Video-level Lung Ultrasound Scoring

A Deep Learning model for LUS Video Score Prediction

Laura Corso [MAT. 230485]

I. Introduction

This document is the report for the assignment of the course *Medical Imaging Diagnostic* (academic year 2022/2023) at University of Trento. The aim of this project is to design a system that allows to extract from a lung ultrasound video its corresponding label indicating the degree of severity for COVID-19 pneumonia.

The rapid outburst of SARS-CoV-2 virus in 2019 has brought attention to Lung Ultrasound Imaging (LUS). In fact, this is a wide available, cost-effective, safe and real-time imaging technique [1] that provided an alternative solution for the diagnosis of COVID-19 during the crisis, when patient inflow exceeded the regular hospital imaging infrastructure capabilities. Several studies, such as [2] and [3], have demonstrated that it's possible to extract LUS image patterns associated with this and other pathologies and train a classifier to detect the severity of the disease based on them.

Furthermore, in [4] a description of a standardize approach to optimize the use of lung ultrasound in patients with COVID-19 is proposed along with a grading system that allows the classification of LUS data into degree of morbidity classes.

At present, studies that tackle the classification problem for COVID-19 bio-markers at video level are unknown. To current knowledge, video classification has been obtained in [1] through aggregation techniques that don't take into account the whole video. Therefore, in this project a classification system for an entire LUS video that follows the labels of [4] is proposed.

II. Task Formalisation

The project consists in the development of an automatic labelling system for lung ultrasound videos.

The label convention described in [4] has been followed and it is here reported for clarity. Therefore, LUS data can be classified into four score classes according to the severity of the lesions due to COVID-19 pneumonia:

- *Score 0*: the pleural line is continuous and regular. Horizontal artifacts are present. These artifacts are generally referred to as **A-lines**. They are due to the high reflectivity of the normally aerated lung surface and characterize the visual representation of the multiple reflections happening between the US transducer and the lung surface itself.
- *Score 1*: the pleural line is indented. Below the indent, vertical areas of white are visible. These are due to local alterations in the acoustical properties of the lung, as, for example, the replacement of volumes previously occupied by air in favor of media that are acoustically much more similar to the intercostal tissue (water, blood, and tissue). This phenomenon opens channels accessible to US, which can explain the appearance of the vertical artifacts called **B-lines**.
- *Score 2*: the pleural line is broken. Below the breaking point, small-to-large consolidated areas (darker areas) appear with associated areas of white below the consolidated area (white lung). The darkening of the consolidated areas signals the loss of aeration and the transition of these areas toward acoustic properties similar to soft tissue over the entire area represented by the consolidation itself. Beyond the consolidations, the appearance of areas of white lung signals the presence of areas not yet fully deaerated, where air inclusions are still present but embedded in tissuelike material. This highly scattering environment can explain this peculiar pattern.
- *Score 3*: the scanned area shows dense and largely extended white lung with or without larger consolidations.

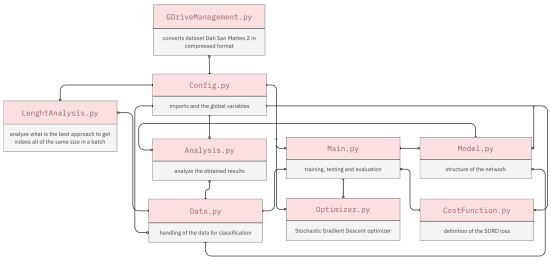

Fig. 1. Structure of the Python code. The arrows circle termination mean that the file imports the file at the other end of the link.

The automatic labelling system is constructed using Deep Learning Networks. In particular, following the example of [5] a *Convolutional Neural Network* (CNN) is used to extract spatial features from the video frames. Then, a spatial attention layer is added to detect the presence of B-line artefacts. Furthermore, the spatial feature maps are input to a *Recurrent Neural Network* (RNN) to extract temporal features. Finally, the so obtained results are input to a *Fully Connected* (FC) layer to obtain the 4-classes classification (more details can be found in Section V).

The aim of the developed model is to combine the spatial features with the temporal ones in order to obtain a classification technique that can be directly applied at video level. In addition, the attention layer is placed in order to refine the extraction of the spatial features needed to obtain a more accurate classification.

III. Archive content

The delivered archive contains:

- this *Report.pdf* file;
- the *Code* folder which contains:
  - the *Python* files that are organized as shown in ***Fig. 1***;
  - the *Annotations.txt* file generated through *AnnotationsCreator.m*;
  - the *fromMatToTxtIF.m* that generate *mIF.txt*;
  - the *requirements.txt* file that contains the Python's libraries to install;
  - the *Results* folder which contains the graphs generated by the analysis and the file *FinalEx.txt* which contains the printing of the last execution of the code;
  - the *README.txt* file that contains all the instructions to execute the code.

IV. Data Description & Analysis

The data used in this project are provided by the course's teacher and are contained in the main folder *Dati San Matteo Dataset 2* in Google Drive. This directory contains two files and 11 subfolders. The two files are: *Read_Data.txt* which explains how the data are organized in the subfolders, and *SanMatteo.mat* which contains a summary of the data.

Regarding the current application, the provided data consists of 239 LUS annotated videos, acquired using the protocol described in [4], during 18 exams performed on 11 patients.

In order to use the hand out data, they are downloaded from Google Drive in a compressed format. Consequently, they are unzipped to form the folder with the structure showed in ***Fig. 2***. As can be seen, the *Dataset* folder contains a folder for each patients. Each patient's folder contains a folder for each exam. Each exam's folder contains a folder for each area of inspection and each area's folder contains the *.jpg* images that will be merged to form the videos of the *Dataset*.

Once the *Dataset* folder is complete, the data are transformed in *Tensors* and organized in the *Dataset* structure of *PyTorch*. For these data, a custom *Dataset* class is defined in order to iterate through the *Dataset* one video at a time.

For the training and the testing of the model described in Section V the data are split into three datasets: the training, the testing and the statistic datasets. The first one is used to train the model. The second one is used to test the performance of the model and the last one is used to extract some statistics of the model further discussed in Section VI. The train and test datasets are build splitting videos at patient level. Therefore, 80% of the patients will be in the train dataset, while the remaining 20% will be in the test set. The splitting is randomly executed but in order to make the code reproducible, in the delivered code the random seed is fixed at 5. The splitting produces the results displayed in ***Table I***.

	Number of videos for				
	Score 0	Score 1	Score 2	Score 3	Total
Train Set	35	31	75	36	177
Test Set	10	14	26	19	69

TABLE I
DISTRIBUTION OF SCORES IN TRAIN AND TEST SETS.

The statistic dataset, instead is comprehensive of all the data without any split applied.

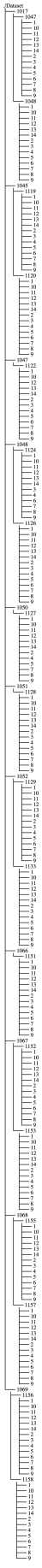

Fig. 2. Structure of the data once downloaded from Google Drive.

Each of the aforementioned *Dataset* is build applying two kind of transformations:

- Frame level transformations which include resizing and normalization of the images that compose the video;
- Video level transformation.

In particular, the latter is a zero-padding in the time dimension. As a consequence, all the videos will have the same length which is the length of the longest video in the provided data. This transformation is needed because during the training of the model the videos are loaded in batches, i.e. subgroups of the videos contained in the *Dataset* structure, that must contains videos all of the same length.

The zero-padding approach is chosen after two length analysis: length of the videos VS scores and length of the videos VS patients. The results of the former are showed in ***Fig. 3***, while the results of the latter are showed in ***Fig. 4***.

From the reported results, it can be seen that the scores are well distributed along the lengths, i.e. the score is independent form the length of the video. In addition, patients n.1051 and n.1047 are the ones with the shortest videos. Therefore, the zero-padding can be applied safely in order to both solve the problem of different lengths and avoid a bias of the classifier on the number of zero-frames added at the end of the shortest videos.

Furthermore, to avoid even more the bias problem, the two aforementioned patients are located deterministically in the test set.

V. Model

As mentioned in Section II, the aim of this project is to develop an automatic system that is able to classify LUS videos in four classes of COVID-19 pneumonia severity. To achieve this, the model depicted in ***Fig. 5*** is implemented.

As shown, the network is composed of:

- Convolutional Neural Network (CNN) that takes as input the frames of each video in a *Dataset* to extract the spatial features. Here a pretrained *AlexNet* is used.
- Attention Layer that is composed, as described in [5], of three convolutional layers that, taken in input the feature map of the ith frame X_i (with $1 \leq i \leq n$, n is the number of frames of a video), return the mask M_i.
  Attention is achieved with element-wise multiplication:
  $$\widetilde{X_i} = X_i \odot M_i$$
  Finally, the attended image feature $\widetilde{X_i}$ represents a region highlighted (or suppressed) version of X_i that is globally pooled, hence delegating any ability to resolve features spatially to the attention mask.
  The purpose of this attention layer is to focus on B-line artefacts in order to increase the classification accuracy by providing additional information to the classifier, as done in [3].
- Bidirectional Recurrent Neural Network that from the spatially attended feature maps extracts the temporal features for a whole video. Here a *Bi-LSTM* is used.

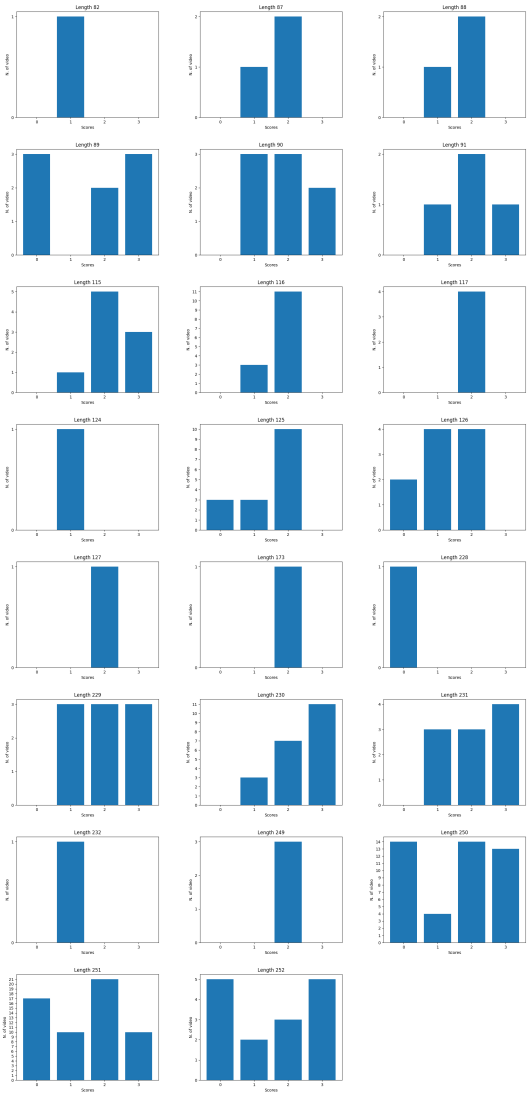

Fig. 3. Lengths VS Scores analysis: each histogram reports the number of videos for each score with a certain length.

- Fully Connected Layer to obtain the classification of a video in the four classes.

The model is trained with the *SORD loss* described in [1] without increasing the distance of $Score\ 0$ from the others. Additionally, *Stochastic Gradient Descent* is used to update the parameters of the network during the training phase. The parameters are the following: $base\ learning\ rate = 0.001$, $weigth\ decay = 0.001$, $momentum = 0.7$. The optimizer is defined by assigning distinct learning rates to the parameters of the CNN and to the ones of the rest of the network. This is done because the AlexNet is pretrained, hence only a finetuning of its parameters is needed. Instead, the learning rate for the rest of the network (Attention Layer + Bi-LSTM) is 10 times higher because its trained from scratch.

In order to carry out the training phase, the model is deployed on the Multi-Process Service (MPS) which enables high-performance training on GPU for MacOS devices with Metal programming framework (for more details see [6]).

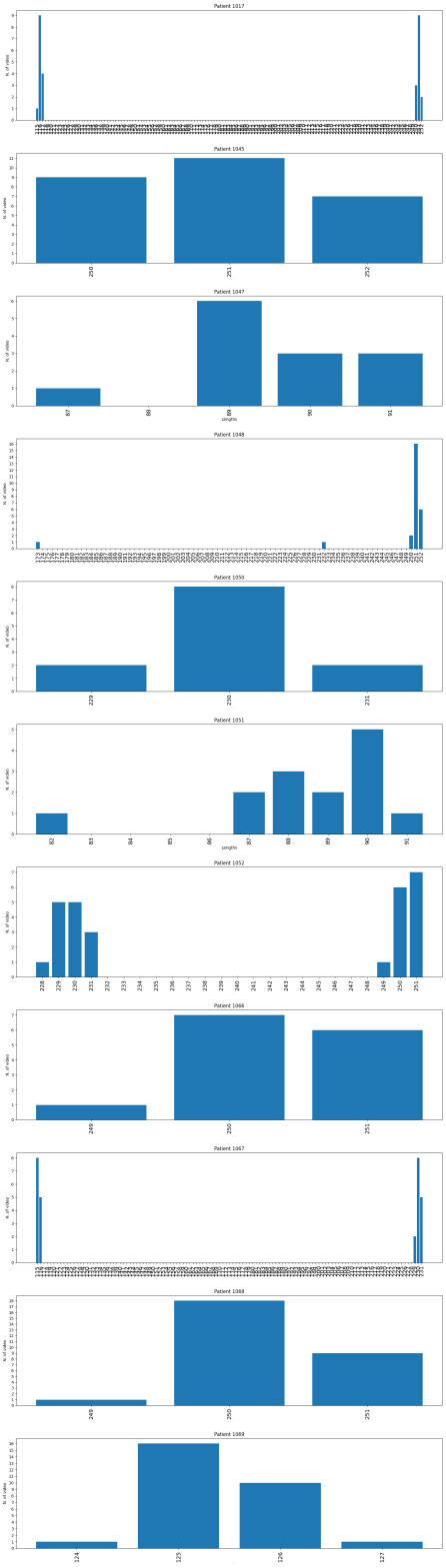

Fig. 4. Lengths VS Patients analysis: each histogram reports the number of videos for each length given a patient.

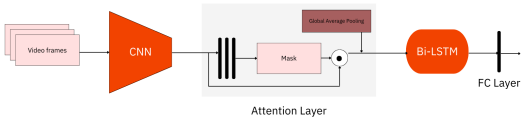

Fig. 5. Network's model.

Given the limitations of the chosen device, the training is carried out with batches of 4 videos due to limited RAM capacity. Furthermore, the number of epochs is fixed empirically at 35. The training with these settings was carried out on a M1 MacBook Pro equipped with 16 GB of RAM in 12 hours.

A. *NaN investigation*

The number of epochs is set to the aforementioned value after some investigations. In fact, after this repetitions, the gradients of the network become Not-A-Number values (NaN).

This phenomenon started to occur from the first stages of development of the project and the decision to use *AlexNet* instead of a deeper network was dictated from the fact that CNNs with an high number of *Convolutional Layers* led to NaN gradients from the first batches in the first epoch. The registration of backward hooks on the network's module revealed that the propagation of the NaN value started after the shallower *Convolutional Layers* of the model (the ones closer to the Attention Layer).

This effect was partially solved with a smaller model that prevent the propagation through many *Convolutional Layers* of gradients close to the zero value due to the small number of videos in a batch. But, if the training was pursued after the fixed number of epochs, the problem presented again.

In order to understand, whether the issue was really the batch size, further analysis were carried out.

Firstly, since the problem seemed to be the vanishing of the gradients, the Sigmoid and ReLu functions of the model was replaced with LeakyReLu but also with this modification the NaN values appeared. Therefore, the activation functions are not the problem.

Secondly, the zero padding was substituted by adding at the end of the video replicas of the last frame. Even in this case, this didn't solve the problem pointing out that the 0 values of the frames are not implicated in the vanishing of the gradients.

Thirdly, the Bi-LSTM was initialized with several combinations of parameters in order to try different starting points in the solution space but still the NaN values appeared.

All this considerations lead to the conclusion that the chosen model is quite deep and, consequently, a more performing hardware is needed to carry out effectively the training using bigger batches.

VI. Results

The above described model is tested at the end of every epoch on the test *Dataset* and in the last instance it reaches the following evaluation values:

- $Accuracy = 44.12\%$;
- $Loss = 0.32$;

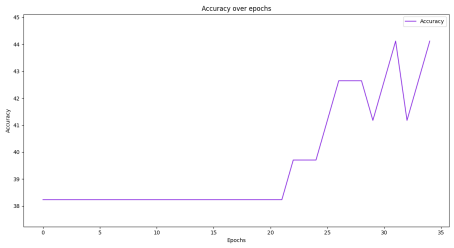

Fig. 6. Accuracies on the test set along epochs.

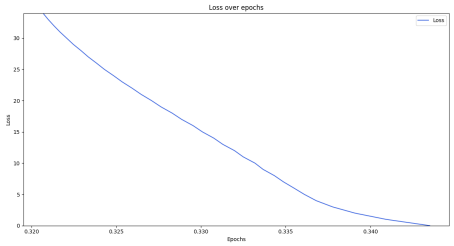

Fig. 7. Losses on the test set along epochs.

- $F1score = 0.44$.

The trends of accuracy and loss values during training on the test set can be observed in ***Fig. 6*** and ***Fig. 7*** respectively.

In addition, the weights of the best performing network (the one that has reached the lower loss value on the test set during the training phase) are saved and used to instantiate a model that is runned on the statistic dataset which comprehend the whole data. This computation results in the confusion matrix depicted in ***Fig. 8*** and in an $Accuracy = 41.06\%$.

In addition, the number of misclassification errors per class, i.e. the number of times a sample of a given class is classified as belonging to another one, is specified in ***Fig. 9***.

To evaluate the performance of the *Attention Layer*, eight attention maps are extracted from the best performing model and applied to the original images. The results can be seen in ***Fig. 10***.

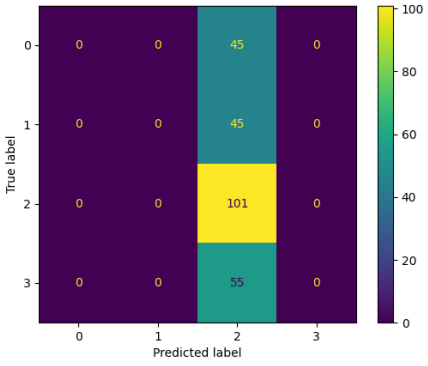

Fig. 8. Confusion Matrix.

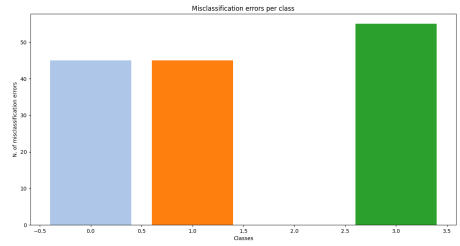

Fig. 9. Misclassification errors per class

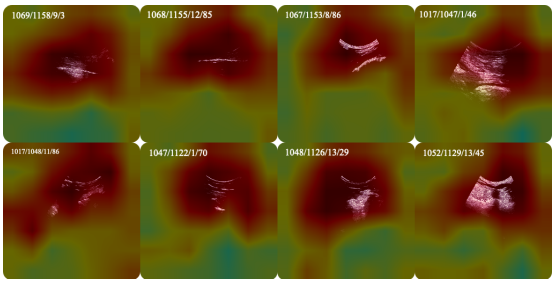

Fig. 10. Attentions Maps for some selected frames. The text in every image is the identification of the frame as: patient_id /exam_id /area_id /frame_id.

VII. Discussion and Conclusions

From the results reported in Section VI it's possible to see that the model tries to maximize the accuracy classifying all the videos as belonging to the $Score\ 2$ class. This class is also the one more represented in the data as can be seen in ***Table I***. Therefore, the classifier classify every sample as the most probable one that, given the unbalanced dataset, is a video of $Score\ 2$.

In addition, as can be seen from ***Fig. 10*** the attention layer is useful only in highlighting the area of the image that actually contains the LUS exam, and the main focus, most of the times, is not on B-lines artefacts.

Even if the aforementioned performances are not so good, additional investigations are needed in order to understand if the model can be further finetuned leading to better results. In fact, the accuracy and loss graphs depicted in ***Fig. 6*** and ***Fig. 7***, respectively, indicate a favorable trend toward a good classifier. Also, the fact that some B-lines are correctly identified (as in frame 1017/1048/11/86) could be a preview of what the model, trained on a hardware that supports bigger batch size, is able to do.

Future works could focus on the implementation of a weighted loss that takes into account the unbalanced dataset and the testing of the model on a bigger dataset to even more finetune the parameters.

Concluding, even if the results are not above chance level, this model has the potentiality to reach good performances with an appropriate hardware and, consequently, deserves further analysis since it is the only one that tries to classify LUS exams at video level.

References

[1] S. Roy, W. Menapace, S. Oei, B. Luijten, E. Fini, C. Saltori, I. Huijben, N. Chennakeshava, F. Mento, A. Sentelli, E. Peschiera, R. Trevisan,

G. Maschietto, E. Torri, R. Inchingolo, A. Smargiassi, G. Soldati, P. Rota, A. Passerini, R. J. G. van Sloun, E. Ricci, and L. Demi, "Deep learning for classification and localization of covid-19 markers in point-of-care lung ultrasound," *IEEE Transactions on Medical Imaging*, vol. 39, no. 8, pp. 2676–2687, 2020.

[2] L. Carrer, E. Donini, D. Marinelli, M. Zanetti, F. Mento, E. Torri, A. Smargiassi, R. Inchingolo, G. Soldati, L. Demi, F. Bovolo, and L. Bruzzone, "Automatic pleural line extraction and covid-19 scoring from lung ultrasound data," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 67, no. 11, pp. 2207–2217, 2020.

[3] O. Frank, N. Schipper, M. Vaturi, G. Soldati, A. Smargiassi, R. Inchingolo, E. Torri, T. Perrone, F. Mento, L. Demi, M. Galun, Y. C. Eldar, and S. Bagon, "Integrating domain knowledge into deep networks for lung ultrasound with applications to covid-19," *IEEE Transactions on Medical Imaging*, vol. 41, no. 3, pp. 571–581, 2022.

[4] G. S. et al., "Proposal for international standardization of the use of lung ultrasound for patients with covid-19: A simple, quantitative, reproducible method," *J Ultrasound Med*, vol. 2020 Jul;39(7):1413-1419, 2020.

[5] H. Kerdegari, N. T. H. Phung, A. McBride, L. Pisani, H. V. Nguyen, T. B. Duong, R. Razavi, L. Thwaites, S. Yacoub, A. Gomez, and V. Consortium, "B-line detection and localization in lung ultrasound videos using spatiotemporal attention," *Applied Sciences*, vol. 11, no. 24, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/24/11697

[6] [Online]. Available: https://pytorch.org/docs/stable/notes/mps.html