Convolutional architectures with LSTM and TCN to embolism classification: exploring dependency between data

Luiz G. K. Zanini[1], **Aldomar P. S. Silva**[1], **Felipe V. de Almeida**[1], **Fátima L. S. N. Marques**[2] , **Anna H. R. Costa**[1]

[1]Escola Politécnica – Universidade de São Paulo (USP)

[2]Escola de Artes, Ciências e Humanidades – Universidade de São Paulo (USP)

{luiz.kasputis,pietrosantana,felipe.valencia.almeida,anna.reali}@usp.br

{fatima.nunes}@usp.br

Abstract. *Pulmonary Embolism is an affection caused by obstruction of the pulmonary artery or one of its branches. This condition imposes a high mortality incidence, in the United States approximately 100.000 deaths per year. Computed Tomography Pulmonary Angiography is a radiologic modality and an essential technology for diagnosing this disease, providing a series of axial images. We trained two Convolutional Neural Networks (Efficient Net B0 and Resnet 3D 18) in the RSNA-STR Computed Tomography Pulmonary Angiography Dataset to identify this affection. After training these Convolutional Neural Networks, we added a new layer to the architecture by exploring the dependency between the images along the exam using Long Short-Term Memory or Temporal Convolutional Networks. With the models trained and tested, we compared these different approaches using different metrics. As a result, the Temporal Convolutional Network approach with Resnet 3D 18 improved significantly compared to the results found in the other methods. The main contribution of this work was to observe how different combinations of architectures can help classify Computed Tomography Pulmonary Angiography.*

1. Introduction

Deep Learning Techniques have been taking place in many fields, such as Medicine and Agriculture and also in the improvement of computer techniques in domains like Cyber Security and Data Mining [Latha et al. 2021]. Since they have shown a considerable capability to handle different types of complex data in an extensive range of problems, also these techniques can be expressed in various types of architectures such as Convolutional Neural Networks (CNNs), Generative Adversarial Networks (GANs) and Graph Neural Networks [Pang et al. 2022]. From these architectures, CNNs specifically are widely used for image classification tasks because they have the flexibility to include layers for feature extraction, achieving reasonable performance rates.

Deep Learning is also being addressed in the task of Anomaly Detection, a general problem that occurs in all kinds of data (such as videos, images and time series) and can be described as the identification of data points that diverge from the rest of the dataset [Pang et al. 2022]. Anomaly Detection is considered one of the most active research areas. However, the development of deep learning in this area is still slow and very challenging due to the unique characteristics of the anomalies.

On the other hand, in the medical context, a problem that has been taking place in medical care centers since a long time ago is the Pulmonary Embolism (PE) Disease. Since 2004, Wittram et al. (2004) details occurrences of PE misdiagnosis and deaths because it often went undetected. Nowadays, Kwok et al. (2022) still reports the misdiagnosis of pulmonary Embolism, the confusion made between PE and other diseases, which ends up causing suboptimal care and fatalities.

Pulmonary Embolism is a life-threatening condition. An important employed method for diagnosing it is Computed Tomography Pulmonary Angiography (CTPA) [Righini et al. 2017], a special kind of exam that provides a series of axial images (Figure 1) of the thorax in a caudocranial direction [Wittram et al. 2004] and end up requiring experience and time to an accurate interpretation.

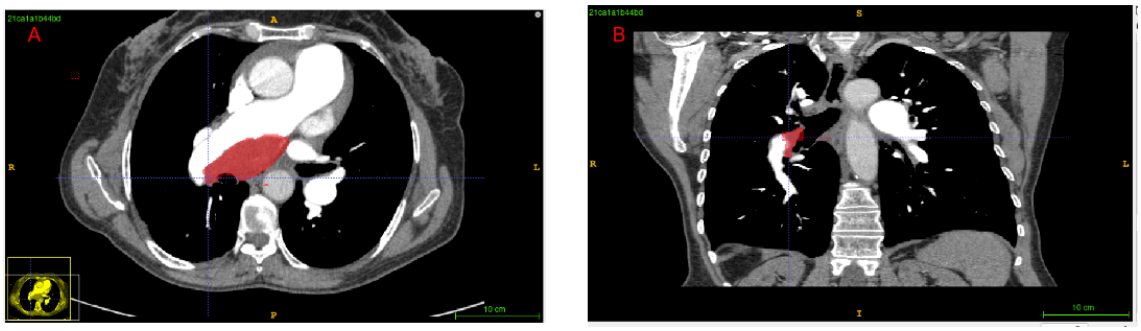

Figure 1. CT pulmonary angiography (CTPA). A. Axial contrasted CT with manual segmentation shows filling defects within the right pulmonary artery with acute pulmonary Embolism. B. Coronal contrasted CT with manual segmentation shows filling defects within the right pulmonary artery with acute pulmonary Embolism. We used axial images from the exam [Colak et al. 2021] in this project.

A solution that has been proposed to solve this problem is the use of Machine Learning to aid in the diagnosis of pulmonary Embolism by identifying anomalies in CTPA images. An anomaly detection field is based on finding heterogeneous characteristics in a data set. It has been taking place in a large group of researchers, including temporal data, visual data and graph data, which makes its techniques reasonable to solve the pulmonary embolism problem.

In this context, to increase the use of Machine Learning in the diagnostic of Pulmonary Embolism, the Radiological Society of North America (RSNA) and the Society of Thoracic Radiology (STR) have presented a dataset with thousands of images [Colak et al. 2021]. This dataset can be used as a testbench for deep learning models, since it is possible to identify research papers that use it. For example, Ma et al. (2022) proposed a deep learning multitask learning method using CNN and the Temporal Convolutional Network (TCN) to predict disease and other classifications, which resulted in good performance using this dataset. Another example is a deep learning experiment using Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM), but using a private dataset [Huhtanen et al. 2022]. According to these two works, we proposed comparing different approaches to explore dependency between the images along the exam using Long Short-Term Memory (LSTM) and Temporal Convolutional Network (TCN).

The Pulmonary Embolism classification task offers different ways of looking at the same problem and, therefore, different approaches to detecting the anomaly. The objective of this work is to compare two different methods comparing techniques of temporal dependence of the data. In the context of pulmonary Embolism, we aim to verify whether the time dependency between the data can help in the classification task in anomaly detection.

To do so, we trained two Convolutional Neural Networks in the RSNA-STR Computed Tomography Pulmonary Angiography Dataset, taken as a case study, and, with the CNN weights freezed after the training, we introduced a Long Short-Term Memory Architecture in it. For the same purpose, we added an architecture that aims to be a better option than LSTMs, the TCN. With both models trained and tested, we compared these approaches using different metrics.

The CNN model learns about each slice of the tomography individually, while the CNN-LSTM model, or the CNN-TCN model, can use the information from previous slices to help in its decision. This kind of task, where temporal dependency in data is taken into account by the models, is called a sequential modeling task.

By the end, with the comparison between these approaches taking different metrics into account, we can have an overall look about how changing the perspective of the problem, looking at it as time-dependent data instead of single observations, can help us to improve our results.

2. Sequence Modeling Architectures

The sequential characteristics of the RSNA-STR Pulmonary Embolism dataset, combined with our hypothesis that taking into account the time dependency in a set of data can increase the classification performance, led us to explore sequence modeling tasks.

We end up finding two architectures, based on different approaches, that propose to solve this problem: one based on recurrent neural networks, the Long Short-Term Memory (LSTM), and one based on convolutional operations, the Temporal Convolutional Network (TCN). So, in this section, more details are given about each one of these architectures.

2.1. Long Short-Term Memory

Recurrent Neural Networks (RNNs) consist of a class of artificial neural networks that have an internal state, representing a memory, which makes them capable of dealing with time dependencies between data [Wang and Tax 2016]. Although very well known, Recurrent Neural Networks have a limitation that often appears when dealing with long sequences: the internal state can be reduced to zero or be enhanced to an outstanding value while moving through the observations, which can make the classification task impossible.

To overcome these problems, Long Short-Term Memory Architecture was proposed, being different from RNNs because of its feedback connections. These connections make LSTM possible to process long sequences instead of single points of the data, keeping valuable information from preceding observations that help to learn from the next ones without the problem of a vanishing gradient problem.

Long Short-Term Memory architectures are composed of a set of neural networks that behave like gates, each one having its own function. There are basically three steps that compose a LSTM, being the first one the process of the forget gate, where a neural network is trained to decide what information present in the long term memory of the architecture (cell state) is relevant given the previous point in time (hidden state). Irrelevant information has a value close to 0 in the output vector and, in the pointwise multiplication that follows, it will have a result that has less influence in the next steps, while relevant information will have values next to 1, being preserved in the next steps.

The second step is related to deciding what information is going to be added to the cell state and has two neural networks: the new memory network, responsible for learning how to combine the previous state and the new input information and the input gate, which responsible for learning what information given by the new memory network is worth retaining; then, the outputs of the networks are pointwise multiplied and added to the cell state.

At last, in the third step, the output gate, another neural network, is trained to decide which information from the newly updated cell state will be given as the new hidden state, ensuring that only important and necessary information is given.

In this work, we used a bidirectional LSTM. Bidirectional LSTMs works just like a regular LSTM architecture, but instead of learning just the sequence of the input provided, it also learns the reverse order, being trained twice. For more information on LSTMs, we recommend reading Huang et al. (2015) work.

2.2. Temporal Convolutional Network

Temporal Convolutional Networks (TCN) was proposed by Bai et al. (2018) as a different way of thinking when dealing with sequential modeling tasks, since this kind of problem is often treated with recurrent neural networks, such as LSTMs. TCN promises to achieve better performance than these other approaches and also solve the problem of a vanishing gradient with the differential of promoting the usage of parallel computation to speed up the data processing.

The architecture composed by TCN has as input 3-dimensional tensors and also outputs tensors of the same shape, being the dimensions (i) *batch-size* (ii) *input_length* or *output_length*, in case of the output, with both being of the same size and (iii) *input_size/output_size*, which sizes can differ depending of the context.

To generate the output vector, the convolution is performed between the input vector and a kernel of learned weights. This means that for each element of the output vector, a dot product is done between the kernel and some subsequent elements of the input vector, where the number of elements taken corresponds to the *kernel_size* parameter. At the end of the operation, the input vector elements are shifted in one position to the right.

To ensure that the output vector has the same size as the input vector, zero-padding is applied on the left of the input vector, since if it is added to the right size, it may difficult the temporal context. In the case of multiple channels (multivariate cases), the same process is made for each channel but with different kernels, and, in the end, these intermediate vectors are summed to form the output vector.

The architecture is built with *input_size* representing the number of neurons in the input layer and *output_size* representing the number of neurons in the last layer, and the intermediate layers depend on the parameter *num_filters*. To reduce the number of intermediate layers, a technique called dilation is applied. That means that the spacing (d) between the elements selected from the input vector is increased in order to include a larger area.

To even lower the number of layers (avoiding it to be very deep), the spacing *d* is calculated as $d = b^i$, being b a constant called *dilation_base*, which should be lower or equal than *kernel_size*, and i being the number of layers below the actual layer. For more details about how forecasting is done and other improvements made to the architecture, such as activation functions and the residual blocks added, we recommend reading the Bai et al. (2018) work paper.

3. Material and Methods

The data used in this project is presented in subsection 3.1. In subsection 3.2 we describe image processing techniques and data transformations. In subsections 3.3 and 3.4, respectively, we explain the convolutional neural networks used in this project. There is an explanation of the training of the LSTM and TCN architecture used in subsection 3.5. Finally, we have a description of the experiments performed in subsection 3.6.

3.1. Dataset

Experiments were performed using the RSNA dataset. The RSNA Pulmonary Embolism CT Dataset was made available for competition on the Kaggle platform to classify Pulmonary Embolism cases on chest CT scans [RSNA 2020]. The available data have 7279 cases of studies (exam) with distribution into 2368 with the disease and 4911 with no disease.

This dataset is composed of exams made from healthy or not people, and each exam is composed by a number of images (slices) that may contain (or not) signals of pulmonary embolism. In the case of exams with the disease, not all images have these signals of pulmonary embolism, which lowers the proportion of images that have the characteristics of the disease compared to the other ones. With this classification, we train the models described in Section 3.6 to predict if a slice has the signal of the disease.

3.2. Preprocessing data

Before training the models, we applied preprocessing techniques in the CTPA images generating a mask under the region of lung tissue as seen in Figure 2. We used this method to prevent the model from learning some non-recurring patterns of the image. So the classifier learned only information about lung tissues.

In the first step in preprocessing, we transformed the raw CTPA data into the Hounsfield Scale [DenOtter and Schubert 2022], which represents the density of tissues. After the transformation, we applied window width 700 and window level 100, which corresponds to a pixel range $[-250, 450]$ on the Hounsfield scale (Figure 2 - Data Hounsfield). These parameters were recommended to help differentiate a trombone and an artifact [Wittram et al. 2004].

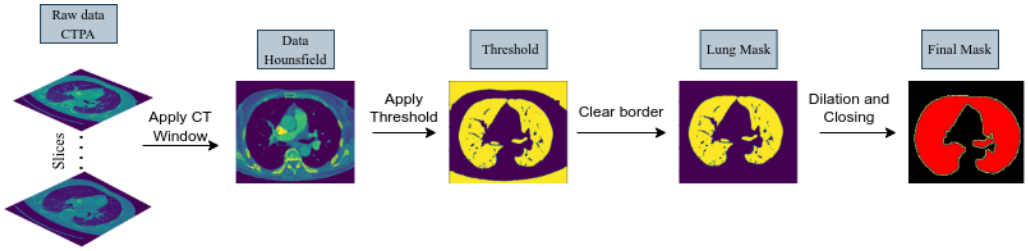

Figure 2. Pipeline corresponding to lung tissue extraction applying preprocessing techniques.

In the next pre-processing steps, we used a fixed threshold ($T_H = -400$) separating the lung tissue from other parts of the body (Figure 2 - Threshold Application). The next step consists in removing the background, extracting all regions close to the border of the image (Figure 2 - Background Removal). Finally, we applied morphological operations using dilatation and closing, to generate a mask that was applied to the image according to Figure 2 - Final Mask.

The input to the CNNs has three channels, so we add the same image but with a different window width and level to the images, corresponding to the input with three channels. Using the recommendation from [Wittram et al. 2004], applying window with and window level: (-600, 1500) lung and (40, 400) mediastinal.

3.3. Training EfficientNet B0

We selected a CNN capable of providing high accuracy and fast learning to classify the data. In this approach, we chose the architecture EfficientNet B0, a CNN that produces good results with a smaller number of training parameters [Tan and Le 2019]. Our focus in this research was on considering different types of architectures that explored temporal dependence, such as LSTM and TCN. The CNN EfficientNet B0 provides fast training and good time prediction to deal with it.

Before we trained the CNN, we used weights pre-trained on ImageNet [Deng et al. 2009], improving the performance and getting a fast training rate. We used data augmentation to avoid overfitting, increasing the number of samples and keeping the proportion of the classes. [Shorten and Khoshgoftaar 2019]. Methods for data augmentation were Gaussian blur, random horizontal flip, and random rotation, from the library PyTorch [MetaAI 2022].

$$BCE = -\frac{1}{N}\sum_{i=0}^{N} y_i \cdot \log\left(\hat{y}_i\right) + \left(1 - y_i\right) \cdot \log\left(1 - \hat{y}_i\right) \tag{1}$$

As for the loss function, we used binary cross entropy (BCE), as shown in Equation 1 where $\hat{y}$ is the predicted value and y is the actual label since we predicted binary values. To train the CNN, we set 10 epochs, a learning rate of 0.0001, and batch sizes of 48. We observed the loss function and selected the best hyperparameters.

3.4. Training ResNet 3D 18

The 3D convolutional neural network uses temporal dependence to extract features through 3D convolution. We used the architecture ResNet 3D 18, which presents good

results in extracting features in videos and 3D images [Tran et al. 2018]. Furthermore, we used the pre-trained network using the Kinetics dataset [Aurelio et al. 2019].

To train the 3D network, we divided the exam into an arbitrary number of windows. Our work aims to classify each of the slices individually, so we split the exam into windows, since the entire exam as input to the neural network is very computationally expensive. The chosen windows value was 32 slices (Figure 3), aiming to extract a significant number of temporal features and obtain a good training time, since increasing the number of frames used in the 3D Network increases the training time [Tran et al. 2018]. The input of this network corresponds to $C \times F \times H \times W$, where F is the number of frames corresponding to slices in our approach, C refers to channels, H and W are the height and width of the image, respectively.

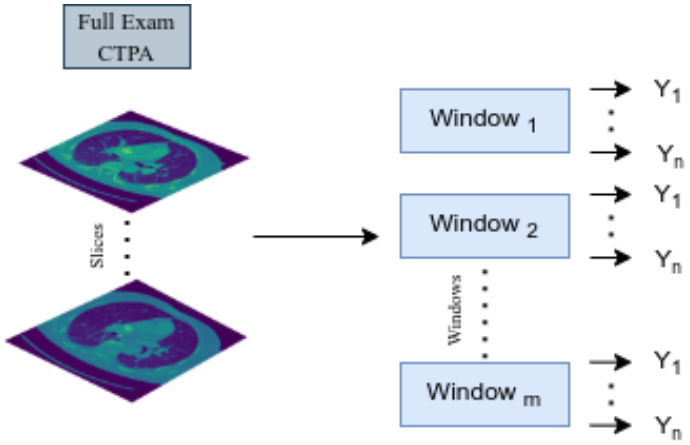

Figure 3. Split exam in windows. The M was the total of windows split of the exam, and N was the number of slices.

Each exam was divided into 32 slices, and for the remaining values, we added padding with zero values. The data augmentation methods were applied to windows; we used the same techniques according to Section 3.3. To train this neural network, we set 20 epochs, a learning rate of 0.0001, and batch sizes were 4. We used BCE for the loss function since we predicted binary values.

3.5. Training LSTM and TCN

To explore the use of slices in CTPA, we used the two convolutional networks defined in Sections 3.3 and 3.4. We combined two sequence modeling architectures, exploring the use of those architectures combined with extracting features from convolutional networks.

For training these sequence modeling architectures, we did not modify the convolutional network weights. The backpropagation algorithm only changes the weights of the LSTM, TCN, and Fully Connected (FC) layers. The purpose of maintaining the weights was to observe whether TCN and LSTM improved the performance of the pulmonary embolism classification. In addition, if we changed the weights of the convolutional layers, the training would involve a much higher computational cost, since it would be necessary to update the weights in each convolutional layer in each training.

We use the LSTM bidirectional model to provide information between slices. The output of LSTM corresponds to 128, a value that corresponds to the number of features

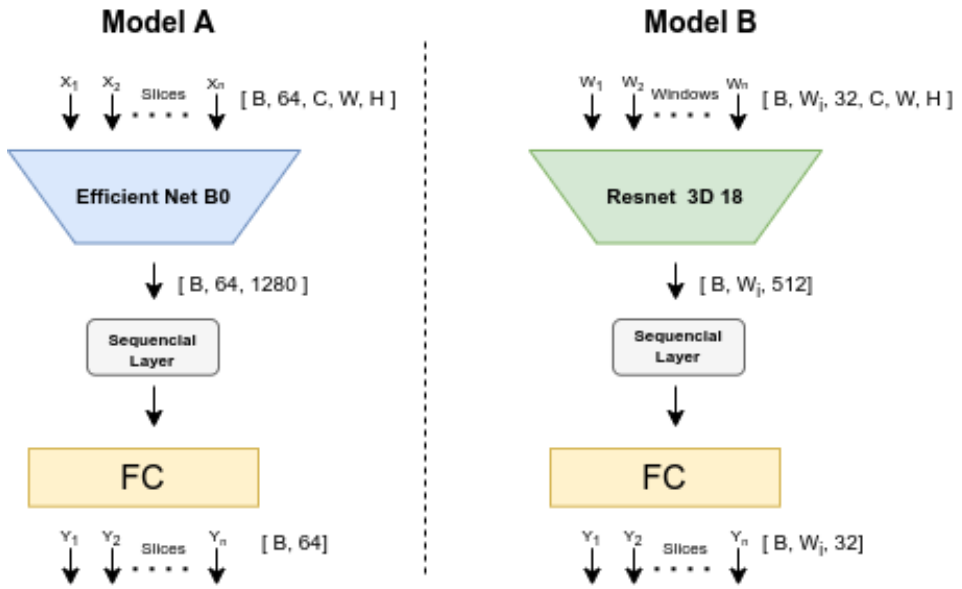

Figure 4. The two types of architectures we used to classify pulmonary embolism (Model A and Model B), showing the Sequential Layer used the LSTM and TCN architecture to learn about the sequence of slices or windows. The inputs and outputs of each layer are: B = batch size, Wi = all windows from exam, C = channels, W = width and H = height.

in the hidden state, which is used as input for the fully connected layer. For the TCN, the output channel corresponds to 128 and the parameters with a kernel size of 5 and number of levels of 3. The parameters were chosen in individual experiments observing the loss function and based on the work of Ma et al. (2022).

In the Resnet 3D 18 with layers LSTM or TCN, we split the exam into windows, that way we concatenate the feature vector before and after the window. The output of CNN3D corresponds to a window vector of 512 characteristics (W_i) we concatenated the values W_{i-1}, W_{i-i}, providing a input to TCN as LSTM with 1516 features. In cases that have no before or after value, zeros are added to the sequence.

For the training of models A and B (Figure 4), we used 100 epochs for each, and the batch size used was 16. The training process was much faster because we did not update the weights of the convolutional networks. The main difference between in these approaches is that we used a complete exam (a whole set of slices) to train in Model B, and Model A learning from 64 slices.

3.6. Experimental Setup

For the experiments, we opted for exams that had a maximum of 320 slices. This number represents 6667 of the 7279 exams, we chose to use these exams because of the padding applied in exams with different sizes. Therefore, we reduced our dataset by less than 10% of the total number of cases.

The training was divided between 20% of a test set and 80% of a training set. In both groups, we distributed the cases to patients with and without embolisms. Thus, the training set has 1728 patients with embolism and 3604 without embolism, while the test

set has 433 patients with embolism and 902 without embolism.

To select an operating threshold, we use the Youden Index [Ruopp et al. 2008]. The threshold calculated by the Youden method provides a good value for splitting classifiers with unbalanced data [Peng et al. 2020]. Threshold selection seeks to optimize the values between sensitivity ($\frac{TP}{TP+FN}$) and specificity ($\frac{TN}{TN+FP}$), where TP are the True Positives, TN are the True Negatives, FP are False Positives and FN are False Negatives.

Table 1. Hyperparameters used for model training

Model	Batch Size	Epochs
Efficient Net B0	48	10
ResNet 3D 18	4	20
Efficient Net B0 and LSTM	16	100
Efficient Net B0 and TCN	16	100
ResNet 3D 18 and LSTM	16	100
ResNet 3D 18 and TCN	16	100

The machine used to run the experiments had an Intel(R) Core(TM) i7-8700K processor with 64 GB of RAM and two Nvidia GTX 1080 TI video cards. The hyperparameters used for each model can be seen in Table 1, the hyperparameters selected were based on previous works [Huhtanen et al. 2022, Ma et al. 2022] and observing the loss function from the model. The models were created using the PyTorch library [MetaAI 2022], and the language for writing the code was Python.

4. Results and Discussion

The results obtained by training the architectures presented in the previous sections correspond to Table 2.

Table 2. Training results, best results from the metrics found, in bold text

Model	Accuracy	Sensitivity	Specificity	Precision	F1 Score	AUC
Efficient Net B0	0,7709	0,7808	0,7687	0,1676	0,2760	0,8328
ResNet 3D 18	0,7893	0,8089	0,7754	0,2213	0,3493	0,8680
Efficient Net B0 and LSTM	0,8289	0,8321	0,8231	0,2643	0,4012	0,8954
Efficient Net B0 and TCN	0,8023	0,8184	0,8015	0,2541	0,3878	0,8759
ResNet 3D 18 and LSTM	0,8343	**0,8706**	0,8169	0,2851	0,4295	0,9136
ResNet 3D 18 and TCN	**0,8790**	0,8665	**0,8702**	**0,3339**	**0,4820**	**0,9213**

From the results obtained according to Table 2, it is possible to observe that in all cases, the precision of the results is a value below 40%. This is due to the fact that the exam (set of slices) is distributed into 66,91% exams without embolism and 33,09% with embolism. In the slices with embolism present (any form of pulmonary embolism is present on the image), we have for training set and validation set an amount corresponding to 4,85% and 4,89% of the data, respectively. Therefore, the precision metric is significantly affected by the imbalance of the data and we use data augmentation only to avoid overfitting, not to balance the classes, since it is unlikely in real scenarios

[Ma et al. 2022]. Our approach is concerned with finding the best results within the conditions using LSTM and TCN, not seeking to optimize the classification of pulmonary embolism.

Huhtanen et al. (2022) presented the model precision at a value of 60% - 70% using the confidence interval. However, the data used in the model were balanced, unlike the data presented in this project and the real scenario. In Ma et al. (2022) work , the metric precision was not presented, only specificity, sensitivity, and AUC using the dataset RSNA-STR.

Comparing the approaches that take a single slice to make the classification (Efficient Net B0 and ResNet 3D 18) alone with the methods that combine them with a sequential architecture (LSTM or TCN), we can see that the association with sequential architectures increases all the metrics collected. This means that taking into account the temporal dependency in datasets, which allows us to do so, maybe an excellent approach to solving the problems.

As for the differences between the LSTM and TCN architectures, we can see that, depending on the context, one architecture may be a better approach. In the 2D approach, using Efficient Net B0, we achieved better results with the association with the LSTM model, while in the 3D approach, we achieved better results with the TCN model. Our hypothesis is that the 2D context, with the Efficient Net B0, does not provide enough characteristics for the TCN model to achieve higher performances. In contrast, the 3D context can provide enough characteristics for the task.

The Resnet 3D 18 convolutional network uses filters that extract features along the slices, unlike the approach used in Efficient Net B0. The use of context in this neural network increases performance compared to the metrics found in the approach that uses conventional convolutional filters. Therefore, Resnet 3D 18 performed better in all selected methods.

5. Conclusion

This work aims to compare the two main approaches in the classification of pulmonary embolism in CTPA ([Huhtanen et al. 2022], [Ma et al. 2022]), analyzing these techniques from the field of anomaly detection. Our focus was to observe convolutional architectures with architectures that use temporal dependence as input. Within the context used, we made some adaptations. However, we observed that there was a performance improvement using sequential architectures.

Thus, TCN's approach with Resnet 3D 18 offered an improvement compared to the results using only CNN. However, this approach is extremely expensive and requires very high computational power. Techniques using the Efficient Net B0 offer smaller performance metrics with much less training time.

Therefore, the main contribution generated by this work was to observe how different combinations of architectures can help classify CTPA, also, how the use of temporal architectures influences the results and can be used in other types of exams and diseases.

For future work, we expect to use the implementation of the architectures in different sets of medical data, analyzing the influence of these architectures on the

classification of other diseases.

Acknowledgments

This work is supported in part by the Brazilian National Council for Scientific and Technological Development (CNPq grant numbers 310085/2020-9, 309030/2019-6, 140253/2021-1), the Coordination for the Improvement of Higher Education Personnel (CAPES Finance Code 001), Brazil, and Itaú Unibanco S.A. through the PBI program of the *Centro de Ciência de Dados* (C^2D) of Escola Politécnica at Universidade de São Paulo, and São Paulo Research Foundation (FAPESP) – National Institute of Science and Technology – Medicine Assisted by Scientific Computing (INCT-MACC) – grant 2014/50889-7.

References

Aurelio, Y. S., de Almeida, G. M., de Castro, C. L., and Braga, A. P. (2019). Learning from Imbalanced Data Sets with Weighted Cross-Entropy Function. Neural Processing Letters, 50(2):1937–1949.

Bai, S., Kolter, J. Z., and Koltun, V. (2018). An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. arXiv preprint arXiv:1803.01271.

Colak, E., Kitamura, F. C., Hobbs, S. B., Wu, C. C., Lungren, M. P., Prevedello, L. M., Kalpathy-Cramer, J., Ball, R. L., Shih, G., Stein, A., Halabi, S. S., Altinmakas, E., Law, M., Kumar, P., Manzalawi, K. A., Nelson Rubio, D. C., Sechrist, J. W., Germaine, P., Lopez, E. C., Amerio, T., Gupta, P., Jain, M., Kay, F. U., Lin, C. T., Sen, S., Revels, J. W., Brussaard, C. C., Mongan, J., Abdala, N., Bearce, B., Carrete, H., Dogan, H., Huang, S.-C., Crivellaro, P., Dincler, S., Kavnoudias, H., Lee, R., Lin, H.-M., Salehinejad, H., Samorodova, O., Rodrigues dos Santos, E., Seah, J., Zia, A., Arteaga, V. A., Batra, K., Castelli von Atzingen, A., Chacko, A., DiDomenico, P. B., Gill, R. R., Hafez, M. A., John, S., Karl, R. L., Kanne, J. P., Mathilakath Nair, R. V., McDermott, S., Mittal, P. K., Mumbower, A., Lee, C., Orausclio, P. J., Palacio, D., Pozzessere, C., Rajiah, P., Ramos, O. A., Rodriguez, S., Shaaban, M. N., Shah, P. N., Son, H., Sonavane, S. K., Spieler, B., Tsai, E., Vásquez, A., Vijayakumar, D., Wali, P. P., Wand, A., and Zamora Endara, G. E. (2021). The RSNA Pulmonary Embolism CT Dataset. Radiology: Artificial Intelligence, 3(2):e200254. Publisher: Radiological Society of North America.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 248–255. ISSN: 1063-6919.

DenOtter, T. D. and Schubert, J. (2022). Hounsfield Unit. In StatPearls. StatPearls Publishing.

Huang, Z., Xu, W., and Yu, K. (2015). Bidirectional LSTM-CRF models for Sequence Tagging. arXiv preprint arXiv:1508.01991.

Huhtanen, H., Nyman, M., Mohsen, T., Virkki, A., Karlsson, A., and Hirvonen, J. (2022). Automated detection of pulmonary embolism from CT-angiograms using deep learning. BMC Medical Imaging, 22(1):43.

Kwok, C. S., Wong, C. W., Lovatt, S., Myint, P. K., and Loke, Y. K. (2022). Misdiagnosis of pulmonary embolism and missed pulmonary embolism: A systematic review of the literature. Health Sciences Review, 3:100022.

Latha, R., R. Sreekanth, G. R., Suganthe, R., and Selvaraj, R. E. (2021). A survey on the applications of Deep Neural Networks. In 2021 International Conference on Computer Communication and Informatics (ICCCI), pages 1–3. ISSN: 2329-7190.

Ma, X., Ferguson, E. C., Jiang, X., Savitz, S. I., and Shams, S. (2022). A multitask deep learning approach for pulmonary embolism detection and identification. Scientific Reports, 12(1):13087.

MetaAI (2022). PyTorch. `https://pytorch.org/`. Acessed on 10-Aug-2022.

Pang, G., Aggarwal, C., Shen, C., and Sebe, N. (2022). Editorial Deep Learning for Anomaly Detection. IEEE Transactions on Neural Networks and Learning Systems, 33(6):2282–2286.

Peng, Y., Li, C., Wang, K., Gao, Z., and Yu, R. (2020). Examining imbalanced classification algorithms in predicting real-time traffic crash risk. Accident Analysis & Prevention, 144:105610.

Righini, M., Robert-Ebadi, H., and Le Gal, G. (2017). Diagnosis of acute pulmonary embolism. Journal of Thrombosis and Haemostasis, 15(7):1251–1261.

RSNA (2020). RSNA STR Pulmonary Embolism Detection. `https://kaggle.com/competitions/rsna-str-pulmonary-embolism-detection`. Acessed on 10-Aug-2022.

Ruopp, M. D., Perkins, N. J., Whitcomb, B. W., and Schisterman, E. F. (2008). Youden Index and Optimal Cut-Point Estimated from Observations Affected by a Lower Limit of Detection. Biometrical journal. Biometrische Zeitschrift, 50(3):419–430.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. Journal of Big Data, 6(1):60.

Tan, M. and Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In International conference on machine learning, pages 6105–6114. PMLR.

Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., and Paluri, M. (2018). A closer look at spatiotemporal convolutions for action recognition. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 6450–6459.

Wang, F. and Tax, D. M. (2016). Survey on the attention based RNN model and its applications in computer vision. arXiv preprint arXiv:1601.06823.

Wittram, C., Maher, M. M., Yoo, A. J., Kalra, M. K., Shepard, J.-A. O., and McLoud, T. C. (2004). CT Angiography of Pulmonary Embolism: Diagnostic Criteria and Causes of Misdiagnosis. RadioGraphics, 24(5):1219–1238. Publisher: Radiological Society of North America.