Exponential-Based Rational Activations Functions

***Abstract*—One of the persistent challenges in designing neural network models—particularly in the field of natural language processing (NLP)—is managing the large number of learnable parameters, which often leads to increased computational costs and a greater risk of overfitting. In this paper, we propose a novel approach to address this issue by leveraging Rational Activation Functions (RAFs), including both previously proposed variants and a new class of non-linear RAFs with learnable parameters. These functions are designed to enhance the representational power of neural networks while significantly reducing their architectural complexity. We demonstrate that RAFs can effectively emulate the behavior of deeper feed-forward neural networks, thereby enabling a reduction in the number of hidden layers required for a given task. This reduction translates into a lower total number of learnable parameters without compromising performance. Our method is evaluated on two fronts: first, through a series of function approximation experiments that highlight the expressiveness of RAFs; and second, on real-world NLP tasks involving text classification in noisy, spoken dialogue environments and an image classification task. The results confirm that networks incorporating RAFs maintain high accuracy while benefiting from increased efficiency and robustness. This work suggests that RAF-based architectures offer a promising direction for building lightweight, high-performance models, particularly in resource-constrained or real-time NLP applications.**

I. Introduction

Today, transformer neural network models have been massively developed for a wide variety of real-life related tasks such as image recognition [29], natural language processing [10] (NLP), voice generation [26], or medical imaging [44]. Generative models based on transformer architectures [42] became state-of-the-art on different NLP related tasks.

Therefore, transformer based models have demonstrated their powerful capabilities to generate relevant data and to express latent and complex grammatical and semantic structures during the learning process of large language models (LLM) [38]. LLMs based on transformers are now state-of-the-art for a wide range of NLP tasks, such as chatbot [12], translation [28] or sentiment analysis [40]. One of the main issues with such large neural networks architectures is that training of those models [39] is time and memory consuming. Therefore, searching for original lightweight architectures with competitive performances became an important challenge in neural networks [23].

These transformers employ both linear and non-linear transformations during the learning process to code latent words dependencies and semantic structures contained in the language by focusing on specific sub-contexts contained in the sequence of words or signals. Among the non-linear transformations required to learn these latent informations, activation functions are fundamental and have to be efficient in terms of both processing time, memory usage and the ability to separate non-linear data on the features sub-space.

A wide range of activation functions based on different mathematics fields have already been proposed such as probability, analysis or statistics [43]. As stated in [2], the interest for activation functions has increased during the recent years. More precisely, activation functions with learnable parameters have been a strong subject of interest for research in neural networks [2] to better capture hidden relations, reduce the number of learnable parameters and the memory used, as well as increasing the performances observed during the generative process. Different activation functions have already been proposed and have demonstrated their ability on different real-life tasks such as the Relu function [33]. The number of parameters required to learn linear layers associated with the Relu function is potentially huge since the word representation (word embeddings [22] for example) is large and requires large memory.

Among novel activation functions, trainable activation functions called "Rational Activation Functions" (RAFs) recently developed [7] have shown promising results on different NLP related tasks. These functions are a quotient of two polynomial functions of different degrees, where each coefficient is a learnable parameter. [7] has demonstrated that those functions are able to approximate Relu-based neural networks very well.

The aim of this paper is to introduce novel architectures with very few learnable parameters based on rational activation functions and with good performances and to propose novel activation functions based on rational activation functions that reduce both the number of parameters required for learning and reach promising results knowing the small size of the neural network. We then evaluate the effectiveness of the proposed activation functions on: functions approximation tasks, noisy spoken conversations classification and images classification, using small neural networks models to better assess the effect of specific activation functions on the model's results. The goal is **not** to reach state-of-the art accuracies on these tasks, but to point out that a rational activation function with a dedicated non-linear transformation (exponential, etc.) allows the neural based system to improve the performances observed, while having fewer parameters than ReLU based-ones.

II. Neural Netwroks

A. Multilayer Perceptron (MLP)

MLP have been first introduced in 1957 [17] and constitute the skeleton of modern neural networks architectures. In this section, we will briefly describe it's forward phase. For a MLP

made of M layers of N nodes of neurons, x being the input to a node and N_l being the number of nodes present in the layer l, with $1 < l < M$. b_n^l is the bias of the neuron n with $1 < n < N_l$. With a given set of P inputs pattern x_p ($1 < p < P$), a set of t_p labels associated to each x_p, the output γ_n^l ($\gamma_n^0 = x_p^n$) of the neuron n of the layer l is defined as follows:

$$\gamma_n^l = \alpha(s_n^l) \tag{1}$$

With

$$s_n^l = \sum_{m=0}^{N_{l-1}} \omega_{nm}^l \times \gamma_m^{l-1} + b_n^l \tag{2}$$

and α being the activation function. Further explanations about activation functions will be developed in SectionIII below.

B. Convolutional Neural Networks

Convolutional Neural Networks (CNNs) is a type of deep learning architecture particularly effective for analysing grid-structured data, such as images. They rely on convolutional layers that apply filters (also known as kernels) to extract local features like edges, textures, or shapes. As more layers are stacked, CNNs progressively learn more abstract and complex representations — starting from basic patterns in early layers to full object representations in deeper ones. Typically, they also include pooling layers to downsample spatial dimensions and fully connected layers for producing final predictions or classifications.

CNN's forward phase is defined as follows: with $S_{a,b}^l$ the pre-activation output at layer l and at the indexes (a, b) of the new feature map and w the weight-filter map of size $f * f$:

$$\gamma_{a,b}^l = \alpha(S_{a,b}^l) \tag{3}$$

With:

$$S_{a,b}^l = \sum_{c=0}^{f-1} \sum_{d=0}^{f-1} w^l \times \gamma_{a+c,b+d}^{l-1} \tag{4}$$

And α defined as in (1).

III. Activation Functions

A. Background

Activation functions play a crucial role in neural networks. Indeed, activation functions introduce non-linearity within the neural network since these functions contain non-linear components. Originally, activation functions are used to introduce non-linear behaviour in the data processing during the learning process of neural networks. Some of the most commonly employed activation functions are the Rectified Linear Unite (ReLU) [24] function, the Gaussian Linear Unite (GeLU) [21] function, the sigmoid [31] or the hyperbolic tangent [41]. Activation functions can be defined as follows:

Let E be a real vector space, let x and θ be vectors in E^n and β be a scalar in E. The activation function $F : E \to E$ operates the transformation:

$$F\left(\sum_{i=1}^{n} x_i\theta_i + \beta\right) = y_i \tag{5}$$

One can notice that activation functions, and more generally neural networks, aren't necessarily defined on a real space, as there exist complex [36] and quaternions [30] neural networks with dedicated activation functions. It is crucial for a neural network to be able to approximate solutions separated with complex hyper-plans in non-linear sub-spaces encountered in real-life related tasks. Another requirement of activation functions is their differentiability. It is a required property for activation functions to be able to be differentiated to compute the gradient descent algorithm (backward process).

Activation functions can be divided into two main categories: non-trainable and trainable activation functions. As their name underlines, non-trainable activation functions apply transformations to incoming data without being altered during the learning process. Conversely, trainable activation functions are composed of learnable parameters and suit to the data during the learning process as those learnable parameters are computed during the backward process (gradient descend algorithm). One of the first trainable function introduced in neural networks was the Adjustable Generalized Sigmoid [16]:

$$AGSig(x) = \frac{\alpha}{1 + exp(-\beta x)}, \tag{6}$$

with α and β are the learnable parameters. Over the years, other trainable activation functions have been introduced, often based on previous non-trainable activation functions, such as Sigmoidal Selector [34] or Flexible ReLU [32].

B. Rational Activation Functions

Rational **A**ctivation **F**unctions (RAF) have been introduced by [7] and are defined as follows:

$$F_{RAF}(x) = \frac{P(x)}{Q(x)} = \frac{\sum_{i=0}^{r_p} a_i x^i}{1 + |\sum_{j=1}^{r_q} b_j x^j|}, \tag{7}$$

where r_p and r_q are the polynomial degrees of the numerator and denominator respectively and $r_p \geq r_q$. Each a_i and b_j coefficient of the P and Q polynomial functions is a learnable parameter. The use of functions with learnable parameters allows the neural network to better capture different features within the model. Neural networks using these functions are called rational neural networks. These networks have been demonstrated to be able to approximate any ReLU networks [7]. The degree of both r_p and r_q are hyper-parameters of the neural network. These polynomial activation functions have shown promising results during different benchmark tasks such as MNIST classification [7] and NLP related architectures such as attention-based transformers [14].

C. Proposed non-linear Rational Activation Function

As detailed in section III-A, different mere non-trainable activation functions are based on the exponential and logarithm functions, such as Tanh, Sigmoid, Softmax or LogSoftmax. Following this design, the trainable activation functions proposed first were based on these non-trainable functions, such as sigmoidal selector [34] or Flexible ReLU [32]. Rational activation functions (equation 7) are not exponential or logarithm based. Therefore, these functions are not able to well express non-linear hidden sub-spaces. To address this, we propose original rational activation functions based on exponential and logarithm functions to represent non-linear informations contained in datasets defined there-after with the scalar hyper-parameter λ.

Exponential Rational Activation Function: This function consider non-linear components contained in datasets by introducing exponential composed function on RAF.

$$F_{exp}(\lambda x) = F(e^{\lambda x}) \quad = \frac{P(e^{\lambda x})}{Q(e^{\lambda x})} \tag{8}$$

Hyperbolic Sinus Rational Activation Function: The paper also proposes a linear combination of exponential transformations for RAF.

$$F_{\text{sinh}}(x) = F(\text{sinh}(\lambda x)) \quad = \frac{P(\text{sinh}(\lambda x))}{Q(\text{sinh}(\lambda x))}, \tag{9}$$

with the hyperbolic sinus function:

$$\text{sinh}(\lambda x) = \frac{e^{\lambda x} - e^{-\lambda x}}{2} \tag{10}$$

Inverse Hyperbolic Sinus Rational Activation Function: This activation function introduces logarithm (exponential inverse function) to study the impact of inverse non-linear function.

$$F_{\text{arsinh}}(x) = F(\text{arsinh}(x)) \quad = \frac{P(\text{arsinh}(x))}{Q(\text{arsinh}(x))}, \tag{11}$$

with the inverse hyperbolic sinus function:

$$\text{arsinh}(x) = \log(x + \sqrt{1 + x^2}), \tag{12}$$

Knowing the structure of rational activation functions, the assumption of this paper is first that with a set of well selected learnable polynomial coefficients, the rational activation functions can replace the usual linear layers with Relu activation functions and drastically reduce the number of learnable parameters of the neural network models; the second hypothesis of the paper is the efficiency and robustness of our customs non-linear rational activation functions described in (8), (9) and (10).

D. Approximation Capability

This section provides a formal justification for the ability of our exponential-based RAF to approximate ReLU-based neural networks, while avoiding unnecessary technical detail. For a concise argument, the demonstration of the capacity of such rational activation functions to approximate ReLU neural networks can be directly derived from the appendix of [7], particularly for F_{sinh} and F_{argsinh}. Specifically, the proof of Lemma 1 in [7], which is based on results from [5], can be adapted to continuous odd functions such as sinh and arsinh.

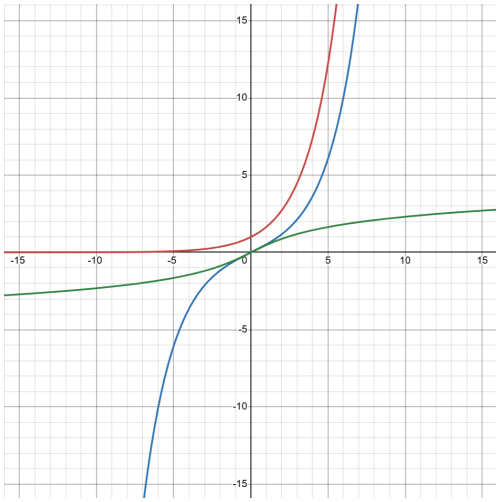


Fig. 1. Graph with $\exp(\frac{1}{2} * x)$ in red, $\sinh(\frac{1}{2} * x)$ in blue and $\text{arsinh}(\frac{1}{2} * x)$ in green.

For a more rigorous approach:
Lemma 1 in [7] states:

Lemma 1. *Let $0 < \epsilon < 1$. There exists a rational network $R : [-1, 1] \to [-1, 1]$ of size $O(\log(\log(1/\epsilon)))$ such that:*

$$||R - ReLU||_\infty := max_{x\in[-1,1]}|R(x) - ReLU(x)| \leq \epsilon \tag{13}$$

Moreover, no rational network of size smaller than $\Omega(\log(\log(1/\epsilon)))$ can achieve this.

The proof of this lemma is provided in the appendix of [7] and is based on Equation (3.3) from [5]. Without delving into the details of Zolotarev numbers, we can derive analogous properties from [5], since its approach involves working with intervals of the form $[-b, -a], [a, b]$ where $0 < a < b < \infty$.

Let F denote either the hyperbolic sine function as defined in (10) or the inverse hyperbolic sine function as in (12). Since F is a continuous odd function, the transformed intervals $[F(-b), F(-a)]$ and $[F(a), F(b)]$ become $[-F(b), -F(a)]$ and $[F(a), F(b)]$, respectively. These can be rewritten as $[-B, -A], [A, B]$ with $F(a) = A$ and $F(b) = B$.

Because applying F preserves the structure of the original intervals, the subsequent development in [5] remains valid under this transformation. This allows us to restate Lemma 1 from [7] in the following form:

Lemma 2. *Let $0 < \epsilon < 1$. There exists a hyperbolic sine/inverse hyperbolic sine rational network $R : [-1, 1] \to [-1, 1]$ of size $O(\log(\log(1/\epsilon)))$ such that:*

$$||R - ReLU||_\infty := max_{x\in[-1,1]}|R(x) - ReLU(x)| \leq \epsilon \tag{14}$$

Moreover, no rational network of size smaller than $\Omega(\log(\log(1/\epsilon)))$ can achieve this.

IV. Experiments

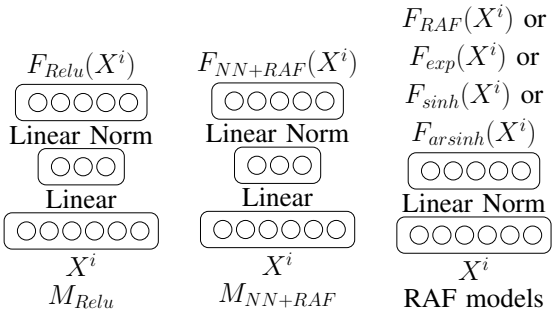
This section evaluates the effectiveness and the robustness of the proposed non-linear rational activation functions throughout three experiments of functions approximation (section IV-A), a theme identification task of spoken dialogues transcribed in very noisy conditions from the DECODA corpus [4] (section IV-B) and an image classification task from the CIFAR10 corpus (section IV-C). The aim of these experiments is **not** to reach state-of-the-art performances, but to compare the results of the simple architectures introduced before. As established in [3] and [37], testing original learnable activation functions for neural networks on small architectures is common practice, and allows a better comprehension and insight on the inner workings of learnable activation functions.

A. Function Approximation

This experiment evaluates different neural and non-neural approaches for functions approximation. Let $X^1, X^2, ..., X^{100}$ be a set of 100 real-valued random vectors of size 50 with each component $0 \leq X^i_j \leq 10$ for all $0 \leq i \leq 100$ and $0 \leq j \leq 50$. We then compute the image of each X^i throughout a function $F : F(X^i) = Y^i$. Our approximation tasks (table I) are to approximate as close as possible each Y^i with the outputs of our models $\tilde{Y^i}$. M_{Relu} and M_{NN+RAF} employ feed-forward neural networks (NN with a Linear layer of size [50, 50] + Norm layer). The RAF degrees are $r_p = 5$ and $r_q = 4$, with $\lambda = 1$ (see (8), (9), (11)):

- M_{Relu} is a NN and a Relu activation function.
- M_{NN+RAF} employs the same NN but with a RAF.
- M_{RAF} is only made of a norm layer and a RAF.
- M_{exp} is composed of a layer norm and the F_{exp} RAF.
- M_{sinh} is a normalized layer with the F_{sinh} RAF.
- M_{arsinh} employs a norm layer and the F_{arsinh} RAF.

TABLE I
Function approximation architectures.



Training has been realized over 100 Epochs, with the L_1 loss function and a learning rate of 0.005. The functions approximated by our six different models are $F(x) = x$, $F(x) = \frac{1}{x}$ and $F(x) = log(x)$. Table II reports the results observed during the approximation task of $F(x)$

TABLE II
Results for the approximation task.

Model	$y = x$	$y = \frac{1}{x}$	$y = log(x)$	#param
M_{Relu}	411.499	79.719	92.015	6,330
M_{NN+RAF}	249.793	77.125	68.934	6,369
M_{RAF}	226.851	69.104	47.590	39
Proposed rational activation functions				
M_{exp}	**62.750**	69.245	**29.841**	39
M_{sinh}	226.453	69.132	47.219	39
M_{arsinh}	227.237	**69.077**	47.950	39

The results show first that M_{Relu} that employs a mere NN with a Relu activation function obtains the worst performance of all models (error on a simple $y = x$ approximation task roughly twice larger than for the other models). The best results observed are those from models employing RAF with a gain of about 2 points compared to the Relu model. We can also underline that the model with Relu (M_{Relu}) reaches a difference among the 100 inputs X^i between real outputs Y^i of 411 compared to the same architecture but with RAF (M_{NN+RAF}) with 226. The difference is even larger for the other functions (92 for M_{Relu} and 47 for the M_{NN+RAF} for example). Table II also depicts the results obtained by the proposed non-linear RAF. We can easily see that the proposed M_{exp} obtains the best result for the first experiment ($y = x$) with a main reduction of the error with 4 times less than the M_{NN+RAF} and more than 6 times better than the M_{Relu}. For the other functions, the same observations can be made. For example, the second function to approximate ($y = \frac{1}{x}$), the three proposed non-linear RAFs obtained roughly the same results with a main gain compared to M_{Relu} and close to the M_{RAF}.

Finally, these approximation functions experiments show that models with few resources in terms of learnable parameters architectures based on rational activation functions can outperform architectures with way more learnable parameters. Indeed, the models based on RAF (M_{NN+RAF}, ..., M_{arsinh}) are composed of only 39 learnable parameters, compared to M_{Relu} with $6,330$ parameters that represents only **0.61%** of the learnable parameters compared to M_{NN+RAF} and M_{Relu}. These findings led us to design an image classification task and a more real-world oriented NLP task.

B. Classification of Noisy Spoken Dialogues

Noisy Spoken Dialogues from the Decoda corpus: This experiment consists of a conversation classification task of spoken dialogues from the DECODA corpus depicted in Figure 2 and concerns the automatic analysis of telephone conversations [15] between an agent and a customer in the call center of the Paris public transport authority (RATP) [4]. This set of spoken dialogues is a corpus of agent/customer telephone conversations in French from the customer care service of the RATP Paris transportation call-center composed of 1,242 telephone conversations corresponding to 74 hours of signal transcribed with the Automatic Speech Recognition (ASR) system LIA- Speeral [25] to keep the noisy context. Each conversation has been manually transcribed and labelled with one theme (of 8 possible themes) corresponding to the main topic. The dataset is split into training(730 conversations), development(171 conversations) and test(321 conversations) datasets. LIA-Speeral, a highly error prone ASR system is employed to keep real life conditions of speech recording to better study the impact of noisy segments during the learning process of rational activation functions.

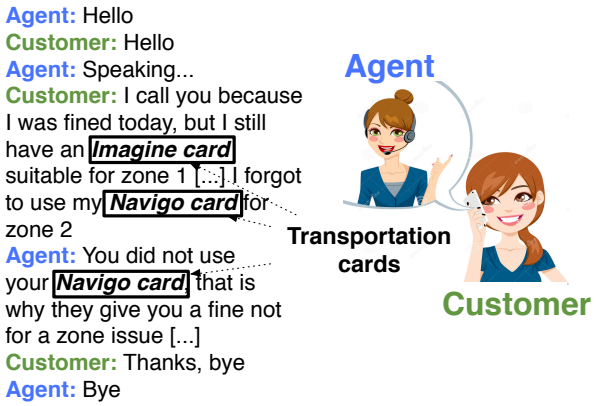


Fig. 2. Exemple of a dialogue from the DECODA corpus for the SLU task of theme identification. This dialogue has been labelled by the agent as "OBJECTS" (Lost & founds objects).

Latent features of corrupted transcriptions from CamemBERT: The output of the LIA-Speeral is labelled, highly noisy, text conversations. The language model used for generating the embeddings is CamemBERT. CamemBERT is a multi-layer bidirectional Transformer [38] French language model similar to RoBERTa that uses whole-word masking and SentencePiece tokenization [19] instead of WordPiece. RoBERTa [27] is an English language model which is a modified version of BERT [13]. CamemBERT has been trained on the french subcorpus OSCAR [35] made of 138 GB of raw French text, that is a significantly smaller corpus than the one employed for RoBERTa (161 GB of English text). CamemBERT's parameters have not been fine-tuned during the learning process. [8] compares the performances of CamemBERT to other French language models: FlauBERT [20], FrALBERT [9] and XLM-R [11] on the MEDIA [6] and ATIS-FR [1] corpus, with the F1 and CER metrics. CamemBERT performs better on the two corpus for the F1 metric (90% on F1 measure compared to FlauBert with 89% for example). CamemBERT is used to encode and extract features from the highly noisy text conversations given by LIA-Speeral system. The CamemBERT encoding has a maximum dimension of 512. This is due to the high number of tokens contained in some conversations. Every encoded vector is truncated to be of maximum size 512. The extracted features from CamemBERT are of size $[1, x, 728]$, with $x \leq 512$. As we need all vectors to be of the same size, padding is applied, so each extracted feature is of size $[1, 512, 728]$. To finish, each extracted feature is flatten to attain the size of $[372736]$. These features are then fed through the different model architectures detailed thereafter and in table III. Learning was conducted on the training dataset during 20 Epochs with a learning rate of $2e-05$ and a hyper parameter $\lambda = 0.5$ (see (8), (9), (11)).

RAF and non-RAF models configurations: Models have been evaluated on the development dataset at each Epoch, and evaluated on the test dataset at the end of training. The loss function used for training is the cross entropy. The models evaluated during the theme identification task of highly corrupted spoken dialogues from the DECODA corpus are similar to those described in sectionIV-A. The difference is due to the classification task itself, and therefore, a linear layer from the input size (372736) to the number of themes (8) is added with a Softmax on the top of each model M. The difference between M_{NN+RAF} and M_{RAF} is as described before with M_{NN+RAF} has an additional NN (linear layer) that gives a two linear NN with the first one being of size $[372736, 384]$ (Linear_1) and the second one $[384, 8]$ (Linear_2) and all M_{RAF} based models only contain a single linear layer of size $[372736, 8]$ (Linear_3).

TABLE III
FUNCTION APPROXIMATION ARCHITECTURES.

$F_{Relu}(X^i)$	$F_{NN+RAF}(X^i)$	$F_{RAF}(X^i)$ or $F_{exp}(X^i)$ or $F_{sinh}(X^i)$ or $F_{arsinh}(X^i)$
SoftMax	SoftMax	
Linear_2	Linear_2	
Linear Norm	Linear Norm	Linear Norm
Linear_1	Linear_1	Linear_3
X^i	X^i	X^i
M_{Relu}	M_{NN+RAF}	RAF models

Experimental results and discussions: Table IV details first the loss observed during the learning process of each model. One can firstly observe that the rational activation functions perform better than Relu function regarding the loss of the model for all RAF based models and corroborates claims from [7] and [14]. Secondly, as the smallest loss is reached by

M_{RAF} by a close margin, table IV also shows that lightweight neural networks are able to reach performances of models with more parameters. These findings match with our first function approximation in subsection III. In this experimental context, models with **2.08**% of the total number of parameters of more heavy neural networks reach better performances in term of loss observed.

TABLE IV
LOSS ON TRAIN DATASET AND ACCURACY ON THE TRAIN, DEVELOPMENT AND TEST DATASET.

Model	Loss	Train	Development	Test
M_{Relu}	01.388	47%	45%	43%
M_{NN+RAF}	06.32e-05	**100**%	54%	46%
M_{RAF}	**02.34e-05**	**100**%	53%	50%
Proposed rational activation functions				
M_{exp}	20e-05	**100**%	**57%**	49%
M_{sinh}	40.40e-05	95 %	48%	**55%**
M_{arsinh}	08.17e-5	**100**%	56%	53%

Results from table IV also shows that rational activation functions perform better than Relu function regarding the accuracy of the model on the training, development or the test datasets and all models reach about 100% accuracy on the training dataset with the same hyper-parameters except for the M_{Relu} (only 47%). In terms of accuracy, the RAF based models reach roughly more than 50% of accuracy compared to the Relu model with a gain of **9%**. Even if the models are quite simple and contain few parameters, one can underline that RAF based models always outperform the M_{Relu} with the best accuracy reached for the development dataset of 57% with the M_{exp} and 55% on the test dataset with the M_{sinh}. The M_{arsinh} obtains equivalent accuracies than the other non-linear RAF for all datasets.

C. Classification of CIFAR10 Images

This experiment is an image classification task on the CIFAR10 dataset [18]. The CIFAR-10 dataset consists of 60,000 color images, each with a resolution of 32×32 pixels and three RGB color channels. These images are evenly distributed across 10 distinct object classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck, with 6,000 images per class. The different models compared on this classification task are detailed below. The hyper-parameters employed during the training phase are the following: the batch size was set at 50, the degrees of the RAFs employed are all set to $r_p = 5$ and $r_q = 4$, with a λ set as 0.9 (see (8), (9), (11)). Training was conducted across 15 epochs, with a learning rate set at $5.e^{-4}$. The loss function used is the cross-entropy. For this classification task, two types of convolutional models were tested. The 2_CONV models with two convolutional layers, and the 1_CONV with only one convolutional layer. These basic architectures are then split regarding the type of activation function employed, being ReLU, RAF or the non-linear RAF (8), (9) and (10). The models' architectures are detailed below.

- 2_CONV$_{Relu}$ is a CNN composed of two convolutional layers followed by one fully connected layer. The network takes as input a 4-channel image and applies a first 2D convolutional layer with 6 output channels and a kernel size of 5, followed by a 2×2 max pooling operation. A second convolutional layer increases the channel depth to 16, again with a kernel size of 5, followed by max pooling and a ReLU activation function. The resulting feature maps are flattened into a 1D vector and passed through a linear layer which outputs a 10-dimensional vector. ReLU activation is applied after the linear layer. Figure **??** illustrates this model.

- 2_CONV$_{RAF}$ employs the same architecture as 2_CONV$_{Relu}$ but with RAFs in place of ReLU activations.

- 2_CONV$_{exp}$ employs the same architecture as 2_CONV$_{Relu}$ but with F_{exp} RAFs in place of ReLU activations.

- 2_CONV$_{sinh}$ employs the same architecture as 2_CONV$_{Relu}$ but with F_{sinh} RAFs in place of ReLU activations.

- 2_CONV$_{arsinh}$ employs the same architecture as 2_CONV$_{Relu}$ but with F_{arsinh} RAFs in place of ReLU activations.

- 1_CONV$_{Relu}$ is a smaller CNN. It begins with a 2D convolutional layer that processes 4-channel input images, producing 6 feature maps using a kernel size of 5 and no bias. A 2×2 max pooling operation reduces the spatial dimensions and passed through a ReLU activation function. The resulting feature maps are then flattened into a one-dimensional vector. The flattened features are then connected to a single fully connected layer with 10 output units. A ReLU activation is applied before the output. Figure **??** illustrates this model.

- 1_CONV$_{RAF}$ employs the same architecture as 1_CONV$_{Relu}$ but with RAFs in place of ReLU activations.

- 1_CONV$_{exp}$ employs the same architecture as 1_CONV$_{Relu}$ but with the F_{exp} RAFs in place of ReLU activations.

- 1_CONV$_{sinh}$ employs the same architecture as 1_CONV$_{Relu}$ but with the F_{sinh} RAFs in place of ReLU activations.

- 1_CONV$_{arsinh}$ employs the same architecture as 1_CONV$_{Relu}$ but with the F_{arsinh} RAFs in place of ReLU activations.

TABLE V
LOSS, ACCURACY AND NUMBER OF PARAMETERS FOR THE CIFAR10 CLASSIFICATION.

Model	Loss	Accuracy	# parameters
2_CONV$_{Relu}$	1238.692	56.901%	67,160
2_CONV$_{RAF}$	1108.076	61.204%	67,178
1_CONV$_{Relu}$	1352.293	52.824%	50,080
1_CONV$_{RAF}$	1157.579	59.534%	50,098
Proposed rational activation functions			
2_CONV$_{exp}$	**1041.097**	**63.615**%	67,178
2_CONV$_{sinh}$	1054.060	63.564%	67,178
2_CONV$_{arsinh}$	1101.215	61.378%	67,178
1_CONV$_{exp}$	**1079.0109**	**62.61** %	50,098
1_CONV$_{sinh}$	1166.147	59.426 %	50,098
1_CONV$_{arsinh}$	1126.435	61.21%	50,098

Results for the CIFAR10 classification are reported in Table V. We first notice that the best performances are reached by the 2_CONV$_{exp}$ model for the loss value, with 1041.097 points and for the prediction accuracy, with 63.516%. This represents a difference in loss value of 197.595 points with the most used ReLU-based model 2_CONV$_{Relu}$ and a difference in accuracy of $6,714\%$.

The superior performances of 2_CONV models were expected over 1_CONV due to larger numbers of parameters in 2_CONV models. Within 2_CONV models, those implementing our custom RAFs outperformed slightly 2_CONV$_{RAF}$ in regard to loss and accuracy, and all clearly outperformed the more established 2_CONV$_{Relu}$. For example, 2_CONV$_{sinh}$ reached a loss 184,632 points lower and an accuracy $6,663\%$ higher. The worst scoring 2_CONV model regarding both loss value and accuracy is 2_CONV$_{Relu}$.

Regarding 1_CONV models, models implementing non-linear RAFs all outperformed 1_CONV$_{Relu}$ and two of those non-linear RAFs models outperformed 1_CONV$_{RAF}$. Both 1_CONV$_{exp}$ and 1_CONV$_{arsinh}$ outperformed 1_CONV$_{RAF}$. The worst scoring 1_CONV model regarding both loss value and accuracy is 1_CONV$_{Relu}$. These findings reinforce our opinion about the better performances of non-linear RAFs for prediction accuracy and loss minimisation.

Amongst the proposed rational activation functions, 1_CONV$_{exp}$ yields the best results in both shallow and deep models. The 1_CONV$_{exp}$, with only 50,098 parameters, outperforms the deeper 2_CONV$_{Relu}$ in both accuracy and loss, highlighting the efficiency of rational functions in compact architectures. While 1_CONV$_{sinh}$ and 1_CONV$_{arsinh}$ also surpass ReLU models, their performances vary slightly.

These results, while close, should be compared regarding the number of parameters: 67,160 for 2_CONV$_{Relu}$ and 50,098 for 1_CONV$_{exp}$, which is approximatively 25.40**%** less parameters. These findings suggest that carefully designed activation functions can yield significant performance improvements, even in models with fewer parameters, and reaffirm the potential of RAFs in lightweight yet efficient architectures. Results emphasise the importance of activation function choice in neural network performance. While deeper architectures naturally benefit from increased capacity, the adoption of RAF — especially in smaller models — can yield competitive or even superior outcomes with fewer parameters. This highlights RAFs as a promising direction for enhancing efficiency and generalisation in neural networks architectures.

V. CONCLUSION

Summary. This paper proposes a promising set of activation functions based on polynomials functions with learnable parameters. These activation functions called rational activation functions (RAF) allow the neural based model to learn the model parameters alongside the activation function parameters. The results observed in an approximation task, a dedicated spoken dialogues classification task as well as an image classification class with a very small number of parameters have shown very interesting results compared to ReLU activation function and the hitherto proposed RAF. Moreover, the experiments have shown that a mere RAF can mimic, even overtake, the performance of a neural network with a ReLU activation function with less parameters (69 for RAF and 6339 for the NN ReLU). This work confirmed our assumptions that: 1) lightweight neural networks models based on RAF can reach equivalent or better performances than more heavy neural networks, especially regarding loss minimisation; 2) injection of non-linear component on the RAF improves both the capability of the model to approximate and process (classification) datasets in controlled and real conditions (imperfect spoken dialogues recorded in noisy conditions).

Limitations and Future Works. Future works will also be related to both theoretical and experimental aspects. Indeed, as was stated in section III-C the capacity of ReLU-based neural networks approximation by functions such as F_{sinh} and F_{arsinh} is derived from the lemma in the appendix of [7]. Regarding F_e, a new demonstration of approximation will be investigated. Investigations regarding the importance of λ and it's role will be conducted, as well as making it a learnable parameters. Experimental works will also be conducted in an attention-based framework, such as transformers using exponential and

logarithm-based rational activations. Finally, the experiments have been conducted in a small range of functions and other non-linear RAF-based activation functions have to be evaluated. Moreover, the spoken dialogue classification task contains 8 themes and a small amount of transcribed spoken dialogues, therefore future experiments will be conducted on larger datasets to further evaluate the proposed approach.

REFERENCES

[1] J.-Y. Antoine and J. Goulian. Word order variations and spoken man-machine dialogue in french: a corpus analysis on the atis domain. *Proc. of Corpus Linguistics, Lancaster, Royaume-Uni*, 2001.

[2] A. Apicella, F. Donnarumma, F. Isgrò, and R. Prevete. A survey on modern trainable activation functions. *Neural Networks*, 138:14–32, 2021.

[3] A. Apicella, F. Isgro, and R. Prevete. A simple and efficient architecture for trainable activation functions. *Neurocomputing*, 370:1–15, 2019.

[4] F. Bechet, B. Maza, N. Bigouroux, T. Bazillon, M. El-Beze, R. De Mori, and E. Arbillot. Decoda: a call-centre human-human spoken conversation corpus. In *LREC*, pages 1343–1347, 2012.

[5] B. Beckermann and A. Townsend. On the singular values of matrices with displacement structure. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1227–1248, 2017.

[6] H. Bonneau-Maynard, S. Rosset, C. Ayache, A. Kuhn, and D. Mostefa. Semantic annotation of the french media dialog corpus. In *InterSpeech*, pages 3457–3460, 2005.

[7] N. Boullé, Y. Nakatsukasa, and A. Townsend. Rational neural networks. *Advances in neural information processing systems*, 33:14243–14253, 2020.

[8] O. Cattan, S. Ghannay, C. Servan, and S. Rosset. Benchmarking transformers-based models on french spoken language understanding tasks. *arXiv preprint arXiv:2207.09152*, 2022.

[9] O. Cattan, C. Servan, and S. Rosset. On the usability of transformers-based models for a french question-answering task. *arXiv preprint arXiv:2207.09150*, 2022.

[10] K. Chowdhary and K. Chowdhary. Natural language processing. *Fundamentals of artificial intelligence*, pages 603–649, 2020.

[11] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.

[12] S. K. Dam, C. S. Hong, Y. Qiao, and C. Zhang. A complete survey on llm-based ai chatbots. *arXiv preprint arXiv:2406.16937*, 2024.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[14] H. Fang, J.-U. Lee, N. S. Moosavi, and I. Gurevych. Transformers with learnable activation functions. *arXiv preprint arXiv:2208.14111*, 2022.

[15] J. J. Godfrey, E. C. Holliman, and J. McDaniel. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, volume 1, pages 517–520. IEEE, 1992.

[16] Z. Hu. The study of neural network control system. *Control and Decision*, 7:361–366, 1992.

[17] T. Jo. Multiple layer perceptron. In *Deep Learning Foundations*, pages 225–246. Springer, 2023.

[18] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[19] T. Kudo and J. Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

[20] H. Le, L. Vial, J. Frej, V. Segonne, M. Coavoux, B. Lecouteux, A. Allauzen, B. Crabbé, L. Besacier, and D. Schwab. Flaubert: Unsupervised language model pre-training for french. *arXiv preprint arXiv:1912.05372*, 2019.

[21] M. Lee. Mathematical analysis and performance evaluation of the gelu activation function in deep learning. *Journal of Mathematics*, 2023(1):4229924, 2023.

[22] S. Li and B. Gong. Word embedding and text classification based on deep learning methods. In *MATEC Web of Conferences*, volume 336, page 06022. EDP Sciences, 2021.

[23] X. Li, Y. Yao, X. Jiang, X. Fang, X. Meng, S. Fan, P. Han, J. Li, L. Du, B. Qin, et al. Flm-101b: An open llm and how to train it with $100kbudget. arXivpreprintarXiv : 2309.03852, 2023$.

[24] Y. Li and Y. Yuan. Convergence analysis of two-layer neural networks with relu activation. *Advances in neural information processing systems*, 30, 2017.

[25] G. Linares, P. Nocéra, D. Massonie, and D. Matrouf. The lia speech recognition system: from 10xrt to 1xrt. In *International Conference on Text, Speech and Dialogue*, pages 302–308. Springer, 2007.

[26] Z.-H. Ling, S.-Y. Kang, H. Zen, A. Senior, M. Schuster, X.-J. Qian, H. M. Meng, and L. Deng. Deep learning for acoustic modeling in parametric speech generation: A systematic review of existing techniques and future trends. *IEEE Signal Processing Magazine*, 32(3):35–52, 2015.

[27] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[28] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, and A. Mian. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*, 2023.

[29] M. Pak and S. Kim. A review of deep learning in image recognition. In *2017 4th international conference on computer applications and information processing technology (CAIPT)*, pages 1–3. IEEE, 2017.

[30] T. Parcollet, M. Morchid, and G. Linares. A survey of quaternion neural networks. *Artificial Intelligence Review*, 53(4):2957–2982, 2020.

[31] H. Pratiwi, A. P. Windarto, S. Susliansyah, R. R. Aria, S. Susilowati, L. K. Rahayu, Y. Fitriani, A. Merdekawati, and I. R. Rahadjeng. Sigmoid activation function in selecting the best model of artificial neural networks. In *Journal of Physics: Conference Series*, volume 1471, page 012010. IOP Publishing, 2020.

[32] S. Qiu, X. Xu, and B. Cai. Frelu: Flexible rectified linear units for improving convolutional neural networks. In *2018 24th international conference on pattern recognition (icpr)*, pages 1223–1228. IEEE, 2018.

[33] J. Schmidt-Hieber. Nonparametric regression using deep neural networks with relu activation function. 2020.

[34] Y. Singh and P. Chandra. A class+ 1 sigmoidal activation functions for ffanns. *Journal of Economic Dynamics and Control*, 28(1):183–187, 2003.

[35] P. J. O. Suárez, B. Sagot, and L. Romary. Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures. In *7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7)*. Leibniz-Institut für Deutsche Sprache, 2019.

[36] C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal. Deep complex networks. *arXiv preprint arXiv:1705.09792*, 2017.

[37] E. Trentin. Networks with trainable amplitude of activation functions. *Neural Networks*, 14(4-5):471–493, 2001.

[38] A. Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[39] Y. Xia, J. Kim, Y. Chen, H. Ye, S. Kundu, C. C. Hao, and N. Talati. Understanding the performance and estimating the cost of llm fine-tuning. In *2024 IEEE International Symposium on Workload Characterization (IISWC)*, pages 210–223. IEEE, 2024.

[40] H. Yang, Y. Zhao, Y. Wu, S. Wang, T. Zheng, H. Zhang, Z. Ma, W. Che, and B. Qin. Large language models meet text-centric multimodal sentiment analysis: A survey. *arXiv preprint arXiv:2406.08068*, 2024.

[41] B. Zamanlooy and M. Mirhassani. Efficient vlsi implementation of neural networks with hyperbolic tangent activation function. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 22(1):39–48, 2013.

[42] E. Y. Zhang, A. D. Cheok, Z. Pan, J. Cai, and Y. Yan. From turing to transformers: A comprehensive review and tutorial on the evolution and applications of generative transformer models. *Sci*, 5(4):46, 2023.

[43] H. ZHANG, Q. ZHANG, and J. YU. Overview of the development of activation function and its nature analysis. *Journal of Xihua University (Natural Science Edition)*, 40(4):1–10, 2021.

[44] S. K. Zhou, H. Greenspan, C. Davatzikos, J. S. Duncan, B. Van Ginneken, A. Madabhushi, J. L. Prince, D. Rueckert, and R. M. Summers. A review of deep learning in medical imaging: Imaging traits, technology trends, case studies with progress highlights, and future promises. *Proceedings of the IEEE*, 109(5):820–838, 2021.