Bone Age Assessment on Parsimonious CNN Architectures

A Comparative Approach on Simple Pre-Processing Pipelines

Andrés Espinal[†], Sergio Postigo[‡]

***Abstract*—Bone Age Assessments (BAA) are an important diagnostic tool to assess the normal growth of bones. Historically performed through manual comparison against predefined standards (which is often considered tedious and time consuming), recent developments have introduced automated techniques based on Convolutional Neural Networks (CNN's) to BAA. These approaches have proved to be a feasible alternative with similar precision to BAA's done by humans and have allowed individuals without domain knowledge to contribute to the field. However, the best approaches usually use neural networks with a vast amount of parameters, which yields to long training times specially when carried out in non-specialized hardware. In this paper, we explore the use of a simple pre-processing pipeline to perform BAA, and compare its performance against 2 Convolutional Neural Network Architectures, a parsimonious *VGG* and a complex *Inception-V4* architecture with more parameters. The findings in our study hint to the fact that the role of data pre-processing is more important on architectures with fewer parameters than it is for complex ones. Furthermore, we have acquired insights that show that simpler architectures can achieve competitive results in less time.**

***Index Terms*—Neural Networks, Convolutional Neural Networks, Supervised Learning, Image Segmentation, Inception V4, VGG**

I. Introduction

Bone Age Assessments (BAA) are an important tool to identify growth anomalies by using X-Ray images to identify differences in growth of bones amongst chronological ages [1]. Historically, this has been done performing manual comparisons, but it is evident that BAA performed through Convolutional Neural Networks (CNNs) can have an accuracy similar to that done by manual methods [2].

In the past, Bone Age Assessments have been done manually either through *Greulich and Pyle's* method which compares the X-Ray of the patients to an atlas, or the *Tanner-Whitehouse* method which uses a scoring system that analyzes 20 specific bones [3]. However these manual methods are often criticized as being *"tedious, time-consuming, and limited by considerable interrater and intrarater variability"* [2]. Additionally, specialized knowledge is required to perform BAA using traditional methods.

It is evident that Machine Learning can play an important role in automating BAA solutions. However it also has its share of setbacks when attempting to achieve human-levels of precision. For instance, models that take a long time to run, such as those proposed by Lee et. al. [3], the complexity required in some pre-processing pipelines to perform image segmentation like the ones on Iglovikov (2018) [1] and Halabi et. al [4] and the use of specialized hardware (Namely, one that allows GPU acceleration). Finally, choosing the right architecture for the problem can be puzzling due to the lack of benchmarks for specialized tasks.

Our motivation with this study is to explore alternatives to be able to run parsimonious CNN models with simple pre-processing pipelines with equivalent results to those achieved by more complex models, in less time and through less computational cost. To achieve this, we explore the use of a simple pre-processing pipeline to perform BAA, comparing a parsimonious VGG architecture against a complex Inception V4 architecture and reflecting on the role of pre-processing in the precision of CNN's.

This paper is structured as follows: In *Section II* we review the literature and current state of the art. *Section III* provides a high-level overview of the pre-processing pipeline employed. *Section IV* elaborates on the pre-processing pipeline details, the data segmentation strategy used and the inputs for the CNN's. *Section V* describes the CNN's configurations tested. Finally, we wrap up with the results of our research and conclusions in *Sections VI* and *VII* respectively.

II. Related Work

Several attempts have been done to solve Bone Age Assessment tasks using Convolutional Neural Networks, to varying degrees of success. In current literature, it seems that the best performing approaches implement *pre-processing pipelines* to denoise the x-ray images and highlight important features of the bones. Furthermore, *resolution* seems to have a significant impact on model performance but a detrimental effect on pre-processing tasks and training time that needs to be balanced. The choice of which *CNN architecture* to use is of vital importance as well, as some architectures are best tuned for specific digital imaging problems and the choice determines the total amount of parameters to be estimated, and hence training times. Finally, many of the best performing approaches implement *Data Augmentation* to further increase the training examples with random flips, crops, contrast modifications and other image transformations.

[†]Department of Information Engineering, University of Padova, email: andresgabriel.espinalhernandez@studenti.unipd.it

[‡]Department of Information Engineering, University of Padova, email: sergiorenato.postigohuanqui@studenti.unipd.it

Many of the techniques discussed on this paper are inspired on these findings.

Larson et. al (2019) [2] implemented a deep residual network architecture composed of 50 layers and trained through TensorFlow. Results were compared against clinical ratings obtained through the Greulich and Pyle standard (Radiographic Atlas of Skeletal Development of the Hand and Wrist) by a series of reviewers. The study concluded that *the mean bone age estimated by the model was not significantly different from that estimated by any of the reviewers*. Training was performed on downsized 256x256 pixels images that were enhanced using *contrast-limited adaptive histogram equalization*. Even though the precision of the models was good, one limitation of this study was that training times ranged between 6-8 hours, running the model with GPU acceleration. In this study, we propose an alternative approach that sacrifices performance in favor of faster training times through a VGG architecture.

Halabi et. al (2019) [4] analyzes the results of *The RSNA Pediatric Bone Age Machine Learning Challenge*, in which contestants employed Machine Learning Techniques to perform Bone Age Assessments. Contestants used different approaches, of which we remark the 1st place; which used 500x500 pixels images trained through an Inception V3 architecture with gender. And the 2nd place, which splitted the original image in 224x224 pixels contrast enhanced patches and trained through a R*esNet-50* architecture. This study hints to the possibility that using a higher resolution might improve the performance of the models, as the first two places implemented an approach that provides higher resolution images to the network, either by resizing or by cropping (Which increases the bone-to-background ratio in the outputted resolution of the image). In this paper we attempt to combine both approaches by cropping the bones to increase bone-to-background resolution in a pre-processing pipeline that later feeds these outputs a CNN with Inception V4.

Lee et. al (2017) [3] used 512x512 pixels images trained through a *LeNet-5* Architecture and used a detection mask for hands using a CNN (*U-Net*) to split into hand and no-hand regions. The error in this study was reported to be between 1 and 2 years. Furthermore, they used the *Input Occlusion Method* and *Attention Maps* to find which parts of the image were significant for classification and found that the features used by the CNN were similar to those being used through manual methods to assess the age of bones. Even though the segmentation mask used was effective, it required to manually label the training set, which can be laborious. We propose an alternative method to increase bone-to-noise ratio without using CNN's, using simple image pre-processing techniques like thresholding and filtering to detect rectangles around the hand at the expense of allowing some noise into the model.

Iglovikov et. al (2018) [1] also implemented a hand mask using a CNN (*U-Net*) to extract the hand and remove extraneous objects, but unlike Lee et. al (2017), did this through *positive mining* (An iterative procedure that uses a combination of manual labeling and automatic processing) to alleviate the burden of doing purely manual labeling. They implemented a *VGG* architecture with 6 convolutional blocks followed by 2 fully connected layers and performed data augmentation. The MAE evaluated was of 4.97 months, and the authors conclude that the approach can be improved by using more powerful deep networks. Our approach attempts to extend on this work by comparing the performance of our simplified processing pipeline against both *VGG* and *Inception V4* architectures.

III. Pre-Processing Pipeline

Our proposed data preprocessing pipeline is illustrated in Figure 1. The purpose of this pipeline is to increase the bone-to-background ratio of the outputted image. It is composed of 4 steps that are run sequentially, which are:

- **Cropping Module:** A source image in its native resolution is fed to the pipeline and is processed through our cropping module. This module applies a series of transformations to make the outermost contours of the hand more visible and selects the biggest contour found in the picture. After this a decision is made on whether to crop the image based on this contour or not. If the area of the contour is made up of more than 10% of the total image resolution a bounding rectangle is fit to be used for cropping, otherwise the original image is returned. This allows us to consistently filter out artifacts like tags from the image.
- **Contrast Enhancement Module:** The cropped image is fed to this module to perform *Contrast Limited Adaptive Histogram Equalization*. This step is done only after cropping, to allow the enhancement to take into account a higher bone-to-background pixel ratio. The output of this step is an image were bone visibility is enhanced, and the image is returned in the original resolution of the crop for the next step.
- **Fill Module:** After contrast enhancement has been performed, the image is padded to account for differences between the horizontal and vertical dimensions. The output of this step is a rectangular image preserving the original aspect ratio of the crop and the dimensions of the biggest axis. Missing information is filled in with zeros which yields an image with a black border as padding.
- **Resizing Module:** After all transformations are done, a final resize of the image is done to 250x250 pixels to standardize the resolution of the images fed to the model. Several resolutions were tested for this module, 250x250 pixels images seemed to work best based on the models and the hardware used in this study.

All transformations were performed on each of the individual images prior to training using Python's OpenCV module for the training, validation and test folders splits. Execution time of the pipeline for all images took around 9 minutes to process. The resulting images were converted to a single color channel image to prevent the CNN to process unnecessary information (as x-ray images are mostly composed of grayscale). Results were exported to a final folder used by the models. Standardization of the image was performed internally

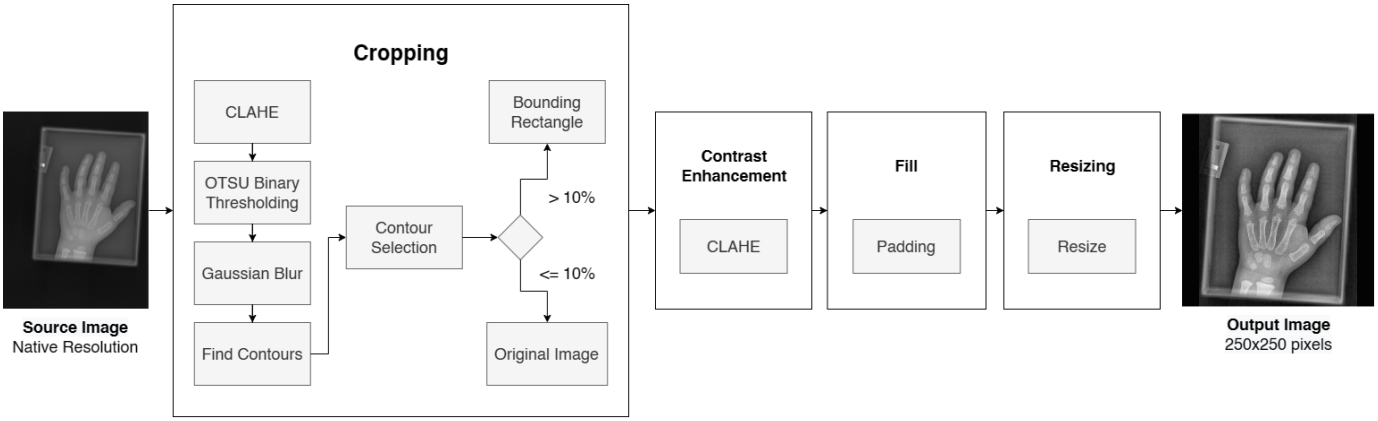

Fig. 1: Pre-processing Pipeline

in the models as an initial stage prior to training. An example comparing the original image against the pre-processed one can be appreciated in Figure 2.

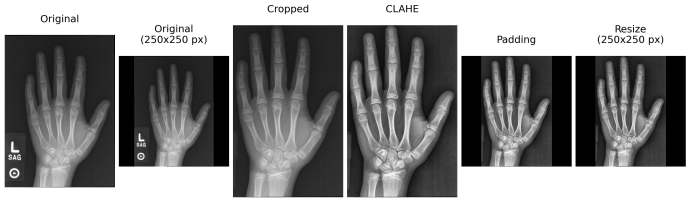

Fig. 2: Example of the Pre-Processing Pipeline Stages

IV. Signals and Features

A. Data Quality Assessment

An exploratory data analysis performed on the training dataset revealed that image quality had high variability, which further complicated the development of a standardized solution for the pre-processing pipeline. The following specific problems were found after EDA:

- **Artifacts:** Overlayed rectangles, tagging labels, casts, catheters, both around and on top of the hand. Furthermore, in some of the images, both hands were visible.
- **Contrast Differences:** Bones overlayed in white backgrounds, dull and dark images, images mostly composed of gray.
- **Proportions:** Images with a hand area significantly smaller with respect to non-hand areas.
- **Resolution:** Different resolution and dimensions of images.
- **Rotation/Mirroring:** Images in different angles, or flipped. This is a problem as BAA only uses left-sided radiographs by convention [3].

B. Solution to Data Quality Issues

Noisy data tends to have a noticeable impact on the performance of CNN's, so we applied a series of pre-processing techniques to address most of these problems through the pipeline presented in Section III. For an illustration of the techniques applied refer to Figure 3. Specifically, the following was done:

- **Artifacts** were removed to an extent by performing cropping through *image segmentation*, which is a technique used to crop images based on a contour. Furthermore, this caused a zoom-in into the picture which solved most **proportions** problems and allowed to account for more bone information in the outputted **resolution**. Several approaches were tested, to varying degrees of success:
  - **Extracting the hand through an outline drawn after *thresholding* and *Gaussian blurring*:** Thresholding allows us to simplify complexity in the image by transforming pixels above a threshold to one color, which aids in detecting hand and no hand regions. Due to the presence of small artifacts in some images, blurring the thresholded mask allowed better recognition of finger tips and filtering of the artifacts. Even though this method managed to recognize the hand outline pretty well, it proved to be inconsistent as finger tips were being clipped in a few examples, even after performing CLAHE.
  - **Fitting an angled bounding-rectangle to the outline extracted in the 1st method:** This method built on top of its predecessor by fitting a rectangle to the detected outline. Since many of the images exhibited hands inside rectangles it proved to be the method that resulted in the best bone-to-background ratio, and it also detected the slope of the hand for images that weren't boxed. However, since the angled bounding box could span an area bigger than the image it added additional artifacts.
  - **Fitting a bounding rectangle disregarding rotation to the 1st method:** The same technique is applied but the bounding box is constrained to remain parallel to the image axis. This method proved to be the most consistent in zooming into the hand and cropping area outside it without clipping. This was the final method chosen for the pipeline. A CLAHE was performed with a clip limit of 2

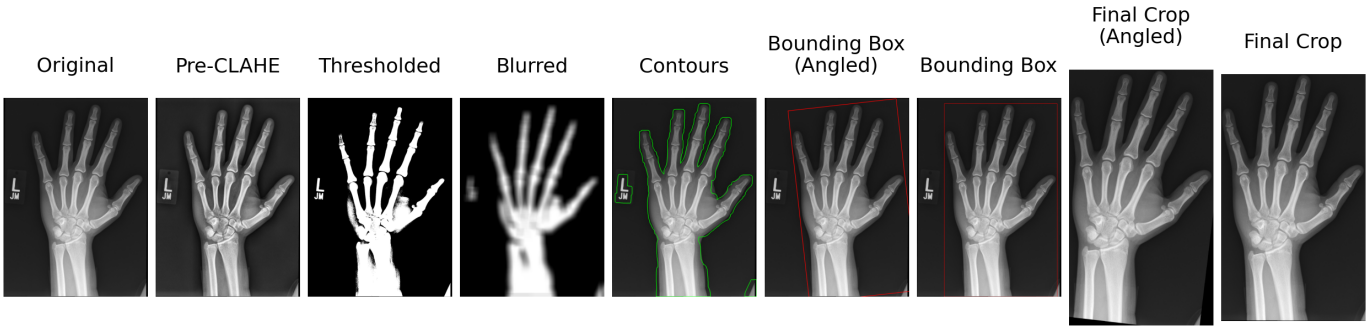

Fig. 3: Overview of Cropping Technique

and a tile grid size of 17, prior to gaussian blurring with a kernel size of 73x73 and a standard deviation of 17. After this a thresholding step was carried on using binarization through the Otsu method to automatically find the best value for the threshold. Finally the outline was detected through the chain approximation rule and the bounding box was used to crop the image.

- **Contrast differences** were massaged by performing *Contrast Limited Adaptive Histogram Equalization (CLAHE)* as a final part of the pipeline, to improve contrast of the image only after removing borders. One thing to remark regarding this step is that Histogram Equalization introduces noise to the image when the enhancement level is strong. A Clip Limit of 2.5 with a Tile Grid Size of 50x50 proved to be a reasonable trade-off between enhancing the visibility of bones and introducing noise for our specific scenario.
- **Rotation/Mirroring** wasn't taken into account for our study as we observed a relatively small number of examples present in the training data. However, Halabi et. al (2019) [4] mentions that methods that *"randomly rotated the images seemed to be less prone to errors when images were rotated in the test data set"*.

C. Model Inputs

The models were tested against two datasets, a baseline of the original images and their pre-processed counterparts. This was done to understand if pre-processing actually had a positive impact in model performance.

- **original_r250p:** This dataset consists of the original images without any modification besides padding them with a black background to transform them into a 1:1 aspect ratio. These images were resized to 250x250 pixels before being passed to the model.
- **preprocessed_r250p:** Contains the images with all the enhancements explained in Sections III and IV. These images were also resized to 250x250 pixels in the final step.

D. Data Splits and Training Features

The input data for this research is composed by three elements: the images with the x-rays of hands, the genders and age labels of the corresponding persons. It is provided in three parts:

- **Training Data:** a folder with 12,611 x-ray images and a CSV file with the genders and age labels associated with the images.
- **Validation Data:** a folder with 1425 x-ray images and a CSV file with the genders and age labels associated with the images. Note that originally, two folders of validation where provided but we decided to merge them into a single one.
- **Testing Data:** a folder with 200 x-ray images and a CSV file with the genders and age labels associated with the images.

Besides the methods listed in Section 1, approaches to further extract additional features from the hand images where analyzed to be included as features for the CNN. The most promising one was to get the hand landmarks and the distances between them using the MediaPipe library from Google [5]. However, the high variability of the images made this unfeasible. Thus, it was decided to continue without further features.

E. CNN Pipeline

In the pre'processing stage, the images were reduced to a resolution of 250x250x1.The decision of re-scaling the images was done based on memory limitations of working with Tensorflow and GPU acceleration. The pipeline was defined as follows:

1) Create a tf.data.Dataset object composed by:
   - **explanatory variables:** a tuple of tensors containing the images and the gender (0 for female and 1 for male).
   - **target variable:** a tensor containing the ages labels.
2) Shuffle the content of the tf.data.Dataset object so that the data is pipelined in a random order.
3) Batch the tf.data.Dataset object so that the GPU doesn't need to load all the data in memory at once but in batches. The batch size will vary according to the model, as it is shown see lines below.

After this, the data was ready to be ingested into models. These are discussed in the following section.

V. Learning Framework

This research aims to explore the application of Deep Convolutional Neural Networks (CNN) to obtain the age of a person based on x-ray images of the hand and gender information. Previous works showed that this is feasible but usually require demanding networks of tens of millions of parameters. The goal in our study is to compare the performance of a network with a low number of parameters against a network with a large number of parameters. The first one is a reduced version of the original VGGNet proposed by Simonyan and Zimmerman, with only 6 convolutional layers [6]. The second one is the Inception-V4 Network proposed by Szegedy, Ioffe, Vanhoucke and Alemi [7]. The details of each of these are presented in the following lines.

VGG-style network (VGG-6): VGG-16 and VGG-19 are networks of 16 and 19 convolutional layers both composed by kernels of size 3x3. They end with two fully connected layers [6]. The approach proposed in this research is inspired in this work since it uses the same principles butreducing the number of convolutional layers to only 6. This change decreases considerably the number of parameters. For simplicity, we will refer to it as VGG-6. The schema of this network is shown in Figure 4. Just as in the original VGG

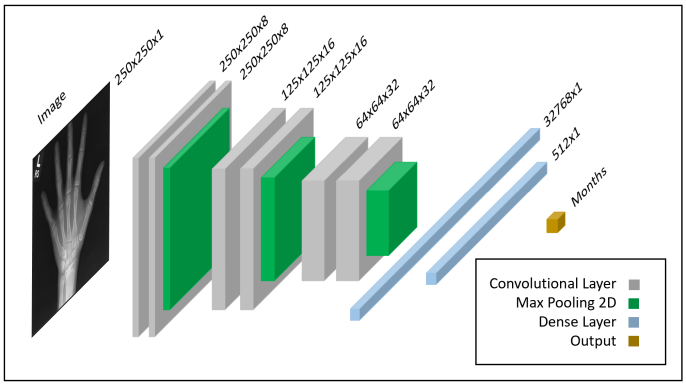

Fig. 4: VGG-6 schema

networks, 3x3 kernels are used in every convolutional layer along with "ReLU" activation functions. Additionally, in every pair of convolutional layers of same dimensions there is a normalization step to set different features to an equivalent scale. Furthermore, after every dense layer a dropout of 20% is applied to avoid over-fitting and make the model balance the contributions from all the neurons. Finally, since this is a regression problem, the output is obtained with a linear activation. For an input image of 250x250x1 this model has to learn 1,067,785 of parameters.

Inception-V4: This is a network composed by several blocks with different configurations of convolutional layers and normalizations with "ReLU" activations. The schema of this network is presented in Figure 5. This network ends with a dense layer with a dropout of 20% that feeds a linear activation

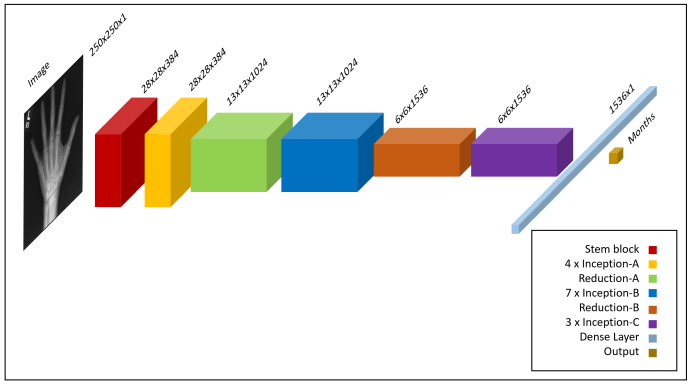

Fig. 5: Inception-V4 schema

in the output. For an input image of 250x250x1 this model has to learn 41,175,361 of parameters. Both of the described models were used to predict the age based on the images only. However, the dataset contains a gender variable, which may also be useful to improve the predictions. Thus, we combined this variable with the presented models as shown in Figures 6 and 7.

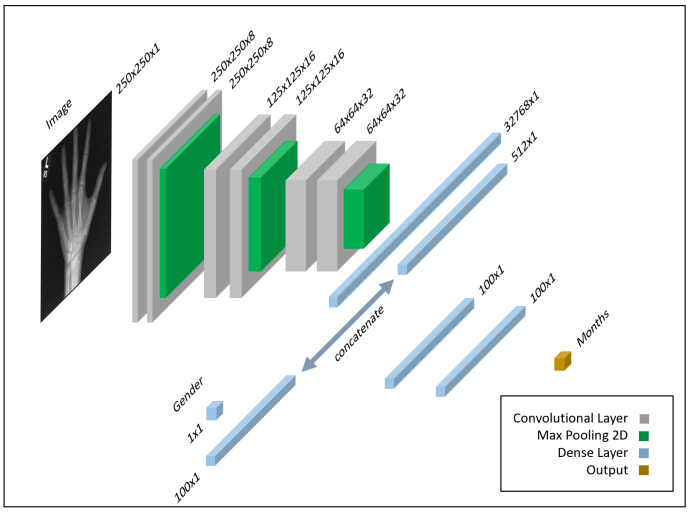

Fig. 6: VGG-6 and gender combination schema

As described in the previous section, the gender variable is 0 for female and 1 for male. It is first converted into a dense layer of 100 neurons, which is concatenated with the dense layers from the VGG-6 and the Inception V4 respectively. After the merge, two fully connected 100 neurons layers are added to finally perform the linear activation in the output to predict the age. In summary, there are 4 different models in total for this research:

1) VGG-6 (1,067,785 parameters)
2) VGG-6 with gender (1,138,973 parameters)
3) Inception-V4 (41,175,361 parameters)
4) Inception-V4 with gender (41,347,925 parameters)

In order to compare their performance, the same optimizer, loss, learning rate and number of epochs was used for all the models. The optimizer chosen was Adam, the number of

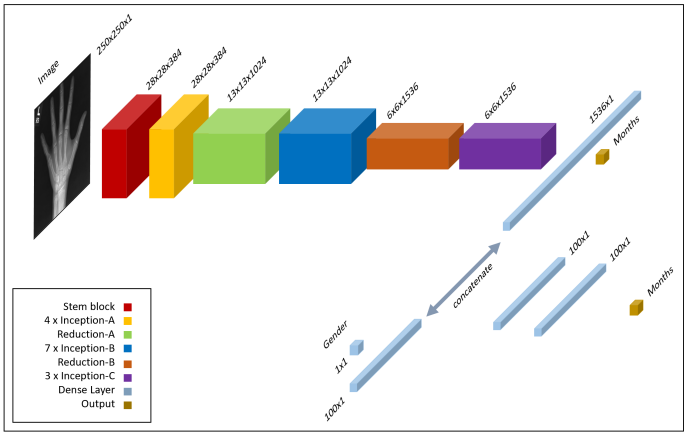

Fig. 7: Inception V4 and gender combination schema

epochs 75, the loss function the Mean Square Error (MSE) and the learning rate set to 0.001 which was set to decrease to 0.0001 after epoch 60. The batch size for models 1 and 2 was 32 and for 3 and 4 was 8. The difference relies on the higher complexity of models 3 and 4. With about 41 times more parameter than model 1 and 2, they required more space in the GPU memory to store the activations that would be used during the backward computation and thus, a smaller batch was used to not collapse the memory.

Finally, the four models were trained using the two preprocessed sets of images: original_r250p and preprocessed_r250p. So, in total 8 different experiments are performed:

- VGG-6 taking images from original_r250p
- VGG-6 taking images from preprocessed_r250p
- VGG-6 with gender taking images from original_r250p
- VGG-6 with gender taking images from preprocessed_r250p
- Inception-V4 taking images from original_r250p
- Inception-V4 taking images from preprocessed_r250p
- Inception-V4 with gender taking images from original_r250p
- Inception-V4 with gender taking images from preprocessed_r250p

The models were run using 2 laptops with the same Hardware configuration. Namely 2 laptops running Windows 11 on 16GB of Ram, with a dedicated NVIDIA GeForce RTX 3050 Ti graphics card with 4GB of VRAM and an i7-11800H CPU. To see the performance of the models in real-time, the Root Mean Square Error (RMSE) metric was included in the model fit instruction. This represents the average error of the predictions in months. Additionally, after every epoch, the validation dataset was tested with the current parameters, allowing to analyze if the model is overfitting or generalizing well.

With all these settings and configurations established the tracking of the experiments was performed and monitored using Tensorboard (visualization toolkit from Tensorflow), where different metrics and graphs were generated, as it will be presented in the following section.

VI. Results

In this section, the metrics of each of the experiments are provided. Furthermore, the performance of the different models is compared to see how they behave with different configurations and data.

TABLE 1: Experiments metrics

Model	Data		Time	RMSE	
	orig.	preproc.		train	valid.
VGG-6	x		1h 20m	9.903	18.87
		x	1h 20m	9.608	17.43
VGG-6 w/ gender	x		1h 9m	6.406	17.23
		x	1h 11m	5.318	13.85
Inception-V4	x		11h 29m	2.571	14.48
		x	11h 30m	2.362	13.69
Inception-V4 w/ gender	x		10h 37m	1.266	12.29
		x	9h 49m	1.508	12.02

Table 1 summarizes the metrics captured in each of the experiments. In particular, the time it took for the model to train for 75 epochs and the Root Mean Square Error (RMSE) both for the train and the validation data. This last metrics quantifies the prediction error in months. Every model is tested with the two versions of data obtained from the preprocessing stage: original_r250p and preprocessed_r250p. From this results, a series of remarks will be covered in the following lines.

Remark VI.1. Adding the gender explanatory variable to the model reduces the prediction error both in VGG-6 and Inception-V4. However this reduction is larger in VGG-6 when training with the preprocessed data. As shown in Table 1 adding the gender improves the RMSE from 17.43 to 13.85 months. In case of Inception-V4, it improves from 13.69 to 12.02 months. This suggests that the parsimonious CNN tends to get more benefits from the preprocessing than the more complex CNN in reducing the prediction error. Furthermore, adding the gender in VGG-6 reduced the error in 2.61 years in average, while in the Inception-V4 it reduced the error in 1.93 years in average.

Remark VI.2. Training the models including Inception-V4 took about 10 times than the equivalent models using VGG-6. As it was mentioned before, Inception-V4 models train almost 41 times the number of parameters than VGG-6 models. Training such a larger amount of parameters improves considerably the RMSE when comparing VGG-6 vs Inception-V4 models except in one case: the VGG-6 w/ gender model vs the Inception-V4 w/gender when trained with preprocessed data. There, the difference in the final validation RMSE is only 1.63 months. This is shown in Figure 8. In that scenario, learning about 40 million more parameters and taking 10x more time led to a small improvement. This suggests that a correct

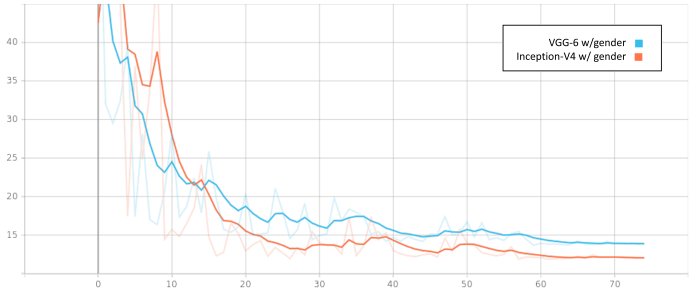

Fig. 8: Validation RMSE comparison with preprocessed_r250p data

preprocessing of x-ray images can remove the necessity of using complex and expensive CNNs.

Remark VI.3. In all the models, training with the preprocessed_r250p dataset showed better results than with the original_r250p. However, the improvements where more significant in the models with VGG-6 than with Inception-V4. For example, Figure 9 shows the validation RMSE per epoch of all the models that include gender. As seen, the models including Inception-V4 don't show a significant difference at the last epoch, which suggests that learning with the data from preprocessed_r250p doesn't improve the performance significantly compared to a training with original_r250p. However, when comparing the two VGG-6 models in Figure 9, it is easy to note the significant improvement of training with the data from preprocessed_r250p compared to the data from original_r250p. These remarks are also valid for the models without gender, as it is shown in Figure 10

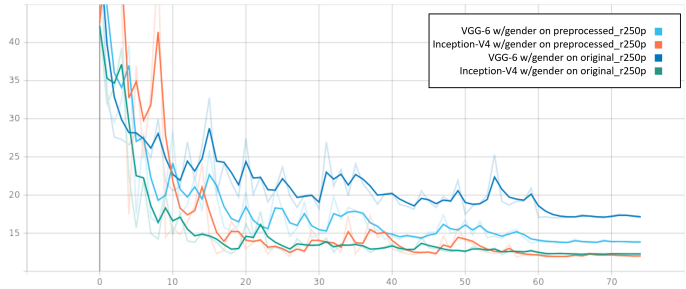

Fig. 9: Validation RMSE for models with gender

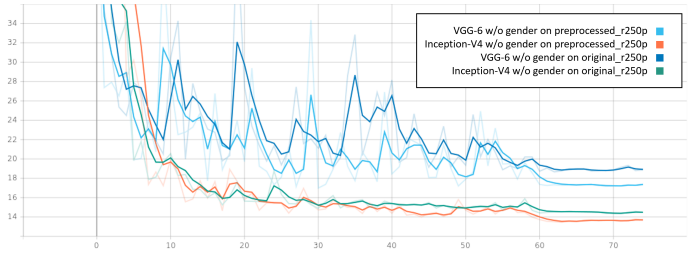

Fig. 10: Validation RMSE for models without gender

To conclude, we tested the models on unseen data provided in the test folder. For this last assessment, we used the Mean Absolute Error (MAE) metric to show the prediction error in months, as it is usually done in the literature explored. From Table 2 it is observed that the model with the lowest

TABLE 2: Assessing the models with test data

Model	Data		MAE (months)
	orig.	preproc.	
VGG-6	x		16.2
		x	14.1
VGG-6 w/ gender	x		15.7
		x	9.2
Inception-V4	x		10.9
		x	10.5
Inception-V4 w/ gender	x		8.1
		x	7.5

age prediction error is the Inception-V4 w/gender trained with the data from preprocessed_r250p. It achieves 7.5 months of MAE. The second best model is the Inception-V4 w/gender trained with the data from original_r250p, giving 8.1 months of MAE. Finally, the third best model is the VGG-6 w/gender trained with preprocessed_r250p, giving 9.2 months of MAE. Of course, this last model has the advantage that it is trained 10x times faster and uses about 41x less parameters.

The code used for the pre-processing pipelines and the models can be found in https://github.com/andresespinalh/Bone-Age-Assessment-HDA.

VII. Concluding Remarks

1) In this paper we have analyzed the impact that data pre-processing has on parsimonious models in comparison to complex models. We have demonstrated that a combination of a good pre-processing pipelines and CNN's with a low amount of parameters can achieve comparable results to those with a vast amount of parameters in the domain of Bone Age Assessments.
2) Through the research done in this paper, we have learned that running CNN models with a high amount of parameters can be tricky due to the amount of hardware resources that need to be allocated. Training models through GPU acceleration is significantly faster than using CPU so this should be favored whenever possible. Furthermore, creating an appropriate input pipeline in the CNN design phase is key to be able to fit all the data in memory through TensorFlow.
3) Training the different CNN's used in this paper proved to be challenging, we found several issues on the way including: a) Finding a pre-processing pipeline that worked well for most of the images, b) calibrating the resolution and batch size of the images to fit in GPU memory, c) Long execution times required for some of the models and d) dealing with NaN's when using Stochastic Gradient Descent as optimizer to train the models.
4) The work on this paper can be extended and improved upon. First, a slightly more elaborate pre-processing pipeline can be implemented by analyzing the histogram

content of the images to perform different contrast enhancements and cropping strategies to better segment hand regions. Second, due to hardware constraints we didn't include Data Augmentation in the input pipeline in our approach, but this is known to improve the performance of the models as explained in Halabi et. al. [4]. Other possibility is to explore a solution that uses Ensemble Learning to combine the predictive power of the models we trained which might also yield smaller MAE values.

REFERENCES

[1] V. I. Iglovikov, A. Rakhlin, A. A. Kalinin, and A. A. Shvets, "Paediatric Bone Age Assessment Using Deep Convolutional Neural Networks," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, (Granada, Spain), Sept. 2018.

[2] 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, pp. 313–322, Apr. 2018.

[3] H. Lee, S. Tajmir, J. Lee, M. Zissen, B. A. Yeshiwas, T. K. Alkasab, G. Choy, and S. Do, "Fully Automated Deep Learning System for Bone Age Assessment," *Journal of Digital Imaging*, vol. 30, no. 1, pp. 427–441, 2017.

[4] S. S. Halabi, L. M. Prevedello, J. Kalpathy-Cramer, P. Artem B. Mamonov, A. Bilbily, M. Cicero, I. Pan, L. A. Pereira, H. H. Thodberg, L. Chen, G. Shih, K. Andriole, M. D. Kohli, B. J. Erickson, and A. E. Flanders, "The RSNA Pediatric Bone Age Machine Learning Challenge," *Radiology*, vol. 290, pp. 498–503, Feb. 2019.

[5] G. Developers, "Hand landmarks detection guide for python."

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014.

[7] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," 2016.