Pediatric Bone Age Prediction CNN Model With Segmentation Preprocessing

Fabio Vaccaro†

***Abstract*—Estimate pediatric bone age is a very important task to determine skeletal maturity and monitor growth process in young patients. Till now doctors had to compare radiographies to standard ones, guessing a number of months. With the steps forward data science and deep learning technologies have reached, we think these technologies could be used to substitute human in this process letting doctors healing patients. Thanks to the Radiological Society of North America (RSNA) many approaches have been presented and amazing results have been obtained. This project tries to combine best ideas taken from prior approaches to obtain even better results in terms of performance. The key part of this project is the focus on preprocessing techniques to clean data set images to let the prediction model learn more relevant features. This will be done using a U-Net architecture preparing images for a second model created with an Inception V3 architecture combined with other layers. We will show how nowadays architectures are able to overcome human specialist in accuracy and reliability, providing and easy and expandable new way of approaching this field.**

***Index Terms*—Inception V3, U-Net, Segmentation, CNNs, Bone age prediction, Radiography, Biomedical imaging**

I. Introduction

Hand radiographs are used in pediatric medicine in patients with growth or hormonal abnormalities to compare their skeletal age with their actual age to ensure they are within normal limits. Although automated bone age prediction systems are already available, few of them are used in the field and this task is still delegated to human radiologists that are stuck using the Greulich and Pyle atlas to estimate the right age [1].

As nowadays computing power is much more easily available and affordable, so approaches using deep learning on images are increasing and their potential should be exploited. For this reason Radiological Society of North America (RSNA) has hold a global machine learning competition to develop the best algorithm to predict skeletal age from pediatric hand radiographs. The best results and winning team's approaches constituted the state of the art for this specific problem [2].

The issue with input data is that radiographies are usually scanned versions of real life ones, so the image contains lots of impurities and could distract a CNN from learning the right features. So the approach used in this project was to train an architecture capable of cleaning images performing hand segmentation. The cleaned dataset will be later used to feed a prediction model. The two different deep learning models had been kept separated providing the reader to use or to improve one or both the models independently.

†Student at Department of Information Engineering, University of Padova, email: fabiovaccaro1@gmail.com

This report is structured as follows. In Sec. II we present the state of the art of technology in this specific field. In Sec. III we explain how data has been collected and how we managed datasets splits, preprocessing, data augmentation and how we implemented Dataset class. In Sec. IV we expose the architectural structure of the two main model's blocks and how the overall pipeline is organized. In Sec. V we present computing environments used for training and how the training phase has taken place for the two models. In Sec. VI we present results of the two training phases and we discuss models performance obtained. In Sec. VII we debate possible further improvements, difficulties encountered and things learnt.

II. Related Work

This work comes from the Radiological Society of North America (RSNA) Pediatric Bone Age Machine Learning Challenge, so many attempts to solve the problem have been made. The best five results are reported in paper [2] and are shown below:

1) The winning approach from Alexander Bilbily, MD, BHSc, FRCPC; Mark Cicero, MD, BESC, FRCPC (Canada) used a combination of a FFNN taking care of the sex data and an Inception V3 architecture on a rescaled 500x500 pixels bones image.
2) The second placed approach from Ian Pan, MA (United States) trained different models based on sex on patches of 224x224 pixels extracted from the original high resolution images. Patches were preprocessed enhancing contrast and were used to fine-tuned ResNet-50 architectures pretrained on the ImageNet dataset.
3) The third placed approach from Felipe Campos Kitamura, MD, MSc; Lucas Araujo Pereira, BSc; Rafael Teixeira Sousa, MSc; Larissa Vasconcellos De Moraes, BSc; Anderson Da Silva Soares, PhD; Nitamar Abdala, MD, PhD; Gabriel Alencar De Oliveira; Igor Rafael Martins Dos Santos, MD (Brazil) has not used a famous architecture (ie, Inception, ResNet), it was made specifically for this task and is composed of a transpose convolution followed by a convolutional and pooling layer bypassed by a residual connection.
4) The fourth placed approach from Hans Henrik Thodberg, PhD (Denmark) was more traditional, not using deep learning, but at the contrary a conventional ML path. Image preprocessing segmented the original high resolution image into 15 bones. Bone age was extracted from each of the 13 bones using hand-crafted features

(shape of the bone, the intensity pattern across the growth zone, and the pattern of Gabor texture energies across the growth zone).

5) The fifth placed approach from Leon Chen, MD; George Shih, MD (United States) used a segmentation mask to preprocess images. About 400 images were manually segmented to give them as input to a convolutional u-net. After having trained this CNN architecture all other training images have been segmented with the help of this architecture. Then an age prediction network was trained on masked images. This was an ensemble of convolution neural networks with a final regression layer and a sex-embedding layer.

In this report we have taken inspiration from these approaches, trying to combine the best ideas of each of them to get the best results possible.

III. Data

A. Data collection

An important step to address these types of data related projects is to understand how data have been collected. For this purpose we rely on [3]. Data is represented in Tab. 1.

A dataset of 14036 radiographs of the left hand from two institutions, Lucile Packard Children's Hospital and Children's Hospital Colorado was used. These images had been interpreted by pedriatric radiologists, giving their bone age estimate on the basis of a visual comparison to Greulich and Pyle's Radiographic Atlas of Skeletal Development of the Hand and Wrist [4]. This dataset was split between the training and validation set, 90% (number = 12611, mean age = 127 months) were randomly selected for the training set and 10% (number = 1425, mean age = 127 months) were used as the validation set.

The first test set is constituted of 200 images, 100 of males and 100 of females subjects (mean age = 132 month). This will be provided to us to evaluate the performance of the model relative to that of human reviewers.

The second test set was used to evaluate the performance on the model relative to that of existing automated software but it has not been provided to us but it is publicly available.

The ground truth skeletal age estimates for all the datasets is based on six separate estimates:

1) Clinical radiology report from their respective institution;
2) Four pediatric radiologists (two from each institution);
3) Second review by one of the pediatric radiologists who reviewed the cases approximately one year after the first review.

The ground truth for each image in the test dataset was determined in two steps: first a preliminary ground truth was calculated as a simple mean of the six reviewers' estimates. Then the performance of each reviewer was calculated as the mean difference and the mean absolute difference (MAD) between the reviewer's estimate and the mean of all the other reviewers ' estimate, which ranged from -0.75 to 1.16 months and from 4.8 to 7.0 months respectively. Next, each reviewer's estimate was weighted with a weight corresponding to the inverse of the MAD (1/MAD). So the final ground truth estimate was determined by calculating the weighted mean of the corrected reviewer estimates [2].

B. Datasets and Data split

Data was provided to us already split into training, validation and test dataset. Respectively of 12611, 1425 and 200 images. We have to point out that these are different from data given to the participants of the challenge. This will be important to later compare out results to those published by the challenge best ranked. Only the 12611 training data was provided with labels. The 1425 were used for the leaderboard phase, so for example the challenge winners used a 85:15 training:validation split resulting in 10,720 training images and 1,892 validation images [1].

To create the segmentation model we took 500 samples from the training dataset and we manually segmented it using Labelbox online tool [5]. We exceed the number or annotated images by the fifth placed team of 100 images as it was not clear if those were all used as training samples or if 400 images were to be split into the training and validation set. For this reason and to be conservative we annotated 500 images, then split with a 80:20 train:validation ratio, obtaining in the worst case the same training images as the fifth placed team.

To creare the prediction model we used the provided dataset without any further split. We only want to point out the train:validation ratio of about 10% which is pretty low. Maybe this could explain the variability of the validation loss jumping up and down at each training iteration.

C. Preprocessing

1) Preprocessing for the segmentation model: Images given are almost always 1514x2044px images (about 3MP), but not all of them; We can find 2044x1514px, same ratio but rotated, but also other formats, like 2570x2040px. For this reason and for the need to feed neural networks with more manageable and compact files we decided to resize the entire dataset (both training, validation and test dataset) to 512x512px (see Fig. 1a). This is the first difference from the winning team's approach. The reason for this change should be searched in the segmentation model applied before the prediction one (not present in the winning approach): the U-Net is made of Conv2D layers in encoder blocks and then Conv2DTranspose in decoder blocks. The input size, being a power of two makes easy to produce a mask which is the same size as the input image. This makes the following preprocessing easier, more precise as there is no need for interpolation methods, less computation intensive as there is no resize to be done. During the rescaling phase the same ratio as the provided images was used adding some side black padding. Not knowing if the proportion between width and height in bones could have some medical sense we were not comfortable to make this hazard.

Variable	Males Images	Females Image	Total Images	Mean Chronologic Age (y)	Mean Estimated Bone Age (y)
Training set					
Stanford	1485	1200	2685	10.7 ± 4.1	10.9 ± 3.9
Colorado	5348	4578	9926	10.8 ± 3.4	10.5 ± 3.3
Total	6833	5778	12611	10.8 ± 3.5	10.6 ± 3.4
Validation set					
Stanford	174	124	298	10.8 ± 4.2	10.9 ± 4.0
Colorado	599	528	1127	10.8 ± 4.2	10.5 ± 3.3
Total	773	652	1425	10.8 ± 3.5	10.6 ± 3.5
Test set					
Stanford test set	100	100	200	11.3 ± 3.8	11.0 ± 3.6
Digital Hand Atlas test set	434	479	913	8.8 ± 3.6	8.8 ± 3.8

TABLE 1: Summary Information for the Training Validation and Test Image Data Sets

2) Preprocessing for the prediction model: Using the previously trained segmentation model we generated segmentation images for each training, validation and test 512x512px image. From each segmented image we took a binary mask keeping every pixel with a probability more than 0.10 (the U-Net generates segmented images with each pixel representing its probability to belong to the segmented area)(see Fig. 1b). The obtained mask was then used to clean the original 512x512px image from distracting elements and background noise, keeping only the hand area and laying it down on a black background (see Fig. 1c).

D. Data augmentation

1) Data augmentation for the segmentation model: To feed the segmentation model we decided to not implement any data augmentation, conscious that U-Net architecture relies on the strong use of data augmentation to use the available annotated samples more efficiently [6]. This choice was taken after a first trial with only 200 samples (160 used for training), that got exceptionally good results. Probably this type of architecture is able to manage far more complex biomedical data and extracting an almost always similar shape as an hand from a relatively simple and constant background is an easy task. Moreover we did not need a very high accuracy as the only need of this phase was to clear the image for the prediction model. For these reasons we increased the annotated dataset and did not implement any data augmentation. Another advantage coming from this choice is that we could implement caching at the end of the dataset implementation also for the training dataset as data should not be augmented, this has given us a good computing boost allowing us to run this phase locally (more details in Sec. V).

2) Data augmentation for the prediction model: In contrast to what said for the segmentation model, for the prediction model accuracy is crucial and the number of samples used for training is so important that we decided to bet in this phase and to improve the winning approach data augmentation taking inspiration from the second ranked approach. The following modifications were applied to every training sample at each iteration:

1) Rotation from -20 to +20 degrees taken uniformly;
2) Translation from -20% to +20% of the image size taken uniformly;
3) Zoom from 0% to 20% of the image size taken uniformly. The resulting cropped image is then randomly placed in the remaining void space;
4) Horizontal flip with a 50% chance to happen;
5) Contrast from 0% to 100% increase (1 to 2 factor increase). For validation and test the contrast is also augmented to a fixed increase of 50%. This has been done as we think that every sample is too low contrast so an overall increase would be beneficial for the prediction. This way validation and test samples should be placed as far as contrast is concerned in the mean point comparing to training ones.

E. Dataset implementation

1) Dataset implementation for the segmentation model: Input and annotations images paths are loaded into two Datasets. With a map operation a read˙images function is applied to each Dataset and png one-channel images are loaded as float32. The two Datasets are then zipped into a single Dataset. A shuffle operation is done if this is a train dataset creation. The Dataset is then batched with batch size of 8, cached and prefetched.

2) Dataset implementation for the prediction model: Dataset creation for the prediction model is far more difficult as it should deal with data augmentation and this force to change the caching system. First we read the csv file with dataset information. We then create two Datasets, the first one with input image path data and sex information, the second one with bone age information. Then using the map operation a read function is applied to the first Dataset and png one-channel images are loaded as float32 and then normalized to -1.0 to 1.0. If in train mode now the Dataset should be cached and preprocessed using the map operation on the

(a) Original hand image (b) Predicted hand mask image (c) Cleaned hand image

Fig. 1: Images of 512x512px size throughout preprocessing

preprocess function (details in Sec. III-D2). If in test mode the Dataset should be preprocessed using the specific test preprocessing (only contrast is modified). Now the second Dataset is normalized (min max scaling) using the normalize˙y function and cached if in train mode. The two Datasets are now zipped together and shuffled (with a buffer size of 0.25 the size of the dataset) if in train mode, otherwise the all Dataset is cached. Finally the Dataset is batched with batch size of 16 and prefetched.

IV. Processing Pipeline

The approach we took was inspired by those presented in the Sec. II, more specifically:

- From approach 1 the integration of the Inception V3 model with sex data was derived. Also the image size and hyper-parameters were initially used, to be then ticked and personalized.
- From approach 2 the addition of the contrast enhancing preprocessing techniques was taken.
- From approach 5 the preprocessing pipeline using a U-net architecture to segment the dataset was derived.

The used processing pipeline could be easily divided into two main blocks:

A. Preprocessing block

The problem with available data was that images were too dirty. Many of them were digitalized versions of radiographies where the borders of the first one were not the borders of the second one, labels with text and background noise were present in the input images. This could reduce the ability of the prediction block to predict accurately the bone age. So the idea was to segment the hand with this block and with the segmentation mask eliminate all the disturbing elements outside the hand's outline.

The model chosen for this task was the U-Net architecture which is a convolutional neural network that was developed for biomedical image segmentation at the Computer Science Department of the University of Freiburg [6]. It was chosen for its focus on biomedical images and for its need for very few annotated images. That was a crucial features as the original dataset did not provided us segmentation annotations, so those should have been done manually by us (details in Sec. III).

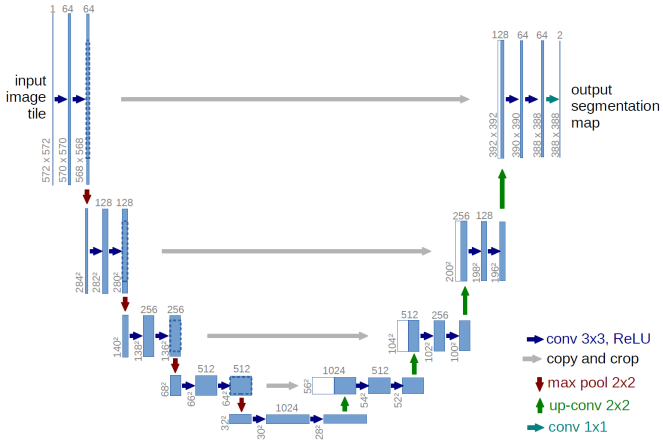

Fig. 2: U-Net architecture

B. Prediction block

Information given as input are radiographies (512x512px images) and sex genre (boolean value) and the bone age (in months) as output. We have chosen to use Inception V3 to deal with image data, completely retrained from scratch. The tricky part is to balance correctly the image information with the genre data. In Fig. 3 we can see the last layers of the combination of the two data information. On the left there are the two last Inception V3 layers. The last one is flattened from (None, 7, 7, 2048) to (None, 100352). On the right genre information is given to the network as a binary information (Male = 1), then it is computed with a 32 neurons layer. The two branches are combined into a single (None 100384). Two dense layers follow the concatenation. The ratio between

100352 and 32 is the one determining the importance of the sex information over those taken from the image.

An important aspect to highlight is that radiographies are grayscale images. So to use a pre-trained Inception V3 on ImageNet dataset the grayscale image should be replicated on the three RGB channels identical, all done on the fly to not multiplicate by 3 the cached dataset. Another approach is to feed the model with only the grayscale one-channel image forcing us to train the system from scratch. We tried both ways and concluded that the pre-trained Inception V3 model fed with a 3-channel image performe quite better than the one fed with a 1-channel image.

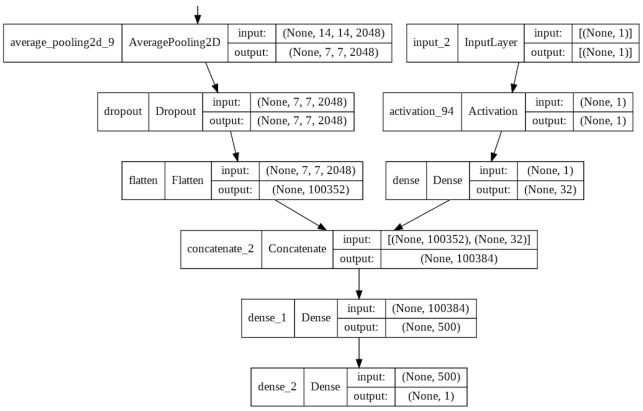

Fig. 3: Last layers of the combined prediction model

C. Overall pipeline

In this Section we will show from start to finish the data and model pipeline describing in a textual way the schema present in Fig. 4.

As said in more details in Sec. III-C1 original input images are big files. First step was to preprocess (in figure "Image preprocessing 1") the entire dataset (training, validation and test) reducing it to 512x512px images. A subset of the training dataset (500 images, of which 400 for training and 100 for validation) had been segmented and then used to train the segmentation model. Using the segmentation model and the original 512x512px images we created masks for the all dataset. With original 512x512px and masks just created we were able to clean the dataset (in Figure "Image preprocessing 2"), obtaining 512x512px cleaned images. Those were used, in combination with age and sex information taken from csv provided files to train the prediction model. The whole structure could be integrated more, with the segmentation model producing segmentation mask on the fly during the prediction model training. We decided to keep the two models completely separate for these reasons:

- To make easier to run training and preprocessing in different computing environment;
- To parallelize design and experimentation with both models at the same time;
- To allow future work on one the two models with no issue on the other as they are completely independent.

V. Training

A. Computing environments

We used three computing environments:

- Google Colab Pro with about 25GB of memory and NVIDIA Tesla P100. Used for training the segmentation model and the last training of the prediction model;
- Google Colab Pro+ with about 50GB of memory and NVIDIA Tesla P100. Used for heavy experimentation for the prediction model;
- Apple Macbook 14" equipped with M1 Pro (8CPU cores and 14GPU cores) and 16GB of memory. Used for preprocessing images (downscaling to 512x512px) and to segment the dataset with the already trained segmentation model.

B. Segmentation block training

Optimizer used for segmentation model training was Adam, our go to choice, trying to reduce the binary cross entropy. The starting learning rate was 1e-02 and it was controlled during training by the callback ReduceLROnPlateau which reduces the learning rate by a 0.8 factor with patience and cooldown of 20 and 10 respectively.

Other two callbacks were used, ModelCheckpoint and CSVLogger. The first one to save a model checkpoint at each iteration (only if the new model was performing better than the last saved one), the second one to save loss and measures at each epoch in a csv file.

C. Prediction block training

Optimizer used for segmentation model training was Adam, trying to minimize the Mean Absolute Error. We created a custom loss function to rescale the MAD in months as the NN output is between 0 and 1. The starting learning rate was 1e-03 at the beginning, but was reduced at 1e-4 at epoch 100.

Two callbacks were use, as before, ModelCheckpoint and CSVLogger. Also ReduceLROnPlateau was tried but with worst results. Maybe having more time to fine tune its parameters would improve results.

VI. Results

A. Results for the segmentation model

Loss monitored on both training and validation was binary cross entropy but it was also tracked Intersection over Union (IoU) as it gives as an easy and clear way to interpret and measure the result. Training epochs limit was set to 2000 and the train process was left unmonitored, but as you can see in Fig. so many epochs were not needed. Training could be stopped at epoch 600-700 without any loss of accuracy. Validation IoU follows easily training IoU, making training easy and straightforward. At epoch 700 binary cross entropy is 0.00099 and 0.06026 and IoU is 0.97251 and 0.96661 for training and validation respectively. We are pretty sure we could improve this result implementing data augmentation even in the segmentation model.

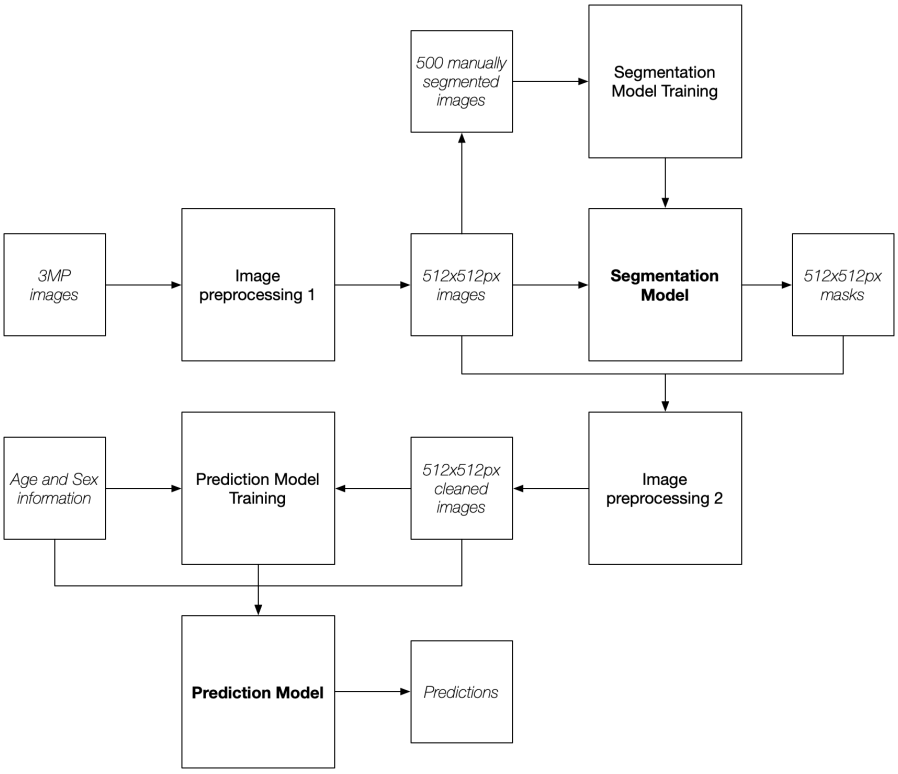

Fig. 4: Overall model pipeline

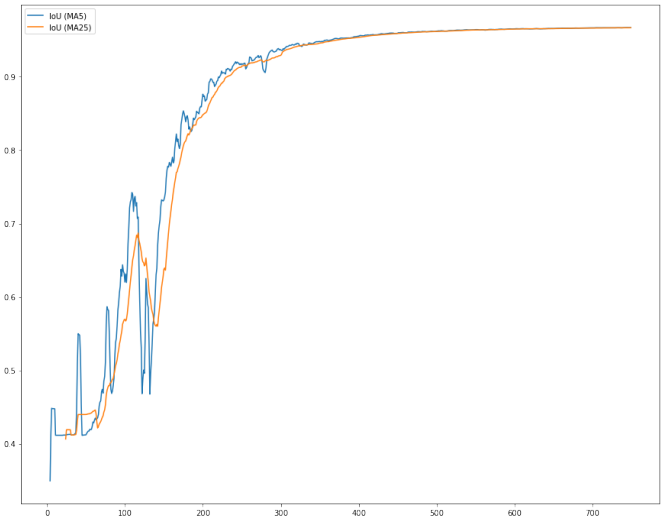

Fig. 5: Segmentation model training (0-700 epochs)

B. Results for the prediction model

Training was stopped after 360 epochs in this trial as monitoring validation we understood that any further improvement was not possible. Actually very small improvements on the validation loss were found after epoch 200-250, as it can be seen in Fig. 6.

Validation loss achieved was 6.6398 and test loss was 5.3018. From [3] we know that test loss for human reviewers was from 6.36 to 8.28 months with a mean of 7.32. This should be considered a success as it shows how automatic deep learning models are able to exceed human accuracy. Our model was able to reduce error from 17% to 36% (1 to 3 months) in months, considering the best and worst reviewer respectively.

We also think that training the same model with more computing resources and time, having the possibility to experiment with hyper-parameters tuning and slightly different architecture this pipeline and model could also compete with winning approaches, despite the already great closeness.

VII. Concluding Remarks

A. Further improvements

In every project we make we would like to continue improving the model, but even more so in context where deep learning on images is applied, time constraint is limiting. For this reason we are used to include in the report a section where to list future possible improvements for ourselves or for the reader.

- Data augmentation in the segmentation model. This could improve the segmentation model a lot as it is trained on only 400 images which, for a neural networks mode, are very little;
- Experiment with data augmentation parameters of the prediction model. There are a lot of parameters like rotation, zoom, contrast range to change and study which could improve learning;

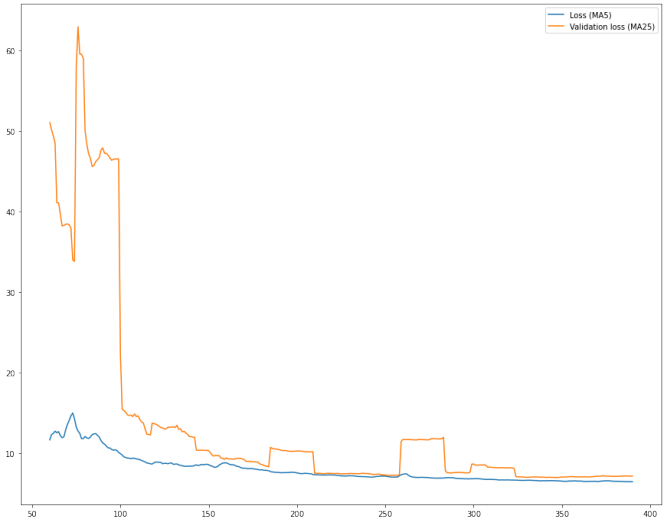

Fig. 6: Prediction model training (30-360 epochs)

- Fine tune the model changing structural things (e.g. last layers of the prediction model, size of the input images) and finding better values for hyper-parameters (e.g. learning rate, scheduler parameters, threshold to consider pixels belonging to the mask and more);
- Integrate the segmentation mask production with the prediction model. This would allow to give the model a noisy image and clean it and predict the bone age simultaneously;
- Integrate into the training phase TensorBoard to ease the training letting the user monitor loss and change parameters during training.

B. Difficulties

Another important aspect to think about when writing a report at the end of some type of project is to realize which had been difficulties found during the project. This allows us to focus on them in the near future.

- Probably the biggest difficulty encountered in this project was the size of the dataset. Images for training are a lot and they are pretty large. Being a computer vision problem, this increases the difficulty as we have to deal with images which are a very intensive data to process from a computational point of view;
- This was the first time dealing with Tensorflow and some difference between PyTorch and Tensorflow were at first problematic as PyTorch is a more pythonic framework;
- To have an affordable way to train a complex network, Google Colab was the only available choice, but not the easiest and more reliable one. Google has the right to disconnect the virtual machine any time or to give us a less powerful machine. For this reason we implemented a checkpoint system which allowed us to restart the training from any epoch the training was left. Even with this method, training was difficult and longer than the normal length in hours as during nights outages were not monitored;
- Dataset implementation being quite different from PyTorch way of doing it, was a bit problematic at first. Anyway we decided to focus the study on it as we thought that making it the right way could improve performances a lot;
- Being training so long and problematic we were in difficulty to fine tune the model trying different architectures and different hyper-parameters. That was a peaty as we think that this has negatively affected performances and the possibility to reach higher results.

C. Things learnt

- First of all Tensorflow was the big new skill obtained with this project. We think that you learn a new framework only when you try yourself on a real life problem and that was the case. Tensorflow Dataset implementation was a particular topics we focused on;
- An architecture for biomedical image segmentation, U-Net, not known before, very useful in many scenarios;
- Writing a report in a more paper-like form;
- How to deal with dataset that could not be contained entirely in memory and more in general how to deal with big sized data inputs;
- This was the first challenge we took part in. Even though we had at our disposal other teams' approaches and we did not have the same time constraint as of the challenge the setup and data were the same.

References

[1] 16Bit.ai, "Machine learning and the future of radiology: How we won the 2017 rsna ml challenge," 2017.

[2] S. S. Halabi, L. M. Prevedello, J. Kalpathy-Cramer, A. B. Mamonov, A. Bilbily, M. Cicero, I. Pan, L. A. Pereira, R. T. Sousa, N. Abdala, *et al.*, "The rsna pediatric bone age machine learning challenge," *Radiology*, vol. 290, no. 2, pp. 498–503, 2019.

[3] D. B. Larson, M. C. Chen, M. P. Lungren, S. S. Halabi, N. V. Stence, and C. P. Langlotz, "Performance of a deep-learning neural network model in assessing skeletal maturity on pediatric hand radiographs," *Radiology*, vol. 287, no. 1, pp. 313–322, 2018.

[4] S. I. P. William Walter Greulich, *Radiographic Atlas of Skeletal Development of the Hand and Wrist*. Stanford University Press, 1959.

[5] "Labelbox," 2022.

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.