Convolutional neural networks with rectified linear unit (ReLU) have been successful in speech recognition and computer vision tasks. ReLU was proposed as a better match to biological neural activation functions compared to sigmoidal non-linearity function. However, ReLU has a disadvantage that the gradient is zero whenever the unit is not active or saturated. To alleviate the potential problems due the zero gradient, Leaky ReLU (LReLU) was proposed. Recently, a parametrized version of ReLU (PReLU) was shown to give superior performance compared to ReLU on large scale computer vision tasks. PReLU is a generalized version of LReLU where the gradient is learned adaptively from the training data. In this paper we investigate PReLU based deep convolutional neural networks for noise robust speech recognition. We report experimental results on Aurora-4 multi-condition training task. We show that PReLU gives slightly better Word Error Rates (WERs) on noisy test sets compared to ReLU. In combination with dropout generalization method we report one of the best WERs in the literature for this noisy speech recognition task.

Index Terms: Parametric Rectified Linear Units, Convolutional Neural Networks, Noise Robustness

1. Introduction

Neural networks are a family of powerful algorithms that can capture complex classification surfaces in the feature space without any strong assumptions about the data structure. They have been used for continuous speech recognition for more than two decades [1]. Recent advances in computing architecture and deep learning methods [2] brought significant success to in acoustic modelling for large vocabulary continuous speech recognition tasks using Deep Neural Networks (DNNs) [3]. Convolutional Neural Networks (CNNs) are one of the oldest and the most successful deep neural network architectures [4] for image recognition and computer vision tasks [5]. Recently they have been applied to speech recognition tasks also [6, 7, 8], showing better accuracy compared to DNNs. CNNs can model spatial and temporal correlations in spectral envelopes while reducing the translational variance caused by different speaking styles [9].

Modelling natural neural activation functions using non-linearity in hidden units is an active research topic. Rectified Linear Units (ReLU) have been shown to overcome the "vanishing gradient" problem of sigmoidal units [10, 11]. Rectified Linear (ReLU) function is mathematically given by,

\[
f(y_i) = \begin{cases} 
  y_i & \text{if } y_i > 0 \\
  0 & \text{if } y_i \leq 0
\end{cases}
\]

Here, \( y_i \) is the input of the nonlinear activation \( f() \) on the \( i \)th channel. Figure 1 shows the shape of ReLU activation function. The partial derivative of ReLU function is 1 when it is switched on and 0 when it is switched off. Although this will result in unbounded hidden layer outputs, vanishing gradients are avoided in even arbitrarily deep networks. A potential disadvantage of 0 gradient is that a hidden unit may never switch on if it was inactive during initialization. To mitigate this, Leaky ReLU (LReLU) was proposed [12]. LReLU has a small gradient when the unit is not active and saturated,

\[
f(y_i) = \begin{cases} 
  y_i & \text{if } y_i > 0 \\
  0.01y_i & \text{if } y_i \leq 0
\end{cases}
\]

Recently a parametric form of ReLU (PReLU) was shown to give better performance than ReLU on large scale computer vision tasks [13]. In this paper we explore the effectiveness of this activation function for noise robust speech recognition. The remainder of this paper is organized as follows. In Section 1 we explain PReLU in detail. Section 3 will focus on acoustic modelling by CNNs. Section 4 will present the experimental setup and analysis the results followed by concluding remarks in Section 5.

2. Parametric Rectified Linear Units

PReLU is a generalized parametric formulation of ReLU. It can be defined as

\[
f(y_i) = \begin{cases} 
  y_i & \text{if } y_i > 0 \\
  ay_i & \text{if } y_i \leq 0
\end{cases}
\]

![Figure 1: Activation functions of ReLU and PReLU. In the case of PReLU, the coefficient \( a \) is learned from the data](#)
Here $a_i$ controls the slope of the negative part. If $0 \leq a_i < 1$, then the negative part is compressed; if $1 \leq a_i < \infty$, the negative part is amplified; if $a_i < 0$, the negative part is fully rectified and scaled. The subscript $i$ in $a_i$ means the every hidden node nonlinear activation has a different slope. When $a_i = 0$, it becomes ReLU; when $a_i$ is a learnable parameter it is referred as Parametric ReLU (PReLU). Figure 1 shows the shape of PReLU activation. If $a_i$ is a small and fixed value, PReLU becomes LReLU ($a_i = 0.01$). Instead of heuristically fixing the slope when the unit is inactive and saturated, it is elegant to adaptively learn the slope of the hidden units. This creates hidden units with specialized activations that are optimized for the data and model topology. The number of additional parameters introduced by PReLU is equal to the number of hidden units, which is insignificant compared to the total number of weights.

As shown in [13] PReLU can be trained using backpropagation [14]. The gradient of $a_i$ for one layer is:

$$\frac{\partial \xi}{\partial a_i} = \sum a_i \frac{\partial \xi}{\partial f(y_i)} \frac{\partial f(y_i)}{\partial a_i}$$

where $\xi$ represents the objective function. The term $\frac{\partial \xi}{\partial f(y_i)}$ is the gradient propagated from the layer above. The gradient of the activation is given by:

$$\frac{\partial f(y_i)}{\partial a_i} = \begin{cases} 0 & \text{if } y_i > 0 \\ y_i & \text{if } y_i \leq 0 \end{cases}$$

The $\sum y_i$ is performed over all position of the feature map. $a_i$ is updated using the momentum method:

$$\Delta a_i := \mu \Delta a_i + \varepsilon \frac{\partial f(y_i)}{\partial a_i}$$

Here $\mu$ is the momentum and $\varepsilon$ is the learning rate.

3. Acoustic Modelling: Convolutional Neural Network

In this section, we explain the architecture and training method of our CNN based acoustic model. The topology of our CNN is as shown in Figure 2. Since CNNs have the capability to capture spatial and temporal correlations in the feature space, we use 40 dimensional logmel-filterbank+delta+double-delta coefficients to train the CNNs. A temporal context of 11 frames is used as input into the CNN, resulting in a 3x11x40 data structure. Here the static, delta and double-delta coefficients are treated analogously to the Red, Green and Blue channels in image processing literature. The features are mean normalized per utterance. There are two convolution layers and three fully connected layers. All the hidden units in the convolutional and fully connected layers have either ReLU or PReLU activation functions. Between the two convolution layers a max-pooling layer is used to sub-sample the dimensionality of feature space. Whereas the convolution layers capture the spatial and temporal correlations, the max-pooling layer introduces translational invariance. Just as in image processing literature, we use feature maps to group and expand the feature space for the convolution operation. There are 40 feature maps, each map representing a 3 dimensional slice of the input 3x11x40 data structure in time, frequency and channel. As shown in Figure 2, the first convolution stage has 40x11x40 feature maps as input; the filter size is 9x9, followed by ReLU/PReLU. Max-pooling stage uses 3x3 tiles. This results in 40x5x20 feature maps for the next convolution stage. The second convolution stage uses 3x3 filters. Subsequent layers are fully connected. We use 3 fully connected layers consisting of 2000 hidden units with ReLU/PReLU non-linearity and the output layer has softmax function. CNNs are trained by optimizing the cross-entropy objective function. The filters and fully connected weights are initialized using “Xavier” [15] initialization. Number of output nodes in the softmax layer is determined by the number of leaves in the GMM/HMM model used for force-aligning the training data [16]. We used Kaldi toolkit [17] to train a boosted-MMI based GMM/HMM model. Entire training data is forced aligned to generate the labels for training the CNN. CNN is implemented using the Caffe deep learning toolkit [18].

4. Experimental Details

We performed experiments with two objectives in mind. Firstly, to investigate if learning the slope of the negative part adaptively for each of the hidden units provides any advantage over ReLU in a noisy speech recognition task. Secondly, to analyse the distribution of the slope coefficient across the layers and also the effect of noise on it.

4.1. Experimental Data

Experiments were conducted on multi-condition training set of Aurora-4 database [19]. Training set consists of 7137 utterances (12 hours) of clean and noisy speech waveforms recorded at 16kHz sampling rate. Two micro-phones (Mic1 and Mic2) are used in recording. There are 83 speakers and 6 different noise conditions with SNR varying randomly from 10dB to 20dB. The noise conditions are car, babble, restaurant, street, airport and train station. The recognition task is a 5K-word dictation task with 14 test sets, with 330 utterances each. Sets 01-07 were recorded with Mic1 adding to sets 02-07 different noises with random SNR from 5 to 15 dB. The same approach was used to obtain sets 08-14 but Mic2 was used instead. These 14 test
sets are grouped according the type of distortion: clean (no distortion), additive noise (Mic1), channel (clean+Mic2) and noise + channel (noise+Mic2). Training and test sets have the same type of additive noise but the SNR varies from 5dB to 15dB. The tri-gram language model with 5000 words provided with the WSJ corpus [20] was used for decoding.

4.2. ReLU vs. PReLU

First a GMM/HMM trained with bMMI criterion is used to obtain the training labels for the CNN. The GMM/HMM was trained using mel-frequency cepstral coefficients+delta+double-delta features. The model set had 2034 tied-states. The training set was forced-aligned to obtain per frame tied-state labels. Next we trained one deep CNN for each type of nonlinear activation function in the hidden layers. All the CNNs have the same architecture as shown in Figure 2 using logmel-filterbank features as described in Section 3. We used a mini-batch size of 64 and each CNN was trained for a maximum of 10 epochs with an initial learning rate of 0.01. Based on the frame error rate on a held out set, the best model was chosen. The resulting CNN acoustic models are used for decoding, where output probabilities of the network, after scaling by the prior probability of each class, are used as HMM emission probabilities. Initially we trained three CNNs, one each for ReLU, LReLU and PReLU based hidden units. We did not constrain the range of $a_i$ so that the activation function may be non-monotonic. We used $a_i = 0.25$ as the initial value. Table 1 shows the word error rate (WER) for the four distortion types. It can be seen that on an average PReLU performs slightly better then ReLU. To test if introducing a small gradient to the negative part of the activation function provides any advantage we performed experiments using LReLU ($a_i = 0.01$) as well. In fact, we observed a slight degradation in performance compared to ReLU. This shows that there is merit to allowing an adaptively learned slope coefficient for the negative part of the activation function. We have not used weight decay ($l_2$ regularization) when updating $a_i$. Weight decay tends to push $a_i$ to zero, and thus biases PReLU towards ReLU.

4.3. Dropout

Dropout is a method introduced in [21] for improving the generalization error of large neural networks. Neural networks with large number of hidden units and hidden layers have a tendency to "co-adapt". Dropout discourages co-adaptations of hidden unit feature detectors by adding a particular type of noise to the hidden unit activations during the forward pass of training. The noise zeros, or drops out, a fixed fraction of the activations of the neurons in a given layer. In a previous work [22], it has been shown that ReLU based neural networks benefited from dropout. Here we test this hypothesis for both ReLU and PReLU. We used a drop-out ratio of 0.5. Table 2 shows the WER of ReLU and PReLU based CNNs trained using dropout regularization method. Both CNNs gave slight improvement compared to the CNNs without dropout.

4.4. Analysis of Slope Coefficients

Here we analyse the learned coefficients of PReLUs for each layer in detail. We also study the effect of noise on the behaviour of the learned coefficients. For that, we trained an-
other PReLU-CNN using the clean training set of the Aurora-4 database [19]. The clean training set has the same utterances as in multi-condition training set, but without noise. The clean PReLU-CNN has identical topology to the the multi-condition PReLU-CNN.

Figure 3 shows the plot of $a_i$ for the entire CNN. The vertical line inside the plot is the initialization value of $a_i$ ($a_{i0} = 0.25$). It can be seen that for the convolutional layers, $a_i$ moves away from the initial value. What is interesting to note is that, some of the $a_i$ in convolution layer 2 ($Conv2$) are negative and some are greater than 1. The units with negative $a_i$ will fully rectify the input and the units with $a_i > 1$ will amplify the input. This gives additional freedom to efficiently encode information in the convolutional layers with limited number of hidden units. Table 3 plots the histogram of $a_i$ for each layer. Rows 2 to 6 show the histogram of $a_i$ for each layer. The median value of learned $a_i$ in layer $Conv1$ for the clean-CNN is closer to 0 than the noisy (multi-condition) CNN. This shows that for clean speech, $Conv1$ is able to compress information more and hence the detected features are more discriminative in the lowest layer. For the noisy speech, due to the intrinsic distortions, lower layers tend to be weak feature detectors. Thus the learned $a_i$ have larger amplitude. $Conv2$ seems to amplify as well as fully rectify the inputs from $Conv1$. Here the median value of $a_i$ is closer to 0 for both clean and noisy CNNs. For the fully connected layers, the distributions are similar across clean and noisy CNNs. As the learned features in deeper layers become more discriminative, the median values of $a_i$ tend to be smaller compared to the one for lower layers.

5. Conclusions and Future Work

In this paper, motivated by successes in computer vision, we have explored using parametric rectified linear units (PReLU) and dropout in deep neural nets for noise robust speech recognition for the first time. PReLU is a generalized version of LReLU where the gradient is learned adaptively from the training data. In conjunction with dropout we report one of the best WER results on the Aurora-4 multi-condition training task. While analysing the distribution of the learned slope coefficient $a_i$, we have found that $a_i$ provides additional flexibility by amplifying and sometimes fully rectifying the output from convolutional layers. The average slope of the negative part of the activation is for the lower convolutional layers is much larger than 0. With limited number of feature maps (e.g., 40), this is an efficient way of exploiting information in the low level feature space. As we go deeper into the fully connected layers of the network, the average slope decreases making the activations more similar to ReLU. Hence we have “more linear” feature detectors in the lower convolutional stages that tend to preserve the information in the input features and in the deeper layers we have non-linear feature detectors that capture complex classification boundaries in the feature space. In future we will study the effect of various input features and noise on the behaviour of learned slope coefficient in further detail on a large scale speech recognition task.

6. Acknowledgements

We would like to thank the selfless and passionate contributors to the Caffe deep learning toolkit [18] for patiently answering our queries regarding usage of toolkit.

7. References


