The Role of Aggregation Functions on Transformers and ViTs Self-Attention for Classification

Joelson Sartori*, Giancarlo Lucca†, Rodrigo de Bem* and Graçaliz Dimuro*
*Center of Computational Sciences (C3), Federal University of Rio Grande (FURG), Brazil
†Center for Social and Technological Sciences (PGEEC), Catholic University of Pelotas (UCPEL), Brazil
Email: joelsonsartori@furg.br, giancarlo.lucca@ucpel.edu.br, rodrigobem@furg.br, gracalizdimuro@furg.br

***Abstract*—Aggregation functions are mathematical operations that combine or summarize a set of values into a single representative value. They play a crucial role in the attention mechanisms of Transformer neural networks. However, Transformers' default aggregation functions, based on matrix multiplication, may have limitations in certain classification scenarios. This function may struggle with the complexity of information present in the input data, resulting in lower accuracy and efficiency. Considering this issue, the present work aims to replace the traditional matrix multiplication operation used in the classical attention mechanism with alternative and more general aggregation functions.**

To validate the new aggregation methods on the attention mechanism, we conducted experiments on two datasets, the recently propose Google American Sign Language (ASL) Fingerspelling Recognition and the well-known CIFAR-10, performing time series and image classification, respectively. Results shed light on the role of aggregation functions for classification with Transformers, demonstrating promising outcomes and potential for further improvements.

I. Introduction

Machine learning has been extensively explored in a variety of domains, with classification being one of its fundamental tasks [1]. In recent decades, convolutional and recurrent neural networks played a dominant role in the fields of image classification and natural language processing [2], respectively. However, the emergence of Transformer neural networks [3] has allowed new groundbreaking state-of-the-art results on both tasks.

Transformers introduced an innovative architecture, with their attention mechanism playing a crucial role in processing complex and long-range dependent sequential information. Through these mechanisms, Transformer neural networks can assign different levels of importance to parts of the input data, enabling a richer and more adaptable representation. With this new architecture, it became possible, for instance, to develop more specific models for both text [4] and image classification [5].

Important components of the Transformer networks, the aggregation functions are mathematical operations used to combine or summarize a set of values into a single representative value. These functions have various applications, such as calculating descriptive statistics, summarizing information, or reducing the dimensionality of a dataset [6], [7]. Despite the advantages of Transformers networks, their default aggregation function based on matrix multiplication may have limitations in certain classification scenarios. This function may struggle with the complexity of information present in the input data, resulting in lower accuracy and efficiency.

This work explores the use of novel aggregation functions in the self-attention mechanism of Transformer neural networks. The goal is to shed light on alternatives to the default aggregation function and evaluate their effects on the performance of Transformers applied to classification problems. Initially, we review the fundamental concepts related to aggregation functions, Transformer neural networks, attention mechanisms, and Vision Transformers (ViT). Following, we present in detail our new aggregation methodology. After that, we show the experiments conducted to evaluate the performance of the new aggregation mechanism on classification using relevant datasets. Finally, results are analyzed highlighting the main conclusions and future research directions in this field.

II. Theoretical foundation

Here we provide the theoretical background related to the main concepts of this study, namely aggregation functions, Transformers, and Vision Transformers.

A. Aggregation functions

Aggregation functions play a pivotal role in the analysis and processing of data in diverse domains of machine learning. These functions are responsible for combining multiple inputs into a single output, enabling the synthesis of information from different sources. It is important for these functions to respect two fundamental properties: the increase property and the monotonicity property.

Definition 1. *Following [8], [9], a function* $A : [0,1]^n \to [0,1]$ *is said to be an aggregation function whenever the following conditions are satisfied:*

- (A1) *A is increasing in each argument: for each* $i \in \{1,\ldots,n\}$, *if* $x_i \leq y$, *then* $A(x_1,\ldots,x_n) \leq A(x_1,\ldots,x_{i-1},y,x_{i+1},\ldots,x_n)$;
- (A2) *A satisfies the boundary conditions:* $A(0,\ldots,0) = 0$ *and* $A(1,\ldots,1) = 1$.

For an increasing (decreasing) function we do not mean a strictly increasing (decreasing) function.

979-8-3503-3872-0/23/$31.00 ©2023 IEEE

Definition 2. *Following [10], an aggregation function* $T : [0,1]^2 \to [0,1]$ *is a t-norm if, for all* $x, y, z \in [0,1]$*, it satisfies the following properties:*

(T1) *Commutativity:* $T(x,y) = T(y,x)$*;*
(T2) *Associativity:* $T(x, T(y,z)) = T(T(x,y), z)$*;*
(T3) *Boundary condition:* $T(x,1) = x$*.*

Definition 3. *Second [11], a function* $\mathfrak{m} : 2^N \to [0,1]$ *is said to be a fuzzy measure if, for all* $X, Y \subseteq N$*, it satisfies the following properties:*

($\mathfrak{m}1$) *Increasing: if* $X \subseteq Y$*, then* $\mathfrak{m}(X) \leq \mathfrak{m}(Y)$*;*
($\mathfrak{m}2$) *Boundary conditions:* $\mathfrak{m}(\emptyset) = 0$ *and* $\mathfrak{m}(N) = 1$*.*

Definition 4. *According to [8], [12], let* $\mu\colon 2^N \to [0, +\infty[$ *be a fuzzy measure. So, the* ***Sugeno integral*** $Su_\mu\colon [0, \mu(N)]^n \to [0, \mu(N)]$ *is defined, for all* $\vec{x} \in [0, \mu(N)]^n$*, by:*

$$Su_\mu(\vec{x}) = \bigvee_{i=1}^{n} \left(x_{(i)} \wedge \mu(A_{(i)}) \right), \tag{1}$$

where (i) *is a permutation on* 2^N *such that* $x_{(i-1)} \leq x_{(i)}$ *for all* $i = 1, \ldots, n$*, with* $x_{(0)} = 0$ *and* $A_{(i)} = \{(1), \ldots, (i)\}$*.*

With the intent of generalizing the expression of the Sugeno integral by replacing the sum and the maximum operator with respective functions F and G, Bardozzo et. al [13] presented the following definition:

Definition 5. *Let* $\mu\colon 2^N \to [0,1]$ *be a symmetric fuzzy measure,* $F\colon [0,\infty] \times [0,1] \to [0,\infty]$ *be a binary function and* $G\colon [0,\infty]^n \to [0,\infty]$ *be an n-ary function. A* ***Sugeno-like FG-functional*** *(FG-functional) is a function* $Su_\mu^{F,G}\colon [0,\infty]^n \to [0,\infty]$ *defined, for all* $\vec{x} \in [0,1]^n$*, by:*

$$Su_m^{F,G}(\vec{x}) = G\left(F\left(x_{(1)}, \mu(A_{(1)})\right), \ldots, F\left(x_{(n)}, \mu(A_{(n)})\right)\right), \tag{2}$$

where (i) *is a permutation on* 2^N *such that* $x_{(i-1)} \leq x_{(i)}$ *for all* $i = 1, \ldots, n$*, with* $x_{(0)} = 0$ *and* $A_{(i)} = \{(1), \ldots, (i)\}$*.*

In this study, we have considered the application of the standard Sugeno integral as well as some FG-functionals which are based on the functions presented in Table I.

TABLE I
FUNCTIONS G AND F USED IN THIS STUDY.

$G(\vec{x})$, $\vec{x} = (x_1, \ldots, x_n) \in [0,+\infty]^n$	$F(x,y)$, $x \in D$, $y \in [0,+\infty]$
$\text{Max}(\vec{x}) = \max_{i=1}^n x_i$	$\text{Min}(x,y) = \min(x,y)$
$\text{Sum}(\vec{x}) = \sum_{i=1}^n x_i$	$\text{Prod}(x,y) = xy$

B. Transformers

Transformers have proven to be highly efficient in a variety of tasks, ranging from natural language processing [4] to computer vision [5]. One of the key features driving the performance of Transformers is the self-attention mechanism.

In the Transformer algorithm, there are two fundamental components, the Encoder, and the Decoder layers. The Encoder is responsible for analyzing and transforming the input data into a suitable format for the model. It incorporates the self-attention mechanism, which captures the relationships between the input elements. On the other hand, the Decoder layer is responsible for generating an output sequence based on the representations obtained from the Encoder and previous contextual information.

As our issue pertains to data classification, our attention will be directed towards the Encoder layer, which is accountable for processing text input and generating representations, as well as the self-attention mechanism. By utilizing only the Encoder layer, we can effectively capture the relevant features from the input data and use them for classification purposes.

The self-attention mechanism enables the model to capture relationships between elements in an ordered sequence. In contrast to traditional approaches such as convolutional or Recurrent Neural Networks [14], the self-attention mechanism allows Transformers to consider global and long-range relationships between words, for instance, rather than relying solely on local information.The self-attention mechanism is grounded in matrix multiplications, applied to three fundamental elements: queries, keys, and values. These elements are obtained through linear transformations of the input representations, which are learned during the model's training.

The self-attention operation may be defined as

$$\textit{Self-Attention}(Q,K,V) = \textit{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{3}$$

where Q is the Query matrix, obtained by multiplying a weight matrix W_q by the input representation; K is the Key matrix, obtained by multiplying a weight matrix W_k by the input representation; V is the Value matrix, obtained by multiplying a weight matrix W_v by the input representation; and d_k is the dimension of queries and keys, which determines the scale of the operation.

The operation begins by calculating the similarity between queries and keys using aggregation function multiplication matrices, before normalizing the result using a softmax function [14]. The value matrix is then adjusted by the resulting similarity and summed, resulting in the final output of the self-attention mechanism.

Self-attention is applied to each input element in a sequence, allowing the model to learn the relative relevance between one element and all the others. This enables the model to identify the relevant element and capture the most important relationships in the problem.

During training, the weights of the matrices W_q, W_k, and W_v are learned through backpropagation [14], adjusting them to maximize performance on a specific task. Thus, the self-attention mechanism learns to assign appropriate weights to the queries, keys, and values, adapting to patterns and relationships in the training data. This process of weight adaptation and learning allows the model to improve its performance over time and generalize well to new unseen examples.

C. Vision Transformers

Based on the Transformer model a new approach called Vision Transformer (ViT) has been introduced to allow the application of Transformers to images [5]. The ViT model extends the Transformer architecture to the domain of computer vision, allowing for the processing and analysis of visual data. Convolutional Neural Networks [15] have traditionally been the dominant approach for image recognition tasks. Vision Transformers, however, offer an alternative solution by leveraging the self-attention mechanism to capture global dependencies and relationships within images.In the ViT model, the image is divided into smaller patches, which are regarded as “words" analogous to the tokens in natural language processing. These patches are then processed by the Transformer architecture, including the self-attention mechanism, to extract relevant features and perform image recognition.

ViT models have demonstrated remarkable performance in various image recognition benchmarks. Moreover, they show the ability to effectively capture long-range dependencies and context in images. The adaptation of the Transformer architecture to computer vision tasks has allowed several advancements in image recognition and comprehension.

III. Related Works

Studies have explored different approaches to enhance the performance of neural networks, including investigating the aggregation functions used by neurons [16]. It has been demonstrated that using neurons that operate in the complex domain instead of the real domain can lead to improvements in the efficiency and performance of models. These complex-valued neurons allow for more expressive representations and can capture richer information in certain applications.

The use of structures that enable learning the best aggregation functions has emerged as a promising strategy to improve the performance of these functions. Pellegrini et al. [17] present an approach that allows the model to learn and adapt the aggregation functions during training, rather than using fixed and predetermined functions. The flexibility provided by learning the aggregation functions is an important aspect of the potential use of these models in new machine learning architectures. By enabling the model to learn the optimal ways to combine and summarize information, it becomes possible to adapt the aggregation process according to the specific characteristics of the data and the problem at hand.

Also in this direction, Rodriguez et al. [18] introduce the modification of max pooling through (a-b)-grouping as an interesting approach that addresses the use of different aggregation functions in convolutional neural networks. They aim to improve the performance of traditional max pooling, which involves selecting the maximum value within a specific region of the image. The (a-b)-grouping is competitive compared to modern pooling operators such as mixed pooling, gated pooling, and attention pooling. These modifications aim to enhance the ability of convolutional networks to capture relevant information and preserve important features in the pooling layers.

In Transformers, aggregation functions are also of great importance. Despite the lack of research evaluating these functions, researchers have explored modifications to the attention mechanism to reduce computational costs. The original self-attention mechanism has a computational complexity proportional to $O(n^2)$, which can be expensive for large sequences. In this line of work, Wang et al. [19] have proposed modifications to the aggregation of multiple attention outputs in the self-attention mechanism. By rethinking the aggregation step, they were able to reduce the computational cost to $O(n)$, making it more efficient for processing long sequences.

Regarding ViTs, Zhang et al. [20] propose the Nested Transformers (NestT) model as an approach to simplifying the Transformer architecture for computer vision tasks. Instead of using a global Transformer across the entire image, NestT suggests nesting basic local Transformers in non-overlapping regions of the image and hierarchically aggregating them. This hierarchical approach allows for a significant reduction in the number of model parameters, resulting in a lighter and more efficient architecture. The study reported a 57% reduction in the number of parameters compared to traditional models.

Despite the related works mentioned above, to the best of our knowledge, no other work in the literature evaluate the effects of different aggregation functions on Transformers and ViT for classification. In the present work, we perform classification on two tasks with diverse spatial and temporal domains, aiming for a wide observation of the aggregation functions' role in this context.

IV. A New self-attention method based on FG-aggregation functions

To investigate the potential of incorporating novel aggregation functions in the self-attention mechanism shown in Eq. 3, we propose a novel approach that employs a combination of FG-functionals. This approach aims to leverage the capability to identify significant connections among elements in a sequence, resulting in more comprehensive and informative representations.

Therefore, we focus on changing the aggregation function between the tensors Q and K, which originally used a matrix multiplication. As a consequence, the original operation is based on a sum of multiplications (rows per column) which can be understood as an FG-functional based on F = sum and G = Prod. The new self-attention method based on the FG-aggregation function replaces this relation with a different combination of functions (see Table I) as stated in Eq. 4

To perform these new combinations, we choose to follow the same pattern as the set of initially proposed aggregation functions, which consists of having a T-conorm as the outer function and a T-norm as the inner function. With this in mind, we formulated our first set of aggregations, which consists of combining the maximum function with the minimum function, resulting in the following equation:

$$\textit{Self-Attention}(Q,K,V) = \textit{Softmax}\left(\frac{Su^{F,G}(Q,K^T)}{\sqrt{d_k}}\right)V. \quad (4)$$

Based on empirical observations, we realize that the aggregation function in Eq. 4 did not require the normalization element, allowing for its suppression. Hence, the normalization component was not employed in any of the proposed aggregation functions, resulting in a streamlined attention mechanism.

V. Experimental Setup

We perform experiments with the new set of self-attention mechanisms introduced in Section IV on both the Transformer and ViT models. The purpose of these tests is to evaluate the performance and effectiveness of different combinations of aggregation functions in each model. For doing so, we have selected a different classification task for each kind of model. To evaluate our methodology on the Transformer networks we apply it to time series classification, in the form of sign language recognition based on body posture keypoints. For the evaluation of the ViT setup, we tackle the single image classification task. Such diverse tasks, concerning their spatial and temporal characteristics, aim for a wider observation of the effects of the new aggregation functions on the self-attention mechanisms of the models.

In the experiments, we employ the original Transformer [3] and ViT architectures [5] as baselines and compare their performance with the respective versions modified with the new aggregation functions. To ensure a reliable experimental setup in which the effects of the different aggregation functions can be accurately measured and compared we set consistent hyperparameters, as shown in Tab. II, and fix a seed for weight initialization, allowing for reproducibility. The data in all experiments is split into sets for training and validation (80%), and testing (20%). Convergence is monitored through the validation error and training is ended using the early-stop strategy. Finally, the Transformer models were implemented in PyTorch and ViT networks in TensorFlow. Experiments ran at Google Colab and the code will be made publicly available.

TABLE II
Hyperparameters

Hyperparameter	Transformers	ViT
Embedding Dropout	0.1	0.1
Epochs	100	30
Batch Size	128	256
Learning Rate	5e-4	1e-3
Heads	4	8
Seed	1601	1601
Dimension Heads	256	64

VI. Experiments: Transformers applied to Sign Language Recognition

Videos of sign language words may be seen as time series, with each frame corresponding to a data point in the sequence. By human pose estimation methods [21], [22], keypoints that encode the body postures of the speaker frame-by-frame can be extracted from videos. Thus, from the original videos, this process derives correspondent time series in which the data points are now sets of 2D keypoints representing the position of the speaker's body parts at each frame of the video sequence.

In this context, a sign language word is analogous to a textual sentence, in a way that each set of keypoints from a frame is analogous to a textual word. Therefore, analogously to word embeddings (word2vec) [23], sign language words require keypoint embedding to be classified with Transformer networks. This approach aims to reduce the dimensionality of the data and formulate an embedding representation. The use of keypoints embedding helps to capture the essential characteristics of the data and provides a more compact and informative representation, facilitating processing and analysis by the Transformers model.

A. Dataset: ASL Fingerspelling Recognition

The data in the ASL Fingerspelling Recognition dataset was recently released in a competition held by Google on the Kaggle platform. The goal is to recognize/classify ASL words into 64 different categories. Each word appears multiple times in the dataset, as it is performed by several subjects on videos. The subjects' body poses are estimated offline for each video and each of these words ends up being composed of 128 keypoints per video frame that correspond to the body postures of the subject along the sequence. Each keypoint is defined by 2D (x, y) coordinates. A sample frame of a dataset sequence is illustrated in Figure 1. The dataset consists of a total of 61,955 samples. Out of this set, 54,719 samples are used for training and validation, while 7,236 examples are allocated for testing. The train and validation sets are further divided, with 10,943 samples for validation and the remaining 43,776 for training.

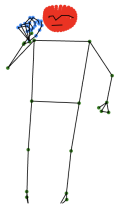

Fig. 1. The set of keypoints correspondent the body pose of a subject performing an ASL word in the Google ASL Fingerspelling Recognition dataset. Keypoints represent face, hands, and body postures in a video frame.

B. Transformer Results on ASL Fingerspelling Recognition

In Figure 2, we are monitoring the increase in accuracy as we progress through epochs. We can observe that the functions behave stably, with two functions standing out for their better performance. The first one is *Vanilla* (our baseline), which refers to the Transformers model found in the literature. The second one is the sum of minimums. Both show promising and superior results in terms of accuracy.

https://www.kaggle.com/

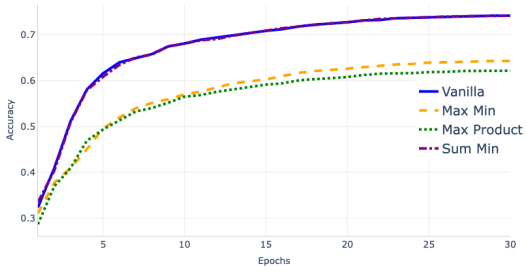

Fig. 2. Accuracy in Validation data

Looking at Figure 3, we can observe the rate of error reduction as we progress through epochs. We can see that the behavior is stable, and the two previous functions continue to have the lowest error rates, remaining similar throughout the validation process. This indicates that these aggregation functions are contributing to the error reduction and good performance of the model.

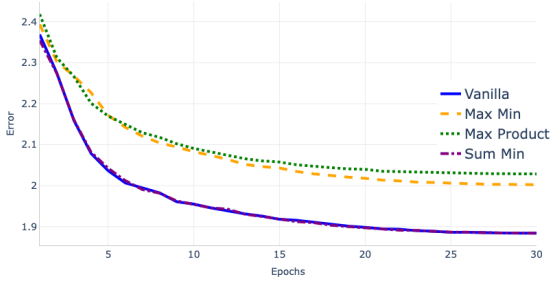

Fig. 3. Loss in Validation data

Lastly, we can observe from Table III that the Transformers with the vanilla function perform better in almost all cases, with lower error rates. However, the sum of minimums functions equals error in terms of error while achieving a higher accuracy metric for the training data. This indicates that the maximum of minimums function may be more effective in capturing relevant information for the correct classification of the training data, although it results in similar error rates.

TABLE III
BEST RESULT IN TRANSFORMERS

Aggregation Functions	Accuracy Train	Accuracy Validation	Loss Train	Loss Validation
Vanilla	80.76	74.13	1.79	1.88
Max Min	72.93	64.25	1.88	2.00
Max Product	72.26	62.13	1.89	2.02
Sum Min	80.82	74.09	1.79	1.88

VII. EXPERIMENTS: VIT APPLIED TO IMAGE CLASSIFICATION

A. Dataset: CIFAR-10

The *CIFAR-10* [24] is an image classification dataset widely used in the field of computer vision and machine learning. It consists of 60,000 color images with a resolution of 32×32 pixels, divided into 10 different classes. Each class represents a specific type of object.*CIFAR-10* [24] is considered a challenging dataset because the images are relatively small and low-resolution, which complicates the classification task. Additionally, the presence of objects in different poses, lighting conditions, and backgrounds adds further variability to the dataset.

The dataset is split into training and testing sets. The training set contains 50,000 images, with 5,000 images for each class, while the testing set contains the remaining 10,000 images, with 1,000 images for each class. This division ensures that the models are evaluated on unseen data during training, allowing for an objective assessment of their performance.

B. ViT Results on CIFAR-10

The data results below refer to the experiments conducted by ViT on the validation process of the CIFAR-10. In Figure 4, we are monitoring the increase in accuracy as we progress through epochs. We can observe that the functions behave remarkably, with the *vanilla* function standing out with better results. However, we can also notice that the maximum of minimums and maximum of product functions are very close to the original function.

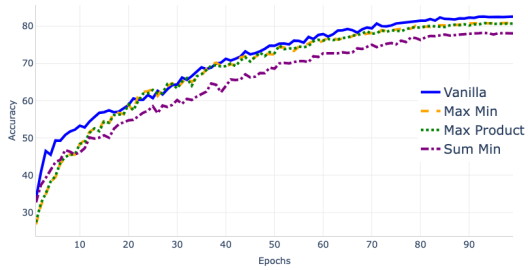

Fig. 4. Accuracy in Validation data

In Figure 5 error reduction, the functions follow a similar behavior as the accuracy functions. Therefore, the vanilla function ends up having the lowest error, while the maximum of minimums and a maximum of product functions show higher error performance. This indicates that the *vanilla* function is more efficient in reducing error and providing more accurate results compared to the other tested aggregation functions.

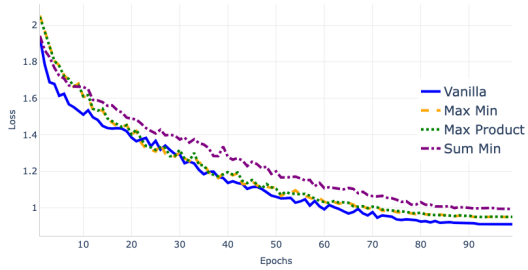

Fig. 5. Loss in Validation data

All these results are reflected in Table IV, where we can observe that the existing Transformers model in the literature shows the best results in all cases. This indicates that the standard model, dubbed here as *vanilla*, has proven to be more effective in handling the CIFAR-10 dataset and achieving

superior performance in terms of evaluation metrics such as accuracy and error.

TABLE IV
BEST RESULT IN VISION TRANSFORMERS

Aggregation Functions	Accuracy Train	Accuracy Validation	Loss Train	Loss Validation
Vanilla	83.74	82.55	0.86	0.91
Max Min	80.76	80.87	0.92	0.94
Max Product	80.65	80.69	0.93	0.94
Sum Min	76.23	78.07	1.02	0.99

VIII. CONCLUSION

A crucial point in Transformers is related to the self-attention mechanism and its default aggregation function based on matrix multiplication.

Having this in mind, this study proposed a modification of this mechanism by considering different aggregation functions known as FG-functionals.

When analyzing the results, it is possible to observe that the aggregation function that seems to perform better is the sum, which has a logical behavior similar to an OR. Regarding the inner function, the product demonstrates better performance in the tests, with a behavior similar to an AND. However, it is also worth mentioning that the minimum function yields considerably similar results, although we cannot establish a direct relationship with a specific logical function in this case.

Based on these results, it appears that the attention mechanism benefits from the combination of an aggregation function similar to an OR (such as the sum) and an inner function similar to an AND (such as the product). This observation can provide an interesting direction for future investigations, exploring sets of functions that follow this approach. These combinations of aggregation functions may potentially improve the performance of the attention mechanism in different tasks and contexts.

It can be observed that in both classification tasks, the *vanilla* function exhibits superior performance in solving the proposed problems. However, we also observed that the performance of each function may vary depending on the specific problem. This is evident in the training of Transformers on the ASL dataset, wherein the sum of minimum functions yielded comparable outcomes to the classical function and outperformed it in terms of validation accuracy. This observation presents a diverse range of possibilities for utilizing alternative aggregation functions in Transformers and modifying them according to the particular matter at hand. In future works, we intend to explore other aggregation functions, as well as modify how we organize the categories of functions within the attention mechanism. This may further improve the performance and flexibility of Transformers and ViTs models.

ACKNOWLEDGMENT

This work is supported by FAPERGS (ARD-21/2551-0000678-1, 23/2551-0000126-8) and CNPq (150160/2023-2).

REFERENCES

[1] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2Nd Edition)*. Wiley-Interscience, 2000.

[2] P. Dhruv and S. Naskar, "Image classification using convolutional neural network (CNN) and recurrent neural network (RNN): a review," *Machine Learning and Information Processing: Proceedings of ICMLIP 2019*, pp. 367–381, 2020.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[4] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, 2019.

[5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

[6] M. Kurzynski and M. Krysmann, "Fuzzy inference methods applied to the learning competence measure in dynamic classifier selection," in *2014 27th SIBGRAPI Conference on Graphics, Patterns and Images*. IEEE, 2014, pp. 180–187.

[7] E. Nikookar, E. Naderi, and A. Rahnavard, "Cervical cancer prediction by merging features of different colposcopic images and using ensemble classifier," *Journal of Medical Signals and Sensors*, vol. 11, no. 2, p. 67, 2021.

[8] M. Grabisch, J. Marichal, R. Mesiar, and E. Pap, *Aggregation Functions*. Cambridge: Cambridge University Press, 2009.

[9] G. Beliakov, A. Pradera, T. Calvo *et al.*, *Aggregation functions: A guide for practitioners*. Springer, 2007, vol. 221.

[10] E. P. Klement, R. Mesiar, and E. Pap, *Triangular Norms*. Dordrecht: Kluwer Academic Publisher, 2000.

[11] T. Murofushi, M. Sugeno, and M. Machida, "Non-monotonic fuzzy measures and the Choquet integral," *Fuzzy Sets and Systems*, vol. 64, no. 1, pp. 73 – 86, 1994.

[12] M. Grabisch, *Set Functions, Games and Capacities in Decision Making*, ser. Theory and Decision Library C. Cham: Springer, 2016, vol. 46.

[13] F. Bardozzo, B. De La Osa, L'ubomíra Horanská, J. Fumanal-Idocin, M. delli Priscoli, L. Troiano, R. Tagliaferri, J. Fernandez, and H. Bustince, "Sugeno integral generalization applied to improve adaptive image binarization," *Information Fusion*, vol. 68, pp. 37–45, 2021.

[14] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[16] B. K. Tripathi and P. K. Kalra, "The novel aggregation function-based neuron models in complex domain," *Soft Computing*, vol. 14, pp. 1069–1081, 2010.

[17] G. Pellegrini, A. Tibo, P. Frasconi, A. Passerini, and M. Jaeger, "Learning aggregation functions," *arXiv preprint arXiv:2012.08482*, 2020.

[18] I. Rodriguez-Martinez, T. da Cruz Asmus, G. P. Dimuro, F. Herrera, Z. Takáč, and H. Bustince, "Generalizing max pooling via (a, b)-grouping functions for convolutional neural networks," *Information Fusion*, p. 101893, 2023.

[19] S. Wang, B. Z. Li, M. Khabsa, H. Fang, and H. Ma, "Linformer: Self-attention with linear complexity," *arXiv preprint arXiv:2006.04768*, 2020.

[20] Z. Zhang, H. Zhang, L. Zhao, T. Chen, and T. Pfister, "Aggregating nested transformers," *arXiv preprint arXiv:2105.12723*, vol. 2, no. 3, p. 5, 2021.

[21] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[22] R. De Bem, A. Arnab, S. Golodetz, M. Sapienza, and P. Torr, "Deep fully-connected part-based models for human pose estimation," in *Asian conference on machine learning*. PMLR, 2018, pp. 327–342.

[23] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

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