ConvNets with Smooth Adaptive Activation Functions for Regression

Le Hou[1] **Dimitris Samaras**[1] **Tahsin M. Kurc**[1,2] **Yi Gao**[1] **Joel H. Saltz**[1,3]
[1]Stony Brook University [2]Oak Ridge National Laboratory [3]Stony Brook University Hospital

Abstract

Within Neural Networks (NN), the parameters of Adaptive Activation Functions (AAF) control the shapes of activation functions. These parameters are trained along with other parameters in the NN. AAFs have improved performance of Convolutional Neural Networks (CNN) in multiple classification tasks. In this paper, we propose and apply AAFs on CNNs for regression tasks. We argue that applying AAFs in the regression (second-to-last) layer of a NN can significantly decrease the bias of the regression NN. However, using existing AAFs may lead to overfitting. To address this problem, we propose a Smooth Adaptive Activation Function (SAAF) with a piecewise polynomial form which can approximate any continuous function to arbitrary degree of error, while having a bounded Lipschitz constant for given bounded model parameters. As a result, NNs with SAAF can avoid overfitting by simply regularizing model parameters. We empirically evaluated CNNs with SAAFs and achieved state-of-the-art results on age and pose estimation datasets.

1 Introduction

Convolutional Neural Networks (CNNs), improved the state-of-the-art on multiple classification tasks (He et al. 2015) and regression tasks (Belagiannis et al. 2015; Szegedy et al. 2013; Bulat et al. 2016). We advocate the use of Adaptive Activation Functions (AAF) in NNs applied to regression problems for two reasons. First, recent studies showed that AAFs improve the classification performance of NNs (Agostinelli et al. 2015; He et al. 2015; Jin et al. 2016). Second, the output of a regression NN should be accurate for a range of ground truth values, as opposed to a binary label. A fixed NN tends to have larger biases in regression tasks, compared to classification tasks. To address this problem, we argue (in Sec. 2) that applying AAFs on the regression (second-to-last) layer can reduce the model bias in regression problems more efficiently than adding more neurons.

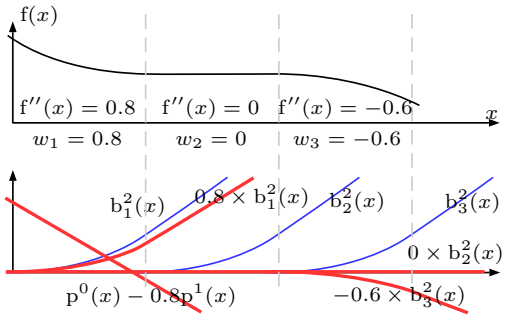

Figure 1: An illustration of the construction of the proposed SAAF with piecewise polynomial form (best viewed in color). Top: a piecewise quadratic SAAF f(x). In each quadratic segment, the second order derivative f''(x) is defined by weight w_i. Regularizing the parameters $w_{1,2,3}$ leads to a small second order derivative and Lipschitz constant, which results in bounded model complexity. Bottom: f(x) equals to the summation of the red curves. Eq. 3 gives the formal definitions of the red curves and basis functions $p^{1,2}(x)$, $b^2_{1,2,3}(x)$.

In contrast to conventional non-adaptive activation functions, AAFs have parameters that are trained along with other parameters in the NN. It is rather challenging to construct and apply AAFs. If an AAF is too simple, it may not be able to approximate the optimal activation function to a desired degree of approximation error, especially for regression problems. On the other hand, complex AAFs might lead to severe overfitting. Designing the AAF with the right approximation power and complexity for each application is contingent on experience and trial-and-error.

AAFs can be defined as combinations of sub-functions or nested sub-functions including the sigmoid, exponen-

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54.

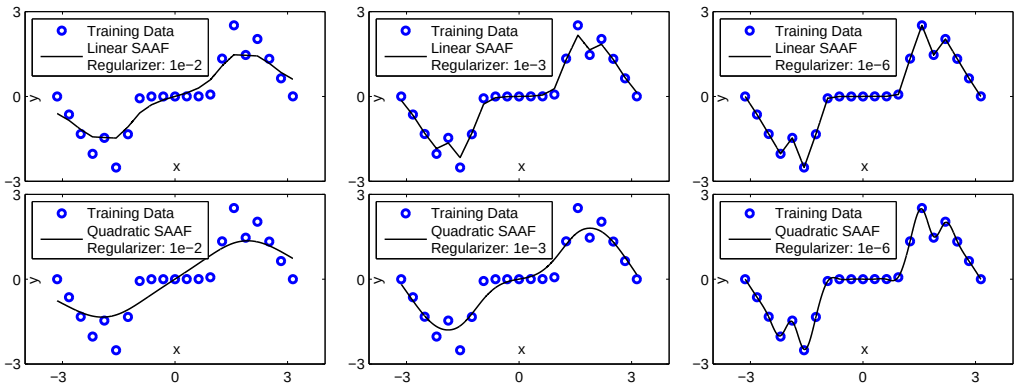

Figure 2: We demonstrate that SAAFs with piecewise polynomial form have regularized smoothness. NNs with SAAFs have bounded model complexity, as explained in Sec. 4. In each plot, there are only 21 training data pairs (input x with ground truth t) but 5000 linear or quadratic segments. Thus, most polynomial segments do not have any data to train on. However, the resulting curve is smooth and not overfitting given a reasonable L2 regularization on the parameters in SAAFs. The parameter regularization affects the magnitude of the first and second order derivatives for the piecewise linear and quadratic SAAFs respectively.

tial, sine, etc. with adjustable parameters (Bai et al. 2009; Xu et al. 2000; Ismail et al. 2013; Trottier et al. 2016). Using these AAFs led to better classification accuracy or architectures with fewer model parameters. Their major drawback is that the set of sub-functions needs to be selected carefully for different datasets, so that the optimal shape of the activation function is approximated to a desired degree of error, without introducing too many parameters. Piecewise polynomials, e.g. Splines, can handle control points implicitly (Guarnieri et al. 1999; Hong et al. 2011; Scardapane et al. 2016; Scardapane et al. 2016), but need complex training processes. Additionally, the many parameters that describe each polynomial segment, increase the probability of severe overfitting. Non-Spline piecewise linear or quadratic parameterizations (Hikawa 2003; Mathias et al. 2012) have issues of discontinuity, non-differentiability or unbounded smoothness.

Note that in most of the methods mentioned so far, one global AAF is applied on all neurons. Recent research in deep CNNs proposed to learn an AAF for each layer of neurons or even individual neurons (Goodfellow et al. 2013; He et al. 2015; Agostinelli et al. 2015; Jin et al. 2016), as an alternative for reducing model bias. Extending the non-adaptive Rectified Linear Units (ReLU) to Parameterized Rectified Linear Units (PReLU) (He et al. 2015) introduce a parameter which controls the slope of the activation function. Activation functions at different layers have the same form, but different slope. A maxout neuron (Goodfellow et al. 2013) outputs the maximum of a set of linear functions. Adaptive Piecewise Linear Units (APLU) (Agostinelli et al. 2015) learn the position of break points and the slope of linear segments simultaneously during training. The Multi-Bias Activation (MBA) (Li et al. 2016) can be viewed as a type of piecewise linear AAF. However, the total number of NN parameters would be increased N times using MBA with N biases, increasing the NN complexity even more. As each neuron learns its own maxout function, APLU, or MBA, the number of parameters in the NNs significantly increases with no clear principles of how to avoid severe overfitting.

There currently exist two types of AAFs: simple AAFs that do not guarantee a bounded approximation ability and complex AAFs that cannot avoid severe overfitting in a principled manner. Viewed in the bias-variance tradeoff paradigm (Bishop 2006), existing AAFs do not guarantee bounded bias and complexity (variance). We propose a novel AAF called Smooth Adaptive Activation Function (SAAF) with piecewise polynomial form. Compared with existing AAFs, given a fixed degree of bias, an SAAF can achieve lower complexity; given a fixed degree of complexity, an SAAF can achieve lower bias. In particular, an SAAF can be regularized under any complexity in terms of the Lipschitz constant of the function and can approximate any function simpler than the given complexity (i.e., with a smaller Lipschitz constant) to an arbitrarily small bias. To regularize an SAAF's Lipschitz constant, one can simply apply L2 regularization on its parameters (shown in Sec. 4.2). In contrast, there are no methods that regularize the complexity of existing AAFs in a principled manner. The Lipschitz constant of an AAF is a good measurement of model complexity because the fat-shattering dimension of a model is bounded by its Lipschitz constant (shown in Sec. 4.3). Figs. 1 and 2 show examples and properties of SAAFs. Our contributions are:

1. Unlike other AAFs, our Smooth Adaptive Activation Function can achieve low model bias and complexity *at the same time*:
   (a) Low model bias: SAAFs can approximate any

one-dimensional function to any degree of error, given sufficient number of polynomial segments.

(b) Low model complexity: NNs with SAAFs have bounded model complexity in terms of fat-shattering dimension, given a bounded magnitude of the parameters, regardless of the number of polynomial segments.

2. We propose to use SAAFs on the regression (second-to-last) layer of regression CNNs. Our method outperforms current state-of-the-art regression CNNs significantly on age and pose estimation datasets.

2 Regression neural networks

In this section we decompose the learning process of a regression NN to a summation of many one-dimensional function learning processes. Based on this, we argue that applying AAFs on the regression (second-to-last) layer can achieve a small model bias using a small number of parameters. Without loss of generality, assume a regression NN has only one output neuron which outputs a real value y as the prediction, expressed as: $y = \sum_{i=1}^{m} h_i o_i + b$, where $o_{1,2,\dots,m}$ are the outputs from neurons in the previous layer, $h_{1,2,\dots,m}$ are the weights of the output neuron's input synapses, and b is a bias term. We train the NN to minimize the expected regression loss on the training set $E = \arg\min_{\theta} \mathrm{E}[(t-y)^2]$, where $\theta \supseteq \{h_{1,2,\dots,m}, b\}$ is the set of trainable parameters of the NN. For a multi-layer NN, the output of a neuron o_i is computed by applying the i-th activation function f_i on the input p_i, expressed as $o_i = \mathrm{f}_i(p_i)$. Note that $\mathrm{f}_{1,2,\dots,m}$ do not necessarily have the same form. Notably, for almost all of the activation functions, there exists a function g_i such that $h_i \mathrm{f}_i(p_i) = \mathrm{f}_i\big(\mathrm{g}_i(h_i, p_i)\big)$ holds for all h_i, o_i. For example, $\mathrm{g}_i(h_i, p_i) = h_i p_i$ for ReLU, Leaky ReLU (LReLU) (Maas et al. 2013) and PReLU. Thus: $y = \sum_{i=1}^{m} \mathrm{f}_i\big(\mathrm{g}_i(h_i, p_i)\big) + b$. In other words, the final prediction of a regression NN is equivalent to the summation of activation function outputs. We denote $\mathrm{g}_i(h_i, p_i)$ as x_i, therefore:

$$y = \sum_{i=1}^{m} \mathrm{f}_i(x_i) + b. \tag{1}$$

We refer to the neurons/layer connected to the output neuron as "**regression neurons/layer**". These regression neurons have activation functions $\mathrm{f}_{1,2,\dots,m}$.

Theorem 1. *We consider the regression loss E and the set of hypotheses functions defined by Eq. 1. We assume on a training set, the loss is minimized such that $y = t$. We also assume $x_{1,2,\dots,m}$ are mutually independent on the training set. Then $\mathrm{f}_i(x_i) = \mathrm{E}[t|x_i] + B_i$ for all i, where $\mathrm{E}[t|x_i]$ is the expectation of ground truth t given x_i and B_i is a constant.*

The proof of Theorem 1 is in App. B. Theorem 1 shows that $\mathrm{f}_i(x_i)$ is a one-dimensional function that approximates t given feature x_i, ignoring constant B. Thus, it is important to be able to learn this one-dimensional function that can achieve small model bias. Although the assumption of mutually independent $x_{1,2,\dots,m}$ in Theorem 1 might not always hold, we found in practice that usually $\mathrm{f}_i(x_i)$ approximates t in real world datasets, as shown in Fig. 3. If all of the activation functions $\mathrm{f}_{1,2,\dots,m}$ have the same simple form, e.g., ReLU, then all of the features x_i must be correlated with t linearly. To generate those features x_i, more neurons and layers are needed. We propose to model $\mathrm{f}_{1,2,\dots,m}$ as AAFs. In this way, we can add a small number of parameters to achieve small model bias efficiently. In our experiments, applying AAF on the regression layer adds less than 1% to the total number of NN parameters and less than 10% to the training time.

3 Smooth adaptive activation function

We introduce the Smooth Adaptive Activation Function (SAAF) formally in this section and show its advantages in Sec. 4. Given $n+1$ real numbers $a_{1,2,\dots,n+1}$ in ascending order and a non-negative integer c, using $\mathbb{1}(\cdot)$ as the indicator function, we define the SAAF as:

$$\mathrm{f}(x) = \sum_{j=0}^{c-1} v_j \mathrm{p}^j(x) + \sum_{k=1}^{n} w_k \mathrm{b}_k^c(x), \tag{2}$$

where

$$\begin{aligned} &\mathrm{p}^j(x) = \frac{x^j}{j\,!}, \quad \mathrm{b}_k^0(x) = \mathbb{1}(a_k \le x < a_{k+1}) \\ &\mathrm{b}_k^c(x) = \underbrace{\int\int \dots \int_0^x}_{c \text{ times}} \mathrm{b}_k^0(\alpha)\, \mathrm{d}^c\alpha. \end{aligned} \tag{3}$$

The SAAF $\mathrm{f}(x)$ is piecewise polynomial. Predefined parameters c and a_k are the degree of polynomial segments and break points respectively. The parameters w_k and v_j are learned during the training stage. b_k^c and p^j are basis functions and $\mathrm{f}(x)$ is a linear combination of these basis functions. b_k^0 is the boxcar function. b_k^1 is the integral of b_k^0, which looks like the step function. b_k^2 is the integral of b_k^1, which looks like the ramp function or ReLU. The degree of the polynomial segments in $\mathrm{f}(x)$ is determined by the degree of the basis functions b_k^c and p^j. Fig. 1 visualizes the construction of $\mathrm{f}(x)$.

Based on this parameterization, we can see that the order of polynomial segments can be defined to an arbitrary number. This allows the SAAF to have a larger variety of forms compared to existing AAFs. Additionally, for each polynomial segment, there is

only one parameter controlling the c-th order derivative within the segment. Derivatives of lower order are guaranteed to be continuous across the entire SAAF, including the locations of break points. By regularizing the parameters $w_{1,2,\dots,n}$, the magnitude of the c-th order derivative is regularized. In other words, the resulting activation function is smooth. As a result, NNs with SAAFs are smooth functions. Fig. 2 gives examples of one-dimensional function learning.

4 SAAF properties

An SAAF can approximate any one-dimensional function to any desired degree of accuracy given a sufficient number of segments. NNs with SAAFs have bounded fat-shattering dimension when the NN parameters are regularized. We now discuss the properties and advantages of SAAFs in detail.

4.1 SAAFs as universal approximators

Piecewise polynomials can approximate any one-dimensional continuous function to any degree of error given sufficiently small polynomial segments (Larson et al. 2013). Because the range of a neuron's input is bounded in practice, an SAAF with a sufficient number of polynomial segments can approximate any function to any degree of error. Moreover, an SAAF with a finite Lipschitz constant can approximate any function that has a smaller Lipschitz constant.

4.2 A NN with SAAF is Lipschitz continuous

We show that because the smoothness of an SAAF can be regularized, a feedforward NN with SAAFs is Lipschitz continuous, given a bounded magnitude of the parameters in the NN. A function f is Lipschitz continuous if there exists a real constant L such that for any α_1 and α_2 in the domain of f, $|\mathrm{f}(\alpha_1) - \mathrm{f}(\alpha_2)| \leq L||\alpha_1 - \alpha_2||$. We use the Euclidean norm in this paper. The constant L is the Lipschitz constant of f.

Assuming a bounded range of input x, it is clear that the maximum derivative magnitude of $\mathrm{f}(x)$ is its Lipschitz constant. Therefore, L can be derived by integrating the parameters $w_{1,2,\dots,n}$ and $v_{1,2,\dots,c-1}$.

$$L = \max_x \left| \underbrace{\iint \dots \int_0^x}_{c-1 \text{ times}} \mathrm{w}(\alpha)\, \mathrm{d}^{c-1}\alpha + \sum_{j=1}^{c-1} \underbrace{\iint \dots \int_0^x}_{j-1 \text{ times}} v_j\, \mathrm{d}^{j-1}\alpha \right|, \tag{4}$$

where $\mathrm{w}(\alpha) = \sum_{k=1}^{n} w_k \mathrm{b}_k^c(\alpha)$. For example, given $c = 1$, $L = \max_k |w_k|$. Given $c = 2$, $L = \max_x |v_1 + \int_0^x \mathrm{w}(\alpha)|$. It has been shown (Anthony et al. 2009) that if the activation functions in an NN are Lipschitz continuous, then the NN is Lipschitz continuous.

Note that NNs with other activation functions such as the Sigmoid, ReLU and PReLU are also Lipschitz continuous given a bounded magnitude of their parameters. Therefore, NNs with combinations of such activation functions and SAAFs are also Lipschitz continuous. However, NNs with these activation functions tend to have large model bias, as argued in Sec 2.

4.3 Bounded model complexity

In this section, we prove an upper bound of the model complexity in terms of fat-shattering dimension (Bartlett et al. 1994) for any Lipschitz continous model (function) such as NNs with SAAFs. Given two models with the same training error, the model with a lower fat-shattering dimension has a better expected generalization error (Anthony et al. 2009). Upper bounds of the fat-shattering dimension have been proven (Bartlett 1998; Anthony et al. 2009) for Lipschitz continuous NNs under assumptions such as the magnitude of NN parameters. We prove an exact upper bound for any Lipschitz continuous regression model with no other model assumptions. The definition of fat-shattering dimension is in App. A.

Theorem 2. *Suppose all data points lay in a d-dimensional cube of unit volume. All functions in function set F have a Lipschitz constant L. Then, the set F has bounded fat-shattering dimension:*

$$\mathrm{fat}_F(\gamma) \leq d + \frac{L^d d!}{\gamma^d \sqrt{2^d(d+1)}}. \tag{5}$$

The proof of theorem 2 is in App. B. From theorem 2, when L decreases, the fat-shattering dimension decreases polynomially. For an NN with SAAFs, since L is bounded by the magnitude of the NN parameters, regularizing the parameters will reduce model complexity polynomially.

5 Experiments

We tested CNNs (implemented in Theano (Bastien et al. 2012)) with our proposed method and other state-of-the-art activation functions on eight real world datasets. The activation functions we tested are:

- Rectified Linear Units (ReLU) (Maas et al. 2013), Sigmoid (Sig), Tanh, and Leaky Rectified Linear Units (LReLU) (Maas et al. 2013). These are widely used non-adaptive activation functions. LReLU has a fixed negative slope α when the input is negative. We set $\alpha = -1/3$ in all experiments.
- Parametric Rectified Linear Units (PReLU) (He et al. 2015). Compared to LReLU, PReLU is an AAF with a learnable α. In CNNs, PReLUs that share the same filter weights also share the same α.

Table 1: Range of error reduction rates using our SAAF compared to other activation functions on all datasets.

Sig	Tanh	ReLU	LReLU
-1 - 13%	5 - 31%	4 - 22%	4 - 21%
PReLU	APLU	MBA	
5 - 15%	7 - 22%	-1 - 15%	

- Adaptive Piecewise Linear Units (APLU) (Agostinelli et al. 2015). APLU is a piecewise linear parameterization that is different from SAAF. In all experiments, we use 5 linear segments as suggested by (Agostinelli et al. 2015).
- Multi-Bias Activation (MBA) (Li et al. 2016). Multiple ReLUs at different biases are applied to each neuron. Thus each neuron has multiple outputs. We use 4 biases as suggested by (Li et al. 2016).
- Piecewise linear and quadratic SAAF (SAAFc1 and SAAFc2), our proposed method. In CNNs, neurons that share the same filter weights also share the same SAAF parameters. We randomly chose 22 as the number of segments, based on our proof that the model complexity can be bounded regardless of the number of polynomial segments. Break points are distributed from -1.1 to 1.1.

According to Sec 2, AAFs on the regression neurons are especially important. Therefore we also tested the variant of applying AAFs only to regression neurons, instead of all neurons. In this case, neurons other than the regression neurons used ReLU or LReLU. We distinguish AAFs only on regression neurons with prefix **R-** such as R-APLU, R-SAAFc1 etc.

We applied the following data augmentation methods. First, we rotated the images by -20 to 20 degrees. Second, input images were cropped from the original images. The height and width of the cropped image were 87.5% of the original image. Third, the colors of the cropped image were slightly perturbed. Fourth, the aspect ratios were adjusted by $+/-0.15$. Finally, images were horizontally flipped. During testing, we only used the center crop and its flip without further augmentation. Then the predictions were averaged. In all experiments, we added 10^{-5} times the L2 norm of all CNN parameters as a regularization term in the loss function. We also used batch normalization (Ioffe et al. 2015) before all activation functions to alleviate the vanishing/exploding gradient problem.

On eight datasets, NNs with proposed SAAFs reduced the error of NNs with ReLU, LReLU, PReLU, APLU, MBA by 11% in average, shown in Tab. 1. Notably, on several age and pose estimation datasets, we followed the training and testing set split scheme in the literature and achieved state-of-the-art results.

Table 2: Results on the Images of Groups dataset (Gallagher et al. 2009) using a 22-layer wide ResNet (Zagoruyko et al. 2016). Our SAAF outperformed other activation functions measured by accuracy of exact match (AEM%) and with-in-one-category-off match (AEO%). (*used facial landmark detection)

Methods	AEM	AEO	Methods	AEM	AEO
			single-CNN* (Dong et al. 2016)	54	90
R-Sig	51.0	91.7	Sig	**54.2**	**93.0**
R-Tanh	50.8	90.9	Tanh	51.3	92.1
R-ReLU	50.7	92.2	ReLU	50.7	92.2
R-LReLU	48.6	92.0	LReLU	49.3	91.9
R-PReLU	46.7	90.0	PReLU	47.7	90.9
R-APLU	49.9	91.4	APLU	50.3	92.0
R-MBA	48.0	90.5	MBA	48.9	91.0
R-SAAFc1	53.3	92.7	SAAFc1	52.3	92.7
R-SAAFc2	53.8	92.3	SAAFc2	52.1	92.2

Table 3: Results on the Images of Groups dataset (Gallagher et al. 2009) using a pretrained VGG 16-layer network (Zagoruyko et al. 2016). Our proposed single-CNN method achieved state-of-the-art results. (*used facial landmark detection)

Method	AEM	AEO	Method	AEM	AEO
			multi-CNN* (Dong et al. 2016)	56	92
R-Sig	56.3	96.1	R-PReLU	59.1	96.0
R-Tanh	47.8	90.4	R-APLU	55.2	95.3
R-ReLU	58.9	96.6	R-MBA	60.9	96.3
R-LReLU	59.2	96.4	R-SAAFc1	61.5	96.3
			R-SAAFc2	**62.0**	**96.7**

5.1 Age estimation

The problem of automatic human age estimation has been well studied in applications such as identity verification (Ramanathan et al. 2006) and social network analysis (Schwartz et al. 2013). We tested four age estimation datasets: the Images of Groups dataset (Gallagher et al. 2009), the ICCV 2015 ChaLearn-AgeGuess challenge dataset (Escalera et al. 2015), the FG-NET dataset (Panis et al. 2014), and the Adience dataset (Eidinger et al. 2014). We predicted age from face images as a regression problem and outperformed existing state-of-the-art methods significantly.

The Images of Groups dataset contains 3.5K training face images and 1K testing face images. We tested two CNN architectures: a 22-layer wide residual network (Zagoruyko et al. 2016) trained from a random initialization, and a VGG 16-layer network pretrained on ImageNet. Results of using a wide residual network are shown in Tab. 2. Results of using a pretrained VGG

Table 4: Single-CNN results on the ICCV 2015 Chalearn-AgeGuess (AgeGuess) challenge validation dataset. We used the standard error metric used in the challenge (Escalera et al. 2015) and report performance on the validation set. We achieved the best single-CNN result. Better results in the challenge used at least 4 CNNs and external datasets that are at least 100X larger than the AgeGuess dataset, for example, the challenge winner Deep Expectation (DEX) (Rothe et al. 2016). We compare with a single DEX CNN on the FG-NET and Adience benchmarks in Tab. 5

Method	Error	Method	Error
Lab219 (Escalera et al. 2015)			47.7
R-Sig	39.8	Sig	41.8
R-Tanh	43.2	Tanh	43.4
R-ReLU	41.7	ReLU	40.4
R-LReLU	42.3	LReLU	42.3
R-PReLU	41.9	PReLU	41.5
R-APLU	42.5	APLU	43.2
R-MBA	40.7	MBA	**38.3**
R-SAAFc1	40.4	SAAFc1	40.0
R-SAAFc2	**38.7**	SAAFc2	**38.7**

network are shown in Tab. 3. When using a pretrained network, we only applied AAFs on the regression neurons that replaced the original classification layer in the pretrained network. Our method outperforms the existing state-of-the-art method significantly.

The Chalearn-AgeGuess (AgeGuess) dataset was used in the ICCV15 age estimation challenge (Escalera et al. 2015). It contains 2.4K training and 1K validation face images. On this dataset, we defined our own 12 layer network which is a smaller version of the VGG architecture and pretrained it on the Adience dataset. The results are shown in Tab. 4. We achieved the best single CNN results on the AgeGuess dataset. Better results in the challenge used extensive prediction fusions (at least 4 CNNs) and very large external datasets (at least 100X larger than the AgeGuess dataset).

The FG-NET is the most popular age estimation dataset that has been studied by over 350 publications (Panis et al. 2014). It contains 1002 images of 82 subjects. We compare our method with the ICCV15 age estimation challenge winner Deep Expectation (DEX) (Rothe et al. 2016). Following the DEX method, we also fine-tuned a VGG 16-layer network. We report the Mean Absolute Error (MAE) obtained from leave one subject out cross-validation, which is the standard evaluation method used in the literature. The results are shown in Tab. 5. We achieved state-of-the-art results on this dataset.

The Adience benchmark contains 26K images in 8 age groups, along with a cross-validation data separation scheme. Following the state-of-the-art DEX method (Rothe et al. 2016), we also used the pretrained VGG 16-layer network. Results using the VGG network are shown in Tab. 5. We achieved state-of-the-art results on this dataset.

Table 5: Results on the FG-NET (Panis et al. 2014) and Adience (Eidinger et al. 2014) age estimation benchmarks using pretrained VGG 16-layer network. Our method outperformed the ICCV15 age estimation challenge winning method Deep Expectation (DEX) significantly, when using no external face datasets (only ImageNet pretrained VGG net). Using the IMDB-WIKI dataset (Rothe et al. 2016) to further pretrain the VGG net, we achieved state-of-the-art results on both datasets. (*Different training and testing set split)

	FG-NET	Adience	
Method	MAE	AEM	AEO
DLA (Wang et al. 2015)	4.26	-	-
ReLU (Levi et al. 2015)	-	50.7	84.7
Cascade CNN (Chen et al. 2016)	3.49*	52.9	88.5
DEX (Rothe et al. 2016)	4.63	55.6	89.7
DEX +IMDB-WIKI	3.09	64.0	96.6
R-Sig	4.26	59.7	94.9
R-Tanh	4.84	58.8	93.8
R-ReLU	4.59	56.5	94.0
R-LReLU	4.17	59.7	94.8
R-PReLU	4.21	59.4	94.9
R-APLU	4.46	51.4	84.0
R-MBA	4.50	54.4	91.6
R-SAAFc1	4.18	61.2	94.6
R-SAAFc2	3.87	61.3	95.1
R-SAAFc2 +IMDB-WIKI	**3.01**	**67.3**	**97.4**

5.2 Pose estimation

Pose estimation is a fundamental problem in computer vision (Chen et al. 2014). We estimate human pose in a single frame. Following (Belagiannis et al. 2015), we regress a set of 14 joint locations: bottom/top of head, left/right ankle, knee, hip, wrist, elbow, and shoulder. Each joint position is expressed by the x and y coordinates. Thus, in total there are 28 real numbers associated with each human image. Also following (Belagiannis et al. 2015), we used the widely used observer-centric ground truth (Chen et al. 2014). We tested our method on the LSP (Johnson et al. 2010) and volleyball (Belagiannis et al. 2014) datasets containing 2000 and 1107 cropped human images respectively. We did not use the other two datasets in (Chen et al. 2014) as they contain only 100 to 200 images. We evaluated with the same metric: Percentage of Correctly estimated Parts (PCP) as in (Belagiannis et al. 2015).

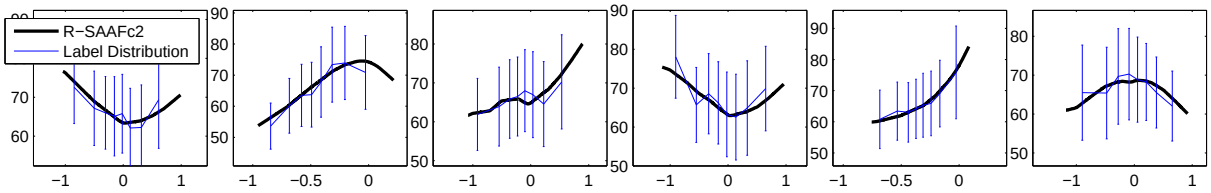

Figure 3: Examples of R-SAAFc2 learned on the LSP pose estimation dataset. Axes are scaled by a constant. We fed training instances into the CNN to compute the inputs of regression neurons (x-axis) and the outputs of the neurons' SAAFs vs. ground truth (y-axis). We see that SAAFs of various shapes correlate with the ground truth.

Table 6: Averaged PCP results of NNs with different activation functions on LSP and Volleyball pose estimation datasets. Our SAAF achieved the best results.

Methods	Datasets LSP	Volly.	Methods	Datasets LSP	Volly.
ReLU by (Belagiannis et al. 2015)				63.9	81.7
R-Sig	64.3	83.5	Sig	63.4	82.1
R-Tanh	59.1	77.7	Tanh	57.4	75.4
R-ReLU	62.5	80.7	ReLU	62.6	80.7
R-LReLU	63.1	81.4	LReLU	63.1	81.4
R-PReLU	62.1	81.8	PReLU	63.2	81.9
R-APLU	63.5	80.3	APLU	61.9	80.0
R-MBA	65.7	82.4	MBA	64.4	82.5
R-SAAFc1	68.3	**84.6**	SAAFc1	**67.2**	**83.9**
R-SAAFc2	**68.6**	84.5	SAAFc2	66.6	82.1

We used the existing regression CNN (Belagiannis et al. 2015) as baseline. We used the same CNN architecture, Tukey's biweight loss function, and just changed the CNN's activation function. In all datasets, NNs with the proposed SAAFs achieved better results than NNs with other activation functions, shown in Tab. 6. In order to examine the effect of adding additional layers with non-adaptive activation functions compared to using SAAFs, we increased the number of layers from 9 to 16 and put more neurons in each layer (324% more NN parameters and 85% more training time) to create a larger ReLU NN. On the LSP dataset, the Percentage of Correctly estimated Parts (PCP) score increased from 62.6% to 66.2%, still lower than the performance (68.6%) of the smaller R-SAAFc2 NN. Using the cascade of four CNNs (Belagiannis et al. 2015), our proposed method (R-SAAFc2 cascade) outperformed the existing regression CNN (Belagiannis et al. 2015), using no external datasets, as shown in Tab. 7. Note that recent approaches (Wei et al. 2016; Bulat et al. 2016) achieved much better results on the LSP dataset, using external datasets (ImageNet, MPII pose dataset (Andriluka et al. 2014), LSP extended) that are at least 100X larger than LSP.

5.3 Facial attractiveness prediction

We learn personal preferences of facial attractiveness on web-quality face images. We built a dataset by applying Viola-Jones' face detection (Viola et al. 2001) on female images downloaded from `hotornot.com`. The dataset contains 2K RGB face images of 120×90 pixels. Three individuals independently rated the images with facial attractiveness scores ranging from 0 to 20. We randomly selected one of the raters to provide the training and testing labels. Labels from the other two raters were only used to compute inter-observer agreement. We used the same CNN architecture in Sec. 5.1. Results are shown in Tab. 8.

5.4 Learning the circularity of nuclei

Hematoxylin and eosin stained pathology images provide rich information to diagnose, classify and study cancer. One diagnostic criterion is the shape of nuclei (Braak et al. 2003). We used CNNs to learn the circularity of nuclei in pathology images of glioma (a type of brain cancer). We used the training set from the MICCAI 2015 nucleus segmentation challenge (Farahani 2015) which contains 1K images of nuclei. We derived the ground truth circularity measurements from the ground truth nuclear segmentation masks and used CNNs to learn the circularity from nuclear images. The results are shown in Tab. 8.

6 Conclusions

We have showed that using Adaptive Activation Functions (AAF) on the regression (second-to-last) layer can improve the performance of a regression CNN. We proposed a novel Smooth Adaptive Activation Function (SAAF) which has multiple advantages. First, it can approximate any function to a desired degree of error. Second, using parameter regularization, an NN with SAAF represents a Lipschitz continuous function which leads to bounded model complexity in terms of the fat-shattering dimension. As a result, the NN can achieve lower model bias and complexity than NNs with other AAFs. We tested different setup of SAAF in various CNN architectures on eight real world datasets and achieved state-of-the-art age estimation results. In the future, we will test SAAF for classification.

Table 7: PCP results of our proposed method vs. the current state-of-the-art regression CNN on two pose estimation datasets. The cascade of 4 CNNs with R-SAAFc2 achieved the best average PCP.

Dataset	Method	Head	Torso	U-Legs	L-Legs	U-Arms	L-Arms	Avg.
LSP	(Belagiannis et al. 2015)	72.0	91.5	78.0	71.2	56.8	31.9	63.9
	(Belagiannis et al. 2015) Cascade	83.2	92.0	79.9	74.3	61.3	40.3	68.8
	R-SAAFc2	76.6	90.9	81.3	74.6	64.5	38.8	68.6
	R-SAAFc2 cascade	**83.9**	**92.8**	**82.6**	**77.8**	**65.6**	**45.8**	**72.0**
Volley-ball	(Belagiannis et al. 2015)	90.4	97.1	86.4	**95.8**	74.0	58.3	81.7
	(Belagiannis et al. 2015) Cascade	89.0	95.8	84.2	94.0	74.2	58.9	81.0
	R-SAAFc2	88.6	**99.3**	94.7	94.7	**82.5**	**60.7**	**85.3**
	R-SAAFc2 cascade	**91.5**	**99.3**	**95.2**	94.8	81.0	54.1	84.1

Table 8: RMSE and Pearson correlation results of facial attractiveness prediction and learning the circularity of nuclei using CNNs with AAFs. For the facial attractiveness dataset, the inter-observer agreement between three individuals is 0.702 Corr.

	Facial Attractiveness		Circularity of Nuclei	
Method	RMSE	Corr.	RMSE	Corr.
R-PReLU	4.189	0.693	0.508	0.631
R-APLU	4.404	0.656	0.618	0.626
R-MBA	4.171	0.701	**0.479**	**0.666**
R-SAAFc1	3.860	0.739	0.483	0.649
R-SAAFc2	3.982	0.723	0.492	0.641
PReLU	4.153	0.706	0.517	0.622
APLU	4.523	0.695	0.587	0.629
MBA	4.045	0.716	0.505	0.644
SAAFc1	3.918	0.734	0.513	0.634
SAAFc2	**3.801**	**0.745**	0.498	0.642

Acknowledgments. This work was supported in part by 1U24CA180924-01A1 from the National Cancer Institute, R01LM011119-01 and R01LM009239 from the National Library of Medicine, and the Subsample Project from the DIGITEO Institute Paris.

Appendix A Fat-shattering dimension

The fat-shattering dimension is a scalar-related dimension defined as follows: Suppose that F is a set of functions mapping from a domain X to $\mathbb{R}$, $D = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_z\}$ is a subset of the domain X, and γ is a positive real number. Then D is γ-shattered by F if there exist real numbers $t_1, t_2, \ldots, t_z$, such that for all $b \in \{0,1\}^z$, there exists a function f_b in F, such that for all $1 \leq i \leq m$,

$$\begin{aligned} \mathrm{f}_b(\mathbf{x}_i) &\geq t_i + \gamma \quad \text{if} \quad b_i = 1, \\ \mathrm{f}_b(\mathbf{x}_i) &\leq t_i - \gamma \quad \text{if} \quad b_i = 0. \end{aligned} \tag{6}$$

The fat-shattering dimension $\mathrm{fat}_F(\gamma)$ is the size of the largest subset D that is γ-shattered by F as scale γ.

Appendix B Proofs of Theorems

Proof of theorem 1. From the assumption of $y = t$, we have $\sum_i \mathrm{f}_i(x_i) + b = t$ on all training data. Taking the conditional expectation of x_1 on both sides, we have $\sum_i \mathrm{E}[\mathrm{f}_i(x_i)|x_1] + b = \mathrm{E}[t|x_1]$. Based on the assumption that $x_{1,2,\ldots,m}$ are mutually independent, $\mathrm{E}[\mathrm{f}_1(x_1)|x_1] = \mathrm{f}_1(x_1)$ and $\mathrm{E}[\mathrm{f}_i(x_i)|x_1] = \mathrm{E}[\mathrm{f}_i(x_i)]$ for every $i \geq 2$. Thus $\mathrm{f}_1(x_1) = \mathrm{E}[t|x_1] + B_i$ where $B_i = -(b + \mathrm{E}[\mathrm{f}_2(x_2)] + \mathrm{E}[\mathrm{f}_3(x_3)] + \cdots + \mathrm{E}[\mathrm{f}_m(x_m)])$. □

Proof of theorem 2. For simplicity, we consider shattering two points $\mathbf{x}_1$ and $\mathbf{x}_2$ with t_1 and t_2 only. Without loss of generality, assume $t_1 \geq t_2$. If $\mathrm{f}_b \in F$ satisfies Eq. 6 when $b_1 = 1$, $b_2 = 0$, then we have $|\mathrm{f}_b(\mathbf{x}_1) - \mathrm{f}_b(\mathbf{x}_2)| \geq |t_1 - t_2 + 2\gamma| \geq 2\gamma$. Denote $s = ||\mathbf{x}_1 - \mathbf{x}_2||$. According to the definition of Lipschitz continuity, we have $2\gamma \leq sL$. In other words the distance s between two points must be no smaller than $2\gamma/L$ in order for F to possibly γ-shatter them. Next we derive $\mathrm{fat}_F(\gamma)$ which is the maximum number of points F can γ-shatter in a cube of unit volume. Those points form a simplex mesh of at least $\mathrm{fat}_F(\gamma) - d$ number of simplexes. The nodes of the simplex mesh are the data points $\mathbf{x}_{1,2,\ldots}$ that we expect F to possibly γ-shatter. The side length of each simplex should be at least $2\gamma/L$. Therefore the total volume of the mesh is no smaller than $V_d(2\gamma/L)^d(\mathrm{fat}_F(\gamma) - d)$, where $V_d = \sqrt{d+1}\,/(d!\sqrt{2^d})$ is the volume of a d-dimensional regular simplex of unit side length. Because we assume the volume of the simplex mesh (where all data points lay) is no greater than 1, we have $\mathrm{fat}_F(\gamma) \leq d + (2\gamma/L)^{-d}/V_d$. If we expand V_d, we derive Eq. 5. □

References

Agostinelli, Forest, Matthew Hoffman, Peter Sadowski, and Pierre Baldi (2015). "Learning activation functions to improve deep neural networks". In: *ICLR workshop*.

Andriluka, Mykhaylo, Leonid Pishchulin, Peter Gehler, and Bernt Schiele (2014). "2d human pose estimation: New benchmark and state of the art analysis". In: *CVPR*.

Anthony, Martin and Peter L Bartlett (2009). *Neural network learning: Theoretical foundations*. cambridge university press.

Bai, Yanping, Haixia Zhang, and Yilong Hao (2009). "The performance of the backpropagation algorithm with varying slope of the activation function". In: *Chaos, Solitons & Fractals*.

Bartlett, Peter L. (1998). "The Sample Complexity of Pattern Classification with Neural Networks: The Size of the Weights is More Important than the Size of the Network." In: *IEEE Transactions on Information Theory*.

Bartlett, Peter L, Philip M Long, and Robert C Williamson (1994). "Fat-shattering and the learnability of real-valued functions". In: *Computational learning theory*.

Bastien, F., P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio (2012). "Theano: new features and speed improvements". In: *NIPS Workshop*.

Belagiannis, Vasileios, Christian Amann, Nassir Navab, and Slobodan Ilic (2014). "Holistic human pose estimation with regression forests". In: *Articulated Motion and Deformable Objects*.

Belagiannis, Vasileios, Christian Rupprecht, Gustavo Carneiro, and Nassir Navab (2015). "Robust Optimization for Deep Regression". In: *ICCV*.

Bishop, Christopher M (2006). *Pattern Recognition and Machine Learning*. Elsevier.

Braak, Heiko, Kelly Del Tredici, Udo Rüb, Rob AI de Vos, Ernst NH Jansen Steur, and Eva Braak (2003). "Staging of brain pathology related to sporadic Parkinsons disease". In: *Neurobiology of aging*.

Bulat, Adrian and Georgios Tzimiropoulos (2016). "Human pose estimation via convolutional part heatmap regression". In: *ECCV*.

Chen, Jun-Cheng, Amit Kumar, Rajeev Ranjan, Vishal M Patel, Azadeh Alavi, and Rama Chellappa (2016). "A Cascaded Convolutional Neural Network for Age Estimation of Unconstrained Faces". In: *Biometrics Theory, Applications and Systems*.

Chen, Xianjie and Alan L Yuille (2014). "Articulated pose estimation by a graphical model with image dependent pairwise relations". In: *NIPS*.

Dong, Yuan, Yinan Liu, and Shiguo Lian (2016). "Automatic age estimation based on deep learning algorithm". In: *Neurocomputing*.

Eidinger, Eran, Roee Enbar, and Tal Hassner (2014). "Age and gender estimation of unfiltered faces". In: *Information Forensics and Security*.

Escalera, Sergio, Jordi Gonzalez, Xavier Baro, Pablo Pardo, Junior Fabian, Marc Oliu, Hugo Jair Escalante, Ivan Huerta, and Isabelle Guyon (2015). "Apparent Age and Cultural Event Recognition Datasets and Results". In: *ICCV workshop*.

Farahani, Keyvan (2015). `http://miccai.cloudapp.net:8000/competitions/37`.

Gallagher, Andrew C and Tsuhan Chen (2009). "Understanding images of groups of people". In: *CVPR*.

Goodfellow, Ian J, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio (2013). "Maxout networks". In: *ICML*.

Guarnieri, Stefano, Stefano Guarnieri, Francesco Piazza, Francesco Piazza, Aurelio Uncini, and Aurelio Uncini (1999). "Multilayer Feedforward Networks with Adaptive Spline Activation Function". In: *Neural Networks*.

He, K., X. Zhang, S. Ren, and J. Sun (2015). "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *ICCV*.

Hikawa, Hiroomi (2003). "A digital hardware pulse-mode neuron with piecewise linear activation function." In: *Neural Networks*.

Hong, X., Y. Gong, and S. Chen (2011). "B-spline neural network based digital baseband predistorter solution using the inverse of De Boor algorithm." In: *IJCNN*.

Ioffe, Sergey and Christian Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *ICML*.

Ismail, A., D.-S. Jeng, L. L. Zhang, and J.-S. Zhang (2013). "Predictions of bridge scour: Application of a feed-forward neural network with an adaptive activation function." In: *Eng. Appl. of AI*.

Jin, Xiaojie, Chunyan Xu, Jiashi Feng, Yunchao Wei, Junjun Xiong, and Shuicheng Yan (2016). "Deep Learning with S-shaped Rectified Linear Activation Units". In: *AAAI*.

Johnson, Sam and Mark Everingham (2010). "Clustered Pose and Nonlinear Appearance Models for Human Pose Estimation." In: *BMVC*.

Larson, Mats G and Fredrik Bengzon (2013). *The finite element method: theory, implementation, and applications*. Springer Science & Business Media.

Levi, Gil and Tal Hassner (2015). "Age and gender classification using convolutional neural networks". In: *CVPR Workshops*.

Li, Hongyang, Wanli Ouyang, and Xiaogang Wang (2016). "Multi-Bias Non-linear Activation in Deep Neural Networks". In: *ICML*.

Maas, Andrew L, Awni Y Hannun, and Andrew Y Ng (2013). "Rectifier nonlinearities improve neural network acoustic models". In: *ICML*.

Mathias, Amanda C. and Paulo C. Rech (2012). "Hopfield neural network: The hyperbolic tangent and the piecewise-linear activation functions". In: *Neural Networks*.

Panis, Gabriel and Andreas Lanitis (2014). "An overview of research activities in facial age estimation using the FG-NET aging database". In: *ECCV*.

Ramanathan, Narayanan and Rama Chellappa (2006). "Face verification across age progression". In: *IEEE Transactions on Image Processing*.

Rothe, Rasmus, Radu Timofte, and Luc Van Gool (2016). "Deep expectation of real and apparent age from a single image without facial landmarks". In: *IJCV*.

Scardapane, Simone, Michele Scarpiniti, Danilo Comminiello, and Aurelio Uncini (2016). "Learning activation functions from data using cubic spline interpolation". In: *CoRR* abs/1605.05509.

Schwartz, H Andrew, Johannes C Eichstaedt, Margaret L Kern, Lukasz Dziurzynski, Stephanie M Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin EP Seligman, et al. (2013). "Personality, gender, and age in the language of social media: The open-vocabulary approach". In: *PloS one*.

Szegedy, C., A. Toshev, and D. Erhan (2013). "Deep neural networks for object detection". In: *NIPS*.

Trottier, Ludovic, Philippe Giguère, and Brahim Chaibdraa (2016). "Parametric Exponential Linear Unit for Deep Convolutional Neural Networks". In: *CoRR* abs/1605.09332.

Viola, P. and M. Jones (2001). "Rapid Object Detection using a Boosted Cascade of Simple Features". In: *CVPR*.

Wang, Xiaolong, Rui Guo, and Chandra Kambhamettu (2015). "Deeply-learned feature for age estimation". In: *IEEE Winter Conference on Applications of Computer Vision*.

Wei, Shih-En, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh (2016). "Convolutional Pose Machines". In: *CVPR*.

Xu, Shuxiang and Ming Zhang (2000). "Justification of a Neuron-Adaptive Activation Function." In: *IJCNN*.

Zagoruyko, Sergey and Nikos Komodakis (2016). "Wide Residual Networks". In: *BMVC*.