Performance Analysis of Artificial Neural Networks in CPU and GPU Platforms Applied in Text Sentiment Analysis

1st Cristiano Alex Künas
Regional University of Northwest of Rio Grande do Sul (UNIJUI)
Santa Rosa, RS – Brazil
cristiano.kunas@sou.unijui.edu.br

2nd Leandro Perius Heck
Regional University of Northwest of Rio Grande do Sul (UNIJUI)
Santa Rosa, RS – Brazil
leandro.h@sou.unijui.edu.br

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

***Abstract*—Nowadays, a lot of information with all kinds of content circulates on the Internet. Analyzing the sentiments exposed can help in understanding what people are talking about a particular company, brand, event or even other people, working as a way to get feedback. In this work, we present an Artificial Neural Network model for sentiment analysis in English language sentences. The Long Short-Term Memory Recursive Artificial Neural Network was implemented for the training of the sentiment analysis model. With the application of ANN developed over a public database with 50,000 movie records using GPU it was possible to reduce the training time of RNAs by up to 91.8% and increase the accuracy to 87.7%.**

***Index Terms*—Sentiment Analysis, Artificial Neural Network, Long Short-Term Memory, Natural Language Processing, GPU**

I. Introduction

Countless information circulates on the Internet through websites, blogs and social networks. The present content ranges from analysis and comments on films to conversations and experiences of its users. In this sense, using technological resources we can store, retrieve and analyze a huge amount of this data efficiently. One thing is that this data is not structured in an understandable way for a single computer system.

Opinions are so important for almost all human activities that, whenever we need to make a decision, we want to hear the opinions of others [1].They are the main influencers of our behavior. This informal or formal opinion, depending on where it was posted, is extremely important, because it will reflect the uncensored feeling of the user. The great growth of social media, such as blogs and social networks has awakened a lot of interest, especially the Sentiment Analysis (SA). However, this growth makes it impossible for any human being to be aware of all the content that interests them in a timely manner.

Individuals and organizations are increasingly using the content of these media for decision making. But, computer systems still find it very difficult to understand how customers feel about the products and services of a certain company. In some situations, only a few points of the text are relevant: about who is spoken about, and whether what is spoken is good or bad [2]. Artificial Intelligence (AI) can be applied to facilitate the understanding of this data, so that we can use a sentence in natural language as input and extract a set of data in output. In this way, the use of an Artificial Neural Network (ANN) can help in the automatic extraction of the feeling or sensation of sentences.

Among the different sub-areas of study that the field of Sentiment Analysis presents, the task most present in the literature and that we will approach in this work is the analysis of polarity of a document, which aims to classify texts on a scale between positive and negative, since this area of research is of great relevance for consumers and organizations. Therefore, the present work aims at: i) creating a recurrent neural network of the Long Short-Term Memory (LSTM) type; ii) performing its training using a public database; and iii) applying it to the recognition of sentiments expressed in texts and/or posts.

The work is organized as follows. Section II discusses the related work. Section III presents the methodology used in the implementation and the execution environment used in the tests. Results are discussed in Section IV, followed by conclusions and perspectives of future work.

II. Related Work

Several works present studies on Sentiment Analysis. This concept was first introduced in 2003 where techniques were used in opinion mining in history. Algorithms based on machine learning were used in [3] to classify the positioning of a person in relation to an object, idea or posture. In this work, people's positioning was classified and applied to political debates. They used 104 debates of double positioning from convinceme.net for 14 different themes and tried to identify the opinion of those involved. The main objective was to determine the potential contribution of dialogue characteristics in classifying debates. They used Support Vector Machines (SVM), Naive Bayes (NB) and a rules-based classifier for classification purposes and were able to specify the choice of the debate side.

In [4], sets of decision trees are used to do sentiment analysis in movie commentaries. The authors set the number of trees to 100 as a reasonable default value. However, for the extraction of attributes from the text, they used deep learning methods. The methodology chosen was Word2Vec which allows the capture of semantic characteristics of words. The vectors obtained by the attribute extraction process were clustered by k-means. Each cluster, from a total of 2000, had an average of 5 words. These were selected for their proximity in vector space. The clusters were used as entries for the classifier. At work, the IMDb database was used. The result of the experiment showed that the approach using the extraction of attributes by deep learning was significantly better than the one using bag-of-words with 5000 entries.

Deep neural networks were used in [5] for sentiment analysis in reviews of electronic products, films and hotels. The authors created a classification framework that uses 3 different methods for attribute extraction: based on frequency, context and labeling of parts of speech. Each method feeds a neural subnet that reduces the dimensionality of the attribute space. The outputs of these subnets feed the main network which is responsible for feeling analysis. The structure formed by the subnets and the main net is called a deep hierarchical neural net. The work compared the performance of the structure proposed by the authors with machine learning methods (NB and SVM). The analysis found that the accuracy of the deep neural network model was superior to NB and SVM for a large number of samples in the training set (over 200000). However, the accuracy of the proposed model decreased for cases where the dimensionality of the inputs is large and the number of samples for training is reduced. The work also showed that regardless of the context of the comments the deep learning approach presents increasing precision with the number of samples in the training. The databases used were amazon.com, IMDb and TripAdvisor.

Sentiment analysis at the sentence level was studied in [6]. The authors created a Chinese language sentence database called Recursive Neural Deep Model (RNDM). Unlike other databases that have a global feeling for each sentence, this has feeling ratings for each sentence within each sentence and for each word within each sentence. The work shows the performance of this system compared to others with machine learning methods (NB, SVM, and Logistics Regression).

These works contemplate important contributions in the use of Artificial Neural Networks for classification of feelings. Following this premise, our proposal is to evaluate the use of a Long Short-Term Memory Recurrent RNA, aiming to increase performance in its training through the use of GPU architecture.

III. Methodology

The ANN implementation of this work is done using Python programming language [7]. This was chosen because of its simplicity and the large number of well documented modules that help from preprocessing data to the application of machine learning models. The main modules used are: Keras [8], TensorFlow [9] and Scikit-Learn [10]. Keras is used to model RNA. This is a high level API, developed with user focus, to allow fast experimentation. It supports several network configurations and works with both CPU and GPU. TensorFlow is Keras standard back-end numerical computing library. This is an open source software library for numerical computing using data flow graphs. It is flexible and allows fast and easy interfacing with CUDA (Compute Unified Device Architecture) [11]. To divide the data used in this work, in training and testing, Scikit-Learn is used. Scikit-learn is a Python module that integrates several state-of-the-art machine learning algorithms for supervised and unsupervised medium scale problems. This package is dedicated to bringing machine learning to non-specialists who use a high-level language for general purposes. Emphasis is placed on ease of use, performance, documentation and API consistency [10].

The database used in the experiments was developed in the work of [12]. This database has a collection of $50,000$ IMDb evaluations written in English. The criticisms are well balanced in 2 classes: positive and negative. The preprocessing of these data is done in stages with the objective of removing noise present in the sentences. They are: i) special characters and punctuation marks are removed; ii) all words are written in lowercase letters; iii) stopwords are removed - these are words that do not add much meaning to the text. They are usually articles, conjunctions and prepositions; iv) sequences are limited to a fixed size (300 words);

The creation of the ANN model is defined as a sequence of layers. The Model class of the Keras library functional API is used in the development of the data model. The compilation of the model configures the learning process. It defines the optimizer (adam), the loss function (binary_crossentropy) and the metrics (accuracy). To train the model, the data are divided into batches (batch_size), iterating repeatedly throughout the data set for a certain number of seasons. The model is evaluated using the *evaluate()* function. This function will generate a prediction for each input and output and will collect scores, take an average over the loss value and accuracy. Figure 1 shows the ANN architecture used in this work, simulated in TensorBoard[1].

The execution environment is composed of a device with an Intel Core i7-9750 processor with 6 cores (12 threads) of 2.60 GHz frequency. This equipment has 16GB of DDR4 RAM, NVIDIA GeForce GTX RTx 2060 GPU with 6GB of GDDR6 and 1920 CUDA cores, used the Linux Ubuntu 18.04.3 LTS operating system with kernel version 5.0.0-37. The NVIDIA CUDA Compiler version used was 10.0.130.

IV. Results

Figure 2 shows the accuracy (y-axis) achieved from the training dataset (40000 records), with different batch sizes, in both CPU and GPU executions. It is noticed that as the number of instances for each gradient update increases, the accuracy of the model is reduced, that is, less ANN weights are updated and consequently the loss rate increases.

[1] https://www.tensorflow.org/tensorboard

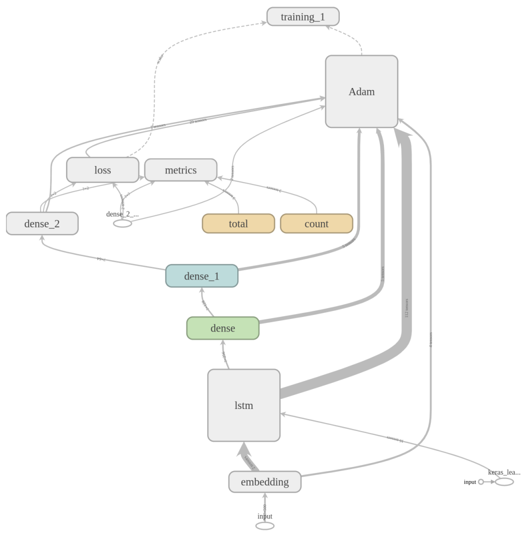

Fig. 1. Simulated ANN architecture in TensorBoard

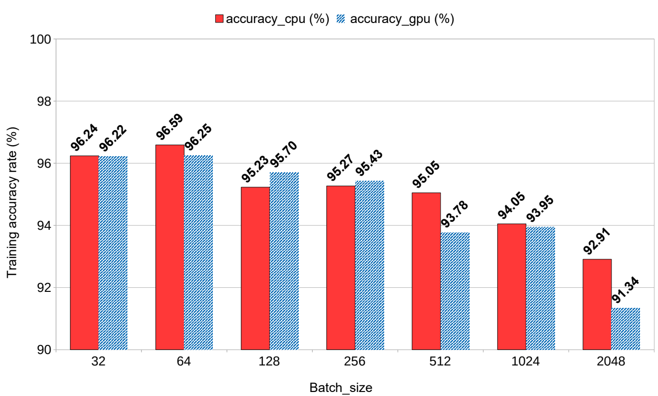

Fig. 2. Metrics for training data of CPU and GPU executions.

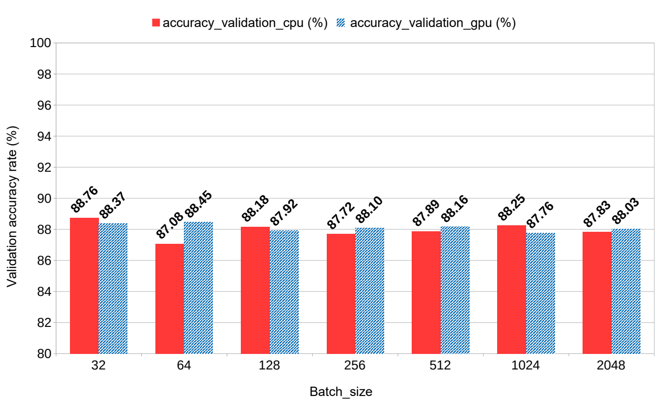

Fig. 3. Metrics for validation data of CPU and GPU executions.

Figure 2 shows the accuracy (y-axis) obtained from the validation dataset with 10000 records by varying the batch_size and running in CPU or GPU. The validation accuracy informs

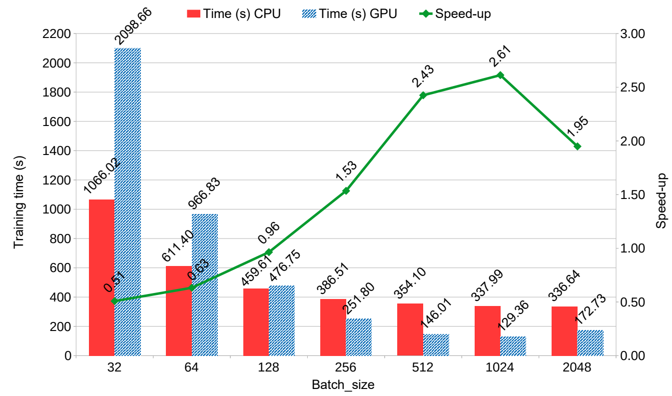

Fig. 4. Training time and ANN speed-up on CPU and GPU architectures.

the percentage that our model has right in the prediction with respect to the label. These tests prove that the implementation accuracy is stable, varying between $87 \sim 89\%$.

Figure 4 shows the training times according to the defined batch_size. It can be seen that as the batch size increases, the runtime is reduced considerably on both architectures. CPU runs showed a gain of 3.17 times. This reduces the execution time from 1066.02 seconds to 336.64 seconds. In GPU executions, the gain reached 12.15 times, jumping from 2098.66 seconds to 172.73 seconds, this represents a gain of 91.8%.

The gain is very expressive if compared to CPU. Yet, in runs with smaller batch_size, it is possible to notice that the execution time in GPU practically doubles in relation to the CPU time. This is because the GPU takes longer to transfer the data than the training itself. When the number of instances increases, it is possible to use more of the capacity the GPU offers.

The speed-up of the application is also demonstrated in Figure 4, presenting its best case in training using batch_size=1024. This represents a gain of 2.61 times over the CPU, reducing the CPU runtime from 337.99 seconds to 129.36 seconds on the GPU. This is due to the GPU devoting more resources to training rather than prioritizing data transfer. However, when the computational resources limit is reached, it also causes loss of performance, as can be seen by increasing the batch_size to 2048. This occurs due to having more processes running and dividing the computational resources.

To validate ANN LSTM, the trained model that presented the lowest loss rate and the highest accuracy rate, both values obtained from the validation dataset, was selected. This resulted in the GPU trained model with batch_size=2048. Figure 5 shows the metrics of the model, with values accuracy = 87.76% and loss = 30.33%. The model is loaded by the ANN algorithm, and then the inputs are submitted for testing. The new entries are random reviews obtained from the official IMDb[2] page about the film Wonder Woman (2017).

[2]https://www.imdb.com/

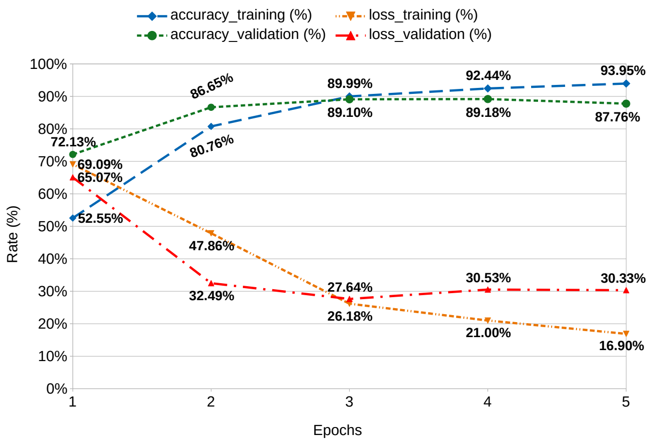

Fig. 5. Training metrics of the selected model.

Review	Evaluation	ANN Prediction
1	1/10	99.22% N
2	5/10	95.80% N
3	7/10	82.83% P
4	9/10	99.74% P
5	1/10	99.54% N
6	10/10	99.71% P
7	9/10	99.93% P
8	5/10	99.01% N
9	8/10	78.01% P
10	3/10	93.40% N

TABLE I

RESULTS OF PROPOSAL VALIDATION. PREDICTION: NEGATIVE (N), POSITIVE (P)

The results obtained are shown in Table I. The first column indicates the selected review. The second column is the number of stars indicated by the author of the review, considering as a negative opinion the evaluations between 1 and 5 and positive opinion the evaluations between 6 and 10. The last column indicates the prediction made by our model. When analyzing the table, it can be seen that the ANN model tested was assertive in all cases submitted.

V. CONCLUSIONS AND FUTURE WORK

This work addressed the use of recurrent artificial neural networks Long Short-Term Memory in the classification of feelings into sentences. Different configurations were used to analyze the performance of the application running on CPU and GPU architectures. Regarding the accuracy rate, for CPU architecture, the best rate is registered for batch size 32. For GPU architecture, the best rate is obtained for batch size 64. The proposal presented good performance in terms of ANN accuracy. Runtime is reduced considerably on both architectures by increasing the batch size. Among the most significant gains, CPU runtime gained 3.17 times. The GPU runtime was reduced by 12.15 times. Comparing the architectures, the GPU runtime showed a reduction of up to 61%. This represents a speed-up of 2.61 times over CPU time. These results indicate that the use of GPU reduces RNAs training time, presenting good yields for larger batch sizes.

As future works, a first initiative could be the use of pre-trained word embeddings. In addition, it is possible to study the influence of other hyperparameters such as learning rate, Dropout rate and Activation Function. It is also possible to evaluate larger databases and in other languages. Another analysis that can be done is to add other classification classes besides positive and negative, such as very positive, very negative and neutral. As well as evaluating other data sources like twitter. Another approach to be studied is a better separation of training, testing and validation bases.

REFERENCES

[1] B. Liu *et al.*, "Sentiment analysis and subjectivity." *Handbook of natural language processing*, vol. 2, no. 2010, pp. 627–666, 2010.

[2] I. P. P. dos Santos, "Análise de sentimento usando redes neurais de convolução," 2017, dissertação (Mestrado em Engenharia Eletrônica), Universidade do Estado do Rio de Janeiro, Rio de Janeiro, Brazil.

[3] M. A. Walker, P. Anand, R. Abbott, J. E. F. Tree, C. Martell, and J. King, "That is your evidence?: Classifying stance in online political debate," *Decision Support Systems*, vol. 53, no. 4, pp. 719–729, 2012.

[4] A. S. Zharmagambetov and A. A. Pak, "Sentiment analysis of a document using deep learning approach and decision trees," in *2015 Twelve international conference on electronics computer and computation (ICECCO)*. IEEE, 2015, pp. 1–4.

[5] Z. Hu, J. Hu, W. Ding, and X. Zheng, "Review sentiment analysis based on deep learning," in *2015 IEEE 12th International Conference on e-Business Engineering*. IEEE, 2015, pp. 87–94.

[6] C. Li, B. Xu, G. Wu, S. He, G. Tian, and H. Hao, "Recursive deep learning for sentiment analysis over social data," in *2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, vol. 2. IEEE, 2014, pp. 180–185.

[7] G. Rossum, "Python reference manual," 1995.

[8] J. Moolayil, "An introduction to deep learning and keras," in *Learn Keras for Deep Neural Networks*. Springer, 2019, pp. 1–16.

[9] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[11] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with cuda," *Queue*, vol. 6, no. 2, pp. 40–53, 2008.

[12] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*. Association for Computational Linguistics, 2011, pp. 142–150.