Extracting high value lung ultrasound images from video for the diagnosis and prognosis of COVID-19.

CS433 2020 - Project 2 - ML4Science

Quentin Müller *M.S. Data Science, EPFL, Switzerland*
Aleandro Eccel *M.S. Data Science, EPFL, Switzerland*
Arnaud Robert *M.S. Data Science, EPFL, Switzerland*

Supervisors
Mary-Anne HARTLEY, iGH, EPFL, Switzerland
Mariko MAKHMUTOVA, iGH, EPFL, Switzerland

December 17, 2020

I. Introduction

The accurate diagnosis of respiratory pathology often relies on expert analysis of lung imaging such as chest X-rays and CT scans. However, in low resource settings neither the equipment nor the expertise are available and in the context of COVID-19 the centralization of such exams created an important burden on the health care system. Lung ultrasound (LUS) is an easy-to-use point-of-care test that has the potential to match the predictive capacity of CT and X-ray but still requires expert analysis of images captured, and-critically-expertly selected images from the live video clips taken at each thoracic site.

Schmutz et al. developed *DeepChest* [1], a deep learning approach for diagnosis and risk stratification in COVID-19 using Lung Ultrasound (LUS) images which were expertly selected from video clips. As with most deep learning approaches, it is likely that the model would benefit from the additional training data extracted from the original videos. Additionally as LUS expertise is rare, a video mining model that could automate expert selection of images with high predictive value from LUS videos would prove to be a practical tool in the democratisation of clinical expertise in low resource settings.

Methods to extract images from videos have been gaining interest in the last years. One of them, keyframe extraction, focuses on the extraction of one image per shot. It aims at keeping the most amount of information in a video using the least amount of images, in other terms, it summarizes videos. Many implementations exist, for example one which computes the smallest correlation between every images output, another one takes a reference image for each scene and compares it to every frame contained in the shot. [2]

However we do not think these methods suit the needs of this project because we are interested in extracting a specific frame of the video, rather than an image that best summarizes it.

II. Aim

Our aim is to automate expert selection of image frames in a LUS video that have high clinical predictive value.

A. Objectives

Pre-processing: Pre-process LUS images and videos for standardized analyses.

Image selection:

1) Explore variations of brightness between images as a selection criteria.
2) Explore image cosine similarity with 2 different encoding methods (convolutional computation of local brightness and a CNN used as encoder).
3) Train a binary image classifier using transfer learning to identify high quality images.
4) Test the predictive capacity of the images mined from the videos in the DeepChest model for the diagnosis of COVID-19.

III. Methods

A. Cohort and Dataset

The dataset comprises 3455 images and 1265 videos from 193 patients recruited at the CHUV emergency room in Lausanne Switzerland between the 6th February 2020 and the 5th August 2020. These dates fell within the first wave of the COVID epidemic [3]. Inclusion criteria were consenting adult patients suspected of lower respiratory tract infection. LUS videos were performed on at most 10 thoracic sites and still images were then selected by expert clinicians either from the exact video recorded or from the region the video represents.

For our binary classifier, we naturally define that the images selected by experts constitute our *True* labels. We do not, however, have data for the opposite *False* label, which represent images that the experts would not select.

B. Data pre-processing

To train our future models, we first need to standardize the images and videos. By inspecting them, the following observations could be made:

- Before performing the LUS exam on a patient, parameters that control the sound wave are tuned to explore different depth and are thus written in a text on the top of the videos. This results in videos with different shapes and scales.
- On the experts selected images, a blue dot with a white B situated on the upper left, as well as a yellow text indicating from which site the image was taken on the bottom are added. Videos do not have these labels.
- Videos and experts images do not have the same shapes. Videos have shapes (1080, 792, 3) whereas images have shapes (1080, 791, 3) or (1080, 790, 3).

To solve the first two points, we decided to manually create a static mask that would remove the title, blue dot, right scale, and lower yellow text from all images while keeping the central image intact. The last point was resolved by resizing the expert images to the format of the videos using the opencv resize method [4].

C. Transfer learning

We explore transfer learning approaches for their benefits of computational efficiency. There is evidence for a successful classification on LUS images, using models pretrained on the ImageNet dataset [5], which supports our approach. In order to train our classifier, we use the ResNet18 model [6] pretrained on the ImageNet dataset. In this implementation, the model uses various convolutional layers to convert an image to 512 features maps, which are then reduced to scalar values by a 2D adaptative average pooling layer, and

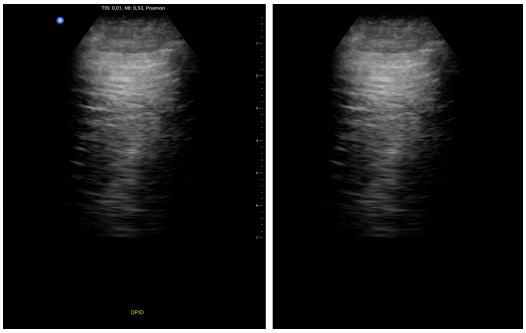

Figure 1. LUS video pre-processing: Before the mask is applied (left) and after (right)

then linked to a 1000-dimensional vector through a fully connected layer. Since our classifier is binary, we modify this last layer into delivering a 2-dimensional vector. Before entering the model, a given image goes through an additional pre-processing pipeline: it is resized to dimensions 349x256, center cropped on a square of dimension 224 and finally normalised channel wise using means $[0.485, 0.456, 0.406]$ and standard deviations $[0.229, 0.224, 0.225]$, which is necessary to correctly make use of the pretrained model [7].

Training is done over 25 epochs, using cross entropy loss and the Stochastic Gradient Descent with momentum ($\beta = 0.9$). Our initial learning rate value is $\gamma = 0.001$ and we use a learning rate optimizer that multiplies it by factor 0.1 every 7 epoch. We fine tune the model during training by allowing all the weights of the model to be retrained.

D. Generating datasets

In order to train our model, we need data. As mentioned earlier, our dataset only contains images with *True* labels. Thus we need to generate data for the opposite label in order to be able to train our model effectively. Our reasoning is the following: among all frames of a given site's video(s), there must be a frame that corresponds, perfectly or by some extent, to the associated expert image, and by extension the other frames should differ from this image and thus possess features that make them *False* images. Hence, we'll be extracting our missing label images from the videos. We tested the following:

1) Selection based on brightness: This method uses the mean brightness of images to identify poor quality images. It would be sensitive to malpositioning of the US device (air/bone acquisition which has extreme values of echogeniety). The mean brightness and its deviation over a subsample of our *True* images S was computed on the masked zone (zones outside the mask are anyway black and would only lower the mean) in order to get a brightness reference point. Based on these values, we selected boundaries that were equal to $[\mu_{up}, \mu_{down}] = \mu_S \pm \sigma_S$. We then select video frames whose brightness level fell outside of these boundaries as our set of *False* images. We added that a frame could not be selected if a nearby frame in the same video was already selected.

2) Selection based on similarity: Cosine similarity has been previously explained [8]. We aim to use it to select a set of *False* images with controlled similarity to their corresponding *True* image. We hypothesize that this would closer emulate the clinical logic of deciding on the most representative image within a set, which may be based on structural (and thus pathological) components of the image as well as their visual characteristics of contrast etc. Here, images are encoded to a lower dimensional space, then converted to vectors on which we compute cosine similarity. Two different methods for encoding were tested, namely a convolutional computation of local brightness and a CNN used as encoder.

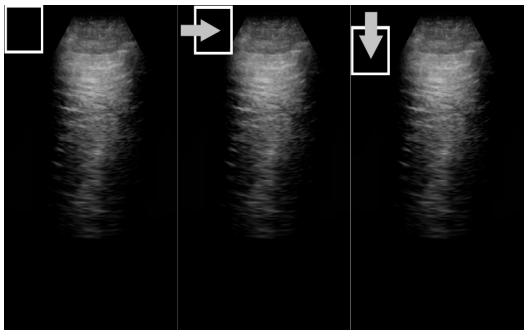

Figure 2. Subsamples brightness : The initial position of the mask (left), the mask after being shifted on the right (middle) and the mask lowered when the end of the first row is reached (right)

Computation of local brightness: we split the image in a couple sub-parts for which we compute the mean brightness. To obtain these sub-parts, a mask of size $N \times N$ was applied on the images and was then shifted by $\frac{N}{2}$ pixels on the right, when the end of the image is reached, the mask returns to the left most x-coordinate and the y-coordinate is increased by $\frac{N}{2}$ pixels. In our case, N was set to 200 pixels, which means that given an image of shape 1080×792, this method outputs a 10×7 matrix. The shifts of $\frac{N}{2}$ act as a way to decrease the loss of information of the image as opposed to shifts of size N, that make all the sub-parts independent of each other and could thus drown some bright parts in a sea of darkness. This method is more resistant to malpositioning than global mean brightness based method, as it will look for a similar distribution of brightness over the image rather than comparing the overall values.

CNN Encoder: Once again, we try to benefit from a transfer learning approach focusing namely on feature extraction. We consider the same pretrained ResNet18 model that we use for training our classifier, except that we do not retrain its weights. In order to preserve only the encoding part of the model, we remove its last two layers and replace them by a 2D adaptative average pooling layer which converts the 512 feature maps to 4×4 matrices. We then take the

channel average over these 512 reduced features maps to obtain a single 4×4 matrix, which is our final encoding of the image. Just like for our classifier, a given image is pre-processed before entering the model, which enables our encoder to take raw images as input and return a 4×4 matrix encoding as output.

We now detail the procedure to extract *False* images. For each image in our original dataset, we find the matching video if it exists, then iterate through its frames. For each frame and our reference *True* image, we compute the similarity as described previously. If the similarity score is included in given similarity boundaries, the frame is selected as a *False* image and is added together with the *True* image to our new dataset, such that labels are balanced in the final dataset. The video iteration then stops and another image from the original dataset is considered. Using this procedure, we create four different datasets based on different similarity boundaries. Those are $[0.75, 0.85], [0.8, 0.9], [0.85, 0.95], [0.9, 0.97]$. These boundaries were selected in order to grasp different types of images. We want to analyze how our model performs when *False* images are selected far, in between, or close to the *True* ones. We select 0.75 as the lower bound because we observed that cosine similarity and a random generated image tends to be around $\simeq 0.55$ for the local brightness and $\simeq 0.77$ for the CNN encoder.

IV. Results

A. Cohort and Dataset

The dataset contains between 350 and 400 images from each site with the exception of the sites QLD and QLG which have roughly 220 images each. The distribution of the videos is unequal, with more QLD and QLG videos than for any other site. Furthermore, images and videos without an anatomical reference (misspelled or unknown sites: UNK and QASP) were removed (n=16 images and n=4 videos).

B. Data pre-processing

Due to the uniformity in the layout of the elements that needed to be removed, the implementation of a simple mask managed to remove artifacts and created standardised images.

C. Transfer learning

The model trained on brightness data achieves high validation accuracy starting from epoch 7 as can be seen on Fig 3 and peaks at 99,74% validation accuracy. The training produced 10,99 gCO2eq.

The last two columns of Table I indicate the best validation accuracy and carbon footprint of all the similarity based model we trained. One can note that most of them reach 100% validation accuracy.

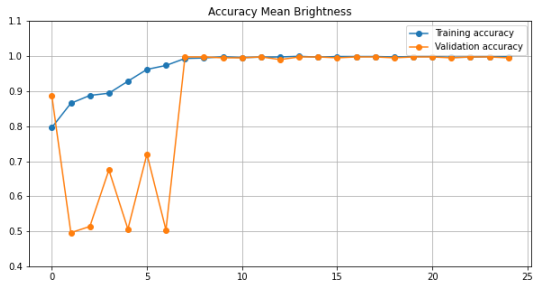

Figure 3. Training & validation accuracies, on brightness based data

Bounds	Encoder	Samples generated (max 1698)	Best Val Accuracy	CO2 footprint
0.75-0.85	Bright matrix	398 (23%)	97,43%	2.29
0.75-0.85	cnn	88 (5%)	100%	0.78
0.8-0.9	Bright matrix	584 (34%)	100%	3.23
0.8-0.9	cnn	222 (13%)	100%	1.52
0.85-0.95	Bright matrix	956 (56%)	100%	5.2
0.85-0.95	cnn	578 (34%)	99.12%	3.27
0.9-0.97	Bright matrix	1100 (64%)	100%	5.89
0.9-0.97	cnn	856 (50%)	100%	4.63

Table I
DATASET GENERATION AND MODEL TRAINING METRICS

D. Generating datasets

1) Selection based on brightness: We create a sub-sample of 1000 *True* images and 1000 *False* images. *True* images are randomly recovered from the original dataset. For the *False* images, we also iterate randomly over the videos and select frames based on the criteria described in III-D1.

2) Selection based on similarity: Each similarity boundaries paired with one of the two encoders (Bright matrix and CNN) generated different number of samples as detailed in the *Samples generated* column of Table I. Since there are 849 matching pairs of images and videos in our dataset, the maximum number of samples that can be obtained through our procedure is 1698.

Each dataset was then split between a train and validation set using a 80/20 train-ratio.

E. Success metrics

In order to test the performance of our models, we created a test set that is composed of 10 patients that had at least 6 videos from different sites. In this subset, there were 131 videos with 89 having matching expert images. We devise two tests.

A first test, based on quantity, is to have a model go through the test videos and extract the single most confident good image ($p > 0.5$) from each one, if he is able to find one. We then see how many good images were extracted in total over the test set.

A second test, based on quality, uses the notion of similarity once again. Iterating through the test videos, we select the most confident image according to our model

for each video (notice that the selected image does not necessarily satisfy the condition $p > 0.5$) and then compute the similarity between this image and the corresponding expert image to the video. We then obtain our metric by computing the mean of these similarity scores over the test set, for each model. This is performed for both similarity metrics. Because this measure is similarity, it is at most equal to 1.

Bounds	Encoder	Test 1 #Good images (max 131)	Test 2 Bright matrix	Test2 CNN
0.75-0.85	Bright matrix	56 (43%)	**0.9441**	0.9718
0.75-0.85	cnn	**93** (71%)	0.9318	0.9694
0.8-0.9	Bright matrix	7 (5%)	0.9336	0.9714
0.8-0.9	cnn	**61** (47%)	0.9347	0.9685
0.85-0.95	Bright matrix	2 (2%)	0.9251	0.9714
0.85-0.95	cnn	47 (36%)	0.9401	0.9699
0.9-0.97	Bright matrix	1 (1%)	0.9249	0.9659
0.9-0.97	cnn	15 (11%)	0.9351	**0.9738**

Table II
TEST METRICS FOR MODELS TRAINED ON SIMILARITY DATA

On the first test, the model trained on brightness data performs poorly by extracting only 2 images out of all the 131 videos from the test set. Considering an expert was able to obtain 89 images out of the videos, this gives us an indication that our model is flawed. The results for the first and the second test for the models trained on similarity data can be seen in Table II. The model trained on CNN with bounds 0.75-0.85 is able to extract 93 good images out of 131 in the test set, which is an encouraging result compared to previously. On the last two columns, we can see that the two different similarities methods respectively designate models BrightMat 0.75-0.85 and CNN 0.9-0.97 as being the ones able to get images most similar to expert images. Interestingly, one can note that the two similarity methods do not agree on designating the best model, and further that none of these two models are the ones able to extract most images. Finally, one also has to consider that the two models that extract images are those trained on the least amount of samples, which could lead to a higher variance in their performance.

1) Test with DeepChest: A last test was performed with the two models that extracted most images, namely CNN 0.75-0.85 and CNN 0.8-0.9. A new test set consisting of 68 videos from 7 patients was fed to our models. The first of these two models generated 37 images from 6/7 patients, not being able to find suitable images for the 7th patient, while the second, the 0.8-0.9 CNN generated 28 images for only 4/7 patients. The images were then given to the *DeepChest* algorithm. Table III contains these results.

	Correct prediction	Correct prediction for all patients
Expert images	6/7 (85%)	6/7 (85%)
0.75-0.85 CNN	6/6 (100%)	6/7 (85%)
0.8-0.9 CNN	3/4 (75%)	3/7 (42%)

Table III
SUCCESSFUL *DeepChest* DIAGNOSIS BASED ON SELECTED IMAGES

V. DISCUSSION

1) Conclusions: Our first model approach, based on a dataset generated through selection based on mean brightness, appears to be a poor image extractor. We suppose that this method of selection is too naive and actually yields *False* images that are not relevant because they are too different from the *True* images, and as such the model can not learn to recognise good images efficiently.

Our second model approach seems to yield better results. We are able to extract more good images from the test set. We hypothesize that giving *False* images that are closer to *True* images enables the model to learn more complex discriminating features, such as the presence or position of the pleural line, in order to select images rather than looking at brightness only. Lastly, images selected by model under this approach allowed the *DeepChest* algorithm to perform a majority of accurate diagnostic, suggesting our model was able to extract high quality images from videos.

2) Limitations: Our work shows potential to automate expert selection of image frames in LUS videos but is limited in many points. Firstly, it could highly benefit from a richer and more standardized dataset. As written in IV-A Cohort and Dataset, the videos dataset does not follow a uniform distribution and is dominated by QLD and QLG videos, while being under represented in the images. Because we create our *False* dataset using videos, a likely consequence is that the models could think that images coming from certain sites are more likely to be bad images thus having a bias towards their predictions. Another big limitation of our work is that some results could be irreproducible (although similar in values). Due to the lack of samples, the models could be influenced by the order the images were fed into the network. We also bring the attention to the fact that, in some cases, our models can return no image for a given video.

Finally, the lack of access to *DeepChest* meant that the images could not be tested as they were selected, and thus the number of verification performed during the whole project was limited.

3) Future work: Future work should try an unsupervised learning approach in order to generate labels for the dataset. Clusters of images could be fed to *DeepChest*, and based on its performance, would acquire the *True* and *False* labels. This, combined with an active learning approach like a CNN, could increase the performance.

ACKNOWLEDGEMENTS

We thank Mary-Anne Hartley and Mariko Makhmutova for the amazing support shown during this whole work and for letting us have the opportunity to work on such an interesting project.

REFERENCES

[1] H. Schmutz, "Deepchest: A neural attention model for interpretable, missingness-resilient diagnosis and risk stratification of covid-19 from lung ultrasound images," 2020.

[2] I. Hadi and T. Alfatlawi, "Key frame extraction methods," in *International Journal of Pure and Applied Mathematics Volume 119 No. 10*, 2018, pp. 485–489.

[3] "La première vague de Covid-19 en Suisse et les soins primaires," https://www.revmed.ch/RMS/2020/RMS-N-713/La-premiere-vague-de-Covid-19-en-Suisse-et-les-soins-primaires, accessed: 2020-12-16.

[4] "Opencv resize method," https://www.tutorialkart.com/opencv/python/opencv-python-resize-image, accessed: 2020-12-17.

[5] P. M. Cheng and H. S. Malhi, "Transfer learning with convolutional neural networks for classification of abdominal ultrasound images," *Journal of digital imaging*, vol. 30, no. 2, pp. 234–243, 2017.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[7] "PyTorch torchvision.models," https://pytorch.org/docs/stable/torchvision/models.html, accessed: 2020-12-14.

[8] L. Yang, Y. Zhang, J. Chen, S. Zhang, and D. Z. Chen, "Suggestive annotation: A deep active learning framework for biomedical image segmentation," 2017.