Performance Analysis of Convolutional Neural Networks on GPU, TPU and CPU Platforms Applied to Facial Expression Recognition

1st Leandro Perius Heck
Regional University of Northwest of Rio Grande do Sul (UNIJUI)
Santa Rosa, RS – Brazil
leandro.h@sou.unijui.edu.br

2st Cristiano Alex Künas
Regional University of Northwest of Rio Grande do Sul (UNIJUI)
Santa Rosa, RS – Brazil
cristiano.kunas@sou.unijui.edu.br

3st Edson Luiz Padoin
Regional University of Northwest of Rio Grande do Sul (UNIJUI)
Ijuí, RS – Brazil
padoin@unijui.edu.br

***Abstract*—Considering the growing interest in the field of human-computer interaction and that this iteration has become something more and more natural and social, along with the increase in computational capacity provided by Central Process Unit (CPUs), Graphics Processing Unit (GPUs) and Tensor Processing Units (TPUs), which are application-specific integrated circuits developed to accelerate machine learning workloads. Areas such as recognition of emotions have proven to be of great interest and relevance to the scientific community. However, even with several works performed, detecting and recognizing emotions computationally and with the same ease that humans recognize is still a relevant problem to be explored. In order to explore this theme, this work has adopted the use of Convolutional Neural Networks (CNN) in the recognition of emotions in humans based on facial expressions. The results showed that with the training of an Artificial Neural Network (ANN) in GPUs, it was possible to reduce computational time by up to 90% and increase accuracy to 65%**

***Index Terms*—Artificial Intelligence, GPU, Artificial Neural Networks, Convolutional Neural Networks.**

I. Introduction

Nowadays there is a growing interest in improving the interaction between humans and computers. For an effective intelligent human-computer interface to be achieved, the computer must naturally relate to the user, similar to the way humans interact [1]. One of the ways to find and promote this kind of integration is with the help of Affective Computing, Artificial Intelligence, Machine Learning and Deep Learning.

Through these approaches it is possible to use computers to recognize, model and express emotions and how they can respond to them. According to Leão et al. [1], Making a machine recognize, model and express emotions is not a simple task. When human beings interact with each other, much of this interaction is based on verbal language and the use of body language through gestures and facial expressions that carry and transmit the emotions of the interlocutors.

In recent decades the scientific community has been taking a growing interest in the recognition of emotions. There are several ways to express human emotions, and these have been studied over the years, and the following sources of data have been explored, such as texts, sending emoticons, voice and facial expressions.

In addition to facial expressions playing an important role in the relationship between people, that it, it provides information that is relevant to an individual's emotions and intentions during dialogue or communication. The recognition of such emotions allows us to create numerous applications of human-computer interaction and has attracted the attention of the scientific community.

As facial recognition technology progresses, new ways of using it in our daily lives also emerge, such as to make payments, see who's watching in class, wake up a sleepy driver, personalized menus, and targeted marketing to the trade. Imagine a store being able to organize all its products, according to the reaction of its customers. This way, you can leave the products that most attract the attention of individuals with a greater prominence in the store or window, thus calling even more customers to the store.

This work aims to realize the recognition of the 6 emotions considered as basic by Ekman [2], which are happiness, sadness, fear, surprise, anger, disgust, plus the neutral emotion. For this task, images of human facial expressions will be used, and for the extraction of the characteristics, computational vision and ANN techniques will be used. But the main objective is to perform the analysis and comparison of CNN times on the different CPU, GPU and TPU architectures. Besides observing the performance of ANN in the different architectures.

II. Related Work

In Bartlett's [3] work a study is presented with the objective of automatically locating faces in a video stream and encoding visual expression in a dynamic way. The authors discuss that face-to-face communication is a real-time operation with a time scale of 40 milliseconds. The system is able to detect seven expressions: neutral, anger, disgust, fear, joy, sadness and surprise, with a differential from other works because it operates in real time. This system has been trained and tested using the CohnKanade AU-Coded Expression Database. This

database contains the facial record of 210 adults between 18 and 50 years of age, 69% female and 31% male, and 81% Euro-American, 13% African-American and 6% other ethnic groups. The experiments performed compared the performance of the automatic detection approach recognition with the manual detection approach, finding no significant difference between them. The system showed a level of accuracy of 93% of recognition, in the selection of one of the 7 options of facial expressions.

The work developed by Tang and Huang [4], aims to recognize the six basic and universal emotions through facial expressions using 3D geometry. This approach extracts characteristics that are invariant under the effect of illumination or posture, characteristics that, the authors consider as obstacles for facial recognition in 2D images. In this work, the BU-3DFE database was used as a training and testing basis. This database is composed of 100 individuals, 60% female and 40% male, with ethnic variety, including white, black, East Asian, Middle East Asian, Latin American among others. For this work it was found that the approach produced an absolute increase of 3.5% in the average recognition rate. This approach produced an average accuracy of 87.1%, the highest rate being 99.2% for the recognition of the facial expression of surprise.

In the work developed by Amin et al. [5], an ANN was elaborated that has the purpose of recognizing emotions through facial expressions using the technique of deep learning. According to the author, the use of convolutional neural networks, in the approach to the identification of emotions, reaches an average precision of 60%. The Facial Expression Recognition 2013 (FER-2013) database was used for the development of the work. This database was developed by Pierre Luc Carrier and Aaron Courville, and has a total of 35887 images, with dimensions of 48x48 pixels. The samples have the six facial expressions of Ekman, added from the neutral expression. The samples in this database are distributed in: 7215 images of joy; 436 of disgust; 4097 of fear; 4965 of neutral; 3995 of anger; 3171 of surprise; and 4830 of sadness. The work presented good results, reaching an average accuracy of 61.05% for the classification of the seven emotions. Analyzing the results, the authors found that the emotion of joy has the highest recognition rate. However, the model cannot perfectly classify the emotions of fear and sadness.

Different from the works presented, our proposal seeks to develop an CNN that recognizes facial expressions from images, and identifies the emotion present in the image. Besides proposing the development of an CNN, the performance and training time in the CPU, GPU and TPU architectures will be analyzed. To obtain a satisfactory result in the shortest possible time.

III. Implementation of CNN

For the implementation of the proposed CNN the development environment Google Colab and the programming language Python were used. Google Colab is a free cloud service hosted by Google for Machine Learning and IA, with free GPU accelerators, pre-installed libraries, built based on Jupyter Notebook, supports bash commands and stores the notebooks in the Drive itself. The main libraries used are TensorFlow 2.0 and Keras, which are focused on deep machine learning. Keras is the most used framework in the area for its easiness. In TensorFlow 2.0, Keras has been ”embedded” into TensorFlow through the module tf.keras. CNN also uses the tools: OpenCV, Scikit-Learn, NumPy, Pandas, Matplotlib.

The database used for this work was *FER-2013* Facial Expression Recognition 2013). This database brings together an open-source data set created by Pierre- Luc Carrier and Aaron Couville, then publicly shared for a Kaggle competition[1]. This data set consists of 35,887 gray tone facial images subdivided into seven emotions, with a dimension of 48 x 48 pixels. The faces were registered automatically so that the face is more or less centered and occupies approximately the same amount of space in each image. The task is to categorize each face based on the emotion shown from the facial expression into one of seven categories, which contain the description of the emotion and its descriptor already defined by the database. The file *FER2013.csv* has two columns that are ”emotions” and ”pixels”. The ”emotion” column contains a numerical code ranging from zero to six. The ”pixels” column contains a sequence of numbers that represents the grayscale values for each pixel of the image.

To start the development of CNN it is necessary to apply some transformations in the information in the file *FER2013.csv*. When reading the *csv* file, the information is in string format, it is necessary to convert it to an array, for this is used the function *tolist()*, which converts the database data to a list. In the elaboration, it was opted to implement a convolutional neural network, which has been applied successfully in the processing and analysis of digital images [6]. For this reason there are basically four steps to be followed for its development [7]:

i) **Step 1 Convolution Operator:** This step works as filters that see small frames and goes through the whole image capturing its most relevant features. For example, on a 48x48 image and a filter that covers a 3x3 area of the image with 1 jump movement, the filter will scroll the entire image. In the 3x3 filter it is taken point by point and multiplication is performed by a feature detector, this one defined by the library that is used, forming at the end a feature map or feature map.

ii) **Step 2 Pooling:** In this step has the function of simplifying the information of the previous layer, in this case, the map of characteristics. As in the convolution, an area unit is defined, usually using a 2x2 matrix, to pass through the entire output of the previous layer. The unit is responsible for summarizing the information of that area in a single value. To select this element it is necessary to choose the way the summarization will be made. The most used method is MaxPooling, which returns the largest number of the unit and passes this value to the output. This data summarization serves

[1]Kaggle is a web platform for Data Science competitions. It is currently owned by Google

to reduce the amount of weights to be learned by the Neural Network, and also avoid overfitting.

iii) **Step 3 Flattening:** This step receives the matrix created in the Pooling step as input, and basically operates a transformation in the image matrix, changing its format to an array, that is, it converts the matrix to a characteristics vector.

iv) **Step 4 Neural Network:** At this stage it receives the data from the Flattening layer and submits the characteristics vector as input for Neural Network training, which is a computational model based on the human central nervous system. In this way, you will be able to recognize patterns in a large mass of data in order to classify them in some category.

IV. Results

For the CNN tests, 10 repetitions were performed in order to obtain the average execution times and Speedup, both in CPU, TPU and the four models of GPUs available in Google Colab, which provides the NVIDIA Tesla K80, T4, P4 and P100 for free.

The best Speedup achieved by CNN compared to CPU is shown in Figure 1, where the Tesla P100 obtained the best result. The Speedup of the algorithm running on GPU shows a 10.71 gain over the CPU. Thus reducing the jumping runtime from 3950.44 seconds to 368.81 seconds, presenting a gain of 90.66%. The result of the Speedup of the other GPUs was also satisfactory. The Speedup of the algorithm running on the Tesla K80 GPU showed a gain of 3.51 times over the CPU. Thus reducing the runtime from 3950.44 seconds to 1125.37 seconds, obtaining a gain of 71.21%. The Tesla P4 showed a 7.16 gain over the CPU. Decreasing the runtime from 3950.44 seconds to 551.60 seconds, achieving a 86.04% gain. And finally Tesla T4 pointed to a 9.45 gain compared to the CPU. Limiting the runtime from 3950.44 seconds to 417.98, earning 89.42%. The comparison of TPU use with the CPU had a gain of 2.04 times, decreasing the execution time from 3950.44 seconds to 1935.49 seconds, presenting a gain of 50.01%.

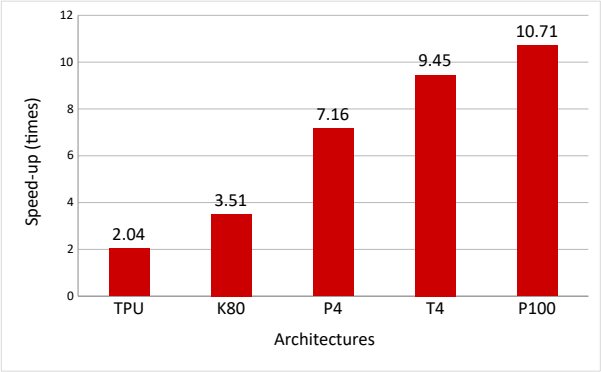

Fig. 1. Speedup of GPU and TPU architectures compared to CPU.

Comparisons were also made between GPUs, for analysis of the gain obtained. The comparison of the Tesla P4 with the Tesla K80, showed a gain of 2.04 times, reducing the execution time from 3950.44 seconds to 551.60 seconds, showing a gain of 86.04%. Tesla T4 was compared to P4 and K80. Tesla T4 earned 1.31 times compared to P4, decreasing the time from 551.60 seconds to 417.98 seconds, making a gain of 24.23%. Compared to K80, T4 earned 2.69 times, limiting the runtime from 3950.44 seconds to 417.98 seconds, earning 89.42%. Finally, the analysis was performed comparing the Tesla P100 with the other GPUs. In the comparison with the T4, a 1.13 times gain was obtained, reducing the execution time from 417.98 seconds to 368.81 seconds, reaching a gain of 11.77%. The conference with P4, came to a 1.49 times gain, reducing the time from 551.60 seconds to 368.81 seconds, coming to a 33.14 times gain. In the confrontation with K80 it reached a 3.05 times gain, decreasing the execution time of 1125.37 seconds to 368.81, presenting a gain of 67.23%.

The comparison of Speedup between GPUs and the Google Colab TPU, illustrated in Figure 2, was also performed. The Speedup of the algorithm running on the Tesla K80 GPU showed a 1.71 times gain over the TPU, reducing the runtime from 1935.49 seconds to 1125.37 seconds, obtaining a 41.86% gain. The Tesla P4 showed a 3.50 gain over the TPU, decreasing the execution time from 1935.49 seconds to 551.60 seconds, reaching a gain of 51.51%. Tesla T4 pointed to a 4.63 gain compared to TPU, limiting the execution time from 1935.49 seconds to 417.98, with a gain of 78.41%. And finally the Tesla P100 GPU showed a 5.24 gain over the TPU, reducing the runtime from 1935.49 seconds to 368.81 seconds, obtaining a 80.95% gain.

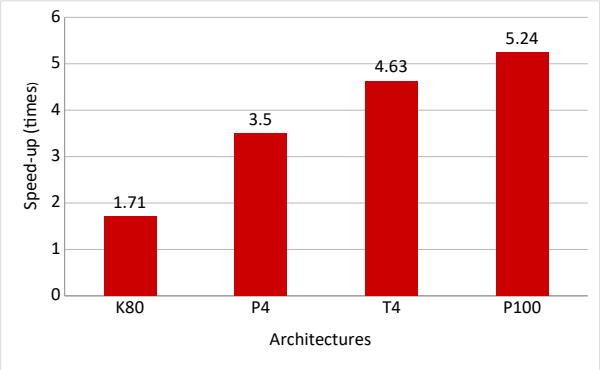

Fig. 2. GPU architecture Speedup compared to TPU .

For the development of CNN, a total number of filters (num_features) was defined as 32, the division of labor, in this case, to perform the adjustments of the RNA weights (batch_size) as 16 and a total of 100 seasons, for the training. A stop metric (EarlyStopping) was also defined, in the period of 15 seasons. In the convolution layer, the ELU (Exponential Linear Unit) activator was used, Dropout was 20%.

The tests were performed using the four video cards available in Google Colab. The accuracy values were compared with the execution performed in CPU and TPU, Figure 3. The presented accuracy values are obtained from the test base and not from the RNA validation base. The GPU that presented the best accuracy was the Tesla K80, obtaining 65.67%. The Tesla T4 obtained an accuracy of 65.39, while the other GPUs obtained a slightly lower accuracy when compared with the

CPU and TPU. The Tesla P4 had 64.05 accuracy and Tesla P100 reached 64.92, TPU had an accuracy of 65.25, while the CPU had an accuracy of 65.17.

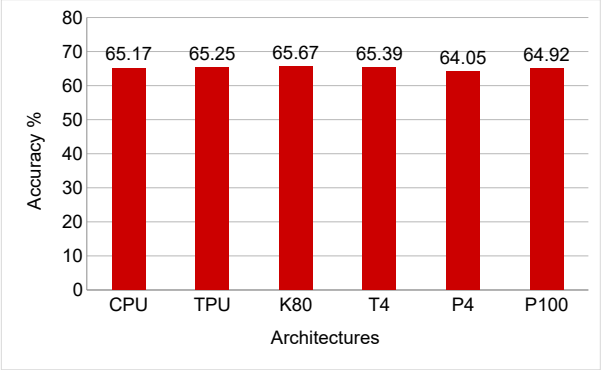

Fig. 3. Result of the accuracy obtained in each architecture.

V. Tests Performed

To perform the tests, before predicting what emotion is present in the image, it is necessary to load an image and recognize the faces. For the recognition was used the OpenCV library, which already has a pre-trained model with the type of attribute to be identified, in this case the faces. The file to be used is called haarcascade_frontalface_default.xml.

With the image and the classifier initialized, it is already possible to perform face identification on the images. However, the classifier used only accepts grayscale images, so it is necessary to convert the original image to grayscale. Having converted the image to grayscale, it is now possible to detect faces and work with the image to recognize the emotion predicted in the image.

CNN tests were performed on all architectures, CPU, GPU and TPU, but to illustrate the test result, Figure 4 shows the result obtained by the architecture that obtained the best precision in the training phase, in this case, the tests were performed on the Tesla K80 GPU.

Fig. 4. Prediction Performed on CNN.

In the comparison made in the tests on all the architectures, CNN produced a confusion in the identification of the emotion of fear and surprise. The fact that confusion occurs between the emotion of fear and surprise is that both have similar facial characteristics, which can produce a confusion at the time of their identification. The similarities between the two emotions are raised eyebrows and open jaw [8] [9].

VI. Conclusions and Future Works

This work approached the development of a convolutional neural network for the recognition of emotions through facial expressions, performing numerous tests to analyze and evaluate the performance of the application running on CPU, and GPU architectures. With the new implementation it was possible to increase the accuracy of RNA reaching an accuracy of up to 65.67%. Regarding computational time, the version developed for Tesla P100 GPU, achieved gains, reducing the execution time by up to 10.71 times compared to the CPU execution.

As future works it is intended to apply the solution developed in other databases to validate their behavior. A second initiative would be to develop the neural network to make the recognition of emotions in real time of people in videos. Also modify the algorithm of the Artificial Neural Network so that you can use the Google Cloud TPU execution environment, which has TPUs that have a better archiving, than the free TPU provided in Google Colab, and also analyze the influence of other hyperparameters, such as learning rate, Dropout and Activation Function.

References

[1] L. P. Leão, J. S. Bezerra, L. N. Matos, and M. A. S. N. Nunes, "Detecção de expressões faciais: uma abordagem baseada em análise do fluxo óptico," *Revista GEINTEC-Gestão, Inovação e Tecnologias*, vol. 2, no. 5, pp. 472–489, 2012.

[2] P. Ekman, "Cross-cultural studies of facial expression," *Darwin and facial expression: A century of research in review*, vol. 169222, no. 1, 1973.

[3] M. S. Bartlett, G. Littlewort, I. Fasel, and J. R. Movellan, "Real time face detection and facial expression recognition: development and applications to human computer interaction." in *2003 Conference on computer vision and pattern recognition workshop*, vol. 5. IEEE, 2003, pp. 53–53.

[4] H. Tang and T. S. Huang, "3d facial expression recognition based on properties of line segments connecting facial feature points," in *2008 8th IEEE International Conference on Automatic Face & Gesture Recognition*. IEEE, 2008, pp. 1–6.

[5] D. Amin, P. Chase, and K. Sinha, "Touchy feely: An emotion recognition challenge," *Palo alto: Stanford*, 2017.

[6] R. C. Gonzalez and R. E. Woods, *Processamento de imagens digitais*. Editora Blucher, 2000.

[7] J. D. P. Massucatto, "Aplicação de conceitos de redes neurais convolucionais na classificação de imagens de folhas," B.S. thesis, Universidade Tecnológica Federal do Paraná, 2018.

[8] P. Ekman and W. V. Friesen, *Unmasking the face: A guide to recognizing emotions from facial clues*. Ishk, 2003.

[9] P. Ekman, "A linguagem das emoções," *São Paulo: Lua de Papel*, 2011.