$ activation. The ELU function is very similar to $ReLU$ in terms of the identity functions for non-negative inputs [34]. Unlike the $ReLU$ which sharply becomes smooth, the ELU becomes smooth slowly until its output equal to a constant negative value. For positive inputs, the ELU [34] can blow up the activation, which can lead to the gradient exploding problem. The $SELU$ adds one scaling fixed parameter in ELU which sustains the bad weight initialization [35] better. The $GELU$ randomly applies the identity or zero map to a neuron's input according to Gaussian fashion [36]. The shape of $GELU$ is in between the $ReLU$ and ELU. The $ABReLU$ is proposed recently which allows the prominent negative values after converting it into positive values by biasing it with the average of all activations [37]. The $ABReLU$ also could not utilize all the negative values due to trimming of values at zero, similar to $ReLU$. Most of these existing activation methods are sometimes not able to take the advantage of negative values which is solved in the proposed $LiSHT$ activation.

Recently, Xu et al. have performed an empirical study of rectified activations in CNNs [39]. Very recently, Ramachandran et al. proposed a sigmoid-weighted linear unit $Swish$ activation function [40]. They tested with many functions and found $Swish$, i.e., $f(x) = x \times sigmoid(\beta x)$ is a promising activation function. The $Swish$ activation function interpolates between $ReLU$ and Linear function, by learning the parameter β. However, the gradient diminishing problem is still present in case of $Swish$ function. Recently, an attempt is made to design the complex and nonparametric activation function for complex-valued neural networks (CVNNs) [41]. It relies over the kernel expansion with a fixed dictionary which can be implemented on vectorized hardware.

The existing activation functions are still weak to replace the $ReLU$ due to inconsistent gains. In this paper, a linearly scaled hyperbolic tangent activation function ($LiSHT$) is proposed to introduce the non-linearities in the neural networks. The $LiSHT$ scales the $Tanh$ function linearly to tackle its gradient diminishing problem. Here, some classification tackle experiments are performed to show the improvement over variety of datasets on three type of neural networks.

The contributions of this paper are as follows,

- A new activation function named non-parametric Linearly Scaled Hyperbolic Tangent ($LiSHT$) is proposed by linearly scaling the $Tanh$ activation function.
- The increased amount of non-linearity of the proposed activation function is visualized from its first and second order derivatives curves (Fig. 1).
- The proposed $LiSHT$ activation function is tested with different types of neural networks, including older Artificial Neural Network (ANN) by Multilayer Perceptron (MLP), Convolutional Neural Network (CNN) by Residual Neural Network (ResNet), and Recurrent Neural Network (RNN) by Long-Short Term Memory (LSTM) network.
- Three different types of experimental data are used 1) $\mathbb{R}$-dimensional data, including Iris and MNIST (converted from image) datasets, 2) 2-D image data, including MNIST, CIFAR-10 and CIFAR-100 datasets, and 3) sentiment analysis data, including twitter140 dataset.
- The impact of different non-linearity functions over activation feature maps and weight distribution has been analyzed.
- The activation maps, weight distributions and optimization landscape are also analyzed to show the effectiveness of the proposed *LiSHT* activation function.

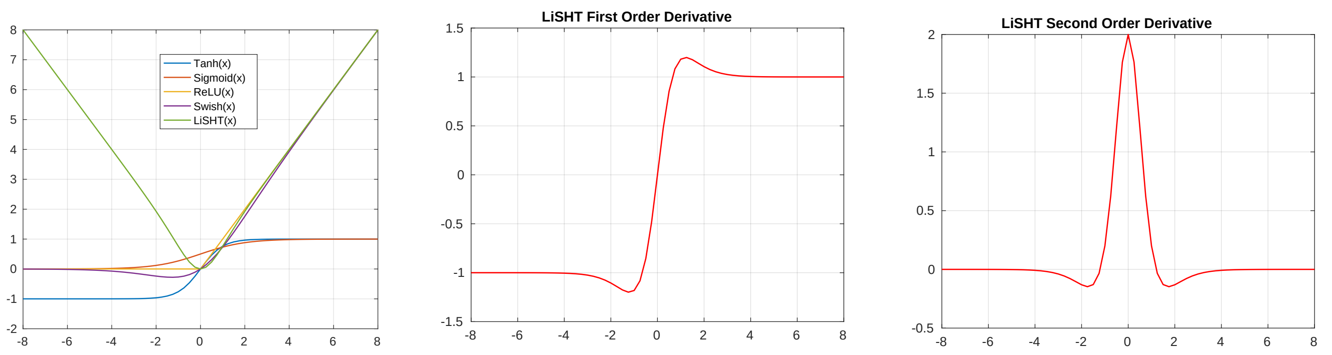
This paper is organized as follows: Section 2 outlines the proposed $LiSHT$ activation; Section 3 describes the experimental setup; Section 4 presents the results and analysis; and Section 5 contains the concluding remarks.

II. Proposed LiSHT Activation Function

A feed-forward Deep Neural Network (DNN) comprises of more than one hidden nonlinear layer. Let an input vector be $x \in \mathbb{R}^d$, and each hidden layer is capable to transform its input vector by applying a linear affine transform followed by a nonlinear mapping from the l^{th} layer to the $(l+1)^{th}$ layer as follows:

$$\left.\begin{array}{ll} \tau^0 & = x \\ s_i^{l+1} & = \sum_{j=1}^{N^l} w_{ij}^l \tau_j^l + o_i^l \\ \tau_i^{l+1} & = \phi(s_i^{l+1}) \end{array}\right\} \quad (1)$$

Here, τ represents the activation volume of any given layer, s_i^l, w_{ij}^l, o_i^l, and N^l represent the vectors of output, weights, biases and number of units in the hidden l^{th} layer, respectively, and a non-linear activation mapping $\phi(z)$. Looking for an efficient and powerful activation function in DNN is always demanding due to the overabundance by the *saturation* properties of existing activation functions. An activation function $\phi(z)$ is said to be saturate [42], if its derivative $\phi'(z)$ tends to zero in both directions (i.e., $z \to \infty$ and $z \to -\infty$,



(a) (b) (c)

Fig. 1: (a) The characteristics of the activation functions by visualization of the proposed *LiSHT* and commonly used activation functions such as $Tanh$, $Sigmoid$, $ReLU$, and $Swish$. It can be noted that the proposed $LiSHT$ introduces more amount of non-linearity as compared to the other activations. Moreover, the proposed activation is symmetric w.r.t. y-axis and converts the negative values into positive values. This property of the proposed $LiSHT$ activation function helps to avoid the gradient diminishing problem of other activation functions. (b) The 1^{st} order Derivative of proposed *LiSHT* activation. The 1^{st} order derivative is positive for the positive input and negative for the negative input. This property of the proposed $LiSHT$ activation function helps to avoid the gradient exploding problem of other activation functions. (c) The 2^{nd} order Derivative of the proposed *LiSHT* activation. Very importantly, it is similar to the opposite of the Laplacian operator (i.e., 2^{nd} order derivative of Gaussian), which is used to find the maximum. Thus, it makes sense to use the activation function like the proposed $LiSHT$ which is similar to the opposite of Gaussian for the better training of the neural networks in terms of the minimization of loss functions.

respectively). The training of a deep neural networks is almost impossible with of $Sigmoid$ and $Tanh$ activation functions due to the gradient diminishing problem when input is either too small or too large [4]. For the first time, the Rectified Linear Unit ($ReLU$) (i.e., $\phi(z) = max(0, z)$) became very popular activation for training the DNN [8], [43]. But, $ReLU$ also suffers due to the gradient diminishing problem for negative inputs which lead to the dying neuron problem.

Hence, we propose a non-parametric linearly scaled hyperbolic tangent activation function, so called $LiSHT$. Like $ReLU$ [8] and $Swish$ [40], [44], $LiSHT$ shares the similar unbounded upper limits property on the right hand side of activation curve. However, because of the symmetry preserving property of $LiSHT$, the left hand side of the activation is in the upwardly unbounded direction, hence it satisfies non-monotonicity (see Fig. 1(a)). Apart from the literature [34], [40] and to the best of our knowledge, first time in the history of activation function, $LiSHT$ utilizes the benefits of positive valued activation without identically propagating all the inputs, which mitigates gradient vanishing at back propagation and acquiesces faster training of deep neural network. The proposed activation function $LiSHT$ is computed by multiplying the $Tanh$ function to its input z and defined as,

$$\phi(z) = z \cdot g(z) \tag{2}$$

where $g(z)$ is a hyperbolic tangent function and defined as,

$$g(z) = Tanh(z) = \frac{exp^z - exp^{-z}}{exp^z + exp^{-z}}. \tag{3}$$

where z is the input to the activation function and exp is the exponential function.

For the large positive inputs, the behavior of the $LiSHT$ is close to the $ReLU$ and $Swish$, i.e., the output is close to the input as depicted in Fig. 1(a). Whereas, unlike $ReLU$ and other commonly used activation functions, the output of $LiSHT$ for negative inputs is symmetric to the output of $LiSHT$ for positive inputs as illustrated in Fig. 1(a). The 1^{st} order derivative (i.e., $\phi'(z)$) of $LiSHT$ is given as follows,

$$\begin{aligned}\phi'(z) &= z[1 - Tanh^2(z)] + Tanh(z)\\ &= z + Tanh(z)[1 - \phi(z)].\end{aligned} \tag{4}$$

Similarly, the 2^{nd} order derivative (i.e., $\phi''(z)$) of $LiSHT$ is given as follows,

$$\begin{aligned}\phi''(z) &= 1 - Tanh(z)\phi(z) + [1 - \phi(z)](1 - Tanh^2(z))\\ &= 2 - Tanh(z)\phi'(z) - \phi(z) - Tanh(z)[\phi'(z) - z]\\ &= 2[1 - Tanh(z)\phi'(z)].\end{aligned} \tag{5}$$

The 1^{st} and 2^{nd} order derivatives of the proposed *LiSHT* are plotted in Fig. 1(b) and Fig. 1(c), respectively. An attractive characteristic of the *LiSHT* is *self-stability* property, the magnitude of derivatives is less than 1 for $z \in [-0.65, 0.65]$. It can be observed from the derivatives of $LiSHT$ in Fig. 1 that the amount of non-linearity is very high near to zero as compared to the existing activations which can boost the learning of a complex model. Another major advantage of proposed activation is due to the nearly linear nature for very small and very large inputs which resolves the gradient diminishing problem of existing methods. It can be seen in Fig. 1(b) that the gradients of proposed $LiSHT$ activation function have both the positive and negative values, which tackles the problem of exploding gradient of $ReLU$. As described in Fig. 1(c) that the 2^{nd} order derivative of proposed $LisHT$ activation function is similar to the opposite of the Laplacian operator (i.e., the 2^{nd} order derivative of Gaussian

Fig. 2: Random sample images taken from each category of MNIST (left) and CIFAR-10 (right) dataset. CIFAR-10 images are taken from https://www.cs.toronto.edu/∼kriz/cifar.html.

operator) which is useful to maximize a function. Thus, due to opposite nature of Gaussian operator, the proposed $LiSHT$ activation function boosts the training of the neural network for the minimization problem of the loss function.

Theoretically, it is difficult to prove why an activation function works well, it may be due to many puzzling factors that directly affect DNN training. We understand that being unbounded in both positive and negative directions, smooth, and non-monotonicity are the advantages of the proposed $LiSHT$ activation. The complete unbounded property makes $LiSHT$ different from all the traditional activation functions. Moreover, it makes use of strong advantage of positive feature space, which is suitable for assessing the "information content" of features in complex classification tasks. Unlike $ReLU$, $LiSHT$ is a smooth and non-monotonic function. In addition $LiSHT$ is a symmetric function and introduces more amount of non-linearity in the training process than $Swish$.

III. Experimental Setup

This section is devoted to the experimental setup used in this paper. First, six datasets are described in detail, then the three types of networks are summarized, and finally the training settings are stated in detail.

A. Datasets Used

The effectiveness of the proposed *LiSHT* activation function is evaluated on five benchmark databases, including Iris, MNIST, CIFAR-10, CIFAR-100 and twitter140. The **Fisher's Iris Flower dataset**[1] [45] is a classic and one of the best known multivariate dataset that contains a total of 150 samples from 3 different species of Iris, including Iris setosa, Iris virginica and Iris versicolor. In Iris dataset, every sample is a vector of length four (i.e., "sepal length", "sepal width", "petal length", and "petal width"). The **MNIST dataset** is widely used for recognizing the English digits from images. This dataset contains a total of $60K$ training samples and $10K$ test samples of dimension 28×28 [46]. In this dataset, one image contains only one digit from 0 to 9. The sample images of the MNIST dataset are shown Fig. 2. The **CIFAR-10 dataset** consists of 32×32 resolution $60K$ color images

[1]C. Blake, C. Merz, UCI Repository of Machine Learning Databases.

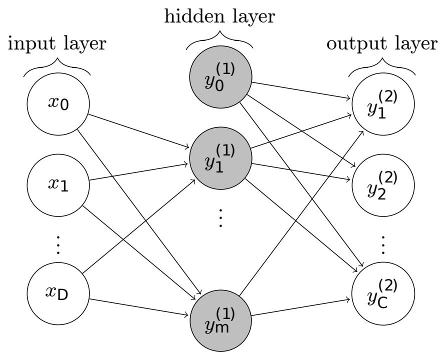


Fig. 3: A multi-layer perceptron (MLP) with C input units, D output units and $m := m^{(1)}$ hidden units, where superscript 1 represents a single hidden layer.

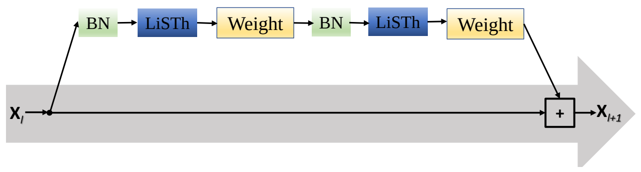


Fig. 4: The Pre-Activated Residual Module: BN, LiSHT, and Weight represent batch normalization, activation, and convolutional (Conv) layers, respectively. The X_l and X_{l+1} are the input and output volumes, respectively. The residual module basically adds the input volume with output volume of the second Conv layer to produce the final output volume.

from 10 classes, with $6K$ images per class [47]. The $50K$ images with $5K$ images per class are used for the training and $10K$ images with $1k$ images per class are used for the testing. The 10 classes of CIFAR-10 dataset are from the'airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', and 'truck' categories, respectively. See Fig. 2 for the sample images of CIFAR10 dataset. The **CIFAR-100 dataset** contains the same CIFAR-10 dataset images (i.e., $50K$ for training and $10K$ for testing). The training and testing images are categorized into 100 classes in CIFAR-100 dataset. The **twitter140 dataset** [48] is used to perform the classification of sentiments of Twitter messages by classifying as either positive, negative or neutral with respect to a query. In this dataset, we have considered 1,600,000 examples, where 85% are used as training set and the rest 15% as validation set.

B. Tested Neural Networks

Three different types of neural networks (i.e., a Multi-layer Perceptron (MLP) with one hidden layer, a widely used deep convolutional Pre-activation Residual Network (ResNet-PreAct) [49], and a Long-Sort Term Memory (LSTM)) were tested to show the performance of activation functions. These architectures are explained in this section. The **Multi-layer Perceptron (MLP)** with one hidden layer is used in this paper for the classification of data. The typical structure of MLP is illustrated in Fig. 3. The internal architecture in MLP uses input, hidden and final $softmax$ layer with 6, 5, and 4 nodes

TABLE I: The classification performance of MLP for different activations over
Iris and MNIST datasets.

Dataset	Activation Function	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
Iris	Tanh	0.0937	97.46	0.0898	96.26
	Sigmoid	0.0951	97.83	0.0913	96.23
	ReLU	0.0983	98.33	0.0886	96.41
	PReLU	0.0318	99.28	0.0629	97.11
	LReLU	0.0273	99.06	0.0891	96.53
	Swish	0.0953	98.50	0.0994	96.34
	LiSHT	0.0926	98.67	0.0862	97.33
MNIST	Tanh	0.0138	99.56	0.0987	98.26
	Sigmoid	0.0064	99.60	0.0928	98.43
	ReLU	0.0192	99.51	0.1040	98.48
	PReLU	0.0184	99.54	0.1179	98.34
	LReLU	0.0361	98.92	0.1034	97.69
	Swish	0.0159	99.58	0.1048	98.45
	LiSHT	0.0127	99.68	0.0915	98.60

for the Car evaluation dataset. For Iris Flower dataset, the MLP uses 4, 3, and 3 nodes in the input, hidden and final $softmax$ layer, respectively. The MNIST dataset images are stretched into one-dimensional vectors when used with MLP. Thus, for MNIST dataset, the MLP uses 784, 512, and 10 nodes in the input, hidden and final $softmax$ layer, respectively. The **Residual Neural Network (ResNet)** is the state-of-the-art for the image classification task. In this the improved version of ResNet (i.e., ResNet with Pre Activation [49]) is used for the image classification over MNIST, CIFAR-10, and CIFAR-100 datasets. The ResNet-PreAct is used with 164-layer (i.e., very deep network) for CIFAR-10 and CIFAR-100 datasets, whereas it is used with 20-layer for MNIST dataset. The pre-activation residual block is illustrated in Fig. 4. The channel pixel mean subtraction is used for preprocessing over image datasets with this network as per the standard practice being followed by most image classification neural networks. In this paper, the **Long Short Term Memory (LSTM)** is used as the third type of neural network, which basically belongs to the Recurrent Neural Network (RNN) family. A single layered LSTM with 196 cells is used for sentiment analysis over twitter140 dataset. The LSTM is fed with 300 dimensional word vectors trained with FastText Embeddings.

TABLE II: The classification performance of ResNet for different activations over
MNIST and CIFAR-10/100 datasets.

Dataset	ResNet Depth	Tanh	ReLU	PReLU	LReLU	Swish	LiSHT
		Activation Functions					
MNIST	20	99.48	99.56	99.56	99.52	99.53	99.59
CIFAR-10	164	89.74	91.15	92.86	91.50	91.60	92.92
CIFAR-100	164	68.80	72.84	73.01	72.24	74.45	75.32

C. Training Settings

The Keras deep learning Python libraries with TensorFlow at the back-end are used for the implementation of activation function. Different computer systems, including different GPUs (such as NVIDIA Titan X, Pascal 12GB GPU and NVIDIA Titan V 12GB GPU) are used at different stages of the experiments. The $Adam$ optimizer [50], [51] is used for the experiments in this paper. The batch size is set to 128 for the training of the networks. The learning rate is initialized to 0.1 and reduced by a factor of 0.1 at 80^{th}, 120^{th}, 160^{th}, and 180^{th} epochs during training. For LSTM, after 10625 iteration on 128 sized mini-batches, the learning rate is dropped by a factor of 0.5 up to $212,500$ mini-batch iterations.

TABLE III: The classification performance of LSTM for different activations over
twitter140 dataset.

Dataset	Tanh	ReLU	LReLU	Swish	LiSHT
	Activation Functions				
Twitter140	82.27	82.47	78.18	82.22	82.47

IV. Results and Analysis

We investigate the performance and effectiveness of the proposed $LiSHT$ activation and compare with state-of-the-art activation functions such as $Tanh$, $ReLU$, and $Swish$. In this section, first the experimental results using MLP, ResNet and LSTM are presented, then the results analysis are done in terms of the accuracy and loss vs epochs, then the analysis of feature activation maps and weight distribution is performed for different activation functions, and finally the effect of activation functions is analyzed using loss landscape.

A. Data Classification Results using MLP

The classification performance using MLP over Cars, Iris and MNIST datasets are reported in Table I in terms of the loss and accuracy over training and validation sets. The training is performed for 200 epochs using the *categorical cross-entropy* loss. In order to run training smoothly in both the dataset, 80% of samples were randomly chosen for training and remaining 20% are used for validation. The proposed $LiSHT$ activation achieves minimum training and validation loss over the datasets. The proposed $LiSHT$ activation function achieves the best accuracies of 97.33% and 98.60% over Iris and MNIST datasets, respectively.

B. Image Classification Results using ResNet

Table II summarizes the validation accuracies of different activations over MNIST, CIFAR-10 and CIFAR-100 datasets with pre-activation ResNet. The depth of ResNet is 20 for MNIST and 164 for CIFAR datasets. The training is performed for the 200 epochs using the *categorical cross-entropy* loss. It is observed that the proposed $LiSHT$ activation function outperforms the other activation functions with a record accuracies of 99.59% and 92.92%, and 75.32% over MNIST, CIFAR-10 and CIFAR-100 datasets, respectively. Moreover, a significant improvement has been shown by $LiSHT$ on CIFAR datasets. The unbounded, symmetric and more non-linear properties of the proposed $LiSHT$ activation function facilitates better and efficient training as compared to the other activation functions such as $Tanh$, $ReLU$ and $Swish$. The unbounded and symmetric nature of $LiSHT$ leads to the more exploration of weights and positive and negative gradients to tackle the gradient diminishing and exploding problems.

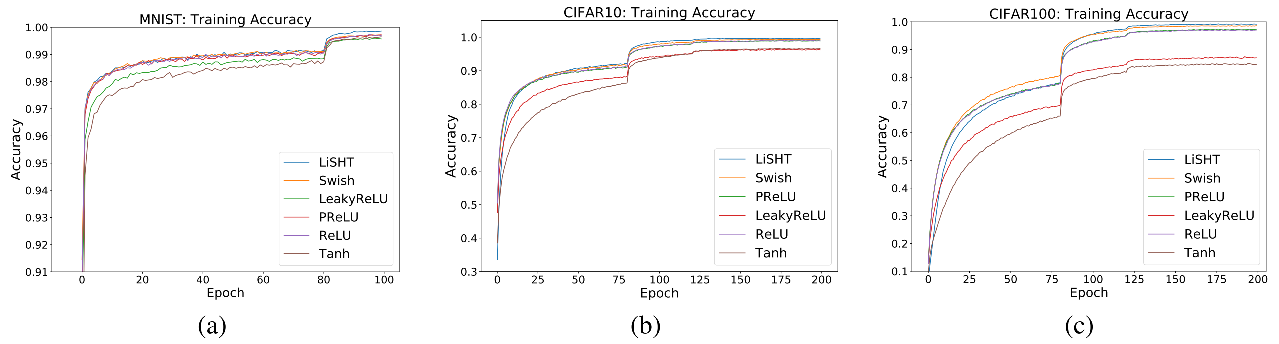


Fig. 5: The classification accuracy for training sets using the $LiSHT$ and state-of-the-art $Tanh$, $ReLU$ and $Swish$ activations with ResNet model over (a) MNIST (b) CIFAR-10 and (c) CIFAR-100 datasets.

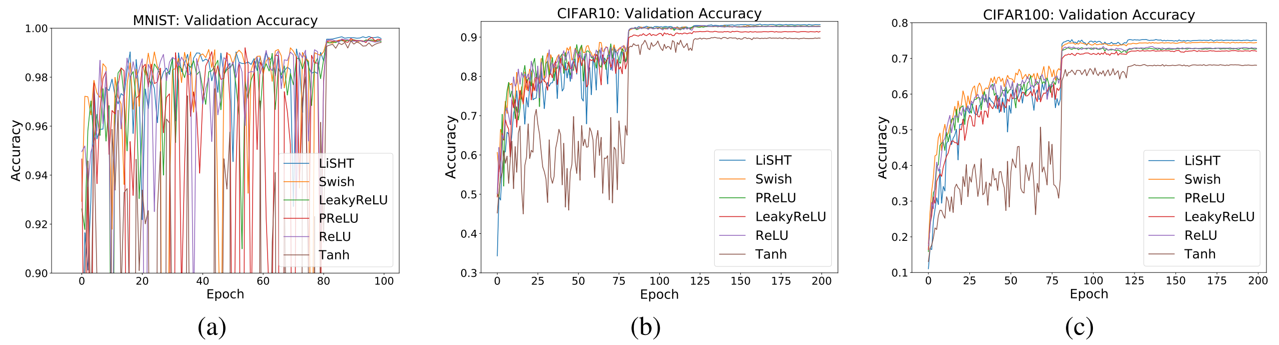


Fig. 6: The classification accuracy for validation sets using the $LiSHT$ and state-of-the-art $Tanh$, $ReLU$ and $Swish$ activations with ResNet model over (a) MNIST (b) CIFAR-10 and (c) CIFAR-100 datasets.

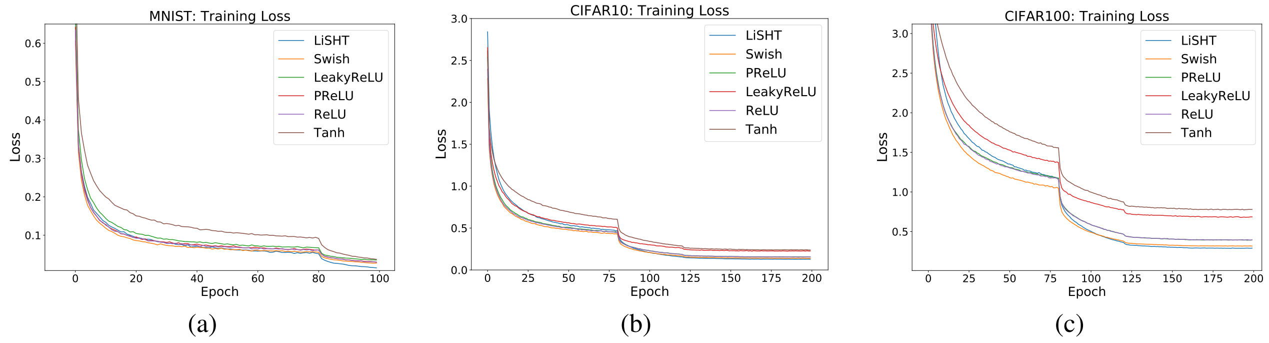


Fig. 7: The convergence curves in terms of loss for training sets using the $LiSHT$ and state-of-the-art $Tanh$, $ReLU$, and $Swish$ activations with ResNet model over (a) MNIST (b) CIFAR-10 and (c) CIFAR-100 datasets.

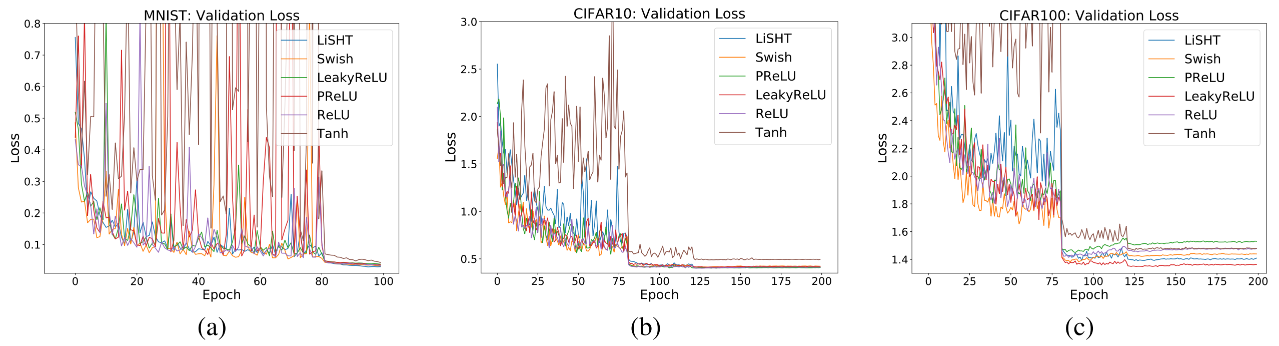
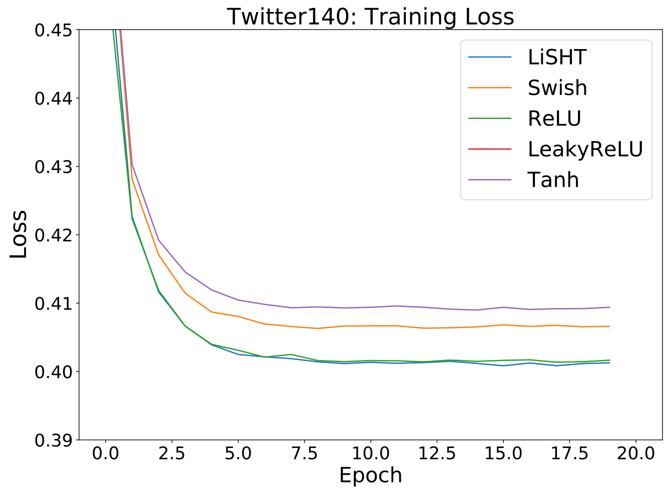
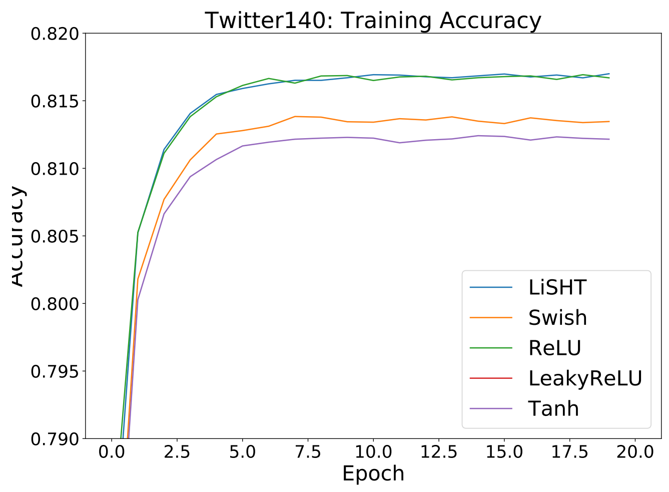


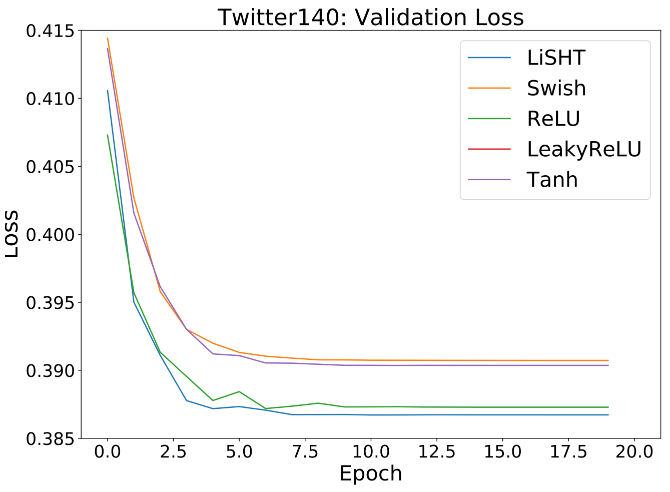
Fig. 8: The convergence curves in terms of loss for validation sets using the $LiSHT$ and state-of-the-art $Tanh$, $ReLU$, and $Swish$ activations with ResNet model over (a) MNIST (b) CIFAR-10 and (c) CIFAR-100 datasets.



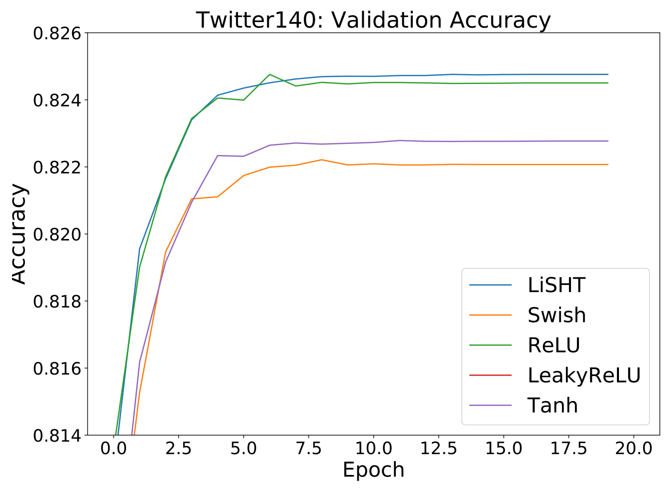
(a)



(b)

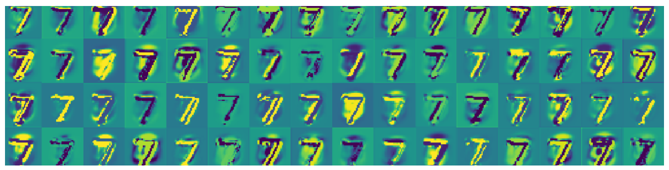


(c)

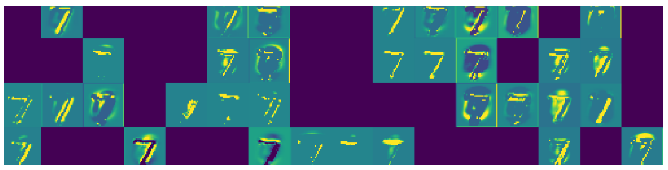


(d)

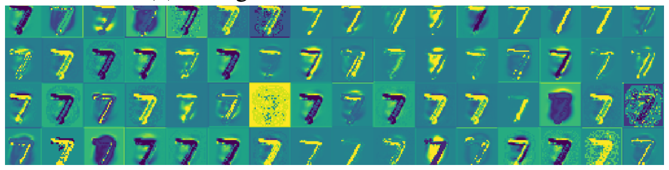
Fig. 9: (a)-(b) The training loss and training accuracy (c)-(d) validation loss and accuracy using LSTM over twitter140 dataset for different activations.



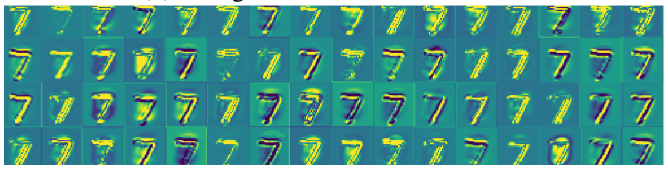
(a) Using $Tanh$ activation function



(b) Using $ReLU$ activation function



(c) Using $Swish$ activation function

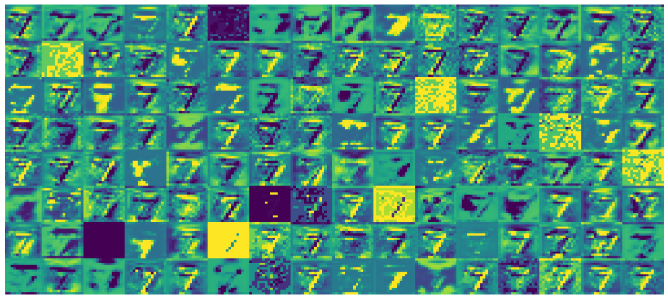


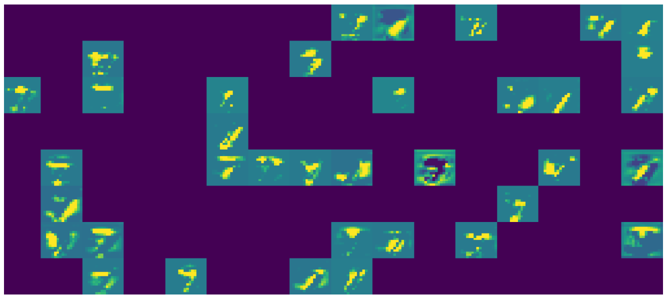
(d) Using $LiSHT$ activation function

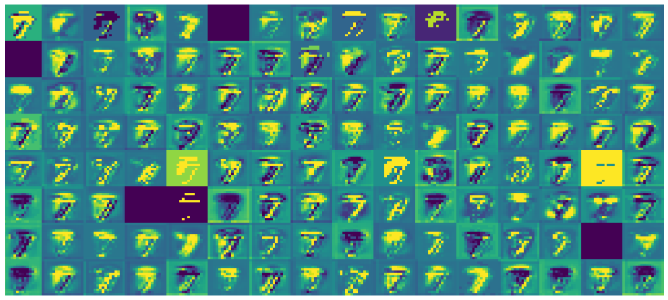
Fig. 10: Visualization of MNIST digit 7 from the 2^{nd} $conv$ layer activation feature maps without feature scale clipping using a fully trained pre-activation ResNet model using the (a) $Tanh$ (b) $ReLU$ (c) $Swish$ and (d) $LiSHT$ activation, respectively. Note that there are 64 feature maps of dimension 32×32 in the 2^{nd} layer, represented in 4 rows and 16 columns.

C. Sentiment Classification Results using LSTM

The sentiment classification performance in terms of the validation accuracy is reported in Table III over twitter140 dataset with LSTM for different activations. It is observed that the performance of proposed $LiSHT$ activation function is better than $Tanh$ and $Swish$, whereas the same as $ReLU$.


(a) Using $Tanh$ activation function


(b) Using $ReLU$ activation function


(c) Using $Swish$ activation function

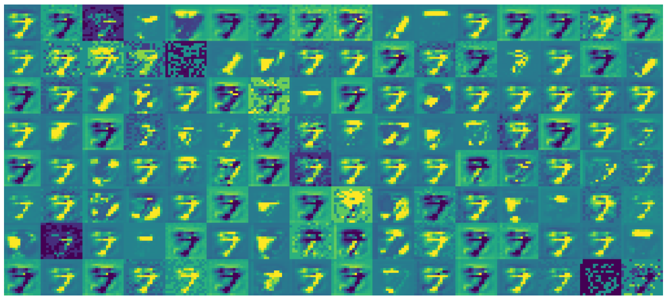
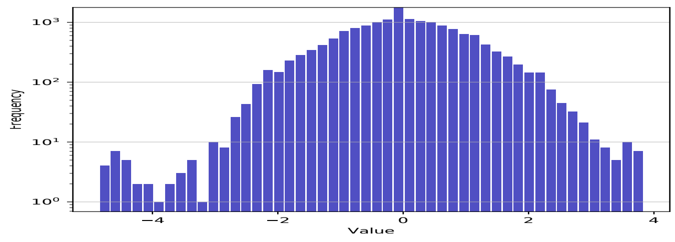
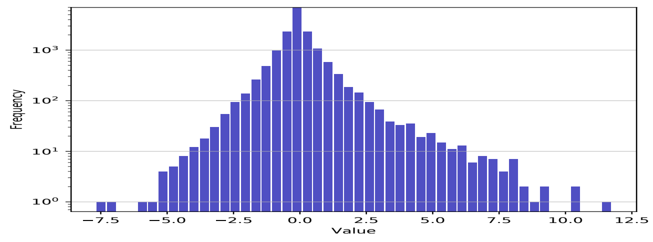

(d) Using $LiSHT$ activation function

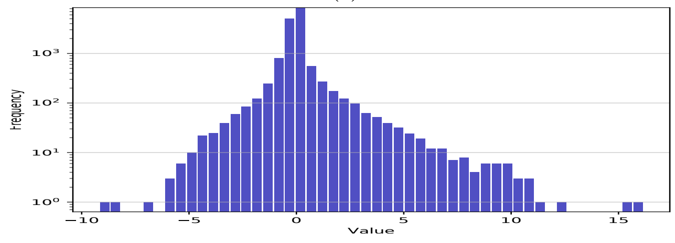
Fig. 11: Visualization of MNIST digit 7 from the 11^{th} $conv$ layer activation feature maps without feature scale clipping using a fully trained pre-activation ResNet model using the (a) $Tanh$ (b) $ReLU$ (c) $Swish$ and (d) $LiSHT$ activation, respectively. Note that there are 128 feature maps of dimension 16×16 in the 11^{th} layer, represented in 8 rows and 16 columns.



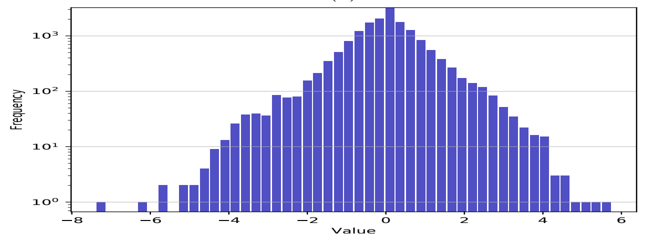
(a)



(b)



(c)



(d)

Fig. 12: Visualizations of the distribution of weights from the final $Conv$ layer of pre-activation ResNet over MNIST dataset for the (a) $Tanh$ (b) $ReLU$ (c) $Swish$ and (d) $LiSHT$ activations, respectively.

It points out one important observation that by considering the negative values as negative by $Swish$ degrades the performance because it leads the $Swish$ activation more towards the linear function as compared to the $ReLU$ activation.

D. Result Analysis

The training and validation accuracy over the epochs are plotted in Fig. 5(a)-(c) and 6(a)-(c) for MNIST, CIFAR-10 and CIFAR-100 datasets using ResNet. The convergence curve of losses is also used as the metric to measure the learning ability of the ResNet model with different activation functions. The training and validation loss over the epochs are plotted in Fig. 7(a)-(c) and 8(a)-(c) for MNIST, CIFAR-10 and CIFAR-100 datasets using ResNet. It can be clearly visualized that the proposed $LiSHT$ activation breakthrough the convergence speed as compared to the other activations. The Fig. 9(a)-(b) shows the convergence of validation loss and accuracy for LSTM network using different activations

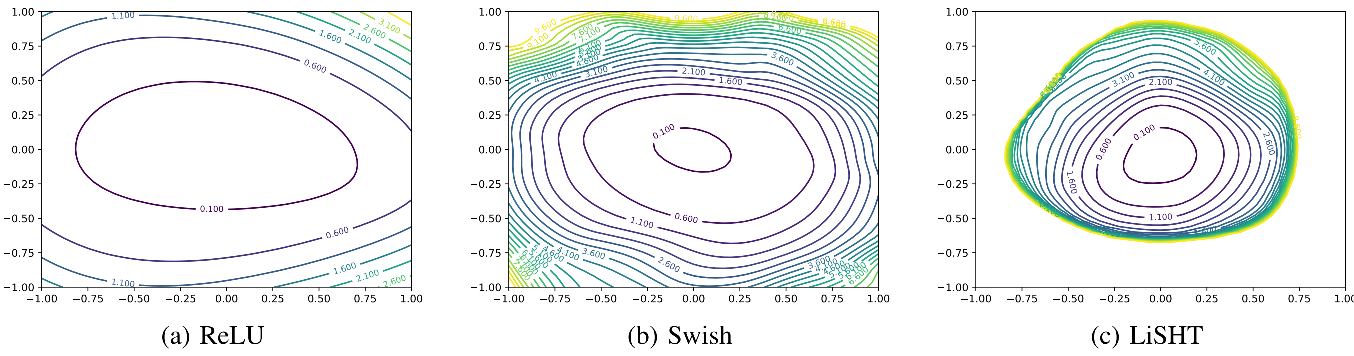


Fig. 13: The vizualization of 2D Loss Landscape plot of CIFAR-10 shown using ReLU, Swish and LiSHT, respectively.

over twitter140 sentiment classification dataset. It is clearly observed that the proposed $LiSHT$ boosts the convergence speed. It is also observed that the $LiSHT$ outperforms the existing non-linearities across several classification tasks with MLP, ResNet and LSTM networks. The training and validation losses converge faster using the $LiSHT$ activation.

E. Analysis of Activation Feature Maps

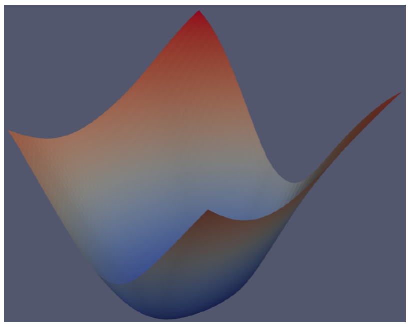
In deep learning, it is a common practice to visualize the activations of different layer of the network. In order to understand the effect of activation functions over the learning of important features at different layer, we have shown the activation feature maps for different non-linearities at 2^{nd} and 11^{th} layer of the pre-activation ResNet of MNIST digit 7 in Fig. 10 and 11, respectively. The number of activation feature maps in 2^{nd} and 11^{th} layers are 64 (each having the 32×32 spatial dimensions) and 128 (each having the 16×16 spatial dimensions), respectively. It can be seen from Fig. 10 and 11 that the images looking deeper blue are due to the dying neuron problem caused by the non-learnable behavior arose due to the improper handling of negative values by the activation functions. The proposed $LiSHT$ activation consistently outperforms other activations. It is observed that the $LiSHT$ generates the less number of non-learnable filters due to the unbounded nature in both positive and negative scenarios which helps it to overcome from the problem of dying gradient. It is also observed that some image patches contain noise in terms of the Yellow color. The patches corresponding to the $LiSHT$ contain less noise. Moreover, it is uniformly distributed over all the patches, when $LiSHT$ is used, compared to other activation functions. It may be also one of the factors that proposed $LiSHT$ outperforms other activations.

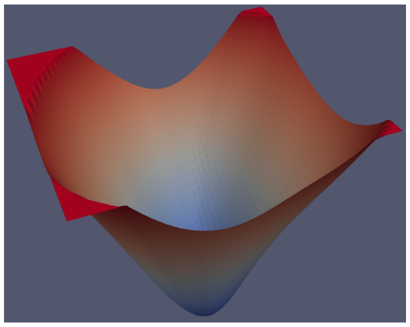
F. Analysis of Final Weight Distribution

The weights of the layers are useful to visualize because it gives the idea about the learning pattern of the network in terms of 1) the positive and negative biasedness and 2) the exploration of weights caused by the activation functions. It is also expected that the well-trained networks usually display well normalized and smooth filters without much noisy patterns. These characteristics are usually most interpretable on the first $Conv$ layer which is looking directly at the raw pixel label data, but it is also possible to show the filter weights deeper in the network to gain the intuition about the abstract level learning of the network. The learning of strong filter weights is highly dependent upon the non-linearity used for the training [52]. We have portrayed the weight distribution of final $Conv$ layer in Fig. 12 for pre-activation ResNet over the MNIST dataset using $Tanh$, $ReLU$, $Swish$ and $LiSHT$ activations. The weight distribution for $Tanh$ is limited in between -5 and 4 (see 12(a)) due to its bounded nature in both negative and positive regions. Interestingly, as depicted in 12(b), the weight distribution for $ReLU$ is biased towards the positive region because it converts all negative values to zero which restricts the learning of weights in the negative direction. This leads to the problems of dying gradient as well as gradient exploding. The $Swish$ tries to overcome the problems of $ReLU$, but unable to succeed due to the bounded nature in negative region (see 12(c)). The above mentioned problems are removed in the $LiSHT$ as suggested by its weight distribution shown in Fig. 12(d). The $LiSHT$ activation leads to the symmetric and smoother weight distribution. Moreover, it also allows the exploration of weights in the higher range (i.e., in between -8 and 6 in the example of Fig. 12).

G. Analysis of Loss Landscape

Training of deep neural networks requires minimizing a non-convex high-dimensional loss function – which is a hard task in theory, but sometimes easier in practice. Simple gradient descent methods [53] are often used to find global minimum/saddle points, where the defined configurations reaches training loss zero or near to zero, even when the data and labels are randomized before training. However, this behavior is desirable, but always not universal. The training ability of DNN is directly and indirectly influenced by the factors like network architecture, the choice of optimizer, variable initialization, and most importantly, what kind of non-linearity function to be used in the architecture. In order to understand the effects of network architecture on non-convexity, we trained the ResNet-152 using $ReLU$, $Swish$ and proposed $LiSHT$ activations and try to explore the structure of the neural network loss landscape. The $2D$ and $3D$ visualizations of loss landscapes


(d) ReLU


(e) Swish

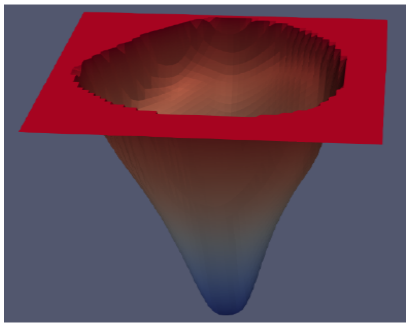

(f) LiSHT

Fig. 14: The vizualization of 3D Loss Landscape plot of CIFAR-10 shown using ReLU, Swish and LiSHT, respectively.

are illustrated in Fig. 13 and 14 by following the visualization technique proposed by Li et al. [54].

As depicted in the $2D$ loss landscape visualizations in Fig. 13(a)-(c), the $LiSHT$ makes the network to produce the smoother loss landscapes with smaller convergence steps which is populated by the narrow, and convex regions. It directly impacts the loss landscape. However, $Swish$ and $ReLU$ also produce smooth loss landscape with large convergence steps, but unlike $LiSHT$, both $Swish$ and $ReLU$ cover wider searching area which leads to poor training behavior. In $3D$ landscape visualization, it can be seen in Fig. 14(a)-(c), it can be observed that the slope of the $LiSHT$ loss landscape is higher than the $Swish$ and $ReLU$ which enables to train deep network efficiently. Therefore, we can say that, the $LiSHT$ decreases the non-convex nature of overall loss minimization landscape as compared to the $ReLU$ and $Swish$ activation functions.

V. Conclusion

In this paper, a novel non-parametric linearly scaled hyper tangent activation function ($LiSHT$) is proposed for training the neural networks. The proposed $LiSHT$ activation function introduces more non-linearity in the network. It is completely unbounded and solves the problems of diminishing gradient problems. Other properties of $LiSHT$ are symmetry, smoothness and non-monotonicity, which play an important roles in training. The classification results are compared with the state-of-the-art activation functions using MLP, ResNet and LSTM models over benchmark datasets. The performance is tested for the data classification, image classification and sentiment classification problems. The experimental results confirm the effectiveness of the unbounded, symmetric and highly non-linear nature of the proposed $LiSHT$ activation function. The importance of unbounded and symmetric non-linearity in both positive and negative regions are analyzed in terms of the activation maps and weight distribution of the learned network. A positive correlation is observed between the network learning and proposed unbounded and symmetric activation function. The visualization of loss landscape confirms the effectiveness of the proposed activations to make the training more smoother with faster convergence.

Acknowledgments

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GeForce Titan X Pascal GPU used partially in this research.

References

[1] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural networks*, vol. 61, pp. 85–117, 2015.
[2] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, “A review on deep learning techniques applied to semantic segmentation,” *arXiv preprint arXiv:1704.06857*, 2017.
[3] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, “Deep learning for computer vision: A brief review,” *Computational intelligence and neuroscience*, vol. 2018, 2018.
[4] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.
[5] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.
[6] J. Tang, C. Deng, and G.-B. Huang, “Extreme learning machine for multilayer perceptron,” *IEEE transactions on neural networks and learning systems*, vol. 27, no. 4, pp. 809–821, 2016.
[7] L.-W. Kim, “Deepx: Deep learning accelerator for restricted boltzmann machine artificial neural networks,” *IEEE transactions on neural networks and learning systems*, vol. 29, no. 5, pp. 1441–1453, 2018.
[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in neural information processing systems*, pp. 1097–1105, 2012.
[9] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
[10] R. Moradi, R. Berangi, and B. Minaei, “Orthomaps: an efficient convolutional neural network with orthogonal feature maps for tiny image classification,” *IET Image Processing*, vol. 13, no. 12, pp. 2067–2076, 2019.
[11] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, “Recurrent neural network based language model,” *Eleventh Annual Conference of the International Speech Communication Association*, 2010.
[12] Y. Wang, M. Huang, L. Zhao *et al.*, “Attention-based lstm for aspect-level sentiment classification,” *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 606–615, 2016.
[13] X. Wang, A. Bao, Y. Cheng, and Q. Yu, “Multipath ensemble convolutional neural network,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2018.
[14] Y. Chen, H. Chang, M. Jin, and D. Zhang, “Ensemble neural networks (enn): A gradient-free stochastic method,” *Neural Networks*, 2018.
[15] S. Wen, H. Wei, Z. Zeng, and T. Huang, “Memristive fully convolutional network: An accurate hardware image-segmentor in deep learning,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 5, pp. 324–334, 2018.
[16] M. Ravanelli, P. Brakel, M. Omologo, and Y. Bengio, “Light gated recurrent units for speech recognition,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 2, pp. 92–102, 2018.

[17] J.-C. Hou, S.-S. Wang, Y.-H. Lai, Y. Tsao, H.-W. Chang, and H.-M. Wang, “Audio-visual speech enhancement using multimodal deep convolutional neural networks,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 2, pp. 117–128, 2018.

[18] S. Basha, S. Ghosh, K. K. Babu, S. R. Dubey, V. Pulabaigari, S. Mukherjee *et al.*, “Rccnet: An efficient convolutional neural network for histological routine colon cancer nuclei classification,” *arXiv preprint arXiv:1810.02797*, 2018.

[19] M. Goyal, N. D. Reeves, A. K. Davison, S. Rajbhandari, J. Spragg, and M. H. Yap, “Dfunet: Convolutional neural networks for diabetic foot ulcer classification,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2018.

[20] K. Zheng, W. Q. Yan, and P. Nand, “Video dynamics detection using deep neural networks,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 3, pp. 224–234, 2018.

[21] V. K. Repala and S. R. Dubey, “Dual cnn models for unsupervised monocular depth estimation,” *arXiv preprint arXiv:1804.06324*, 2018.

[22] C. Nagpal and S. R. Dubey, “A performance evaluation of convolutional neural networks for face anti spoofing,” *arXiv preprint arXiv:1805.04176*, 2018.

[23] B. Li and D. Pi, “Learning deep neural networks for node classification,” *Expert Systems with Applications*, 2019.

[24] S. K. Roy, G. Krishna, S. R. Dubey, and B. B. Chaudhuri, “HybridSN: Exploring 3-D-2-D CNN feature hierarchy for hyperspectral image classification,” *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 2, pp. 277–281, 2020.

[25] S. K. Roy, S. Chatterjee, S. Bhattacharyya, B. B. Chaudhuri, and J. Platoš, “Lightweight spectral–spatial squeeze-and- excitation residual bag-of-features learning for hyperspectral classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 8, pp. 5277–5290, 2020.

[26] S. K. Roy, S. R. Dubey, S. Chatterjee, and B. B. Chaudhuri, “Fusenet: fused squeeze-and-excitation network for spectral-spatial hyperspectral image classification,” *IET Image Processing*, vol. 14, no. 8, pp. 1653–1661, 2020.

[27] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, “A deep learning approach to network intrusion detection,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[28] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, “Progressive neural architecture search,” *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 19–34, 2018.

[29] S. Basha, S. K. Vinakota, V. Pulabaigari, S. Mukherjee, and S. R. Dubey, “Autotune: Automatically tuning convolutional neural networks for improved transfer learning,” *arXiv preprint arXiv:2005.02165*, 2020.

[30] F. Agostinelli, M. Hoffman, P. Sadowski, and P. Baldi, “Learning activation functions to improve deep neural networks,” *arXiv preprint arXiv:1412.6830*, 2014.

[31] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier nonlinearities improve neural network acoustic models,” *Proc. icml*, vol. 30, no. 1, p. 3, 2013.

[32] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

[33] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

[34] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, “Fast and accurate deep network learning by exponential linear units (elus),” *arXiv preprint arXiv:1511.07289*, 2015.

[35] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, “Self-normalizing neural networks,” *Advances in Neural Information Processing Systems*, pp. 971–980, 2017.

[36] D. Hendrycks and K. Gimpel, “Bridging nonlinearities and stochastic regularizers with gaussian error linear units,” *arXiv preprint arXiv:1606.08415*, 2016.

[37] S. R. Dubey and S. Chakraborty, “Average biased relu based cnn descriptor for improved face retrieval,” *arXiv preprint arXiv:1804.02051*, 2018.

[38] V. S. Bawa and V. Kumar, “Linearized sigmoidal activation: A novel activation function with tractable non-linear characteristics to boost representation capability,” *Expert Systems with Applications*, vol. 120, pp. 346–356, 2019.

[39] B. Xu, N. Wang, T. Chen, and M. Li, “Empirical evaluation of rectified activations in convolutional network,” *arXiv preprint arXiv:1505.00853*, 2015.

[40] P. Ramachandran, B. Zoph, and Q. V. Le, “Swish: a self-gated activation function,” *arXiv preprint arXiv:1710.05941*, 2017.

[41] S. Scardapane, S. Van Vaerenbergh, A. Hussain, and A. Uncini, “Complex-valued neural networks with non-parametric activation functions,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2018.

[42] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks,” *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 315–323, 2011.

[43] Y. Guo, L. Du, and J. Chen, “Max-margin multi-scale convolutional factor analysis model with application to image classification,” *Expert Systems with Applications*, vol. 133, pp. 21–33, 2019.

[44] S. Elfwing, E. Uchibe, and K. Doya, “Sigmoid-weighted linear units for neural network function approximation in reinforcement learning,” *Neural Networks*, 2018.

[45] L. Zhang and P. N. Suganthan, “Random forests with ensemble of feature spaces,” *Pattern Recognition*, vol. 47, no. 10, pp. 3429–3437, 2014.

[46] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[47] A. Krizhevsky, “Learning multiple layers of features from tiny images,” Citeseer, Tech. Rep., 2009.

[48] A. Go, R. Bhayani, and L. Huang, “Twitter sentiment classification using distant supervision,” *Technical report, Stanford*, 2009.

[49] K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual networks,” *European conference on computer vision*, pp. 630–645, 2016.

[50] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[51] S. R. Dubey, S. Chakraborty, S. K. Roy, S. Mukherjee, S. K. Singh, and B. B. Chaudhuri, “diffgrad: An optimization method for convolutional neural networks,” *IEEE Transactions on Neural Networks and Learning Systems*, 2019.

[52] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” *European conference on computer vision*, pp. 818–833, 2014.

[53] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, “Understanding deep learning requires rethinking generalization,” *arXiv preprint arXiv:1611.03530*, 2016.

[54] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, “Visualizing the loss landscape of neural nets,” *Advances in Neural Information Processing Systems*, pp. 6389–6399, 2018.