IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 30, 2016; accepted November 19, 2016, date of publication December 22, 2016, date of current version January 27, 2017.

Digital Object Identifier 10.1109/ACCESS.2016.2643678

Improving the Robustness of Neural Networks Using K-Support Norm Based Adversarial Training

SHEIKH WAQAS AKHTAR[1], SAAD REHMAN[1], MAHMOOD AKHTAR[1], (Senior Member, IEEE), MUAZZAM A. KHAN[1], FARHAN RIAZ[1], QAISER CHAUDRY[2], AND RUPERT YOUNG[3]

[1]National University of Sciences and Technology, Islamabad 44000, Pakistan
[2]Georgia Institute of Technology, Atlanta, GA 30332, USA
[3]University of Sussex, Brighton BN1 9RH, U.K.

Corresponding author: S. W. Akhtar (waqasakhtar81@ce.ceme.edu.pk)

ABSTRACT It is of significant importance for any classification and recognition system, which claims near or better than human performance to be immune to small perturbations in the dataset. Researchers found out that neural networks are not very robust to small perturbations and can easily be fooled to persistently misclassify by adding a particular class of noise in the test data. This, so-called adversarial noise severely deteriorates the performance of neural networks, which otherwise perform really well on unperturbed dataset. It has been recently proposed that neural networks can be made robust against adversarial noise by training them using the data corrupted with adversarial noise itself. Following this approach, in this paper, we propose a new mechanism to generate a powerful adversarial noise model based on K-support norm to train neural networks. We tested our approach on two benchmark datasets, namely the MNIST and STL-10, using muti-layer perceptron and convolutional neural networks. Experimental results demonstrate that neural networks trained with the proposed technique show significant improvement in robustness as compared to state-of-the-art techniques.

INDEX TERMS K-Support norm, robutness, generalization, convolutional neural networks, adversarial.

I. INTRODUCTION

Deep neural networks have attracted a lot of interest since their inception in 2006 by Hinton and co-workers in their seminal paper [1]. They have remarkable capability of learning rich, high level features which results in better classification and low generalization error. Over the last ten years deep neural networks have outperformed other classifiers on many benchmark datasets related to images, speech, and text based applications. However, Szegedy *et al.* [2] demonstrated that even the neural networks that have very good generalization properties and near human performance in classification tasks, are not robust to perturbation in the dataset. They utilized images corrupted with an adversarial noise based on min-max optimization algorithm which maximizes the misclassification loss yet keeps the magnitude small and constrains it to be within a norm ball. Visually, the difference between the original and corresponding perturbed image is indiscernible for any human observer. Thus humans cannot not possibly be fooled by such examples. However,the technique can easily fool deep neural networks to fail consistently.

Szegedy *et al.* [2] argued that in general small perturbations do not change class label because of an underlying smoothness of the data space in the near vicinity of input examples. This local generalization is a characteristic of many kernel based methods. For deep neural networks this smoothness assumption does not hold because of the presence of so called blind spots that are low probability (high dimensional) pockets in the manifold which are otherwise hard to find by just randomly perturbing the input data. This peculiar behavior of adversarial examples has intrigued a lot of researchers.

Naturally, questions were raised as to why blind spots exist or, indeed, whether they exist at all? Are they just randomly dispersed over the data space or follow some pattern? Why do neural networks misclassify adversarial examples? Why do adversarial examples have properties such as being cross model (changing hyper-parameters such as number of layers, regularization, initial weights etc do not change or improve misclassification) and cross training set-generalization (training on disjoint sets does not change or improve misclassification). This intrigued a lot of researchers and motivated

them to study and improve robustness of neural networks in the presence of adversarial perturbations. However this was not the emergence of an entirely new avenue of research as there have been previous research efforts in the domain of robustness of classifiers. Xu et.al [20] introduced the term ''algorithmic robustness'' to refer to a performance characteristic of classifiers, such that, if a testing sample is similar to training sample then testing error should also be close to training error. The authors have derived generalization bounds of learning algorithms based on their robustness and claimed that improvement in robustness should also benefit generalization in a positive way. Other researchers [21]–[23] have studied robustness as an optimization problem in Support Vector Machines (SVM), linear regression and logistic regression respectively. Biggio *et al.* [24] studied SVM under adversarial label flip noise, where a specified number of labels are allowed to be flipped in sign. Results showed improvement in robustness and generalization as compared to vanilla SVM.

Goodfellow *et al.* [3] have argued that the reason for neural network to misclassify is the linearity of model in high dimensional space. Neural networks try their best to keep the output of individual neurons to operate in linear region. This simplifies matters in terms of optimization of hyper-parameters but at the same time imparts the problem that neural networks make a very high confidence prediction even in unfamiliar situations. Fawzi *et al.* [4] seem to disagree. They claim that misclassification of adversarial examples is not restricted to neural networks only, rather it is an inherent problem in every classifier. Whether a particular classifier is robust to adversarial examples or not lies in the distinguishability measure which is defined as the distance between the mean of two classes for linear classifiers while for quadratic classifiers it is defined as the distance between the matrices of second order moments of two classes. The authors also prove that linear classifiers are more robust to uniform random noise as compared to adversarial noise by a factor of $\sqrt{d}$ (where d is the dimension of the input signal).

Our contribution in this paper is to improve the robustness of neural networks against different adversarial noise models that can remarkably deteriorate the performance of a neural network which otherwise perform really well on normal (unperturbed) test data. In this work, we have proposed a new adversarial noise model based on K-support norm [7]. We show that neural networks trained using data corrupted with our K-Support norm based adversarial noise become more robust against many other powerful noise models. We further study the effect of uniform random noise on robustness and show that that training a neural network with data augmented with K-support noise achieves greater robustness as compared to training with data corrupted with uniform random noise, thus establishing the significance of adversarial noise over uniform random noise. Finally we empirically demonstrate that improvement in robustness may not necessarily also improve generalization error.

The remainder of the paper is organized as follows: in section 2 we mention some of recent work that analyze the problem of robustness of neural networks in the presence of adversarial examples thus providing local stability. In section 3, we briefly discuss K-Support norm and propose a framework for generating K-Support norm based adversarial examples and training neural networks with it. In section 4, we discuss experimental results and in section 5 we give some concluding remarks to our paper.

A. NOTATIONS

We denote the labeled training dataset by $(x_i, y_i)_{i=1}^m$ where x_i is a d dimensional feature vector and y_i is its corresponding label. Suppose K to be the number of classes in a classification task. The loss function of a network with parameters θ on (x, y) is denoted by $L(\theta, x, y)$. $L_{(\theta,y)}(x)$ represents the loss function with respect to x with θ with y fixed. $\Delta_x \in R^d$ represents a small perturbation of dimension d in data sample x. $\tilde{x} = x + \Delta_x$ represents an adversarial/perturbed example. $\langle a, b\rangle = a^T b = \sum_i x_i y_i$ represents the inner product of a and b vectors. Given a norm $\|.\|$, its dual norm is represented by $\|.\|_*$ such that $\|a\|_* = max_{\|b\|\leq 1}\langle a, b\rangle$. Given a function $p(x, y)$, $\nabla_x p(x, y)$ denotes its gradient with respect to vector x.

II. RELATED WORK

Szegedy *et al.* [2] first discovered that deep neural networks learn input output mappings that are discontinuous to a large extent. Thus, it is possible to cause a network to misclassify an image by adding a perturbation in the image that is hardly perceptible and distinguishable by the human eye. Such a perturbation was obtained by solving the following box constrained optimization problem using L-BFGS.

$$min_{\Delta_x} c\|.\|_2 + L(\theta, x + \Delta_x, y^{'}) \quad subject\ to\ x + \Delta_x \in [0, 1] \tag{1}$$

where $y^{'} \neq y$. The solution finds perturbations Δ_x that are small yet force the network to misclassify the example x with label $y^{'}$. Goodfellow *et al.* [3] proposed a very fast and easy way to generate adversarial examples. Their method is fast because it does not have to solve for an auxiliary optimization problem as in [2] and involves computing gradient of loss function L, using backpropagation. It linearly approximates the loss function around an original training example with a small perturbation Δ_x.

$$L_{\theta,y}(x + \Delta_x) \approx L_{\theta,y}(x) + \langle \nabla L_{\theta,y}(x), \Delta_x\rangle \tag{2}$$

To maximize the loss function in case of perturbed examples $(x + \Delta_x)$, the right side of equation (2) is maximized with respect to Δ_x contained with in a ℓ_∞ ball of radius ϵ. The Δ_x that maximizes the right hand side equation is given by

$$\Delta_x = sign(\nabla_x L(\theta, x, y)) \tag{3}$$

Miyato *et al.* [8] proposed an adversarial training method based on local distributional smoothing. Local distributional smoothing is the negative of Kullback-Leibler divergence (KL-divergence) between the predicted distribution of labels

$p(y \mid x)$ and $p(y \mid x + \Delta_x)$. The adversarial example was computed as the perturbation that gives the maximum KL divergence such that the ℓ_2 norm of the perturbation is less than equal to ϵ. Nokland [9] proposed a modified backpropagation algorithm to improve generalization by adding an adversarial gradient to the learning objective function. This adversarial gradient was defined to be the difference between adversarial and standard backpropagation. Experiments were performed using different activation functions such as logistic, tanh and rectified linear units (RELU).

Shaham *et al.* [10] adopted a similar approach to Goodfellow *et al.* [3] to generate adversarial noise by linearly approximating the loss function around the original training data, then adding this noise to training data to get perturbed images and then doing an additional forward backward pass with this perturbed data to update the weights. Adversarial noise was constraint to be with in the ℓ_1, ℓ_2 or ℓ_∞ norm ball. They showed that adversarial noise constrained with ℓ_∞ norm proved to be better than the other two in terms of robustness and generalization against adversarial examples.

Gu and Rigazio [11] proposed to improve model smoothing and thus robustness by using deep contractive autoencoders in which the loss function was penalized by adding Frobenius norm of the Jacobian of the neural network output y with respect to input x. However, instead of computing a full Jacobian which is computationaly very expensive, they approximated it with the sum of the Frobenius norm of the Jacobian over every adjacent pair of hidden layers.

Tabacof and Valle [5] have studied the space of adversarial examples and observed that adversarial examples do not just exist sporadically in the input space, rather there are large adversarial pockets which can be found in the vicinity of input samples and are continuous in structure.

Sabour et al. [12], studied the feature representation of hidden layers of deep neural network in an effort to understand the misclassification of adversarial examples and most interestingly discovered that although the adversarial example of an image may be indiscernible to the original, the hidden features of these image learnt by neural network look entirely different.

III. PROPOSED TRAINING FRAMEWORK

A. ADVERSARIAL EXAMPLE GENERATION

Adversarial examples can be found by solving the following maximization-minimization problem [2].

$$min_\theta \tilde{L}(\theta, x, y) = min_\theta \sum_{i=1}^{m} max_{\tilde{x} \in B_i} L(\theta, \tilde{x_i}, y_i) \tag{4}$$

Equation (4) represents a method to learn network parameters θ with respect to the worst case data instead of original data, where the worst case example $\tilde{x_i}$ belongs to a certain set B. B can be a unit norm ball. This optimization problem can be solved iteratively, first by fixing parameters θ and finding the worst case data $\tilde{x_i}$,. This involves finding Δ_{x_i} for every training example x_i such that $x + \Delta x \in B$ and Δx_i is given by

$$\Delta_{x_i} = argmax_{\Delta : x_i + \Delta \in B_i} L_{\theta, y_i}(x_i + \Delta) \tag{5}$$

This maximization step is referred to as adversarial example generation. Parameter θ is then updated with respect to the worst case data $\tilde{x_i}$. Finding the exact solution of Δ_x is intractable. Therefore we approximate equation (5) with first order Taylor approximation

$$\Delta_{x_i} \approx argmax_{\Delta : x_i + \Delta \in B_i} L_{\theta, y_i}(x_i) + \langle \nabla L_{\theta, y}(x), \Delta \rangle \tag{6}$$

To maximize this equation, we need to maximize the second term on the right hand side

$$argmax_{\Delta : x_i + \Delta \in B_i} \langle \nabla L_{\theta, y}(x), \Delta \rangle \tag{7}$$

We know by the definition of a dual norm that $\|a\|_* = max_{\|b\| \leq 1} \langle a, b \rangle$ or $c\|a\|_* = max_{\|b\| \leq c} \langle a, b \rangle$ where c is the scaling factor. Thus, it is evident that equation (7) is equal to the dual norm of $\nabla L_{\theta, y}(x)$, i.e

$$argmax_{\Delta : x_i + \Delta \in B_i} \langle \nabla L_{\theta, y}(x), \Delta \rangle = c \|\nabla L_{\theta, y}(x)\|_* \tag{8}$$

Remark 1: The dual of ℓ_2 norm is ℓ_2 norm itself. We can use it in equation(8) to get an ℓ_2 constrained noise model given by the following equation

$$\Delta^*_{x_i} = c \frac{\nabla L_{\theta, y}(x_i)}{\|\nabla L_{\theta, y}(x_i)\|_2} \tag{9}$$

Remark 2: The dual of ℓ_∞ norm is ℓ_1 norm. We can use it in equation (8) to get an ℓ_∞ constrained noise model given by the following equation.

$$\Delta^*_{x_i} = c * sign(\nabla L_{\theta, y}(x_i)) \tag{10}$$

B. K-SUPPORT NORM BASED ADVERSARIAL EXAMPLES

We now propose to use the K-support norm in equation(8) to compute an adversarial perturbation.The K-support norm was introduced by Argyriou et al. [7] as a better alternative to ℓ_1, ℓ_2 and elastic net norm. It provides a tighter convex lower bound than elastic net [13] and has been shown to achieve better predictive performance than ℓ_1, ℓ_2 and elastic net norm in numerous classification and regression applications [28], [29], [30].This motivated us to study K-Support norm as the basis of an adversarial noise model, in order to improve robustness of neural networks. The reader is referred to [7] for a detailed discussion on the K-support norm. K-support norm is computed using the following formula [7] For every $\theta \in R^d$

$$\|\theta\|^{sp}_k = \left(\sum_{i=1}^{k-r-1} (\mid \theta \mid)^2 + \frac{1}{r+1} \left(\sum_{i=k-r}^{d} \mid \theta \mid \right)^2 \right)^{\frac{1}{2}} \tag{11}$$

where letting $\mid \theta \mid^{\downarrow}_0$ denote $+\infty$, r is the unique integer in {0,...,k-1} satisfying

$$\mid \theta \mid^{\downarrow}_{k-r-1} > \frac{1}{(r+1)} \sum_{i=k-r}^{d} \mid \theta \mid^{\downarrow}_i \geq \mid \theta \mid^{\downarrow}_{k-r} \tag{12}$$

and k represents the cardinality i.e number of non-zero entries in weight vector.

Now, let us briefly explain the K-Support norm. We know from [6] that when ℓ_1 norm is used as a regularizer in learning tasks, it induces sparsity in weight variables. ℓ_1 norm applies fixed shrinkage equal to τ to all weights, so weights smaller than the threshold τ become zero, while larger weights are reduced by a fixed amount $\theta - \tau$. Whereas ℓ_2 norm penalizes weights proportional to their size, i.e larger weights are penalized more and smaller weights are penalized less. But unlike ℓ_1 norm, weights are not reduced to zero. Both norms are used as regularizers to improve generalization error. However, ℓ_1 norm has been observed to shrink too many weight variables to zero. This might be problematic if a group of variables is highly correlated, because making some of them zero may reduce predictive accuracy. K-Support norm provides a solution for this problem. It enacts a trade-off between ℓ_1 and ℓ_2 norm in a way that it combines the weight θ proportional shrinkage of ℓ_2 norm on the $(k-r)$ largest weight variables and sparse shrinkage of ℓ_1 norm on the $(d-k+r)$ smallest variables. Thus, it imparts sparsity as well as avoids highly correlated variables to become zero. Authors in [7] show that the dual of K-Support norm is equal to the ℓ_2 norm of k largest elements of θ represented as

$$\|\theta\|_k^{sp*} = \| \mid \theta \mid^{\downarrow}_{1:k} \|_2 \tag{13}$$

Using Cauchy-Schwarz inequality, the solution of equation (8) becomes

$$\Delta_x^* = \begin{cases} c\dfrac{\nabla L_{\theta,y}(x_i)}{\| \mid \nabla L_{\theta,y}(x_i) \mid^{\downarrow}_{1:k} \|_2}, & \text{if } \nabla L_{\theta,y}(x_i) \text{ is in largest k} \\ & \text{entries of } \nabla L_{\theta,y}(x). \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

Note that noise matrix Δ^* is k-sparse. This perturbation or noise is then added to the original sample to create adversarially perturbed examples.

$$\tilde{x_i} = x_i + \Delta^*_{x_i} \tag{15}$$

C. PROPOSED TRAINING ALGORITHM

We now propose the learning algorithm based on adversarial perturbation generated using K-support norm as explained in section 3.1. Given we have approximate solution of equation (6), we can use SGD to find local solution of equation (4) iteratively until convergence is achieved.The pseudo-code for this is shown below as Algorithm 1.

IV. EXPERIMENTAL EVALUATION

In this section we experiment with our proposed training algorithm on two benchmark datasets:MNIST [14] and STL-10 [15]. We compare the performance of Algorithm 1 based on K-support norm based adversarial noise against methods which have shown state of the art results in terms of making neural networks more robust in the presence of adversarial noise.

Algorithm 1 Training Based on Adversarial Examples

Input:(x_i, y_i) for $1 \leq i \leq N$; Initialize θ
Output: θ

for iter = 1,2,..,T **do**
  for every batch b_i of training data **do**
    for every input (x_i, y_i) in batch b_i **do**
      Perform one forward-backward pass to compute $\frac{\partial \theta}{\partial x}$
      Compute perturbation Δ^* using method proposed in equation (14)
      Compute perturbed input sample $(\tilde{x_i}, y_i)$ using equation (15)
    end for
    Update parameter θ using forward-backward pass on perturbed samples $(\tilde{x_i}, y_i)$ of current batch
  end for
end for
return θ

A. DATASETS AND PREPROCESSING

The MNIST dataset [14] contains 28×28 grey scale images of handwritten digits. It contains 50,000 samples for training and 10,000 samples for testing. We normalized pixel value of samples to be in range 0 to 1. STL-10 dataset [15] contains 96×96 pixel RGB (color) images of 10 different objects classes. We cropped each image to be of size 48×48 pixels, converted them to grey-scale and normalized it to be in range of −1 to 1. In this dataset too, we have 5000 samples for training and 8000 for testing. MXNET [16] was used to train all models. Some random samples from MNIST and STL-10 datasets are shown in figure 1.

B. EXPERIMENTAL SETUP

The Experimental evaluation consisted of three main parts 1) Generation of adversarial samples. 2)Training of the network using perturbed samples. 3)Testing of the network using a normal and perturbed test set. We tested our technique against three different training methods:

1) Normal (no noise in training data)
2) Dropout (dropout noise in training data [18])
3) Goodfellow's method (adversarial noise in the training data based on Fast gradient sign method by Goodfellow et al. [3])

We compared the above mentioned methods with the K-Support algorithm (Algorithm 1) and a modified version of K-support method, in which we randomly drop a fixed percentage of inputs in one or more fully connected layers before the output layer.Similarly, for testing purpose, we used a normal as well as a perturbed test set. Perturbed test sets were generated using three methods:

1) Perturbation based on a loss function with ℓ_∞ norm constraint (equation(10))
2) Perturbation based on a loss function with ℓ_2 norm constraint (equation (9))

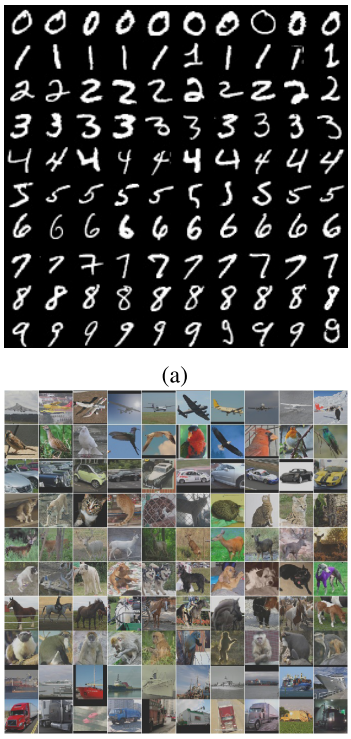


FIGURE 1. Datasets Used in Experiments. (a) Examples of the MNIST dataset. (b) Examples of the STL-10 dataset.

3) Proposed perturbation based on loss function with K-support norm (K-Sup Adv) (equation (14)

C. VISUALIZATION OF PERTURBED SAMPLE IMAGES OF MNIST AND STL-10 DATASET

In the figure 2 and 3, we have randomly taken some samples of MNIST and STL-10 datasets repectively and perturbed them using different adversarial noise models (ℓ_∞, ℓ_2 and K-sup) and uniform random noise (URN). We can see that adversarial noise perturbed images are quite similar to normal samples. A human observer cannot possibly be confused in recognizing them, yet we show in next few sections that these adversarial noised samples can easily fool neural network and significantly deteriorate their prediction performance. We can also see uniform random noise (URN) has corrupted images so much that they are not even recognizable for a human observer, even though the magnitude of perturbation in both adversarial noise and uniform random noise was kept same. However, we have shown in later section that URN perturbed training still imparts robustness but not quite as much as adversarial training.

D. EXPERIMENTS ON THE MNIST DATABASE

1) USING CONVOLUTIONAL NEURAL NETWORK

We trained a convolutional neural network [17] having two convolution and two pooling layers. A convolution layer kernel of size (5,5) was used and for pooling layer, max polling

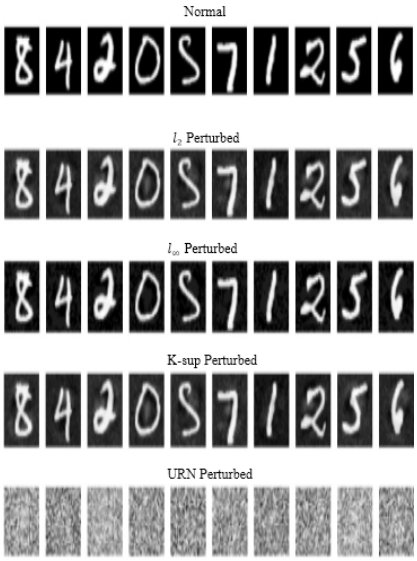


FIGURE 2. Randomly taken samples of MNIST dataset perturbed with various noise models.

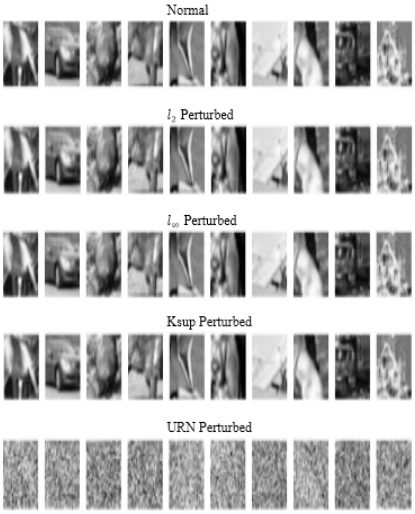


FIGURE 3. Randomly taken samples of STL-10 dataset perturbed with various noise models. Classes Shown in the images are deer, car, bird, horse, bird, dog, airplane, cat, truck, ship (from left to right).

with kernel size (2,2) was used. One fully connected layer with 400×400 units was employed. Finally we have a fully connected output layer with 10 output units. In all methods, rectified linear unit (RELU) was used as activation function. The results obtained are given in Table 1.

Our proposed method has outperformed all other methods on the normal and perturbed test sets. Compared to Goodfellow's method, training with proposed K-Support method achieved an improvement from 99.2% to 99.3% on normal test set, 98.1% to 98.6% on test set perturbed with ℓ_∞ norm based noise, 93.6% to 96.9% on test set perturbed with ℓ_2 norm based noise and 93.7% to 96.8% on test set perturbed with K-Support norm based noise.

TABLE 1. Classification accuracies for the LeNet trained on the MNIST database: the best performance on each adversarial sets are shown in bold. The magnitude of perturbation is 1.5.

	Test set accuracy (%) of normal and perturbed data			
Training Methods	Normal	ℓ_∞ norm	ℓ_2 norm	Ksup-adv
Normal	99.0	85.7	68.8	69.5
Dropout[18]	98.8	88.3	70.8	72.2
Goodfellow[3]	99.2	98.1	93.6	93.7
Ksup	**99.3**	98.3	96.0	96.0
Ksup+Dropout25%	99.1	**98.6**	96.7	**96.8**
Ksup+Dropout50%	99.1	98.5	**96.9**	**96.8**

TABLE 2. Classification accuracies for the 2-hidden-layers neural network on the MNIST database: the best performance on each adversarial sets are shown in bold. The magnitude of perturbation is 1.5.

	Test set accuracy (%) of normal and perturbed data			
Training Methods	Normal	ℓ_∞ norm	ℓ_2 norm	Ksup-adv
Normal	98.1	28.9	20.5	21.4
Dropout[18]	98.3	39.3	23.1	23.8
Goodfellow[3]	**98.9**	92.5	71.7	71.7
Ksup	98.7	93.6	85.7	85.8
Ksup+Dropout 50%	98.4	**95.6**	**90.0**	**90.0**

2) USING MULTI-LAYER PERCEPTRON

We trained a multi-layer perceptron (MLP) with two fully connected layers of 400×400 units and an output layer of 10 unit, using the different training methods mentioned in 4.2. We experimented with two variants of the proposed method. 1) K-Support noise with (K=50%). This means that in the noise matrix we selected 50% of the largest values and rest of values are zero and 2) Modified K-Support Method with (K=50%) and dropout percentage 50% in second fully-connected layer. Rectified linear unit (RELU) has been used as activation function in all methods.

We summarize the results of the tests in table 2. Training with the normal dataset results in good accuracy on normal test set but it is quite poor on perturbed datasets. Dropout improves both accuracy on the normal test set as well as on perturbed test set but there is still lot of room for improvement. Goodfellow's method provide a big leap in improving accuracy on the perturbed test set as compared to dropout. Our proposed K-Support method shows better performance than all other methods. The margin of improvement gets even bigger when we augment K-Sup method with dropout regularization. A comparison with state of the art Goodfellow's method shows an improvement from 92.5% to 95.6% on test set perturbed with ℓ_∞ norm based noise; 71.7% to 90.0% on test set perturbed with ℓ_2 norm based noise and from 71.7%to 90.0% on test set perturbed with K-Support noise. For the normal test set, only Goodfellow's method shows better results than the proposed method, giving an accuracy of 98.9% as compared to our method with 98.7% accuracy.

3) EFFECT OF UNIFORM RANDOM NOISE ON ROBUSTNESS

We now study the question whether the use of adversarial training is absolutely necessary? Do we really need an engineered noise model such as adversarial noise to improve robustness of neural network or we can achieve the same or better robustness using uniform random noise (URN)? To answer this, we generated uniform random noise of mean 0 and standard deviation 1 and trained 2-layer MLP (400×400 units) with this noise. We compared its performance on test sets perturbed with different noise models. The results of these tests are shown in table 3.

Results show that training with URN perturbed data does indeed improve robustness significantly as compared to training with normal data. However training with the proposed K-support perturbed data imparts even greater robustness and generalization than URN. Compared to URN, we achieved an improvement from 94.2% to 98.6% on normal test set, 80.6% to 93.1% on test set perturbed with ℓ_∞ norm based noise,

TABLE 3. Comparison of classification accuracies for the 2-layer MLP on the MNIST database trained with normal(unperturbed) data and data perturbed with uniform random noise and K-Sup noise.

	Test set accuracy (%) of normal and perturbed data				
Training Methods	Normal	ℓ_∞ norm	ℓ_2 norm	Ksup-adv	URN
Normal	98.1	28.9	20.5	21.0	25.6
URN	94.2	80.6	66.6	66.7	**63.1**
Ksup	**98.6**	**93.1**	**85.2**	**84.5**	35.0

66.6% to 85.2% on test set perturbed with ℓ_2 norm based noise and 66.7% to 84.5% on test set perturbed with K-Support norm based noise. However, our method achieved only 35% accuracy on test set perturbed with uniform random noise, whereas for the same test set URN trained network achieved 63.1%. This revealed a shortcoming that K-Support method is not very robust against uniform random noise.

4) EFFECT OF MAGNITUDE OF PERTURBATION ON ROBUSTNESS

We now study the effect of a tunable parameter c in equation (10, and 14) which represents the magnitude of perturbation. We used two-layer MLP with 100×100 hidden units and trained the network using normal data. For testing, we generated perturbed data by different methods outlined in section 4.2. The magnitude of perturbation is varied from 0 to 3.8. Results in figure 4 show that testing accuracy decreases with increase in magnitude of perturbation. Testing accuracy of K-Support noise perturbed data falls relatively sharply as compared to ℓ_∞ perturbed test data. This shows that ℓ_∞ noise is a weaker noise model that causes less performance degradation on increasing magnitude of noise, as compared to K-support noise.

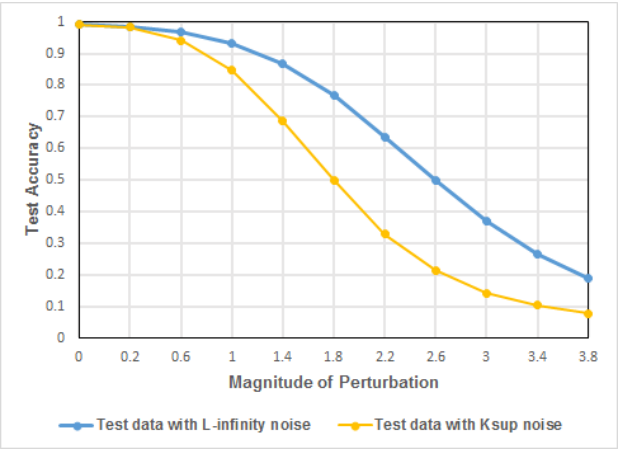


FIGURE 4. Test Accuracy of various noise models plotted against magnitude of perturbation.

5) SOME MORE RESULTS

To further establish the significance of our proposed technique, we compare the micro-averaged precision-recall curves of competing techniques with Ksup adversarial training. We used 2 hidden layered MLP with 100x100 units in each layer. RELU was used as activation function. Results are shown in figure 5 to figure 11.

Observing figures 6 to figure 11, it is evident the performance of K-support norm based training method is better than all other training methods, when test data is perturbed with ℓ_2, ℓ_∞, K-support and Uniform random noise. The large margin of improvement in area under the curve of K-support technique establishes its significance in imparting robustness against different noise models. However, as shown in figure 5, Ksup did not perform better than normal training when test data was also normal. The reason for this anomalous behavior can be described as an inherent limitation of not just K-support method but all regularization methods that involve adding noise in either data samples or weights.

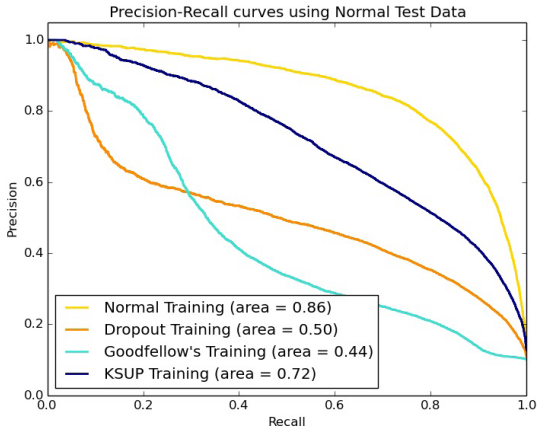


FIGURE 5. Precision recall curves using normal test data.

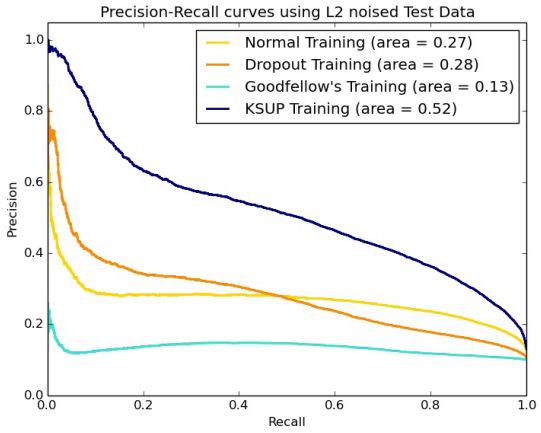


FIGURE 6. Precision recall curves using ℓ_2 noised test data.

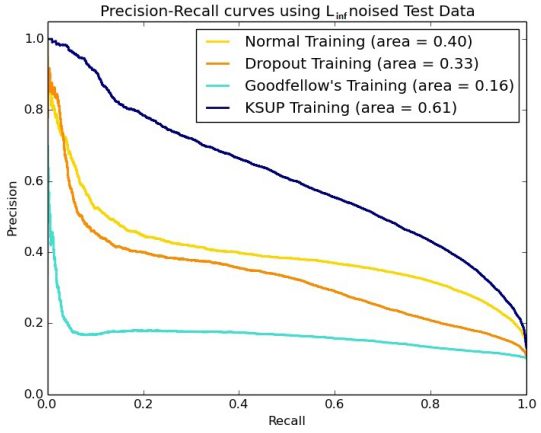


FIGURE 7. Precision recall curves using ℓ_∞ noised test data.

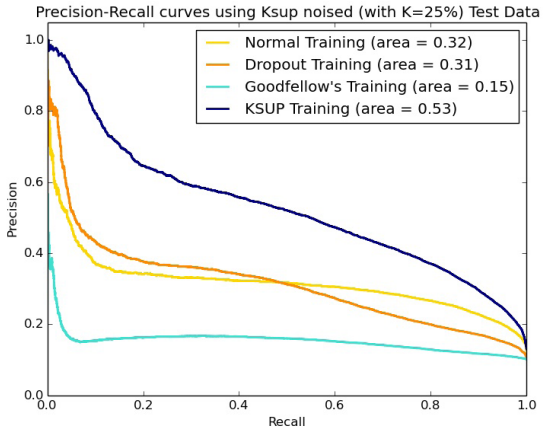


FIGURE 8. Precision-recall curved using Ksup noised data (with K=25%).

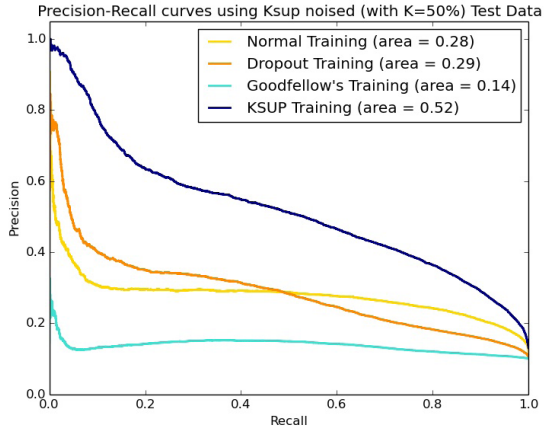


FIGURE 9. Precision-recall curved using Ksup noised data (with K=50%).

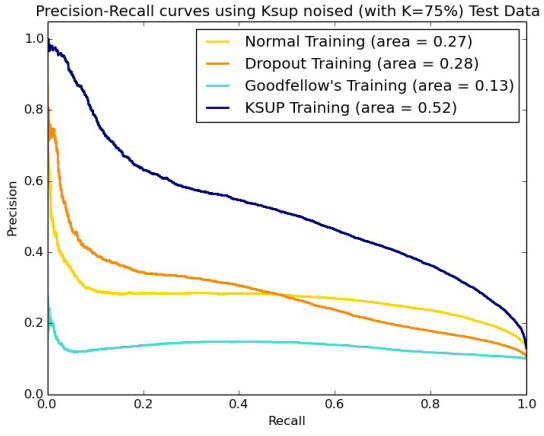


FIGURE 10. Precision-recall curved using Ksup noised data (with K=75%).

This problem can be depicted pictorially in figure 12. We represent a high dimensional image dataset in 3-dimensional space for ease of understanding. When we perturb the original data manifold ('a' in Fig.12) with some noise, images no longer remain at original position in space and move to another position, thus effectively relocating the manifold to another location ('c' in Fig.12). When we train

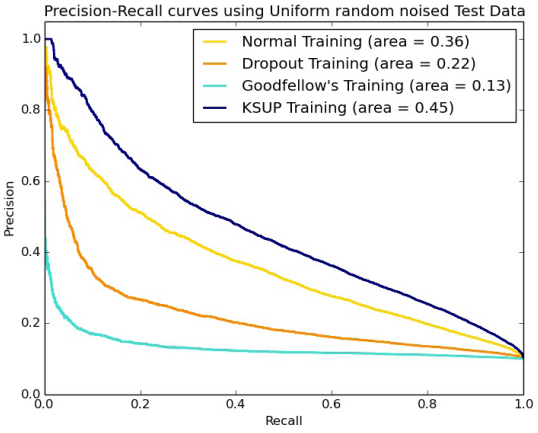
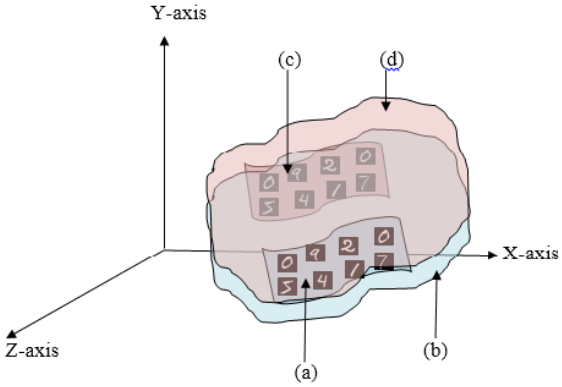


FIGURE 11. Precision-recall curved using URN noised data.



(a) Original data manifold (blue colored manifold)
(b) Vicinity around original manifold in which classifier is uncertain
(c) Data manifold perturbed with noise (pink colored manifold)
(d) Vicinity around perturbed manifold in which classifier is uncertain

FIGURE 12. Illustration of the problem of uncertain region around manifold which occurs when manifold is perturbed by adding noise.

our classifier with perturbed dataset, we make it learn/fit the perturbed manifold. As a result, we achieve our aim of making it more robust in the vicinity of original manifold ('b' in Fig.12) but the original manifold becomes an unknown region for the classifier, because it (original manifold) is now in the region of uncertainty of perturbed manifold ('d' in Fig.12). This describes why Ksup and other noise based training methods did not perform better than normal training on normal test data (Figure 5). To rectify this issue, we need a technique that would make the classifiers robust in the vicinity of original manifold without compromising its knowledge about the original manifold. It is an open problem for research community interested in improving generalization performance of classifiers.

E. EXPERIMENTS ON THE STL-10 DATABASE

For this challenging dataset, we used a convolutional neural network inspired by the VDD-D Network [19]. Our network consists of 2-stage convolutional layers followed by

TABLE 4. Classification accuracies on the STL-10: the best performance on each adversarial sets are shown in bold. The magnitude of perturbation is 1.5

	Test set accuracy (%) of normal and perturbed data			
Training Methods	Normal	ℓ_∞ norm	ℓ_2 norm	Ksup-adv
Normal	48.2	7.4	3.5	3.69
Dropout[18]	**49.1**	20.7	15.8	16.7
Goodfellow[3]	48.6	33.5	27.7	28.0
Ksup	36.9	26.1	22.4	22.7
Ksup+Dropout25%	39.4	33.2	30.2	30.3
Ksup+Dropout50%	40.5	**35.1**	**32.3**	**33.0**

3 fully connected layers. For each convolution stage, we have 3 convolutional layers with a (3, 3) kernel size followed by max polling layer with kernel size (3, 3) and stride of (2, 2). The 1st and 2nd convolution stage has 40 and 80 kernels respectively. For the three fully connected layer, number of units are 400, 400 and 10 respectively. A rectified linear unit (RELU) has been used as the activation function in all convolutional and fully-connected layers. We compare our proposed method with 1. Normal 2. Dropout 3. Goodfellow's method.The results obtained are summarized in Table 4.

On STL-10 dataset, proposed K-Support norm based training achieved better accuracy on perturbed test sets than all other methods. Compared to Goodfellow'*s* method, we got an improvement from 33.5% to 35.1% on test set perturbed with ℓ_∞ norm based noise, 27.7% to 32.3% on test set perturbed with ℓ_2 norm based noise and 28.0% to 33.0% on test set perturbed with K-Support norm based noise model. However, contrary to the performance of K-Support method on MNIST dataset where it had achieved improvement in accuracy on normal test set, it did not show improvement on STL-10 dataset and gave an accuracy of 40.5% compared to 48.2% achieved by normal training. Also, we have seen previously in Table I that Dropout training could also not improve accuracy on normal test set on MNIST dataset and reported an accuracy of 98.8% as compared to 99% achieved by normal training but on STL-10 dataset, it showed best performance with an accuracy of 49.1% on normal test set. The results of our experiments on MNIST and STL-10 dataset suggest that training neural network with a noise model may not always improve accuracy on both perturbed as well as normal test set. However this issue needs further theoretical investigation, as our results are empirical.

V. CONCLUSION

The reported work contributes in the recent efforts made by deep learning community to enhance robustness of neural networks against adversarial noise. Adversarial noise is generated using a min-max optimization algorithm that maximizes the networks misclassification loss. We proposed a mechanism to generate a powerful adversarial noise based on K-Support norm. We experimented using the MNIST and STL-10 datasets using two neural network architectures (multi-layer perceptron and convolutional neural networks). The performance of neural network trained using proposed noise model was better than several other training methods. The margin of improvement was further increased when we augmented our algorithm with dropout noise. We also empirically validated the significance of adversarial training by comparing the robustness imparted by a uniform random noise with that of our algorithm. We showed uniform random noise does imparts robustness against different kinds of perturbations but its performance is far below than network trained with K-Supp adversarial noise. Finally, we demonstrated empirically that an improvement in robustness may not improve generalization performance as well. As future work, a mathematical framework to support our empirical results would be an interesting and important advancement. Another direction could be to study and improve the robustness of K-Support norm based method against uniform random noise. Further, investigation of different structured norms such as fused lasso [25], trace lasso [26], and simultaneous lasso [27] etc. would be valuable. These could then be used in the framework of new and efficient adversarial noise models that would improve robustness and possibly also generalization of deep learning neural networks.

REFERENCES

[1] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[2] C. Szegedy *et al.*, "Intriguing properties of neural networks," in *Proc. Int. Conf. Lear. Represent.*, 2014. [Online]. Available: https://arxiv.org/abs/1312.6199

[3] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Represent.*, 2015. [Online]. Available: https://arxiv.org/abs/1412.6572

[4] A. Fawzi, O. Fawzi, and P. Frossard. (Feb. 2015). "Analysis of classifiers' robustness to adversarial perturbations." [Online]. Available: https://arxiv.org/abs/1502.02590

[5] P. Tabacof and E. Valle. (Oct. 2015). "Exploring the space of adversarial images." [Online]. Available: https://arxiv.org/abs/1510.05328

[6] T. Hastie, R. Tibshirani, and J. Friedman, "The elements of statistical learning: Data mining," in *Inference and Prediction*. Berlin, Germany: Springer-Verlag, 2001.

[7] A. Argyriou, R. Foygel, and N. Srebro, "Sparse prediction with the *k*-support norm," in *Proc. 25th Adv. Neural Inf. Process. Syst. Conf.*, 2012, pp. 1457–1465.

[8] T. Miyato, S. Maeda, M. Koyama, K. Nakae, and S. Ishii, "Distributional smoothing with virtual adversarial training," in *Proc. Int. Conf. Learn. Represent.*, 2016. [Online]. Available: http://arxiv.org/abs/1507.00677

[9] A. Nokland. (Oct. 2015). "Improving back-propagation by adding an adversarial gradient." [Online]. Available: https://arxiv.org/abs/1510.04189

[10] U. Shaham, Y. Yamada, and S. Negahban. (Nov. 2015). "Understanding adversarial training: Increasing local stability of neural nets through robust optimization." [Online]. Available: https://arxiv.org/abs/1511.05432

[11] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015. [Online]. Available: https://arxiv.org/abs/1412.5068
[12] S. Sabour, Y. Cao, F. Faghri, and D. Fleet, "Adversarial manipulation of deep representations," in *Proc. Int. Conf. Learn. Represent.*, 2016. [Online]. Available: https://arxiv.org/abs/1511.05122
[13] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *J. Roy. Statist. Soc. B (Statistical Methodology)*, vol. 67, no. 2, pp. 301–320, 2005.
[14] Y. LeCun, C. Cortes, and C. Burges. (1998). *The MNIST Database of Handwritten Digits*. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[15] A. Coates, H. Lee, and A. Y. Ng, "An analysis of single layer networks in unsupervised feature learning," in *Proc. Artif. Intell. Statist. Conf. (AISTATS)*, 2011, pp. 215–223.
[16] C. Tianqi *et al.*, " Mxnet: A distributed deep learning framework for efficiency and flexibility," in *Proc. Learn. Syst. Workshop, 28th Adv. Neural Inf. Process. Syst. Conf. (NIPS)*, 2015. [Online]. Available: http://learningsys.org/papers/LearningSys_2015_paper_1.pdf
[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.
[19] K. Simonyan and A. Zisserman. (Sep. 2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556
[20] H. Xu and S. Mannor, "Robustness and generalization," *Mach. Learn.*, vol. 86, no. 3, pp. 391–423, 2012.
[21] H. Xu, C. Caramanis, and S. Mannor, "Robustness and regularization of support vector machines," *J. Mach. Learn. Res.*, vol. 10, pp. 1485–1510, Jan. 2009
[22] H. Xu, C. Caramanis, and S. Mannor, "Robust regression and lasso," in *Proc. 21st Adv. Neural Inf. Process. Syst. Conf. (NIPS)*, 2008, pp. 1801–1808.
[23] J. Feng, H. Xu, S. Mannor, and S. Ya, "Robust logistic regression and classication," in *Proc. 27th Adv. Neural Inf. Process. Syst. Conf. (NIPS)*, 2014, pp. 253–261.
[24] B. Biggio, B. Nelson, and P. Laskov, "Support vector machines under adversarial label noise," in *Proc. Workshop Proc. Asian Conf. Mach. Learn.*, 2011, pp. 97–112.
[25] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight, "Sparsity and smoothness via the fused Lasso," *J. Roy. Statist. Soc. B*, pp. 91–108, 2005.
[26] E. Grave, G. Obozinski, and F. Bach, "Trace Lasso: A trace norm regularization for correlated designs," in *Proc. 24th Adv. Neural Inf. Process. Syst. Conf. (NIPS)*, 2011, pp. 2187–2195.
[27] B. Turlach, W. Venables, and S. Wright, "Simultaneous variable selection," *Technometrics*, vol. 47, no. 3, pp. 349–363, 2005.
[28] M. Misyrlis *et al.*, "Predicting cross-task behavioral variables from FMRI data using the k-support norm," in *Proc. 2nd Int. Workshop Sparsity Techn. Med. Imag.*, 2014. [online]. Available: http://stmi2014.ece.cornell.edu/papers/STMI-O-4.pdf
[29] H. Sidahmed, E. Prokofyeva, and M. B. Blaschko, "Discovering predictors of mental health service utilization with k-support regularized logistic regression," *Inf. Sci.*, vol. 329, pp. 1–15, Feb. 2015.
[30] E. Belilovsky *et al.*, "Predictive sparse modeling of FMRI data for improved classification, regression, and visualization using the k-support norm," *Comput. Med. Imag. Graph.*, vol. 46, pp. 40–46, Dec. 2015.

SHEIKH WAQAS AKHTAR received the B.Sc. degree in electrical engineering from the University of Engineering and Technology, Lahore, in 2010. He is currently pursuing the M.S. degree in computer engineering with the College of Electrical and Mechanical Engineering, National University of Sciences and Technology, Islamabad. His research interests include machine learning, signal processing, and optimization.

SAAD REHMAN received the Ph.D. degree from the University of Sussex, Brighton, U.K. He is currently an Associate Head of Department with the College of Electrical and Mechanical Engineering, National University of Sciences and Technology, Islamabad. His research interests include pattern recognition, and image and voice processing.

MAHMOOD AKHTAR (SM'09) received the B.Sc. degree (Hons.) in electrical engineering from the University of Engineering and Technology, Lahore, Pakistan, in 2003, the M.S. degree in computer engineering from the National University of Sciences and Technology, Pakistan, in 2005, and the Ph.D. degree in electrical engineering from the University of New South Wales (UNSW), Australia, in 2008. He was with UNSW, as a Post-Doctoral Research Associate in public health bio-surveillance: early detection of infectious disease outbreaks. He was with the University of Sydney, Australia, as a Post-Doctoral Research Fellow of Medical Imaging (2009–2012), where he investigated challenges in reconstructing positron emission tomography images of freely moving small animals. He is currently an Assistant Professor of Computer Engineering with the National University of Sciences and Technology, Pakistan (2012 to date).

Dr. Akhtar has authored or co-authored around 35 publications and has served as a Reviewer for several IEEE, IET, and other journals and numerous conferences. His research interests include biomedical imaging and signal processing, computer graphics and vision, and bioinformatics and computational biology.

MUAZZAM A. KHAN is currently an Assistant Professor with the College of Electrical and Mechanical Engineering, National University of Sciences and Technology, Islamabad. His research interests include wireless sensor networks, routing, security, localization, and quality of service.

FARHAN RIAZ is currently an Assistant Professor with the College of Electrical and Mechanical Engineering, National University of Sciences and Technology, Islamabad. His research interests include biomedical signal, image processing, and machine learning.

QAISER CHAUDRY received the Ph.D. degree from the Georgia Institute of Technology, USA. His research interests include biomedical image analysis, bioinformatics, and image processing.

RUPERT YOUNG is currently a Reader of Engineering with the University of Sussex, Brighton, U.K. His research interests include computer vision and image processing, pattern recognition, genetic algorithms, holography instrumentation, medical informatics, and photonics.