Natural Speech Technology

Parameterised Sigmoid and ReLU Activation Functions for DNN Acoustic Modelling

Chao Zhang and Phil Woodland

Cambridge University Engineering Department

UNIVERSITY OF CAMBRIDGE

Abstract

- Sigmoid and ReLU are most commonly used hidden activation functions with fixed function shapes and no adaptive parameters.
- Parameterised Sigmoid and ReLU with learnable parameters, and their integration with std. acoustic modelling were investigated.
- Parameterised Sigmoid and ReLU resulted in 3.4% and 2.0% relative WER reductions on a challenging Mandarin CTS task.
- Method requires an increase in the number of parameters in training by 0.06% and no extra parameters in the final model.

Parameterised Sigmoid Function

- The generalised form of Sigmoid, or the *logistic function* is

$$f_i(a_i) = \eta_i \cdot \frac{1}{1 + e^{-\gamma_i a_i + \theta_i}},$$

where $f_i(\cdot)$ and a_i are the activation function and its input associated with node i, denoted as $p-$Sigmoid$(\eta_i, \gamma_i, \theta_i)$.
- η_i, γ_i, and θ_i have different effects on $f_i(a_i)$:
  - η_i defines the boundaries of $f_i(a_i)$, which allows positive, zero, or negative contributions from each hidden unit.
  - γ_i controls the steepness of the curve.
  - θ_i applies a horizontal displacement to $f_i(\cdot)$.

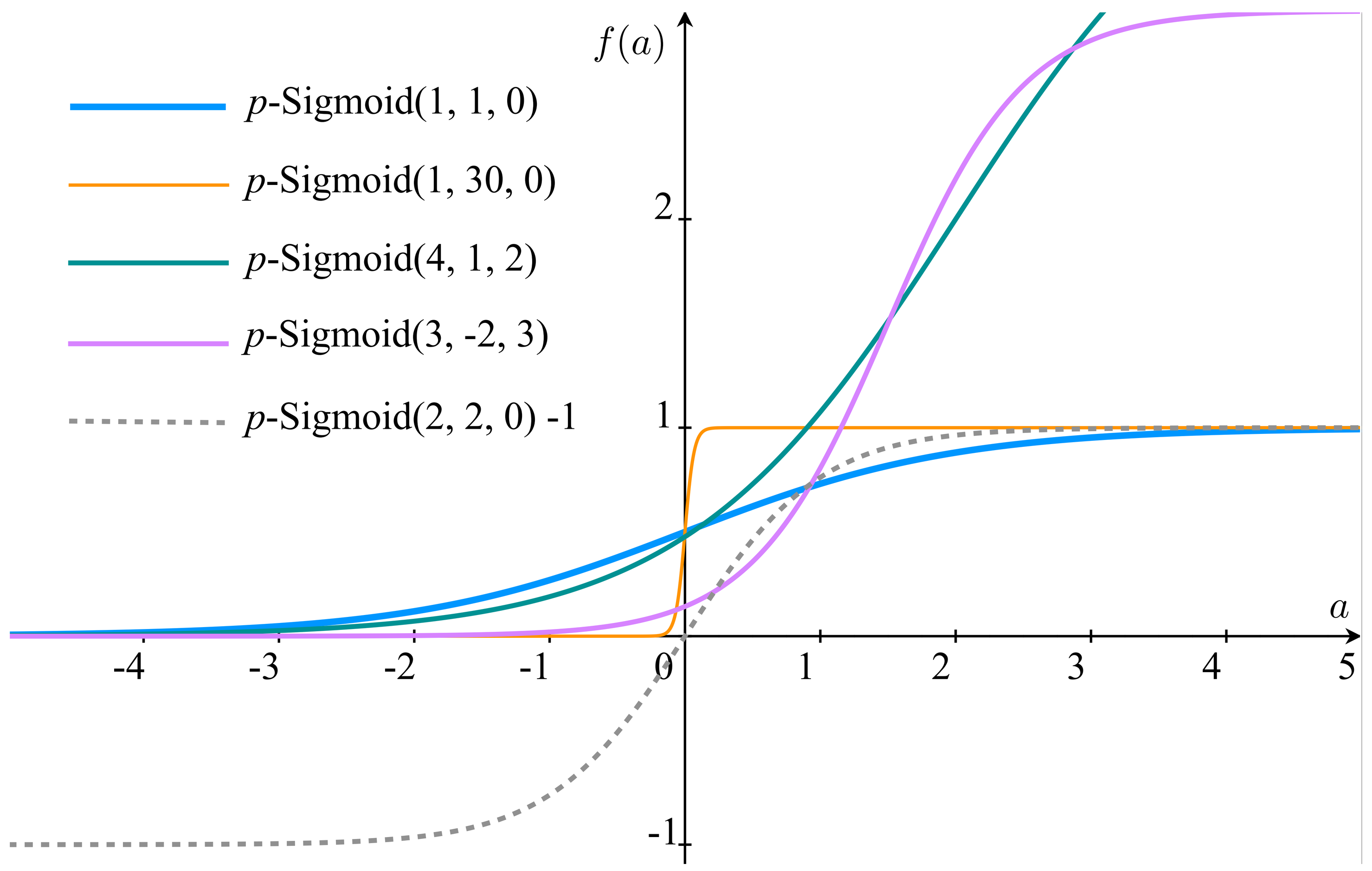

Figure 1: Piecewise approximation by $p-$Sigmoid functions.

- By varying the parameters, $p-$Sigmoid$(\eta_i, \gamma_i, \theta_i)$ can do piecewise approximation to other functions, e.g., when $a \in [-5, +5]$,
  - $p-$Sigmoid(1,30,0): step function;
  - $p-$Sigmoid(4,1,2): soft ReLU function;
  - $p-$Sigmoid(3,-2,3): std. ReLU function.
- If all parameters are learnt properly, many $p-$Sigmoid units can behave as several commonly used activation functions.

Parameterised ReLU Function

- Associate a scaling factor to either part of ReLU, to enable the 2 ends of the "hinge" to rotate separately around the "pin", i.e.,

$$f_i(a_i) = \begin{cases} \alpha_i \cdot a_i & \text{if} \;\; a_i > 0 \\ \beta_i \cdot a_i & \text{if} \;\; a_i \leqslant 0 \end{cases},$$

where α_i and β_i are any real numbers weighting the contributions of the positive and negative parts, respectively.
- The generalised ReLU function is denoted as $p-$ReLU(α_i, β_i).

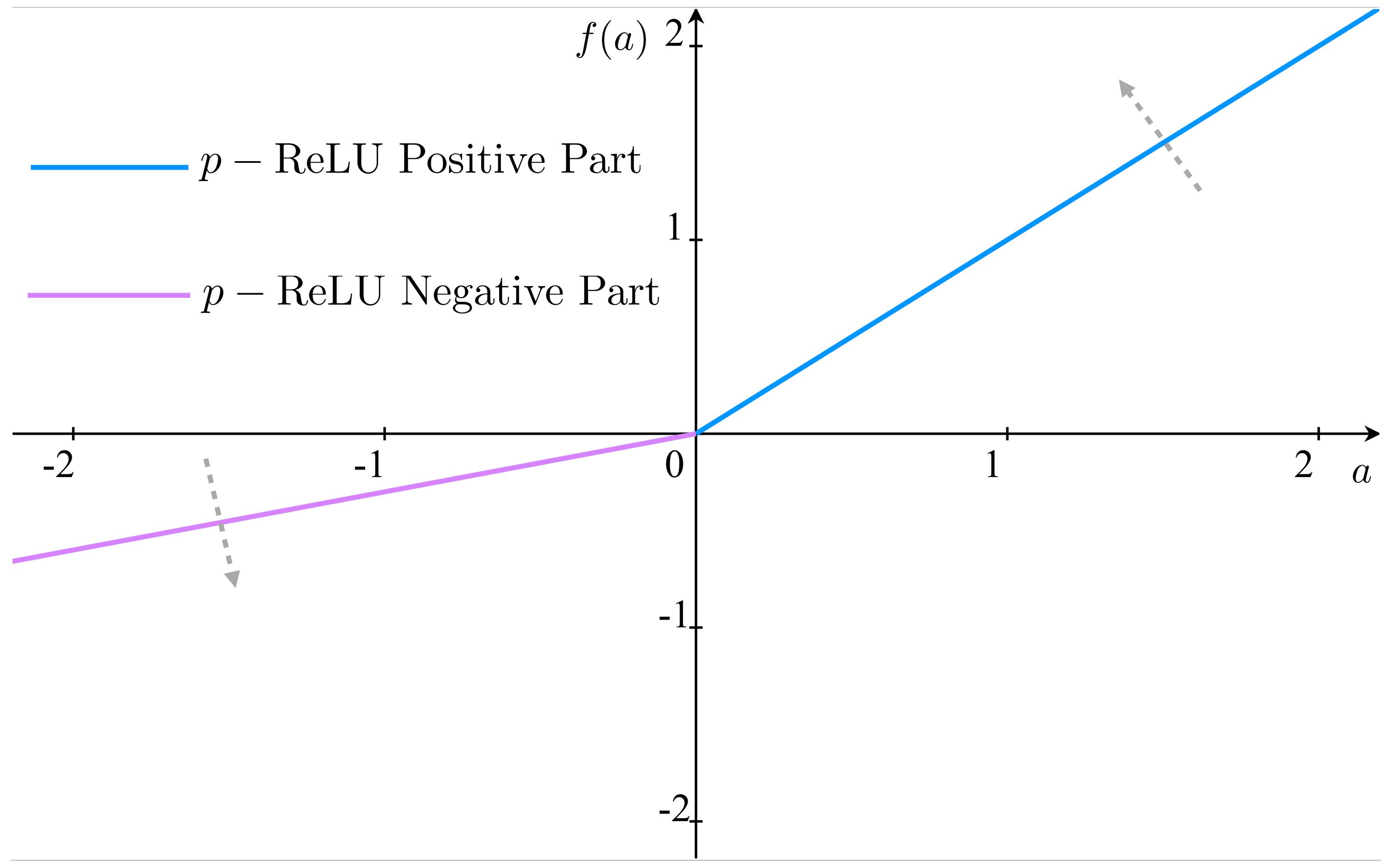

Figure 2: Illustration of the hinge-like shape of $p-$ReLU functions.

Implementations

- Involved activation functions are implemented in HTK V3.5, for both speaker independent & dependent cases.
  - $p-$Sigmoid$(\eta_i, \gamma_i, \theta_i)$ and $p-$Sigmoid$(\eta_i, 1, 0)$ are implemented as *ParmSigmoid* and *PSigmoid*.
  - $p-$ReLU(α_i, β_i) and $p-$ReLU$(\alpha_i, 0)$ are denoted as *ParmReLU* and *PReLU*.
- Implementation of $p-$Sigmoid$(\eta_i, 1, 0)$ or $p-$ReLU$(\alpha_i, 0)$ can be simplified to save only $f_i(a_i)$ but not a_i, by forcing $\partial f_i(a_i)/\partial \eta_i$ or $\partial f_i(a_i)/\partial \alpha_i$ to 0 when η_i or α_i equals 0.

Experimental Setup

- CE DNN-HMMs were trained on 72 hours Mandarin CTS data.
- Three test sets were used, *dev04*, *eval03*, and *eval97*.
  - 42d CMLLR(HLDA(PLP_0_D_A_T_Z)+Pitch_D_A) features.
- 63k word dictionary and trigram LM trained using 1 billion words.
- DNN structure $378 \times 1000^5 \times 6005$.
- η_i, γ_i, θ_i, α_i, and β_i are initialised as 1.0, 1.0, 0.0, 1.0, and 0.25.

Experiments

- Experiments with $p-$Sigmoid are given in the left part of Table 1.
  - Learning η_i, γ_i, and θ_i in PT & FT, and FT only.
  - Using multiple parameters is no better than using individual ones.
- Experiments with $p-$ReLU are listed in the right part of Table 1.
  - For ReLU DNNs, it is not necessary to freeze the parameters in the first epoch.
  - α_i has more impact on training than β_i.

ID	Activation Function	dev04	ID	Activation Function	dev04
S0	Sigmoid	27.9	R0	ReLU	27.6
$S1^+$	p-Sigmoid$(\eta_i, 1, 0)$	27.6	R1	p-ReLU$(\alpha_i, 0)$	26.8
$S2^+$	p-Sigmoid$(1, \gamma_i, 0)$	27.7	R2	p-ReLU$(1, \beta_i)$	27.0
$S3^+$	p-Sigmoid$(1, 1, \theta_i)$	27.7	R3	p-ReLU(α_i, β_i)	27.1
S1	p-Sigmoid$(\eta_i, 1, 0)$	27.1	$R1^-$	p-ReLU$(\alpha_i, 0)$	27.4
S2	p-Sigmoid$(1, \gamma_i, 0)$	27.5	$R2^-$	p-ReLU$(1, \beta_i)$	27.0
S3	p-Sigmoid$(1, 1, \theta_i)$	27.4			
S6	p-Sigmoid$(\eta_i, \gamma_i, \theta_i)$	27.3			

Table 1: *dev04 %WER for the p-Sigmoid (left) and p−ReLU (right) systems. $^+$ means the activation function parameters were trained in both PT and FT. $^-$ indicates the activation function parameters were frozen in the 1st epoch.*

- Results on all test sets are listed in Table 2.
  - S1 and R1 had 3.4% and 2.0% lower WER than S0 and R0, by increasing the number of parameters by only 0.06%.
  - p-Sigmoid gains by making Sigmoid similar to ReLU.

ID	Activation Function	eval97	eval03	dev04
S0	Sigmoid	34.1	29.7	27.9
S1	p-Sigmoid$(\eta_i, 1, 0)$	32.9	28.6	27.1
R0	ReLU	33.3	29.1	27.6
R1	p-ReLU$(\alpha_i, 0)$	32.7	28.7	26.8

Table 2: *%WER on all test sets.*

- An equivalent model with Sigmoid or ReLU can be obtained by removing activation function parameters of $p-$Sigmoid$(\eta_i, 1, 0)$ or $p-$ReLU$(\alpha_i, 0)$ and scaling the next layer weights accordingly.

Conclusion

- In this paper, we found a linear scaling factor with no constraint imposed is the most useful for parameterised Sigmoid and ReLU.
- Experiments showed DNNs trained with parameterised Sigmoid and ReLU resulted in 3.4% and 2.0% relative reductions in WER, without increasing any extra parameters in the final model.