Improving Radiographic Diagnosis with Deep Learning in Clinical Settings

Aloïs Pourchot

THÈSE DE DOCTORAT DE
SORBONNE UNIVERSITÉ

École Doctorale Informatique,
Télécommunications, et Électronique (Paris)

Présentée par :

Aloïs POURCHOT

Pour obtenir le grade de
DOCTEUR de SORBONNE UNIVERSITÉ

Improving Radiographic Diagnosis with Deep Learning in Clinical Settings

---

Soutenue publiquement le 22/09/2022
Devant le jury composé de :

Diana MATEUS Professeure, Centrale Nantes	**Rapportrice**
Caroline PETITJEAN Professeure, Université de Rouen Normandie	**Rapportrice**
Stéphanie ALLASSONNIÈRE Professeure, Université Paris Cité	**Examinatrice**
Christian WOLF Principal Scientist, Naver Labs Europe	**Examinateur**
Kévin BAILLY Maître de Conférences, Sorbonne Université	**Co-encadrant**
Olivier SIGAUD Professeur, Sorbonne Université	**Directeur de Thèse**
Alexis Ducarouge Scientific and Technical Director, Gleamer	**Invité**

Abstract

The impressive successes of deep learning over the course of the past decade have reinforced its establishment as the standard *modus operandi* to solve difficult machine learning problems, as well as enabled its swift spread to manifold domains of application. One such domain, which is at the heart of this PhD, is medical imaging. Deep learning has made the thrilling perspective of relieving medical experts from a fraction of their burden through automated diagnosis a reality. Over the course of this thesis, we were led to consider two medical problems: the task of fracture detection, and the task of bone age assessment. For both of them, we strove to explore possibilities to improve deep learning tools aimed at facilitating their diagnosis. With this objective in mind, we have explored two different strategies. The first one, ambitious yet arrogant, has led us to investigate the paradigm of neural architecture search, a logical succession to deep learning which aims at learning the very structure of the neural network model used to solve a task. In a second, bleaker but wiser strategy, we have tried to improve a model through the meticulous analysis of the data sources at hands. In both scenarios, a particular care was given to the clinical relevance of our different results and contributions, as we believed that the practical anchoring of our different contrivances was just as important as their theoretical design.

Acknowledgements

I would first like to greatly thank the three people without whom the completion of this PhD would not have been possible. Olivier Sigaud, for his infallible backing and encouragements, for believing in me from start to finish, and for guiding me through difficult times. Kévin Bailly, for accepting to effectively take the role of co-supervisor as I was quickly diverging from Olivier's field of expertise, and for the fruitful discussions that emerged from his critical thinking and analyses. Alexis Ducarouge, for the initial idea that led to the onset of this project, for believing that I could see it through, and for helping me reconsider the scope of this PhD at times when I needed to find practical meaning in my work.

I could not have hoped for a better setting to pursue this doctorate than as part of Gleamer. The Gleamer team is full of vibrant and brilliant people, and I am very grateful to be part of it. I would like to warmly thank Gabriel, Nini, Zekun, Vincent, Lauryane, Jeanne, Violette, Chris, Audrey, Léo, Louis-Maxime, Yassine and many others, for their help, support, enthusiasm, and sense of humor. I am thrilled to be able to pursue my adventures with them.

I further sincerely thank Diane Mateus and Caroline Petitjean for accepting to review this PhD, Christian Wolf and Stéphanie Allassonnière for accepting to be part of my defense jury, as well as Ludovic Denoyer and Matthieu Cord for the advice that they offered through the monitoring committees.

Finally, I would like to thank my family and friends for their unconditional love and support.

Contents

Acronyms

AI artificial intelligence.

AUC Area Under the Curve.

BAA bone age assessment.

CAD Computer-Aided Detection.

CNN Convolutional Neural Network.

CNNs Convolutional Neural Networks.

CT Computed Tomography.

DCNN Deep Convolutional Neural Network.

DCNNs Deep Convolutional Neural Networks.

DHAD Digital Hand Atlas Database.

DL Deep Learning.

EDs emergency doctors.

FCNNs Fully-Connected Neural Networks.

FN false negative.

FP false positive.

FROC Free-response Receiver Operating Characteristic.

G&P Greulich and Pyle.

GNNs Graph Neural Networks.

GPU Graphics Processing Unit.

GPUs Graphics Processing Units.

HPO Hyperparameter Optimization.

IOU Intersection Over Union.

MAE Mean Average Error.

MDP Markov Decision Process.

ML machine learning.

MRI Magnetic Resonance Imaging.

MSE Mean Squarred Error.

NAS Neural Architecture Search.

NB-101 NAS-Bench-101.

NEAT NeuroEvolution of Augmenting Topologies.

R-CNN Region Based Convolutional Neural Network.

RHPE Radiological Hand Pose Estimation.

RL reinforcement learning.

RMSE Root Mean Square Error.

RNNs Recurrent Neural Networks.

ROC Receiver Operating Characteristic.

ROI Regions of Interest.

RPN Region Proposal Network.

RSNA Radiological Society of North America.

SGD Stochastic Gradient Descent.

TN true negative.

TP true positive.

TTA Test Time Augmentations.

TW Tanner-Whitehouse.

WS Weight-Sharing.

Chapter 1

General Introduction

In the last decades, the advent of Deep Learning (DL) has led to tremendous breakthroughs in several hard machine learning (ML) problems (LeCun et al., 2015). Tasks which required ML practitioners to acquire and leverage advanced field expertise, can now be readily tackled by experienced DL specialists, owing to the ability of DL models to extract features on their own. This paradigm shift has enabled the ML community to considerably widen their scope of application. The literature of computer vision, a discipline that is interested in the high-level understanding of images, videos, 3D point clouds and volumes through computer programs, has been particularly prolific since the advent of DL. Computer vision researchers have proposed a plethora of DL-based solutions to challenging practical problems in robotics (Arulkumaran et al., 2017), autonomous driving (Grigorescu et al., 2020), image generation (Z. Pan et al., 2019), and perhaps most notably, in medical imaging (J.-G. Lee et al., 2017; Litjens et al., 2017).

1.1 Medical Context

The discipline that uses medical imaging to diagnose and treat diseases is called radiology, and medical doctors that specialize in their analysis radiologists. Modern imaging techniques can help radiologists identify a plethora of pathologies, from fractures and other outcomes of traumatic events, to viral and bacterial infections, or even cancers (Sutton, 1987). In most countries, radiologists must undergo an intense and lengthy medical training and are a scarce resource. Yet, the ever expanding availability and use of medical imaging has considerably increased their workloads, resulting in a shortage of radiologists (Rimmer, 2017) and invariably leading to higher rates of misdiagnosis, a leading cause of morbidity and mortality (James, 2013). Common external sources of error are overstrain induced tiredness or limitations in the time committed to the interpretation of a single exam.

Nonetheless, even in ideal work conditions, radiologists are prone to several cognitive biases (Busby et al., 2018). Notorious examples are satisfaction of search, where a radiologist might stop looking for pathologies after one has already been found, or satisfaction bias, where a radiologist might overlook an exam based on prior belief that a patient is healthy.

Computer software are inherently unaffected by aforementioned impediments. As such, they have long been considered as tools to improve patient care (Castellino, 2005). The availability and performances of those programs has soared with the onset of DL and they are now used in diverse scenarios such as the segmentation of brain lesions (Ghafoorian et al., 2016), the segmentation of lung nodules (Jaeger et al., 2020), the detection of breast cancer (Shen et al., 2019) or the detection of fractures (Duron et al., 2021; Guermazi et al., 2021). Several meta studies suggest that radiologists greatly benefit from the usage of AI tools in their clinical workflow (Kuo et al., 2022; Zheng et al., 2021). AI technologies for medical diagnosis are coming of age and their use in clinical routine is getting increasingly essential.

This work was conducted in partnership with Gleamer, a French company which develops AI solutions to assist radiologists in their daily work. The goal of this PhD was to study several DL-based tools for diagnosis aid, and explore ways to improve them. Two particular medical tasks were of interest. The task of **fracture detection**, which consists in spotting fractures in a patient after a traumatic event, and the task of **bone age assessment**, which consists in estimating the advance of the skeletal development of children.

Both of those tasks are commonly diagnosed through radiography, an imaging technique that uses X-rays to reveal the inner form of a region of interest. To create radiographs (also referred to as X-rays), X-rays are sent through patients around the region of interest. Placed oppositely to the radiation source, a receptor measures the quantity of energy that is absorbed by the subject's tissues. Because this absorption is directly dependent on the density of the anatomy traversed by the beam, the result image produces a 2d projection of the density of the crossed regions. A contrast is thus created between regions of different projected densities. Consequently, bones emerge quite clearly due to the relatively high density of their calcium minerals, as opposed to air and other softer tissues which are mostly comprised of oxygen, carbon, hydrogen and nitrogen. We present in Figure 1.1 examples of radiographs used for the fracture detection and BAA diagnosis.

1.2 Contributions

The contributions of this PhD were separated into the following chapters:

Figure 1.1: Examples of radiographs for the fracture detection and the BAA tasks. On the left, a radiograph of a hip reveals a quite obvious fracture of the femoral head. On the right, the radiograph of a children's left hand typically used by radiologists to assess skeletal maturity.

- In Chapter 2, **Improving Fracture Detection Performance with Deep-Learning**, we introduce the fracture detection problem, and the rationale behind its coveted automation. After a succinct literature review, we present the AI software of Gleamer, and delve into the protocol used to determine its clinical effectiveness. We briefly describe the inner workings of the algorithm, introduce the clinical validation dataset, and describe the setup of the study, and how medical experts were assisted by the AI in their diagnosis. Along the way, we provide a critical analysis of performance indicators typically used to measure the performance of readers on the fracture detection task, and introduce what we argue are better alternatives. Finally, we detail the results of the study, revealing that AI assistance has a significant positive impact on the performance of medical experts. The work within this chapter is linked to the following journal publication:
  - Loïc Duron, Alexis Ducarouge, André Gillibert, Julia Lainé, Christian Allouche, Nicolas Cherel, Zekun Zhang, Nicolas Nitche, Elise Lacave, Aloïs Pourchot, Adrien Felter, Louis Lassalle, Nor-Eddine Regnard, Antoine Feydy: Assessment of an AI aid in detection of adult appendicular skeletal fractures by emergency physicians and radiologists: a multicenter cross-sectional diagnostic study. In *Radiology* 2021.
- In Chapter 3, **Neural Architecture Search**, we present Neural Architecture Search (NAS), a recent learning paradigm which delegates the search of not only the features, but also the architecture used to

extract those features, to the AI algorithm. We briefly describe motivations for the application of NAS to our setting, and put forward key papers of the NAS literature. We present the optimization processes typically encountered in NAS, and close the chapter by describing the Weight-Sharing (WS) paradigm, an elegant solution to NAS, that aims at learning all the architectures of a search space of interest at once.

- In Chapter 4, **Weight-Sharing on the NAS-Bench-101 Dataset: A Practical Case Study**, we take advantage of NAS-Bench-101 (NB-101), a dataset of architecture evaluations, to challenge the efficiency of a uniform-sampling based WS variant on several representative search spaces. After reviewing previous studies on WS and highlighting several of their shortcomings, we introduce our own experimental setup, from which we extract several good practices that one should keep in mind when evaluating WS. With our experiments we first establish that, given the correct evaluation procedure, WS is able to produce accuracy scores decently correlated with standalone ones. We then provide evidence that on some search spaces, this WS variant is able to rapidly find better than random architectures, whilst it is equivalent or sometimes even worse than a baseline random search on others. We find that when given the same budget, the probability of superiority of an architecture found using WS over an architecture found through random search can vary between 7% and 78% depending on the search space. We present evidence that the search space itself has an intricate effect on the capabilities of WS and can bias weight-sharing towards certain architectural patterns with no clear accuracy advantage. We conclude that the impact of WS is heavily search-space dependent and difficult to anticipate for a given problem.

  – Aloïs Pourchot, Kévin Bailly, Alexis Ducarouge, Olivier Sigaud, An Extensive Appraisal of Weight-Sharing on the NAS-Bench-101 Benchmark. In *Neurocomputing*, 2022.

- In Chapter 5, **Neural Architecture Search for Fracture Classification**, we try to assess the efficacy of tailoring DCNN architectures to traumatic radiographs. As an alternative to the fracture detection problem, we start by introducing a fracture patch classification task on which NAS is tractable. Then, after reaffirming the importance of using transfer learning from natural images, and emphasizing the apparent incompatibility between NAS and pre-training, we introduce an efficient scheme based on weight sharing that makes it possible to exploit both conjointly. Using a plain genetic algorithm and a search space of efficient architectures, we show that it is possible to find architectures that are better suited to our problem than their counterparts selected on ImageNet, both in terms of diagnostic performance, and

inference-time computational efficacy. The work accomplished in this chapter has led to a publication which is under review at ICIP.

  - Aloïs Pourchot, Kévin Bailly, Alexis Ducarouge, Olivier Sigaud, Neural Architecture Search for Fracture Classification. Under review at *ICIP*, 2022.

- In Chapter 6, **Estimating Bone Age with Deep Learning**, we introduce the bone age assessment (BAA) problem and the different methods used by radiologists to perform its diagnosis. We then carry out a brief review of recent deep learning solutions to BAA, with a particular focus on the results of the RSNA pediatric bone age challenge. To analyze the generalization capabilities of a BAA model trained on the RSNA dataset, we introduce a BAA clinical dataset of our own. Starting from the solution of the winners of the challenge, we modernize the training setup to slightly alleviate the computational resources, and reveal that the resulting model is biased towards certain age and sex categories. We propose a simple weighting mechanism to counteract this bias and display its efficiency on our internal dataset. Making use of another public dataset for which the chronological age of the patients were available, we demonstrate that incorporating this information as an input to the model decreases the global average error, but further biases the model, deteriorating its performance in the detection of pathological patients. We argue that this effect has been ignored in the literature, and show that by correcting the two biases at the same time using the weighting procedure, we are able to significantly improve the performance of our baseline model, which performs substantially better than a reference radiologist tasked to perform BAA on our clinical dataset.

Chapter 2

Improving Fracture Detection Performance with Deep-Learning

Contents

In this chapter, we introduce the fracture detection problem, and the rationale behind its coveted automation. After a succinct literature review, we present the AI software of Gleamer and briefly describe its inner workings. We then delve into the protocol used to determine its clinical effectiveness: we introduce the clinical validation dataset, and describe the setup of the study, and how medical experts were assisted by the AI in their diagnosis. We take advantage of this study to provide a critical analysis of performance indicators typically used to measure the performance of readers on the fracture detection task, and introduce what we argue are better alternatives regarding clinical adequacy. Finally, we detail the results of the study, and reveal that AI assistance has a significant positive impact on the performance of medical experts. We conclude by showing that improvements of the AI could lead to fully automated fracture detection systems.

2.1 Introduction

Traumatic skeletal injuries are a leading source of emergency departments consultation, with an annual incidence of 1.3% in the United States of America (DiMaggio et al., 2017), and 0.32% in China (W. Chen et al., 2017). Radiography is the first-line imaging tool for the diagnosis of these lesions and the most used imaging modality worldwide (Arasu et al., 2015; Willett, 2019). However, the global increase in radiographic exams has resulted in a shortage of available radiologists (Rimmer, 2017). In turn, emergency physicians and radiology trainees, who often lack expertise in orthopaedic imaging, are regularly required to make patient management decisions prior to the availability of a senior radiologist report, with an increased risk of interpretation error (Scepi et al., 2005). Missed fractures, an important cause of morbidity, account for 80% of emergency department diagnostic errors (Guly, 2001), and are a frequent cause of patient claim.

Studies report that two of the main sources of misdiagnosis in radiology are the under-reading of exams, and satisfaction of search (Y. W. Kim et al., 2014). The former refers to practitioners overlooking a visible anomaly for various reasons, such as tiredness, distraction or lack of training. The latter is a well-known cognitive bias in radiology (Busby et al., 2018), and refers to a decrease in a person's alertness after a first finding was uncovered. In addition, there is an inherent difficulty to X-rays analysis due to its projective nature. Some abnormalities simply cannot be identified using a single X-ray, despite their presence being confirmed by additional X-rays with different projection angles or other imaging modalities such as Computed Tomography (CT) or Magnetic Resonance Imaging (MRI). In Figure 2.1, we provide an example of a radiograph where satisfaction of search might typically occur, and another example were an anomaly is present but impossible to describe from that image only.

Supplying physicians with real-time and reliable information about the presence and location of fractures could therefore facilitate radiograph analysis, and help decrease overall false negative rates. In fields such as the detection of breast cancer (Fenton et al., 2007) or the detection of lung nodules in CT-scans (Shaukat et al., 2019), Computer-Aided Detection (CAD) systems have been available for around 20 years. Similar technologies have been applied to fracture detection unsuccessfully, on account of the wider variety of morphological aspects and image patterns involved (Donnelley et al., 2008; Cao et al., 2015). Only the onset of deep learning allowed the emergence of systems capable of tackling this challenge.

In this chapter, we dive into the fracture detection system of Gleamer and its clinical validation as a fracture detection AI-assistant, which led to a publication in the Radiology journal (Duron et al., 2021). The critical focus

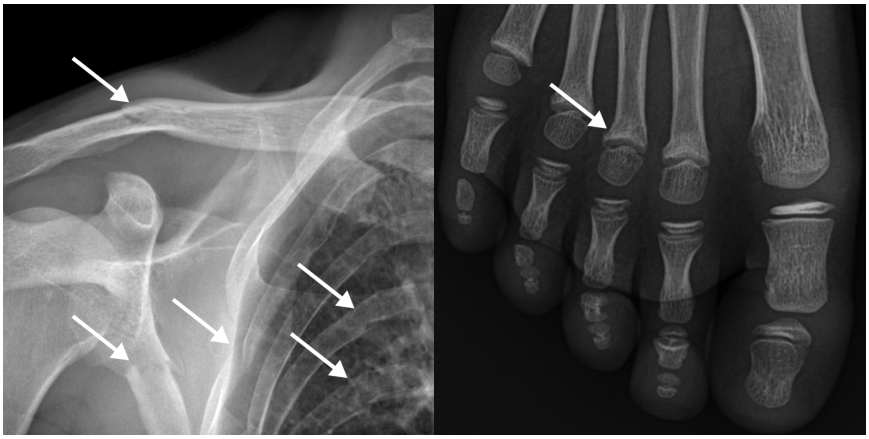


Figure 2.1: Examples of common radiographic diagnostic errors. Fractures are marked with white arrows. On the left image, the radiologist reported the fractures on the clavicle and the scapula, but failed to notice the three fractures on the rib cage. On the right, an example of a very subtle fracture of the third metatarsus, which was missed by the radiologist.

of this study was to demonstrate that Gleamer's AI tool could improve the fracture diagnosis performance of medical professionals in realistic settings. To perform this feat, 12 medical experts were enrolled to analyse 600 exams consecutively gathered from several French centers. Exams were comprised of radiographs of various anatomic body parts, often containing several fractures. Each reader examined a subset of the exams on their own, and the other with the predictions of the AI displayed in their workflow.

This study was a team effort which individual contributions are hard to disentangle. Part of this PhD was dedicated to the setup and design of the AI, preliminary statistical analyses, literature review and manuscript drafting. As such, the structure of this chapter is close to that of Duron et al., 2021, and some paragraphs directly transcribed from earlier drafts of the manuscript. Nonetheless, we try in this thesis to provide more insights into technical aspects of the study that were sidelined from the original publication due to editorial choices, the targeted public of the Radiology journal being radiologists rather than computer vision researchers.

We pursue this chapter with a short review of the literature of fracture detection with DL in Section 2.2. In Section 2.3, we then present the different aspects of our methods, including the AI model, the clinical dataset, and the reported performance metrics. Finally, in Section 2.4, we introduce the results of the study and compare the performance of the medical experts, with and without AI assistance, revealing that help from the system enhances their diagnostic performance and reduces their reading time.

2.2 Related Work

Prior to this work, no clinically relevant system existed to help physicians detect bone fractures on extremity radiographs. However, a few DL algorithms were developed with encouraging results.

Several proof of concept studies have considered the use of various DCNN architectures to predict whether an X-ray contains a fracture. Different bones or body parts were investigated, including the wrist, (D. Kim et al., 2018), the proximal humerus (S. W. Chung et al., 2018), the ankle (Kitamura et al., 2019), the femur, (Adams et al., 2019) or the hip (Cheng et al., 2019; Gale et al., 2017). However, their clinical relevance were hindered by several aspects. First, the different anatomical focuses critically narrowed their range of application. Secondly, because networks were trained in a classification paradigm, they were unable to readily explain their malignancy scores by providing localization information, which is arguably what matters the most to medical experts. Finally, clinical validations themselves were limited or inadequate, as most approaches evaluated AI algorithms on their own, without addressing the advantages that they could readily provide to medical experts.

Closest to a suitable and practical solution is the work of Lindsey et al., 2018, in which the authors develop a DCNN system capable of pinpointing fractures in radiographs. Based on the U-Net architecture (Ronneberger et al., 2015), their model learns to predict whether a fracture is present in the image, and if applicable, a heatmap of its spatial extent. They develop their algorithm using 135,409 radiographs, including 100,855 images of body parts other than the wrist which were used during pre-training, and 31,490 wrist X-rays, used to perform the final fine-tuning. The authors perform a clinical study revealing that AI-assistance reduces the propensity of several emergency doctors to misdiagnose images. The study is however limited to the analysis of wrist radiographs. Besides, no mention is made of the integration of the DL algorithm into radiologists workflows.

2.3 Methods

In this section, we briefly describe the AI framework on which our model is based on. We then detail the clinical dataset used to measure the performance of the readers, as well as the reading process itself. We finally dive into the metrics used to report the performance of the readers, and explain why they must be carefully selected.

2.3.1 AI System

The goal of object detection is to accurately localize instances of objects belonging to prespecified classes within images. Typically, the expected output of a detection model is a list of bounding box, each of them coming with an estimated class for the region it encloses. Unlike the classification and regression paradigms which are fairly straight-forward, DL object detection approaches are quite diverse due to the fundamental difficulty of predicting a varying number of localized outputs per image. For a taste of the variety of existing approaches, we refer the reader to the survey of Li Liu et al., 2020. Hereafter, we very briefly describe a family of object detection models brought about by the Region Based Convolutional Neural Network (R-CNN) approach of Girshick et al., 2014.

The idea behind R-CNN is to treat the detection problem as a patch extraction and classification problem. This implies the presence of two fundamental image processing modules. The first one automatically extracts a list of bounding boxes of interest. Note that bounding boxes obtained from the extraction module do not need to be quite precise, and only serve as a good prior for the position of objects. The goal of this extraction module is to cover all of the actual objects of the images, with the least amount of boxes possible. The second one takes as input the image patches of interest and assesses whether they are "background" patches, or if they contain instances of the sought after objects, and when they do, additionally predict offsets on the coordinates of the patches so that they better surround the object of interest. To perform the former, the authors utilize the Selective Search (Uijlings et al., 2013) algorithm, which outputs a list of boxes of interest based on clusters of pixels in the image. The latter is a plain DCNN trained to classify image patches and regress offsets on positive examples.

Unfortunately, performing a complete forward pass for each patch is very time-consuming and scales poorly with the number of Regions of Interest (ROI). In a subsequent research, Girshick, 2015 introduce the Fast R-CNN algorithm as a successful attempt to mitigate this issue. Instead of extracting patches from the image and performing a forward pass of the model for each patch, the authors propose to first perform a forward pass of the complete image, and then extract the features associated to the patches from the resulting feature maps using a ROI pooling layer. Patches classification and offsets regression are then performed using the extracted feature maps.

In another work (S. Ren et al., 2015), the same authors demonstrate the Selective Search is the new time-limiting factor of Fast R-CNN. They propose instead to automatically learn which regions to consider using another network called the Region Proposal Network (RPN). The designed approach, Faster R-CNN, can be dissected as such: (i) First, the input image is pro-

cessed by a DCNN to produce a global feature map; (ii) Each spatial unit of the resulting feature map is associated to a list of several ROI called anchor boxes, which vary in sizes and aspect ratios, such that they span the entirety of the image all together; (iii) Using additional convolutional layers on top of the extracted image features, the RPN learns to predict for each anchor box whether they contain an object. For anchor boxes that do contain an object, the RPN, also learns to predict an offset to the anchor as a first refinement for the object's location; (iv) Anchor boxes for which the "objectness" score gets above a fixed threshold are kept as ROI, and the features associated to the region are pooled from the set of features, following the Fast R-CNN scheme; (iv) Those features are fed to a second set of layers that learn to classify the object present in the ROI, and further improve the estimation of its boundaries. Due to the two successive classification and regression modules, this type of detection models are referred to as "two-stage" object detectors.

To tackle the fracture detection problem, we make use of the Mask R-CNN framework of He et al., 2017 which is a direct successor of Faster R-CNN. The main difference lies in the introduction of the "ROI Align" layer, a refinement of the ROI Pooling layer of Faster R-CNN. Additionally, instead of directly extracting features from the DCNN backbone, Mask R-CNN makes use of a Feature Pyramid Network (Lin et al., 2017) to extract features at different spatial resolutions. We trained the Mask R-CNN on a dataset of 60,170 trauma radiographs gathered from twenty-two French public hospitals and private radiology departments between January 2011 and May 2019. This development dataset was split into a 70% training set, a 10% validation set and a 20% test set. The training was performed using the Detectron2 (Y. Wu et al., 2019) code base. Standard data augmentation strategies were applied during training, including the random rotation, flipping, translating, cropping and resizing of input images. The model was trained for 270,000 iterations using SGD with a batch size of 4.

Depending on the classification score returned by the AI for each ROI, the whole pipeline is capable of running at different (sensitivity, specificity) operating points. For the reading task of the study, the AI system displayed two types of boxes: "DOUBT-FRACTURE" dashed boxes and "FRACTURE" plain boxes. These two categories respectively correspond to two different operating points of the algorithm: the first one with high sensitivity and the second one with high specificity. The threshold values from which the algorithm had to trace a bounding box during the reading phase of the study had been previously defined on the development test set (independent from the clinical test set) to correspond to a sensitivity of 96.2% for the "DOUBT-FRACTURE" boxes and a specificity of 99.2% for the "FRACTURE" boxes. Boxes were displayed with a confidence level expressed in percentage, as illustrated in Figure 2.2.

2.3.2 Dataset & Readings

To appraise the performances of the AI, a clinical testing set was retrospectively built from interpretations performed between January 2016 and December 2017 in 17 French medical centers. Exams of adult patients were included if they contained a radiography of the lower limbs, the upper limbs or the pelvis performed after a recent trauma. From the resulting list were discarded studies containing only obvious fractures (displaced and commuted fractures), containing an image featuring a body part which was not an extremity (e.g. spine, head and ribs) or containing images of poor quality precluding human interpretation. The inclusion procedure was stopped after 50 exams without any fracture and 50 exams with at least one fracture were selected for each of the six considered body parts: hand, arm, shoulder, pelvis, leg and foot, resulting in a dataset of 600 exams, totalling 2,441 images for an average of 4 radiograph per exam. None of the exams had previously been considered to design the AI. This exam dataset was then split into two 300-patient subsets, with stratified randomization so that the resulting subsets were similar in terms of median age, female-to-male ratio, and body part and fracture prevalence.

We assessed the diagnostic performance and the radiograph reading time of 12 medical experts with and without AI assistance. Six of the participants were radiologists, and the other six were emergency doctors (EDs). Readers were asked to detect and report the localisation of fractures on lossless radiographs by pinpointing them with an annotation tool designed for the study and running on a dedicated workstation. When assisted by the system, readers were first presented with untouched radiographs and were then required to manually activate the AI, which revealed boxes around suggestions of fracture location. They were free to change their diagnosis or not based on the suggestions. Each medical expert analysed one of the two aforementioned subset with the help of the AI, and the other one without. Half of the readers were helped by the AI on the first subset, and the other half on the second subset, and vice-versa. The splitting of the readers across the different groups ensured an even distribution of speciality and experience.

2.3.3 Metrics

The reported performance metrics of the studies introduced in Section 2.2 focus on a classification paradigm in which only the binary output "fracture"/"no fracture" is of interest. All in all, the result of the image-wise (iw) examination of a reader can either by a true positive (TP), true negative (TN), false positive (FP) or a false negative (FN) based on the following definitions:

- $\mathbf{TP}_{iw}$: There is at least one fracture in the image, and the reader reports that a fracture is present.
- $\mathbf{TN}_{iw}$: There is no fracture in the image, and the reader reports that no fracture is present.
- $\mathbf{FP}_{iw}$: There is no fracture in the image, but the reader reports that at least one fracture is present.
- $\mathbf{FN}_{iw}$: There is at least one fracture in the image, and the reader reports that no fracture is present.

Authors then commonly report the image-wise sensitivity and specificity of readers as:

$$\mathrm{SE}_{iw} = \frac{\sum_i \mathrm{TP}_{iw}}{\sum_i \mathrm{TP}_{iw} + \mathrm{FN}_{iw}}, \quad \mathrm{SPE}_{iw} = \frac{\sum_i \mathrm{TN}_{iw}}{\sum_i \mathrm{FP}_{iw} + \mathrm{TN}_{iw}}, \tag{2.1}$$

where summations are performed over the images of interest, and which respectively measure the proportion of patients for whom the reader accurately assessed on the presence of at least one fracture among patients having at least one fracture, and the proportion of patients for whom no fracture was reported by the reader among patients having no fracture.

Usual classification metrics facilitate results understanding by researchers of the medical field, who typically communicate their findings in terms of positive and negative predictive value. Nonetheless, they are fundamentally lacking, as they fail to measure exactness in fracture localization, a key component of patient outcome. Moreover, in its usual description, the classification paradigm introduces no difference between cases with exactly one and cases with strictly more than one fracture, and thus cannot measure the effects of biases such as satisfaction of search. Finally, clinical evaluations of the aforementioned studies are performed image-wise, a patient being associated to a single radiograph, in contrast with clinical settings where radiologists are rarely provided with a unique X-ray to diagnose a patient. Assessing the performance of medical experts in an image-wise setting creates an under-evaluation bias, as some missed fractures could have been easily spotted with the coverage provided by a proper amount of images. To circumvent those limits, we have spent great effort in coming up with appropriate metrics definitions.

Keeping in mind the ease of understanding of classification metrics, we propose to transcribe the fracture detection paradigm into a patient-wise (pw) classification problem, using stricter definitions of patient-wise results. For each patient the result of a reader, can either be a TP, TN, FP, or FN, as defined below:

- $\mathbf{TP}_{pw}$: There is one or more fracture in the exam, and the reader

correctly pinpoints all of them by putting a marker in at least one of their associated ground truth regions.

- $\mathbf{TN}_{pw}$: There is no fracture in the exam, and no marker is created by the reader.
- $\mathbf{FP}_{pw}$: There is no fracture in the exam, but the reader reports the presence of a fracture by creating one or more marker.
- $\mathbf{FN}_{pw}$: There is one or more fracture in the exam, but the reader fails to report at least one of them by not creating a marker inside one of the corresponding ground truth regions.

From there, we introduced the patient-wise sensitivity SE_{pw} and the patient-wise specificity SPE_{pw} as:

$$\mathrm{SE}_{pw} = \frac{\sum_p \mathrm{TP}_{pw}}{\sum_p \mathrm{TP}_{pw} + \mathrm{FN}_{pw}}, \quad \mathrm{SPE}_{pw} = \frac{\sum_p \mathrm{TN}_{pw}}{\sum_p \mathrm{FP}_{pw} + \mathrm{TN}_{pw}}, \tag{2.2}$$

where summations are performed over the patients of interest. SE_{pw} measures the proportion of patients for whom the reader accurately pinpointed all fractures among patients having at least one fracture, whilst SPE_{pw} measures the proportion of patients for whom no fracture was reported by the reader among patients having no fracture. We argue that those definitions are more sensible for three reasons: (i) To be correct, readers must not only assess on the presence of fractures, but also correctly pinpoint them. This is more appropriate, because knowing their localization is required to begin patient treatment. (ii) To be correct, readers must pinpoint **all** the fractures present in an exam. This is rational, as any fracture missed might lead to further morbidity, regardless of how many were already found; (iii) By construction, these metrics account for the exam-structure connecting the different images. This allows the evaluations to be realistic, and diagnosis to be performed with a more adequate amount of information.

Still, this approach suffers some practical defects. First, when every fracture of an exam is reported, any additional marker created is ignored, even if it was created through the incorrect belief that an additional fracture might exist. This scheme is thus intrinsically lax towards false positives. Secondly, because of its strictness towards true positives, this scheme fails to acknowledge the propensity of readers to detect at least some of the fractures, rather than none at all. A missed fracture is treated as if all fractures were missed, even if the reader had accurately spotted several of them.

Those discrepancies are inherent to the transcription from object detection to classification. However, it is possible to correct them by examining object-detection metrics rather than classification metrics. To do this, one

must consider object-wise, i.e. fracture-wise (fw) definitions of TP, FN, and a marker-wise (mw) definition of FP:

- $\mathbf{TP}_{fw}$: A marker was created by the reader inside at least one of the regions associated to the fracture.
- $\mathbf{FN}_{fw}$: No marker was created by the reader inside any of the regions associated to the fracture.
- $\mathbf{FP}_{mw}$: The marker created by the reader did not fall into any of the regions belonging to any fracture described in the exam.

TPs and FNs are then aggregated over all fractures and averaged over patients to compute the fracture-wise sensitivity SE_{fw} of the readers, whilst the marker-wise FPs are leveraged to compute the average number of false positive per patient, FPP_{pw}:

$$\mathrm{SE}_{fw} = \frac{1}{N_p} \sum_p \frac{\sum_f \mathrm{TP}_{fw}}{\sum_f \mathrm{TP}_{fw} + \mathrm{FN}_{fw}}, \quad \mathrm{FPP}_{pw} = \frac{\sum_p \sum_m \mathrm{FP}_{mw}}{N_p}, \tag{2.3}$$

where summations are performed over ground-truth fractures when indexed by f, patients when indexed by p, and markers when indexed by m. N_p is the number of patients considered. The reader will have noticed that it is impossible to define TN in this context, which limits the range of classical metrics that can be considered. SE_{fw} measures the average over patients with fractures, of the proportion of fractures which were accurately pinpointed by the reader, whilst FPP_{pw} measures the average over patients of the number of incorrect markers created by the reader. This approach has the advantages of the previous one, as missed fractures are still penalized, but is more fair as the costs induced are proportional to the average number of fractures actually missed. FPP_{pw} is an informative additional measure of the performance of readers that reflects their tendency to report fractures wrongly.

As both sets of metrics are relevant, we report one and the other in the results. However, we focus on the patient-wise metrics SE_{pw} and SPE_{pw} for their practicality. When evaluating the model on its own, the previous definitions still apply, with two differences: (i) Instead of markers, the model outputs bounding boxes. A box and a ground-truth region are considered to match if their Intersection Over Union (IOU) is above 0,5; (ii) Unlike readers, the AI can function with different levels of sensitivity and specificity. Indeed, bounding boxes are associated with scores. Filtering boxes with small confidences commonly increases the model specificity whilst reducing its sensitivity, and vice-versa. Instead of considering a single functioning point, the whole spectrum is explored by computing the different metrics associated with different threshold values. When in the (SE_{pw}, 1 - SPE_{pw})

Metric	Unaided	Aided	Absolute Difference	
SE_{pw} (%)	70.8 ± 12.5	79.4 ± 7.4	+8.7[3.1,14.2]	$p = .003$
SE_{fw} (%)	73.7 ± 11.1	81.2 ± 6.5	+7.5[2.8,12.2]	$p = .005$
SPE_{pw} (%)	89.5 ± 6.5	93.6 ± 4.6	+4.1[0.5,7.7]	$p = .0006$
FPP_{pw} ($\varnothing$ fract)	0.113 ± 0.069	0.066 ± 0.048	-0.047[-0.086,-0.009]	$p = .02$
FPP_{pw} (fract)	0.082 ± 0.055	0.045 ± 0.028	-0.037[-0.073,0.000]	$p = .05$
Reading Time	67.0 ± 26.2	57.0 ± 24.8	-10.0[-23.1,3.0]	$p = .12$

Table 2.1: Average performance of the readers according to different metrics, with, and without the help of the AI software. Mean values are reported with standard deviation ($\pm$). Absolute differences are reported with their 95% confidence intervals, and reported p-values are for one-sided t-tests. (fract) and ($\varnothing$fract) respectively refer to exams with and without fractures.

space, the resulting curve is referred to as a Receiver Operating Characteristic (ROC) curve, whilst the (SE_{fw}, FPP_{pw}) space creates a Free-response Receiver Operating Characteristic (FROC) curve (Bandos et al., 2009). The former is bounded in the $[0, 1]^2$ square, and the corresponding Area Under the Curve (AUC) is often reported as an overall measure of performance. The AUFROC equivalent for the FROC curve is not as straight-forward to derive, given that the FPP_{pw} coordinate is not bounded (Bandos et al., 2009). A simple alternative is to integrate over a predefined range of FPP_{pw} values that encompasses the typical range of radiologists performance.

2.4 Results

We report the evolution of the average metrics with and without AI-assistance in Table 2.1. The help of the AI had a global positive effect on all the performance metrics of the readers. AI-assistance improved the average patient-wise sensitivity SE_{pw} from 70.8% to 79.4% (+8.7%, $p = .003$), and the average patient-wise specificity SPE_{pw} from 89.5% to 93.6% (+4.1%, $p = .0006$). The mean number of false positive per patient FPP_{pw} in patients with no fractures was reduced from 0.113 to 0.066 (-41.9%, $p = .02$). The fracture-wise sensitivity SE_{fw} improved by 7.5% ($p = .005$). Finally, the mean reading time was reduced from 67.0 to 57.0 seconds (-10.0, $p = .12$). We provide examples of AI results in Figure 2.2

Interestingly, The help of the AI was more valuable on some body parts than others, as the increase in SE_{pw} was the largest for the hand (+20.5%, $p = .009$) and foot (+15.1%; $p = .001$) body parts. Other body parts observed a non-significant increase in SE_{pw}. The SPE_{pw} was significantly non-inferior with the help of the AI for all body parts except pelvis and shoulder.

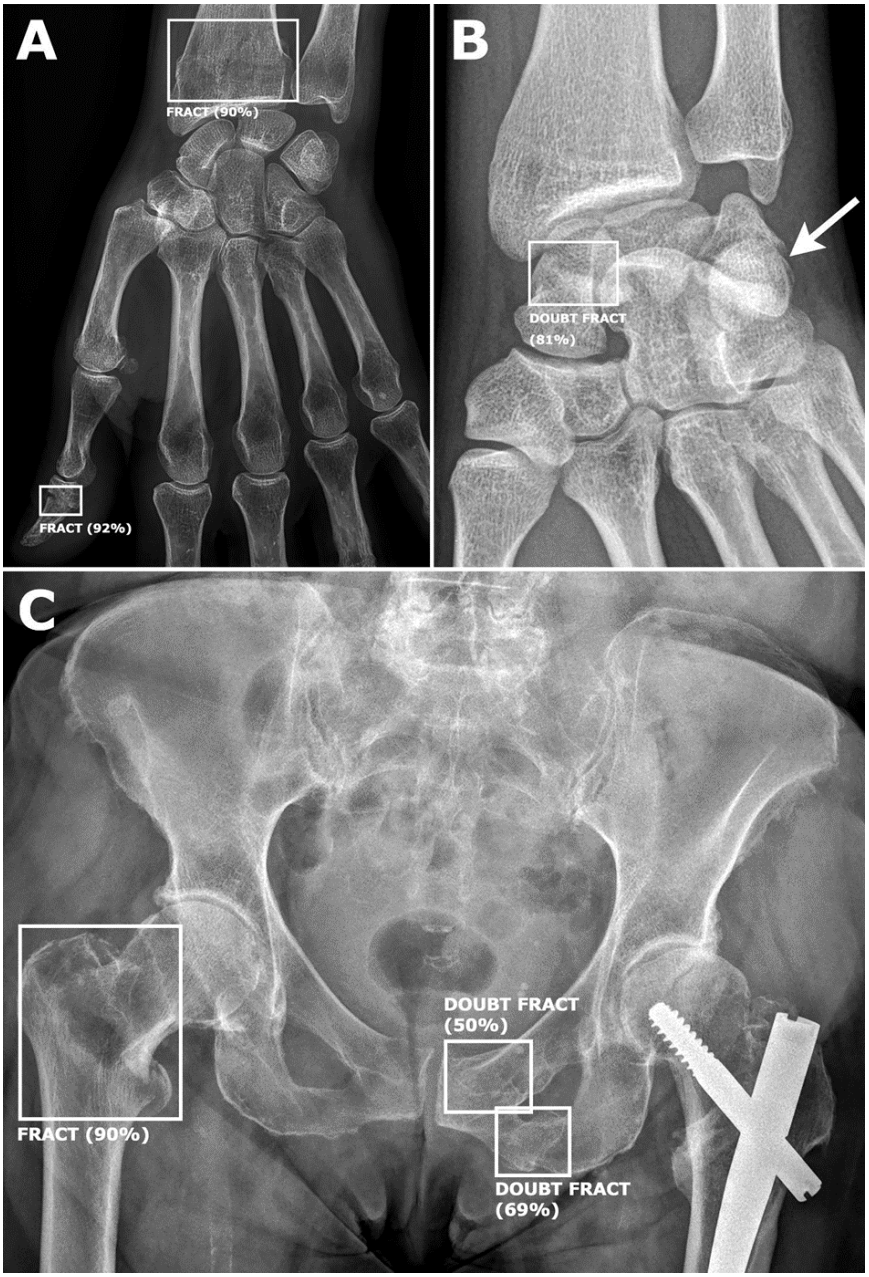


Figure 2.2: Examples of multiple and severe fractures on the clinical dataset. The result of the AI is displayed in boxes with labels "FRACT" and "DOUBT FRACT" with the corresponding confidence level. In image A, a fracture of the right distal radius, associated with a fracture of the distal phalange of the thumb. In image B, a fracture of the scaphoid, and of the triquetrum. The triquetrum fracture was missed by the AI. In image C, a fracture of the right femoral head, along two fractures of the pubis.

The average unaided SE_{fw} was 73.9% for patients with single fractures and 73.4% for patients with multiple fractures with a non-significant difference of (-0.4%, $p = .85$), showing no evidence of satisfaction of search errors. The unaided average number of false positive fractures per patient FPP_{pw} was 38.1% higher ($p = .006$) for patients with no fracture than for patients with one or more fractures suggesting the paradoxical existence of "dissatisfaction of search" errors where readers tried to find fractures when none existed. This suggests that the setting of the study was not appropriate to measure effects on the satisfaction of search bias. We hypothesize that a retrospective study would be more pertinent for this matter.

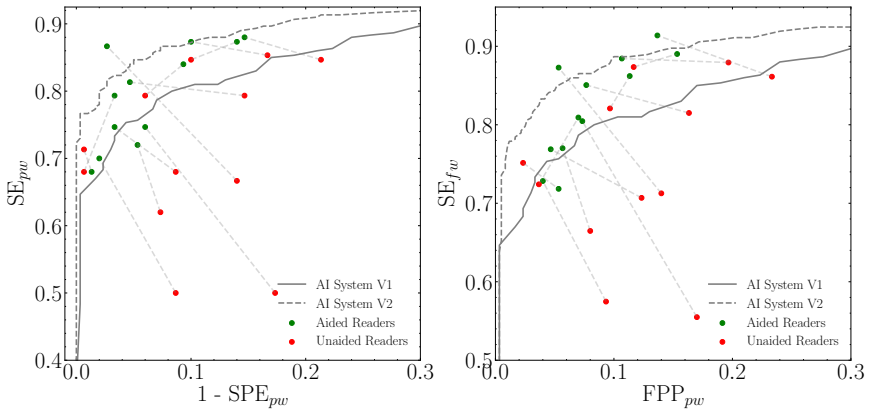
The sensitivity of emergency doctors (EDs) improved from 61.3% to 74.3% (+13%, $p = .03$) with AI-help while the sensitivity of radiologists improved from 80.2% to 84.6% (+4.3%, $p = .03$). The difference in sensitivity gain between EDs and radiologists was of +8.7% ($p = .08$). After adjustment on the unaided sensitivity, this difference in sensitivity gain was reduced to -5.6% ($p = .28$) for EDs vs radiologists. The SE_{pw} of aided EDs (74.3%) and unaided radiologists (80.2%) were not significantly different (-5.9%, $p = .15$) while the SPE_{pw} of aided EDs (96.6%) was higher (+8.1%, $p = .03$) than that of unaided radiologists (88.4%). Those results suggest that AI-help has the potential to reduce misdiagnosis in scenarios where the former has to take premature decisions regarding patient-care in the absence of the latter.

ROC and FROC curves were derived using standalone AI performance. The overall ROCAUC was 0.908. As improvement updates of the AI algorithm are regularly released, a post hoc analysis was also performed to assess the latest release on the same external dataset. This new version of the model was able to outperform every unaided reader and showed a ROCAUC of 0.935. ROC curves, FROC curves and readers performance are illustrated in Figure 2.3.

2.5 Discussion

With this work, we have made clear that the assistance of Gleamer's AI system significantly improves the radiographic diagnostic performance of medical experts in a realistic setting. Remarkably, we revealed that further iterations of the algorithm were able to outperform almost all human readers, hinting at the possibility of letting AI handle the fracture detection task partially on its own.

Previous deep learning approaches to bone fracture detection have only been assessed in restrained conditions. Most studies focused on a single body part, with clinical evaluations solely considering standalone performance of the algorithms, using an inadequate image-wise classification paradigm. In



(a) ROC (b) FROC

Figure 2.3: ROC and FROC curves of the AI system used during the study (AI System V1) and a subsequent version of the AI (AI System V2). In each figure, we report the corresponding aided and non-aided performance of the readers. A thin line between two points indicates that they correspond to the same reader. The new version of the outperforms all unaided readers, and almost all aided readers.

contrast, we: (i) Gathered a clinical dataset containing radiographs of six different anatomical regions, from a wide variety of clinical sources; (ii) introduced several metrics to reliably describe reader performance, accounting for the importance of localization and not classification, accounting for the presence of several fractures in exam, and considering the exam nature of the radiographs; (iii) introduced detection metrics to further inquiry the average proportion of fractures reported by readers, and their tendency to wrongly describe the presence of fractures.

There are however several limitations to our clinical validation. First, readers and AI were evaluated on their ability to make decisions based on images alone, without knowledge about the patients' physical examination or medical history, creating a context bias. Clinical data can be crucial to make decisions (Loy et al., 2004). However, in practice, radiologists often lack relevant clinical data which partially justifies the setup of the study. Second, a Hawthorne effect may have affected the performances of readers and, prevented us from observing cognitive biases such as satisfaction of search. Third, as exams containing only obvious fractures were excluded, the sensitivity of unaided readers was probably underestimated. Finally, the stratification of fractures, leading to an artificial 50% prevalence, made it impossible to calculate negative and positive predictive values, and amplified the context bias.

Chapter 3

Neural Architecture Search

Contents

In this chapter, we present Neural Architecture Search (NAS), a recent learning paradigm which delegates the search of not only the features, but also the architecture used to extract those features, to the AI algorithm. We briefly describe motivations for the application of NAS to our setting, and put forward key papers of the NAS literature. We present the optimization processes typically encountered in NAS, and close the chapter by describing the Weight-Sharing (WS) paradigm, an elegant solution to NAS, that aims at learning all the architectures of a search space of interest at once.

3.1 Deep Convolutional Neural Networks

Convolutional Neural Networks (CNNs) have become the *de facto* way to solve computer vision problems. As is custom in the neural network literature, their design birthed from analogies with the mammalian brain. Their primordial structure was loosely inspired by the hierarchical layout of simple and complex cells found in visual cortices (Hubel et al., 1968). While both of those neurons respond mostly to the presence of oriented edges, the receptive field of simple cells are relatively localized, whilst the response of complex cells are to some extent spatially invariant.

In an attempt to describe this pattern with the semantics of neural networks, Fukushima et al., 1982 introduced the Neocognitron, a hierachical compound of two types of layers: the convolutional layers, and the down-sampling layers. Convolutional layers convolve an input image with a set of learned kernels. Because of the limited receptive fields of the convolution kernels, convolutional layers can be seen as the neural network equivalent to simple cells. Down-sampling layers on the other hand reduce the spatial resolution of an input image by applying a permutation invariant operator (e.g. a maximum or an average) to sets of neighbouring pixels and are thus more akin to complex cells. Interestingly, this convolutional/down-sampling dichotomy is still omnipresent in the description of contemporary CNNs.

Whats is perhaps the first modern usage of CNNs is presented in the work of LeCun et al., 1989, where convolutional layers are trained to recognize handwritten zipcodes using the back-propagation algorithm. Their results suggest a clear superiority of CNNs over Fully-Connected Neural Networks (FCNNs) for image analysis. The authors explain the performance gap by three core aspects of the CNN design:

1. The number of weights of a typical fully-connected layer scales linearly with the number of input variables, where it is constant for convolutional layers. Images, which commonly include tens of thousands of pixels, burden FCNNs with a sizeable amount of parameters, making them prone to overfitting.

2. The properties of the convolution operator inherently grants CNNs with equivariance to translations, and robustness to local alterations. FCNNs could theoretically learn such behaviors, but introducing them as prior knowledge within CNN is far more efficient.

3. FCNNs are oblivious to image topology as their inner computations are permutation-invariant. Images are processed as though they were flat, with no preferential reading order. CNNs on the other hand, are able to exploit the widespread local-correlations found in images, by

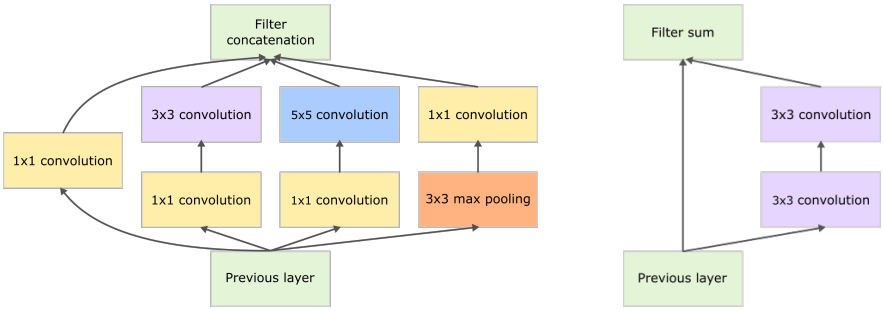
restricting the receptive field of their layers to local neighbourhoods.

Although the networks derived by LeCun et al., 1989 were much more efficient than their FCNNs counterparts, their training on higher resolution images was still excedingly expensive. In 2012 Krizhevsky et al., 2012 proposed to solve this problem by training CNNs using Graphics Processing Units (GPUs). Owing to their efficient convolution implementation, the authors were able to scale their network up to 60 million parameters. With their newly designed AlexNet architecture, they were able to reach a top-5 test error of 15.3% on the ImageNet dataset, effectively winning the ILSVRC-2012 competition, beating the second best entry by more than 10%. Although AlexNet was not the first instance of such large CNNs (Ciresan et al., 2011; Chellapilla et al., 2006), the design of Krizhevsky et al., 2012 effectively marked the beginning of the Deep Convolutional Neural Networks (DCNNs) era.

From there, several iterations were made to improve the AlexNet design. In 2014, (Simonyan et al., 2014) introduced the VGGNet architecture, which improves over the AlexNet design by reducing the receptive fields of the convolution filters to 3×3 and greatly increasing the depth of the network to 19 layers. Szegedy et al., 2015 pushed the depth of their GoogLeNet architecture by 3 additional layers, and replaced the plain linear architecture of VGGNet and with their Inception module, a design pattern that introduces a notion of width by merging the output of convolutional layers with different kernel sizes, 1×1, 3×3 and 5×5. A diagram of the Inception module can be observed in the left portion of Figure 3.1. Attempts at further increasing the depth of DCNNs were met with a degradation in performances. He et al., 2016 identified that this drop was due to the optimization process getting more complex as the network depth increased. To circumvent the complication, the ResNet architecture and its deep residual learning paradigm were introduced. Given a layer meant to learn to map an input x to $\mathcal{H}(x)$, residual learning suggests instead to learn the mapping $\mathcal{F}(x) = \mathcal{H}(x) - x$, such that $\mathcal{H}(x) = \mathcal{F}(x) + x$. The added identity function to the mapping is often called a residual or skip connection. This adjustment to the optimization allowed the authors to train architectures up to 151 layers deep without blows and whistles. A diagram of a residual layer can be observed in the right portion of Figure 3.1.

3.2 Searching for Architectures

As the design complexity of DCNNs architectures further increased (Ioffe et al., 2015; Szegedy et al., 2016; Saining Xie et al., 2017), so did the level of expertise required to select and apply them to new tasks. Although architecture selection is arguably less burdensome than feature description,



(a) GoogLeNet (b) ResNet

Figure 3.1: On the left, the inception module, fundamental block of the GoogLeNet architecture (Szegedy et al., 2015). On the right, a residual module, fundamental building block of the ResNet architecture (He et al., 2016). The left figure is a reproduction from Szegedy et al., 2015.

its reliance on advanced DL knowledge hinders the utilization of DCNNs technologies by non-experts. The less architectural adjustments are required for a model to perform optimally, the more outsider-friendly, and widespread AI approaches will be. A promising direction for the DL framework would be to automatically tune the design of DCNNs architectures, a process coined Neural Architecture Search (NAS) by Zoph et al., 2017a.

Optimizing neural networks architectures is not a recent idea, and several approaches have attempted to learn both the weights and the topology of neural networks. In particular, evolutionary and genetic algorithms researchers have displayed a keen interest in this paradigm, and several approaches related to this literature have been introduced (Angeline et al., 1994; Gruau, 1994; Yao, 1999). An honorable mention is the NeuroEvolution of Augmenting Topologies (NEAT) algorithm introduced by Stanley et al., 2002 that was shown to be able to both optimize and complexify neural networks during learning through evolution, resulting in overall faster training and better final performance on multiple reinforcement learning tasks.

As a research field, NAS is closely related to Hyperparameter Optimization (HPO). Unlike classical parameters, hyperparameters are parameters that describe the learning process, and are usually fixed during training. They have a critical impact on the performances of ML approaches, and tuning them is essential to achieve great performance. Examples of common hyperparameters are the learning rate or the batch size used to optimize a neural network with a gradient descent algorithm. A plethora of methods have been introduced to solve the HPO problem (Feurer et al., 2019), from model-free black box optimization methods like random search, evolutionary

algorithms (Simon, 2013) or particule swarm optimization (Eberhart et al., 1998), to model-based Bayesian optimization (Jones et al., 1998).

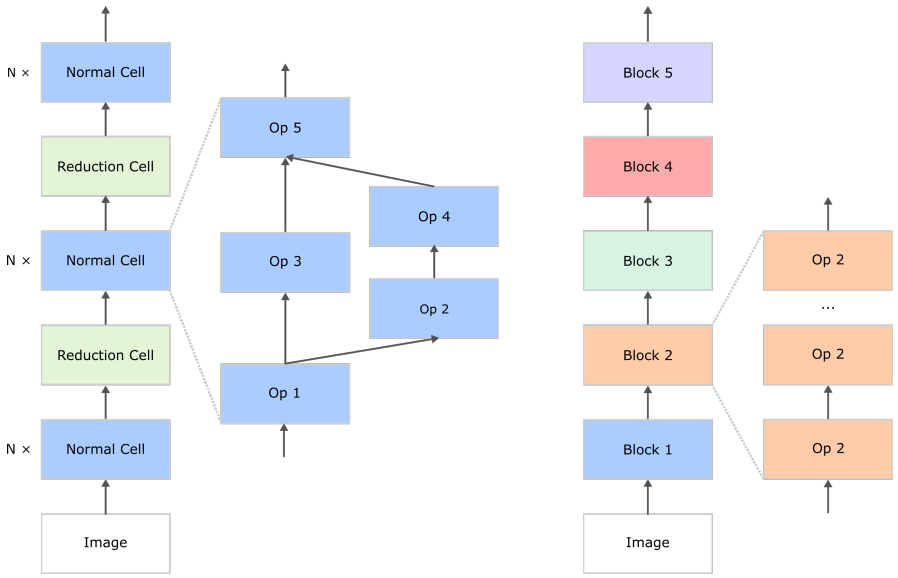
At a high level, the difference between HPO and NAS is thin. Since network architecture on its own is a rightful hyperparameter NAS algorithms belong to the family of HPO algorithms. Nonetheless, whilst most HPO algorithms are agnostic to the type of hyperparameter explored, NAS approaches tend to incorporate the graph topology of architectures as a prior in their inner workings. Conventional HPO approaches work best with continuous hyperparameters and can hardly handle the complex and discrete structure explored by architecture search spaces.

3.3 Search Spaces

A search space is a set of architectures that can be considered by a NAS algorithm. It constitutes one of the three fundamental aspects of a NAS approach (Elsken et al., 2019b). There are thus roughly as many search spaces as there are NAS algorithms. Still, some key design patterns can be identified. We describe below a few approaches which we consider valuable to the literature for their introduction of such design patterns. This list is by no mean exhaustive, and we refer the reader to surveys such as Elsken et al., 2019b or P. Ren et al., 2020 for a comprehensive list.

In their pioneer work, Zoph et al., 2017a considered the optimization of both CNNs and Recurrent Neural Networks (RNNs). We confine our summary to the former. The search space introduced by the authors is quite simple. They propose to represent an architecture by a succession of convolutional blocks, with a fixed maximum total length. Each block is composed of a ReLU non-linearity, a convolutional layer parameterized by its filters, number, sizes and strides, and a batch-norm layer (Ioffe et al., 2015). Inspired by the skip connections of the ResNet architectures (He et al., 2016), and the different pathways of the GoogLeNet (Szegedy et al., 2015), the authors introduced shortcuts connections between the different blocks, and each block can have a potential residual connection with all of its preceding blocks.

Noticing a growing trend in DCNNs architectures to be hierarchically constructed, such as with Inception module (Szegedy et al., 2015) or the residual block (He et al., 2016) represented in Figure 3.1, Zoph et al., 2018a propose to search only for sub-structures of interest, rather than optimizing the complete architectures at once. The authors introduced two types of patters. They dubbed those pattern cells, and searched for two of them: “normal” cells, and “reduction” cells. The latter differs from the former in that it must reduce the spatial dimension of its incoming inputs. The complete architecture is then obtained by stacking those cells in a prede-



(a) Cell-based (b) Block-based

Figure 3.2: (a) Example of an architecture obtained from the cell-based search space of Zoph et al., 2018a; (b) Example of an architecture obtained from the block-based search space of Tan et al., 2019a.

fined, fixed way. Each cell is represented by a directed acyclic graph which transcribes the different operations meant to be performed. We provide a graphical illustration of an architecture following this pattern in Figure 3.2a. The main advantages of this cell-based search space is that by searching for smaller sub-structures the optimization is made easier both because the cardinality of the search space is drastically reduced, and because its design enforces some structural priors which maximizes the likelihood of finding good architectures within their search space. The cell-based search space was particularly fruitful and was reused or adapted in several works (Real et al., 2019a; C. Liu et al., 2018; Pham et al., 2018; H. Liu et al., 2019a).

In Tan et al., 2019a, the authors propose to search for architectures with optimal trade-offs between performance and inference latency. They assert that cell-based approaches are too restrictive because they require the same layers to be replicated at all depth and argue that architectures designed for optimal inference latency should be able to perform different operation at different depth. For instance, they explain that earlier layers process much more data due to the relatively high spatial dimension, and have an overall higher impact on latency than later layers. Instead of the cell-based search space, Tan et al., 2019a introduce a factorized hierarchical search space which divides CNNs into several blocks, with the end of each block

reducing the spatial resolution of the feature map, whilst increasing the channel dimension. Each block is composed of several instances of the same layer, which parameters and structure are searched for. Available choices for the design of a block are the type of the convolutional layer, its kernel size, its Squeeze-and-Excitation ratio (Iandola et al., 2016), the size of the outputs filters and the number of layers in the block. Blocks are then stacked one atop the other to complete the architecture, as visually transcribed in Figure 3.2b. Because of its simplicity, this search space and some of its variants have gained popularity in the NAS literature (Cai et al., 2018; Stamoulis et al., 2019; Howard et al., 2019). A notable example is the family of EfficientNet models (Tan et al., 2019b), which smaller variant was found using NAS on the aforementioned search space.

3.4 Architecture Optimization

Given a search space $\mathbb{A}$, with each of its architecture A coming with a set of possible weights $\mathcal{W}(A)$, NAS can be formulated as a bi-level optimization problem (Colson et al., 2007), which takes the following generic form:

$$\begin{aligned} &\text{Find } A \in \arg\min_{A \in \mathbb{A}} \Gamma(A), \text{ where} \\ &\Gamma(A) = \inf \left\{ F(A, w^\star), w^\star \in \arg\min_{w \in \mathcal{W}} f(A, w) \right\}. \end{aligned} \tag{3.1}$$

The evaluation of the variable of interest depends on a second variable (here A and w respectively) that is constrained to be optimal with respect to another optimization problem. F and A (resp. f and w) are called the outer (resp. inner) objective and variable. In the typical ML setting, f would be the loss of a model on a training dataset, whereas F would be the loss of the same model on a held-out validation dataset. During training, the model has only access to the training dataset to update its parameters in order to minimize f. However, we would rather like the architecture to be selected so as to improve the model generalization, that is to decrease the loss of the model on unseen data, represented by the validation dataset.

3.4.1 Dynamic Formulation of the Inner Objective

Under its general form, the problem presented in Equation (3.1) creates constraints on the inner variable w that cannot be explicitly formulated in terms of A. Additionally, there is no guarantee for an optimal w to be unique. As a result, $w^\star \in \arg\min_{w \in \mathcal{W}} f(A, w)$ cannot be expressed as a function of A, which is cumbersome for practical resolution algorithms. To circumvent this difficulty, it is often implicitly assumed that each possible value of $w^\star$ corresponds to the same value of the outer objective F, and that at least one set of optimal weights can be reached through an iterative

process, which dynamics can be described analytically in terms of A. In practice, Equation (3.1) is thus essentially replaced by:

$$\text{Find } A \in \arg\min_{A \in \mathbb{A}} F(A, w_T(A)), \tag{3.2}$$
$$\text{u.c.} \left\{ \begin{array}{lcll} w_0 & = & \Phi_{\text{init}}(A, \epsilon) & \\ w_t & = & \Phi(w_{t-1}, A, f), & t = 1, \ldots, T \end{array} \right\},$$

where the initial weights w_0 are initialized by a deterministic or random process Φ_{init}. Each w_t is obtained by applying the operator Φ to w_{t-1} and the current architecture A. Typically, Φ takes the form of gradient descent on the inner objective f:

$$\Phi(w, A, f) = w - \alpha \nabla_w f(A, w), \tag{3.3}$$

where α is the learning rate. If the inner optimization behaves well, then as $T \to \infty$ the solutions of Equation (3.2) should get closer to the solutions of Equation (3.1). Franceschi et al., 2018 show that this is true under reasonable assumptions.

3.4.2 Discrete Optimization

In NAS problems, the search space $\mathbb{A}$ is usually discrete. Some approaches directly solve the outer optimization problem in Equation (3.2) using this discrete representation. When an architecture A is to be evaluated, the corresponding weights $w_T(A)$ are computed, and the resulting fitness of the individual $F(A, w_T(A))$ is used directly. Because those approaches do not consider interactions between the optimization of the architecture and the optimization of its inner weights, it is common to think of them as black-box optimization solutions. In Zoph et al., 2017b; Zoph et al., 2018a, the authors describe the process of building an architecture as a Markov Decision Process (MDP), and translate the problem of searching for an optimal architecture into the problem of searching for of an optimal policy to build architecture. They propose to solve their introduced reinforcement learning (RL) problem using Policy Search. Likewise, other approaches frame NAS as a RL problem, which they solve using Q-Learning (Zhao Zhong et al., 2020), or Monte Carlo Tree Search (Linnan Wang et al., 2018; Linnan Wang et al., 2021; Wistuba, 2017; Negrinho et al., 2017). Another successful line of approaches has been with the use of genetic algorithms. In their work, Real et al., 2017 introduce architectural mutations taking the form of operations on computational graphs, and optimize the performance of a population of architectures using a regularized form of the tournament selection algorithm (Goldberg et al., 1991), and so do Real et al., 2019a and H. Liu et al., 2017. Other approaches use genetic programming (Suganuma et al., 2017), Lamarckian evolution (Elsken et al., 2019a), or even a variant of the NEAT algorithm (Miikkulainen et al., 2019).

3.4.3 Continuous Relaxation

To avoid discrete variables, some works rely on a form of continuous relaxation of the search space. In Shin et al., 2018, the authors propose to search for convolutional layers hyperparameters by replacing plain convolution kernels $W_{i,j}$ with weighted averages over all possible resulting kernels $\sum_{i,j} \alpha_i \beta_j W_{i,j}$, with $\lambda = (\alpha_i, \beta_j)_{i,j}$ as learnable parameters. In Ahmed et al., 2018, the i-th convolutional layer learns which of the previous outputs (y_j) to combine as input feature x_i by optimizing the scales $\lambda = (m_{i,j})$ of a weighted average of all previous outputs $x_i = \sum_{j=1}^{i-1} m_{i,j} y_j$. In H. Liu et al., 2019b, elementary operations $(o_i)_i$, such as convolutions and pooling layers with different kernel sizes or skip connections are replaced by a weighted average operation: $\tilde{o_j} = \sum_i \lambda_{i,j} o_i$, parameterized by $\lambda = (\lambda_{i,j})$. Those approaches make the original problem differentiable with respect to the architecture parameters λ:

$$\begin{aligned} &\text{Find } \lambda \in \arg\min_{\lambda \in \Lambda} F(\lambda, w_T(\lambda)), \\ &\text{u.c.} \left\{ \begin{array}{rcll} w_0 & = & \Phi_{\text{init}}(\lambda, \epsilon) & \\ w_t & = & \Phi(w_{t-1}, \lambda, f), & t = 1, \dots, T \end{array} \right\}. \end{aligned} \tag{3.4}$$

Although very straightforward, these relaxations are not resource-efficient, since previously unitary computations are replaced by weighted averages of all possible computations. Besides, the optimization problem in Equation (3.4) is still quite complex, as the gradient $\nabla_\lambda F(\lambda, w_T(\lambda))$ is far from trivially estimated. Rather than unrolling the full optimization path of w_T with respect to λ, H. Liu et al., 2019b; Ahmed et al., 2018; Shin et al., 2018 propose to alternately optimize the inner and outer variables:

$$\text{Find} \left\{ \begin{array}{rcl} \lambda^\star & \in & \arg\min_{\lambda \in \Lambda} F(\lambda, w^\star) \\ w^\star & \in & \arg\min_{w \in \mathcal{W}(\mathbb{A})} f(\lambda^\star, w) \end{array} \right\}, \tag{3.5}$$

where $\mathcal{W}(\mathbb{A})$ represents the set of all possible weights, shared between the different architectures. H. Liu et al., 2019b additionally propose to solve the slightly more complex following problem:

$$\text{Find} \left\{ \begin{array}{rcl} \lambda^\star & \in & \arg\min_{\lambda \in \Lambda} F(\lambda, \Phi(w^\star, \lambda, f)) \\ w^\star & \in & \arg\min_{w \in \mathcal{W}(\mathbb{A})} f(\lambda^\star, w) \end{array} \right\}. \tag{3.6}$$

In Equation (3.6), the architecture parameters are optimized to be one-step ahead of the network parameters. The resulting gradient of the global objective with respect to λ leads to second-order optimization in λ (H. Liu et al., 2019b). In Equation (3.5), a first-order approximation is taken, and the dependency of w in λ is completely omitted. In practices, thoses problems are solved by alternating gradient descent approaches. Once the search is complete, the resulting architecture is made discrete, either by mapping the continuous results to the closest discrete element, or by ensuring the sparsity of the weighted averages using regularization.

3.4.4 Stochastic Relaxation

Numerous other authors rather consider a stochastic relaxation of the problem (Sirui Xie et al., 2019; Casale et al., 2019; Shirakawa et al., 2018). Instead of working with $\mathbb{A}$ directly, a family of probability distributions $\mathcal{P} = \{P_\theta : \theta \in \Theta \subseteq \mathbb{R}^{n_\theta}\}$ is introduced over the set of architectures (Akimoto et al., 2019). It is assumed that each $P_\theta \in \mathcal{P}$ has a density function p_θ which is differentiable with respect to θ. Working with this family, the outer objective in Equation (3.2) is replaced by:

$$\text{Find } \theta \in \arg\min_{\theta \in \mathbb{R}^{n_\theta}} \mathbb{E}_{A \sim P_\theta}\big\{F(A, w_T(A))\big\}. \tag{3.7}$$

The idea behind this relaxation is that optimality is reached if P_θ concentrates all its mass in a Dirac centered around the optimal architecture. This results in the following optimization problem:

$$\text{Find } \theta \in \arg\min_{\theta \in \Theta} \mathbb{E}_{A \sim P_\theta}\big\{F(A, w_T(A))\big\}, \tag{3.8}$$
$$\text{where } \left\{ \begin{array}{rcll} w_0(A) & = & \Phi_{\text{init}}(A, \epsilon) & \\ w_t(A) & = & \Phi(w_{t-1}, A, f), & t = 1, \dots, T \end{array} \right\}.$$

The constraints of the original formulation on θ disappear because of the dummy variable in the expectation and the objective becomes trivially differentiable in θ. In this case, the resulting gradient is a REINFORCE-like (Williams, 1992) gradient:

$$\nabla_\theta \mathbb{E}_{A \sim P_\theta}\big\{F(A, w_T(A))\big\} = \mathbb{E}_{A \sim P_\theta}\big\{\nabla_\theta \ln p_\theta(A) F(A, w_T(A))\big\}. \tag{3.9}$$

Notice that this black-box approach is similar to the gradient-free methods used in the discrete case, and is in particular very akin to evolutionary algorithms such as evolution strategies (H.-G. Beyer et al., 2002). Similarly to the continuous relaxation case, most authors rather optimize the objective over both θ and w simultaneously, by dropping the optimization path $w_T(A)$ and considering the currents weights w directly (Pham et al., 2018; Casale et al., 2019; Sirui Xie et al., 2019):

$$\text{Find } \left\{ \begin{array}{rcl} \theta^\star & \in & \arg\min_{\theta \in \Theta} \mathbb{E}_{A \sim P_\theta}\big\{F(A, w^\star)\big\} \\ w^\star & \in & \arg\min_{w \in \mathcal{W}(\mathbb{A})} \mathbb{E}_{A \sim P^\star_\theta}\big\{f(A, w)\big\} \end{array} \right\}. \tag{3.10}$$

Likewise, these are solved using alternating gradient descent, approximating expectations with Monte-Carlo integration. Finally, it is worth mentioning that some approaches based on a paradigm called HyperNetworks (Ha et al., 2017), try to directly predict the optimal weights of architectures $w^\star(A)$ through another learnable predictive model $\mathcal{H}(A)$ (Brock et al., 2018; C. Zhang et al., 2018). The advantage of such approaches is that it removes the necessity of the hypothesis that the parameters of all architectures can be described as a subset of the set of all possible parameters $\mathcal{W}(\mathbb{A})$. Thus more complex architectures distributions can thus be explored.

3.5 Performance Estimation

The last key concept of NAS (Elsken et al., 2019b) has to do with how the final score of an architecture is evaluated. Equation (3.2) suggests that the score associated to an architecture A is obtained by optimizing its weights, up to a certain points $w_T(A)$. Unfortunately, because of expensive training requirements, evaluating a single DCNN using this simple methods can take days to weeks. First NAS approaches (Real et al., 2019b; Zoph et al., 2018a; Zoph et al., 2017b) were carried out this way, and in turn required thousands of GPU days worth of computing. In light of this concern, many approaches have been explored to reduce the time required to come-up with architecture scores by proposing proxy-evaluations.

3.5.1 Proxy Training

An elementary approach to reduce the computational burden incurred by the evaluation of an architecture is to substitute the original training setting with a less precise, but more affordable surrogate one. In Zela et al., 2018, the authors limit the total training time of individual architectures to 3 hours, and make use of BOHB (Falkner et al., 2018), an HPO algorithm capable of terminating the evaluation of architectures that display sub-par intermediate performances. In a similar way, Klein et al., 2017 propose to perform HPO by training networks on a fraction of the original dataset. Several NAS approaches additionally consider training smaller variants of architectures, which size they increase when the search converges to a final model (Zoph et al., 2018b; Real et al., 2019a). Finally, (Dong et al., 2020) propose to work on down-sampled versions of image datasets. In a summary work, Zhou et al., 2020 propose to perform a systematic study of the fidelity of the scores obtained when combining the four aforementioned techniques: reducing the resolution of inputs, the number of epochs, the data ratio, and the size of the architectures considered. They conclude that some non-trivial combinations exist that preserve the ranking of the evaluations whilst further reducing training costs. Making use of a hierarchy of proxy that allowed them to increase fidelity as sampled architectures improved, combined with the evolutionary algorithm of Real et al., 2019a, they were able to find better performing architectures with 400 times less computations.

3.5.2 Model-Based Prediction

Other approaches try to predict the score $F(A, w_T(A))$ of a model either directly from the architecture A, or from intermediate results, such as the evolution of the scores $(F(A, w_i(A))_i$. Domhan et al., 2015; Klein et al., 2016; Baker et al., 2017 propose to extrapolate performance curves to try and predict the final score, terminating evaluations that are expected to reach poor

final performances. Most performance predictions directly try to learn the relationship between architectures, and their scores in a supervised learning fashion. Using a small dataset $\{(A_i, F(A_i, w_T(A_i)), i = 1 \dots, N\}$ a model $\widetilde{F}$ is learned such that $\widetilde{F}(A) \approx F(A, w_T(A))$. The form of the model $\widetilde{F}$ can be quite diverse, from classical FCNNs working on pre-determined representations of architectures (White et al., 2019; Ning et al., 2020), Graph Neural Networks (GNNs) (Z. Wu et al., 2020) directly working on the graph representation of the architecture (H. Shi et al., 2020; Dudziak et al., 2020). Most approaches only use the resulting models solely to extrapolate the score of a new architecture. In Luo et al., 2018 however, the authors learn to embed architectures using an encoder-decoder structure and optimize architectures by directly performing gradient ascent on the learned architecture manifold. White et al., 2021 perform a systematic study of model-based predictors, assessing their generalization ability on a suite of different benchmarks. We refer readers to their comprehensive work for a global picture on this family of models.

3.5.3 Weight-Sharing

In Section 3.4, we have introduced several approaches which proposed to transform the original discrete problem of Equation 3.2 into continuous alternatives. In order to create practical solutions, some authors further suggested to decouple the optimization of the architecture from the optimization of the weights, as described in Equation (3.5) and (3.10). However, such a feat can only be considered if weights can be factorized across different architectures, meaning that different architectures can share parts of their weights. In our different equations, we have captured this constraint by introducing $\mathcal{W}(\mathbb{A})$, the set of parameters shared by all the architectures present in $\mathbb{A}$.

This approach, which was first popularized by Pham et al., 2018, was adequately coined Weight-Sharing. Typically, an architecture can be seen as a graph of operations, with nodes representing operators applied to input feature maps, and edges indicating the flow of feature maps through the network. At each node, one of the several candidate convolutional operators is selected. Pham et al., 2018, suggest to consider that architectures which apply the same operator at a given node, can share the parameters of said operator at at the specified position. For greater clarity, we illustrate this behavior in Figure 3.3. The global network which is obtained by considering all individual node operations and connectivity pattern allowed by the search space is referred to as the "super-net". The WS approach has gained incredible popularity in the NAS literature (H. Liu et al., 2019b; Sirui Xie et al., 2019; Casale et al., 2019; Stamoulis et al., 2019). The main reason behind this ubiquitous acceptance is its impressive efficiency. Indeed, in

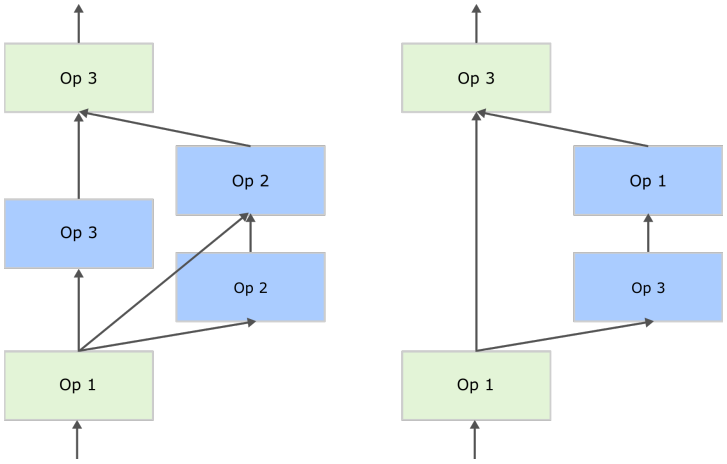


Figure 3.3: Examples of two cells from a cell-based search space with different connectivity patterns and operators at each node. Nodes that are in green indicate that the same operator is used at the same location and thus that their weights can be shared.

substance, WS allows to replace the weights $w_T(A)$ obtained by the costly training of an architecture A, by the set of weights readily extracted from a single trained super-network. Where the evaluation of N architectures induces N learning processes in traditional NAS approaches, adopting the WS paradigm allows to only train the super-net once, and extract the weights of the N networks from the result. Because of this remarkable efficiency, NAS algorithms which make use of WS are traditionally called One-Shot NAS algorithms.

Different approaches combine the optimization of the architecture and the optimization of the shared weights of Equation (3.5) and (3.10) in different ways. Most alternate between the two, with mini-batches of data respectively coming from training and validation sets (H. Liu et al., 2019b; Sirui Xie et al., 2019; Casale et al., 2019; Pham et al., 2018). In turn, the shared weights are optimised so as to better fit the architectures selected by the NAS algorithm. Other approaches, such as Bender et al., 2018 or Guo et al., 2020, instead propose to first learn a shared set of weights adapted to all the architecture of the search space, and only then use the obtained weights to perform the architecture optimization process. This paradigm facilitates the analysis of correlations, as methods training both weights and architectures together induce a bias towards architectures with good early evaluations, as we will see in the next chapter.

3.6 Conclusion

In this chapter, we have introduced the idea behind the NAS paradigm, and how it was inspired by the history of DCNN. We summarized and presented relevant literature for three core aspects of NAS approaches: the search space, the optimization process, and the performance estimation strategy. In particular, we have tried to present and describe the different optimization approaches using the same framework. The work of this chapter was primarily meant as an introduction to NAS, and we will abundantly refer to the concepts presented here in the two following chapters.

Chapter 4

Weight-Sharing on the NAS-Bench-101 Dataset: A Practical Case Study

Contents

In this chapter, we take advantage of NAS-Bench-101 (NB-101), a dataset of architecture evaluations, to challenge the efficiency of a uniform-sampling based WS variant on several representative search spaces. After reviewing previous studies on WS and highlighting several of their shortcomings, we introduce our own experimental setup, from which we extract several good practices that one should keep in mind when evaluating WS. With our experiments we first establish that, given the correct evaluation procedure, WS is able to produce accuracy scores decently correlated with standalone ones. We then provide evidence that on some search spaces, this WS variant is able to rapidly find better than random architectures, whilst it is equivalent or sometimes even worse than a baseline random search on others. We find that when given the same budget, the probability of superiority of an architecture found using WS over an architecture found through random search can vary between 7% and 78% depending on the search space. We present evidence that the search space itself has an intricate effect on the capabilities of WS and can bias weight-sharing towards certain architectural patterns with no clear accuracy advantage. We conclude that the impact of WS is heavily search-space dependent and difficult to anticipate for a given problem.

4.1 Introduction

Because of expensive training requirements, evaluating a single DNN architecture can take days to weeks. In turn, original NAS approaches (Real et al., 2019b; Zoph et al., 2018a; Zoph et al., 2017b) required thousands of GPU days worth of computing, only to find conformations slightly better than expert-designed ones. In light of this concern, many methods have been explored that could drastically cut the resources required to perform NAS, and today's literature is blooming with approaches requiring less than a day of computations Pham et al., 2018; H. Liu et al., 2019b; Sirui Xie et al., 2019; Casale et al., 2019.

Most of these efficient methods rely on a computational trick called Weight-Sharing (WS), an approach first popularized by Brock et al., 2018 and Pham et al., 2018, which we introduced in Chapter 3. Despite a growing literature, the effect of WS on the performance of NAS is still poorly understood. A particular concern is the quality of the scores obtained with the super-net. Employing WS implies substituting metrics obtained after standalone training with metrics derived from the shared set of parameters. Both quantities thus need to be correlated: if networks with excellent standalone performance are under-evaluated by the super-net or vice-versa, the process could be pointless and even detrimental. A possible reason for a lack of studies on the matter is the cost required by the training of a proper amount of architectures. As described in Section 4.2, several works mitigate this issue by assessing the correlations between evaluations of the super-net and true evaluations in reduced settings, either evaluating few architectures or studying a drastically reduced search space.

In this chapter, we leverage the architecture evaluations of the NAS-Bench-101 (NB-101) (Ying et al., 2019) dataset to scrutinize the correlations offered by a simple WS variant and investigate whether it can improve the efficiency of baseline NAS algorithms. We perform this comparison in a realistic setup in which we account for all the computational costs incurred by WS. To strengthen the results of our different analyses, we consider seven different search spaces with specific structural properties and reproduce this analysis on each of them. Besides, to grant the results clear statistical significance, we compare the different methods with an appropriate number of seeds given a pre-specified effect size, totaling around 25 WS runs per search space. To our knowledge, this study is the first to consider analysing WS on such a wide variety of search spaces, using a fair comparison protocol that explores all sources of computational cost, and with strong enough statistical power to draw meaningful conclusions, albeit in the reduced setting of NB-101.

We summarize the key take-aways of our experimental setup in a list of

several good practices, which we believe are essential to accurately evaluate WS. They are the following: evaluating WS using a wide variety of search spaces, avoiding the different sources of biases that could come up during the estimation of correlations, visualizing said correlations, properly assessing computation duration, and guaranteeing statistical significance of the NAS results.

Based on these observations and the resulting good practices, our experimental results show that, on the NB-101 dataset: (i) with the correct methodology, one can get decent Spearman's rank correlations between super-net proxy evaluations and real evaluations on several search spaces containing hundred thousands of architectures, which can range between 0.46 and 0.71 depending on the search space; (ii) WS is often able to quickly spot better than random architectures, but is sometimes on par or even worse than a random search baseline, as the probability of superiority of WS over random search under a one-shot setting can vary between 7% and 78% depending on the search space; (iii) global correlations evidenced in (i) are not the limiting factor of uniform-sampling based WS, as search spaces offering better global correlation do not always offer better WS performances; (iv) the search-space can bias the super-net training and the resulting evaluations, making WS fairly unreliable.

4.2 Related Work

In the first part of this section we describe several works trying to measure the efficacy of WS. In the second part, we present several tricks and WS training variants which have been introduced in the literature to improve the correlations granted by super-nets, and which we explore in our experiments in Section 4.3.2 and 4.4.1.

4.2.1 Evaluating Weight-Sharing

In Bender et al., 2018, the authors train a super-net on a search space of their own. Path dropout is applied during training to randomly cut some portions out of the super-net. A random search is then used to find a good architecture. To validate the use of the super-net as a proxy, $20,000$ architectures are sampled from the chosen search space and evaluated with the resulting super-net. This set is then partitioned into several bins based on the obtained proxy scores. For each bin, 4 architectures are sampled and trained from scratch for a small number of epochs (around 10% of the length of baseline training) before being evaluated. The authors note visually satisfying correlations between the two proxies, but do not report any metrics. For computational reasons, correlations with full budget standalone accuracy are not reported. Moreover, because the few models evaluated are

evenly spread across the range of possible proxy accuracies, the produced appealing correlations plots might not be representative of the whole search space.

In K. Yu et al., 2020b, the authors quantify the impact of WS on NB-101, and on a small language modeling task. They find that a simple random search baseline is competitive with and often outperforms several NAS algorithms exploiting WS such as DARTS (H. Liu et al., 2019b), ENAS (Pham et al., 2018) and NAO (Luo et al., 2018), which furthermore display high variance in their results. They report poor correlations between the ranks obtained using a super-net and with standalone evaluations on the considered search spaces. We note however that the used algorithms were not specifically designed to produce good correlations at the end of training, but rather exploit them to rapidly converge to seemingly good architectures.

Y. Zhang et al., 2020b explore another small search space of 64 architectures dedicated to computer vision. They train several super-nets using different seeds, and report high variance in the relative rankings of the architectures obtained with WS. They notice that during super-net training, strong interactions exist between architectures, as updates in some models can either improve or deteriorate the performance of others. They reach correct correlations with standalone rankings, albeit the important variance seems to hinder the practical implications of the super-net. They propose several approaches to reduce the amount of WS between architectures, such as fine-tuning parts of the super-net before evaluation or grouping architectures into different sets according to different strategies.

In X. Chu et al., 2019, the authors argue that WS is limited by uneven sampling of individual weights throughout the learning process. Although they are seen equally often on average, some might locally be overrepresented due to chance, effectively biasing the weights of the super-net. To prevent this, they propose to average the gradient updates of the shared parameters over n samples, chosen such that each of the n elementary operations of the super-net appears exactly once. They combine their super-net trained with the aforementioned strategy with a multi-objective genetic algorithm, to build a Pareto front of accurate architectures with adequate numbers of parameters and multiply-add operations. To justify their approach, they sample 13 models equally distant from the found Pareto front and compare the accuracy obtained with WS against standalone ones, and reveal that rankings are well preserved. However, details regarding this experiment are lacking and it is likely that the result does not hold for the whole search space.

Luo et al., 2019 also note a strong variance in the results of a few NAS algorithms exploiting WS, and impute the poor results to meager correla-

tions, which they illustrate using 50 randomly sampled architectures. After identifying several factors that could hinder performances, such as short training times and bias towards simple architectures, they propose straightforward solutions for each of them, and eventually demonstrate improved performances.

In A. Yang et al., 2020, the authors benchmark 8 NAS methods, 6 of which are based on WS, on 5 different datasets. Their careful analysis reveals that many NAS algorithms have trouble outperforming a random search. They further argue that the use of training tricks in the evaluation protocol have a greater influence over the final performance than the NAS algorithm itself. Besides, they note that search spaces typically used in NAS have a very narrow accuracy range and are thus already tuned to the considered tasks.

Zela et al., 2020 use NB-101 to evaluate NAS algorithms exploiting WS. However, because the authors choose to study DARTS (H. Liu et al., 2019b) variants, they create their own search spaces to perform evaluations, whereas we directly use the whole NB-101 search space. In one of their experiments, they report the evolution during training of the correlations between evaluations obtained using the super-net, and evaluations queried from the NB-101 dataset. They report poor or nonexistent correlations for most algorithms, which contradicts our findings.

Y. Zhang et al., 2020a perform experiments with an intent similar to that of our study. Instead of considering several sub-spaces of NB-101, the authors rely on different search-spaces typically found in the NAS literature. On NB-101, the authors consider a relatively small sub search-space, introduced by Zela et al., 2020, which contains around $42,000$ architectures, around 10 times less than the number of available unique architectures. For each search space, they train several super-nets using the same uniform-sampling variant of WS (Guo et al., 2020). Their results suggest that it is possible to reach correct correlations, but that there is an important variance with respect to the search space considered. This variance of behavior is more preeminent in their work, given the greater diversity of search spaces considered. Because the search space that they consider are not all based on available NAS benchmarks, some of those correlations are obtained by training only a few tens of architectures. They additionally reveal that on some search spaces uniform WS is biased towards certain operators, and that although WS cannot be used to accurately select top architectures, it is paradoxically quite capable of finding the worst ones. Unlike this work, the authors conclude that the positive effect of WS on NAS is clear cut and positive. We explain this discrepancy by the fact that they do not account for super-net training times when comparing against random search.

For clarity purposes, we gather in Table 4.1 a quick summary of each work

AUTHORS	SPACE SIZE	#SPACES	#EVALS	UNBIASED	UNIFORM	GOOD CORRS.	WS ↗
Bender et al.	$\sim 10^9$	1	$\sim 10^2$	×	✓	✓	✓
K. Yu et al.	$\sim 10^5$	2	200	✓	×	×	×
Y. Zhang et al.	$\sim 10^1$	1	64	✓	✓	✓	n.a.
X. Chu et al.	$\sim 10^{19}$	1	13	×	✓	✓	✓
A. Yang et al.	$\sim 10^{25}$	1	n.a.	n.a.	n.a.	n.a.	×
Luo et al.	$\sim 10^{25}$	1	50	✓	✓	×	n.a.
Zela et al.	$\sim 10^5$	3	100	✓	✓	×	n.a.
Y. Zhang et al.	variable	5	variable	✓	✓	✓	n.a.
Ours	$\sim 10^5$	7	10,000	✓	✓	✓	×

Table 4.1: For each reference in Section 4.2, we gather the conclusions of the authors on super-net correlations (GOOD COORS.), and whether WS can consistently improve NAS (WS ↗). In each entry, we specify the approximate number of architectures in the search space (SPACE SIZE), the number of search spaces considered (#SPACES), the number of architecture evaluations performed (#EVALS), whether the selection of the architecture and/or their evaluation was unbiased (UNBIASED EVALS), and whether the considered WS variant used uniform sampling (UNIFORM WS). We highlight in red what we think are shortcomings of the different analyses. "n.a." stands for not applicable, meaning that the column was not relevant to the study.

described in this section and their conclusions on super-net correlations and whether WS can improve NAS consistently. We embed each paper using five dimensions: the approximate size of the considered search spaces, the number of studied search spaces, the number of architecture evaluations performed, whether the architecture evaluations are biased and whether the WS variant used uniform sampling. We additionally highlight in red what we think are shortcomings of the different studies, which we briefly describe here. The authors of Y. Zhang et al., 2020b consider an unrealistic search space of merely 64 architectures. All studies except Zela et al., 2020; Y. Zhang et al., 2020a and ours consider less than three search spaces, which we think is too narrow to get an understanding of the various behaviors of WS. Almost all authors evaluate below 1,000 architecture when analysing correlations, resulting in limited statistical significance. The authors of Bender et al., 2018; X. Chu et al., 2019 perform biased evaluations which are likely to overestimate the quality of the correlations offered by WS. On the contrary, authors of K. Yu et al., 2020b report their results for WS based NAS algorithms that do not use uniform sampling and are therefore likely to underestimate the quality of the correlations offered by WS.

All in all, the authors of Bender et al., 2018; Y. Zhang et al., 2020b; X. Chu et al., 2019; Y. Zhang et al., 2020a report reaching correct correlations whilst the authors of K. Yu et al., 2020b; Luo et al., 2019; Zela et al., 2020 report poor correlations, both sides coming with various shortcomings in the methodology. Regarding NAS performances when exploiting WS, authors

of K. Yu et al., 2020b; A. Yang et al., 2020, suggest that when properly evaluated, WS-based NAS is no better than random search, whilst most of the NAS literature tacitly agrees on the opposite. With this study we aim at reaching a satisfactory conclusion on those two aspects in a specific setup, which is using the single-path one-shot approach of Guo et al., 2020 on the search space described by the NB-101 benchmark Ying et al., 2019.

4.2.2 Enhancing Weight-Sharing Correlations

When evaluating a super-net to obtain a score for a given architecture, it is possible to directly exploit the whole super-net and perform a standard forward pass on the impending data whilst activating the graph corresponding to the evaluated net. However, several works report the benefit of adapting the statistics of the inherited batch normalization layers (Bender et al., 2018; Guo et al., 2020) to the architecture at hands.

The weights w of the super-net are updated through gradient descent with respect to the objectives described in Chapter 3. The resulting gradient takes the form of an expectation which is commonly approximated by an empirical average. However, in practice Guo Guo et al., 2020 only use a single architecture to estimate the expectation. Although this process is unbiased, it results in high variance updates of w. Decreasing this variance by sampling more models could be a straight-forward way to improve the super-net optimization, at a higher computational cost.

In Stamoulis et al., 2019, the authors propose to not only share the weights of the basic operations between architectures, but to further merge the weights of all basic operations at a given node into a single set of parameters. For instance, if two basic operations were a $x \times x$ convolution and a $y \times y$ convolution with $x > y$, instead of representing each operation with its own set of kernels, one could use a single set of size $x \times x$, and apply the $y \times y$ convolution by extracting the sub-kernels of size $y \times y$ from the the shared set.

Authors of Luo et al., 2019 identify in their work a bias towards architectures with fewer parameters, as they get trained faster. They propose to correct this bias by sampling architectures pro-rata to their number of parameters, sampling complex architectures more often.

4.3 Methods

In this section, we describe the protocols used to address the following questions: Are WS-based proxy-accuracies significantly correlated with standalone ones? How do correlations vary with respect to the training and evaluation regimes of interest? Can the super-net find better than random

architectures? Can it find competitive architectures? How do the results vary between search spaces? Outcomes of the experiments mentioned hereafter are described in Section 4.4

4.3.1 Impact of Search Spaces on NAS Performances

To measure the impact of the search space on WS, we consider for each experiment several sub-sets of the NAS-Bench-101 (NB-101) search space, which we now introduce.

In NB-101, feature maps going to the output of a cell are concatenated. Given that the shape of the output is fixed across all possible cells, and that different cells might have different number of outputs nodes, this means that the kernel depth of a cell's nodes varies over architectures, possibly hindering the use of WS. A reasonable splitting strategy is to consider the sets $(\mathcal{A}_i)_{i=1,\dots,4}$, in which architectures contain exactly i nodes connected to the output (beside the input node, which is added and not concatenated), resulting in compatible nodes with feature maps of equal sizes [1].

We also consider the full NB-101 set, which we call $\mathcal{A}_{\text{FULL}}$: we solve the aforementioned problem by dynamically adapting the number of feature maps used for each node depending on the architecture: each node, as seen by the super-net, contains the maximum number of filters for the given layer, but sampled architectures only inherit the first n filters of the filter-bank, where n is determined so as to satisfy the constraints on the output size of the architecture's cell. Other inheritance strategies could be considered, such as randomly selecting the correct number of filters from the available filter bank, but K. Yu et al., 2020a notice that this scheme is both simple and efficient.

Early results additionally compelled us to study the influence of residual connections on WS. It is well known that edges connecting the input node to the output node of a cell, known as residual connections He et al., 2016, significantly improve the training of convolutional architectures. Suspecting that this is also true for sub-nets of a supernet, we consider two additional search spaces: $\mathcal{A}^{\text{RES}}_{\text{FULL}}$ and $\mathcal{A}^{\text{NRES}}_{\text{FULL}}$, containing all architectures of NB-101 with and without residual connections respectively. Figure 4.1 sketches the structural properties of the different search spaces.

4.3.2 Ranking Architectures with Weight-Sharing

With this experiment, we want to estimate the achievable correlation level between super-net based proxy accuracies, and accuracies obtained with

[1] A difference of 1 feature map can still appear in $\mathcal{A}_3$ since the number of final feature maps after concatenation is rarely divisible by 3. In such case, one branch may end up with one more or one less feature map, e.g. $[42, 43, 43]$ for 128 output feature maps

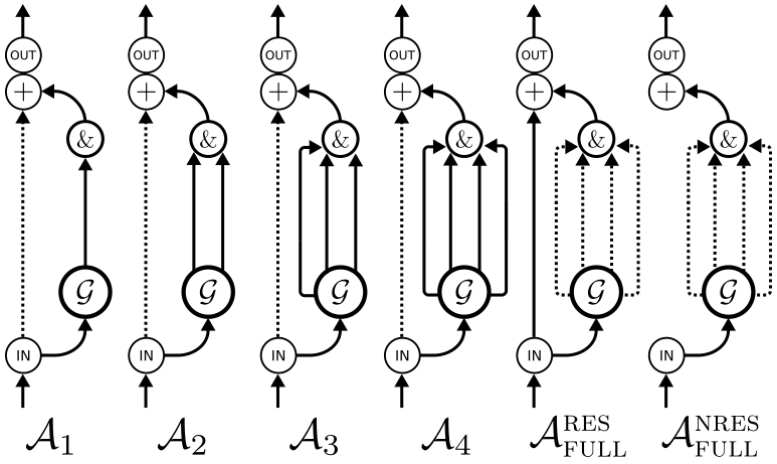


Figure 4.1: Structural properties of the different search-spaces. On each graph, "IN" and "OUT" denote the input and output of the cell, while "+" and "&" denote the sum and the concatenation of incoming feature maps. We only display the edges discriminating at least one search space and represent the rest of the graph with node $\mathcal{G}$. If an edge does not discriminate a specific search space, we represent it with a dotted line. $\mathcal{A}_1$ to $\mathcal{A}_4$ are characterized by the number of edges concatenated after $\mathcal{G}$, and $\mathcal{A}^{\text{RES}}_{\text{FULL}}$ and $\mathcal{A}^{\text{NRES}}_{\text{FULL}}$ by the presence or absence of a residual connection.

standalone training of architectures.

We train several super-nets on the CIFAR-10 dataset. Following Guo et al., 2020, for each mini-batch of data seen during training, a single architecture is uniformly sampled from the search space. The weights of the super-net are then updated according to the computational graph generated by the activation of this architecture.

We reuse the hyper-parameters of Ying et al., 2019 and train the super-nets with the RMSPROP optimizer. The initial learning rate is set to 0.2 and decayed to 0 using a cosine annealing schedule over 432 training epochs, which is four times the original time budget used in NB-101. As noted by Luo et al., 2019 longer training times are often required for super-nets to converge. The momentum is set to 0.9, weight decay to 10^{-4}, and ϵ to 1. The batch size is kept to 256 and the momentum and ϵ of the batch normalization layers are respectively set to 0.003 and 10^{-5} [2]. We do not tune hyper-parameters as doing so would require the computationally expensive training of several architectures in a realistic setting.

We however reduce the initial number of filters to 16 compared to the original 128, in order to accelerate training and evaluations. Earlier iterations

[2]Following PyTorch convention.

of our experiments revealed that using the default 128 value requires fairly longer training times for the super-nets, without significantly improving the resulting correlations. Although setting it to 16 is somewhat arbitrary, Zela et al., 2020 also note that accuracies obtained after training architectures with 16 initial filters are greatly correlated with those obtained using the baseline 128 filters. We thus consider that the number of filters is not the limiting factor of our different WS experiments. This setup additionally mimics one-shot NAS approaches such as H. Liu et al., 2019b; Casale et al., 2019; Pham et al., 2018, where the architecture found by the search is up-scaled and retrained to further improve accuracies.

We train 5 different super-nets on each search space. We then randomly sample 1,000 unique architectures from the search space and compute their proxy accuracies on the held-out validation dataset. We match those accuracies with the average validation accuracies returned by NB-101. To quantify the quality of the correlations between the two, we use Spearman's rank correlation coefficient.

We estimate accuracies with the super-nets performing either no fine-tuning (NO-FT) or after fine-tuning the batch-norm statistics (BNS-FT). Without fine-tuning, we directly use all the parameters and batch-norm statistics of the super-net. When adapting the batch-norm statistics of the super-net to a specific architecture, we simply estimate them using a single mini-batch of training data (i.e. 256 images).

We additionally study the effect on correlations of the different training variations mentioned in Section 4.2.2. We follow the same protocol but always fine-tune the batch-norm statistics (BNS-FT). We consider four variants: averaging the gradients over 3 sampled architectures (AVG-3), sampling architectures pro-rata to their number of parameters (PRO-RATA) Luo et al., 2019, following the single-kernel approach of Stamoulis et al., 2019 (SINGLE-K), and combining the single-kernel and pro-rata approaches (S-K + P-R).

4.3.3 Impact of Weight-Sharing on NAS Performances

Quantifying the correlation level obtained with WS on realistically sized search-spaces is of interest on its own, but is insufficient to conclude on the efficiency of WS itself. Indeed what eventually matters are not the correlations as such, but rather the quality of the architectures that can be found by exploiting them. With the experiment of this section, we aim at characterizing the interest of substituting super-net evaluations to the standalone evaluations when performing NAS.

To investigate this, we consider two very simple search algorithms: random search (RS) and a greedy local search (GS), which we both use in two

different settings. In the baseline NAS setting (BS), we directly maximize the validation accuracy of models as returned by the benchmark. In the weight-sharing (WS) setting, we maximize the performance of architectures obtained from the super-net after fine-tuning batch-norm statistics (BNS-FT).

When performing a RS, we directly sample and evaluate a fixed number of unique architectures from the given search space. The GS is a plain hill-climbing method that has been shown to be very effective on NAS benchmarks White et al., 2020; Ottelander et al., 2020. A random architecture is initially sampled from the search space. All of its neighbours are evaluated, and the neighbour with the best score is selected to be the new current architecture. This process is repeated until a fixed point (an architecture whose best neighbour is itself) is reached, after which a new random architecture is sampled. The algorithm stops when the total evaluation budget is reached. The set of neighbours of a given architecture is comprised of the architecture itself, all architectures obtained by modifying a single node operation, all valid architectures obtained by adding or removing a single edge in its adjacency matrix, as well as all valid architectures that can be obtained by adding a single node to the architecture.

In the baseline NAS setting, each search is given a budget of $10,000$ architecture evaluations. For each algorithm, we report the evolution of the test regret. The test regret is computed after each model evaluation by comparing the mean test accuracy of the model with the running best validation accuracy and the best mean test accuracy of the considered search space:

$$\mathcal{R}_{test}(t) = \max_{a \in \mathcal{A}} acc_{\text{test}}(a) - acc_{\text{test}}(\arg \max_{a \in (a_i)_0^t} acc_{\text{valid}}(a)), \tag{4.1}$$

where $\mathcal{A}$ is the considered search space, $a_1, \ldots, a_t$ the architectures trained on their own and evaluated on the validation dataset at time t, $acc_{\text{valid}}(a)$ and $acc_{\text{test}}(a)$ respectively refer to the validation and test accuracies of the architecture a.

When exploiting WS, the strategy is slightly different. We first optimize the architecture based on the super-net proxy accuracy. Although more evaluations could be considered, we find $10,000$ to be enough for an efficient algorithm such as GS to converge. Each evaluation requires performing an inference on a validation dataset. Although this is orders of magnitude cheaper than training architectures, we still find the process to be quite costly, requiring around 4 seconds per architecture. A search performed using the proxy thus requires around ten hours, which, although reasonable on its own, has to be multiplied by the number of search spaces and considered seeds.

Once the optimization is finished, we assess the quality of the found solutions by querying their true validation accuracies on the benchmark. To obtain regret curves such as those of the baseline methods, we evaluate the 10,000 intermediate models considered during the WS searches in order of decreasing proxy accuracy. Again, we report the evolution of this test regret as a function of time as described in Equation (4.1).

The different regrets reported are plotted as a function of search time. This search time is well approximated by the duration of the training and evaluation of the different models. For a fair time-wise comparison of the regrets, we account in the WS-based paradigm for both super-nets training time and the duration of the evaluation of the 10,000 models. Unfortunately, we were not able to perform experiments using the same hardware as Ying et al., 2019. As a result, using the training times reported by NB-101 would be biased, as theirs was much more efficient. To circumvent this, we used a common setup comprised of two NVidia K80 GPUs and estimated for each distinct architecture the time required to perform a single forward and a single backward pass. We averaged this quantity over three independent measures. To estimate the duration of a training, we simply multiply those quantities by the appropriate number of forward and backward passes. Besides, given that during super-net training each sampled architecture only activates the necessary parts of the network, we approximate the super-net training time by the average training time of the architectures of the considered search space.

The original purpose of WS is to perform NAS in a one-shot paradigm L. Xie et al., 2020 by quickly selecting a single architecture based on proxy accuracy scores, and re-train it from scratch. Given a search space, we note for each WS run the average time spent before the first real evaluation (super-net training and evaluation, and training of the selected architecture) and report the difference between the obtained regret with the regret achieved by the baseline methods under the same time budget. To get an insight on the size of the effect, we report Cohen's d for the considered measures, which is defined as the average difference divided by the pooled standard deviation. Cohen's d Cohen, 1988 reflects how the measured mean absolute difference relates to the standard deviations of both populations. An effect is usually considered important if $|d|> 0.8$, mild if $|d|\approx 0.5$ and small if $|d|<0.2$ Cohen, 1988.

We assess the statistical significance of the differences using a Student's t-test. For the baseline searches, since there is almost no computational requirements (because a run is a simple query of the NB-101 dataset), we follow Ying et al., 2019 and perform 100 runs. As explained above, WS-based search takes a non-negligible amount of time. Given our computational budget, we settle on ensuring that any effect of at least medium size can be

properly measured, with a statistical power of $\beta = 0.8$ and a significance level $\alpha = 0.05$. We estimate [3] that around 25 runs of the WS-based approach should grant such guarantees.

4.3.4 Good Practices

As pointed out in Section 4.2 and Table 4.1, existing analyses of WS in the literature bear several limits. In this section we present what we think are good practices for evaluating the impact of WS on NAS. With those guidelines, we hope to pave the way for future research and possible analyses of other benchmarks. We further succinctly indicate in Table 4.2 the strategy that we employed to avoid each pitfall.

- **Search Space Variability**: When evaluating WS, we suggest not relying on a single search space, but rather explore various search spaces. This diversity gives an idea of the intrinsic variance associated with the procedure. As discussed in Section 4.4.3 this multiplicity is key as WS performance varies greatly with the explored search space.
- **Unbiased Estimation of Correlations**: To fairly assess the quality of correlations between super-net and standalone scores, one must limit biases that could emerge from the super-net in three respects: sampling biases during training that result in some architectures being trained more often than others, biases in the selection of architectures during evaluation that result in correlations not representative of the whole search space, and biases in the evaluation procedure itself that result in poor non-representative individual performance.
- **Visualization of Correlations**: Correlation metrics such as Spearman's rank correlation coefficient cannot capture the complexity of a plain scatter plot of predicted against true score. Such scatter plots additionally reveal over or under evaluation biases created by the super-net. Examples of such figures can be observed in Figure 4.3 and Figure 4.4.
- **Assessment of Computation Times**: When reporting the evolution of regrets as a function of computing time, all sources of computations must be included. This includes the time required to train the super-net, as well as to perform the individual evaluation of all the architectures of interest.
- **Satisfying Statistical Power**: The proper statistical comparison of two NAS approaches is crucial to reach any conclusion. Unfortunately, it is quite common in the NAS literature to claim the superiority of NAS method based on a single run. One must make sure that enough

[3] using the STATSMODELS python package (Seabold et al., 2010)

runs are considered to get statistically significant comparisons Colas et al., 2018.

4.4 Results

We now describe the results of the above studies. We then discuss the influence of the search space on WS.

4.4.1 Ranking Capabilities of Weight-Sharing

For each search space, we report in the left part of Table 4.3 the average over 5 different super-nets of the rank correlation between the standalone accuracy returned by NB-101 and the proxy accuracies obtained after applying the two evaluation protocols described in Section 4.3.2. NO-FT refers to performing no fine-tuning, and BNS-FT to fine-tuning the batch-norm statistics. Without any fine-tuning, correlations are poor across all search-spaces, with substantial variances. With batch-norm statistics fine-tuning, the average correlation increases by 270% over the NO-FT scheme, granting almost 3 times better results on average.

In the right part of Table 4.3, we present the average rank correlations obtained from training the super-net with the different variants described in Section 4.2.2, and applying batch-norm statistics fine-tuning during evaluations. SINGLE-K refers to exploiting the single kernel variant of Stamoulis et al., 2019, PRO-RATA to sampling architectures pro-rata to their number of parameters (Luo et al., 2019), and S-K + P-R the combination of the two. AVG-3 refers to averaging gradients during the training of the super-net over three architectures. We notice that all approaches lead to a small improvement of the correlations, as well as a slight variance reduction.

These simple results show that it is possible to get correlations between proxy evaluations performed with WS and full-budget evaluations as long as batch-norm statistics are adapted to the evaluated architectures. We notice that all the works mentioning poor correlations in Section 4.2 do not detail their evaluation setup, and we suspect that they do not adapt batch-norm statistics. Additionally, it is possible to further improve the resulting correlations by modifying the super-net training in various ways.

4.4.2 Can Weight-Sharing Improve NAS?

The regret curves of the different NAS experiments described in Section 4.3.3 are reported in Figure 4.2. The relative position of the different methods largely depends on the considered search space, but one can see that given sufficient training time, baseline GS outperforms all other algorithms. In a one-shot setting, (i.e. given a time budget equivalent to the evaluation

Good Practice	Our Approach	Observed Results
Search Space Variability	We explored seven different sub search-spaces of NB-101.	Correlations and WS performance vary greatly with the considered search space. For instance, the Spearman coefficient can range between 0.46 and 0.71.
Unbiased Estimation of Correlations	We used the uniform sampling scheme of Guo et al., 2020.	Uniformly sampling architectures during training is a baseline that allowed us to measure the contribution of other more advanced techniques.
	We uniformly sampled architectures from the considered search space.	Uniformly sampling architectures during evaluation allowed us to conclude of the correlation capabilities of WS, as well as identifying some biases in the evaluation of certain architectures
	We tuned batch-norm statistics during evaluation.	The average correlation over the different search spaces increases by 270% when fine-tuning batch-norm statistics.
Visualization of Correlations	We introduced scatter plots with colors indicating specific properties.	We identified some biases in the evaluation of certain architectures.
Assessment of Computation Times	We accounted for super-net training and evaluation times.	Training super-nets takes a non-negligible amount of time and accounting for this duration makes a baseline random search equivalent to a random search employing WS on search spaces.
Satisfying Statistical Power	We met our statistical power and significance goals with 25 runs.	This allowed us to conclude on the efficiency of WS against a random search baseline.

Table 4.2: For each good practice introduced in Section 4.3.4, we indicate the strategy that we employed and the observed results.

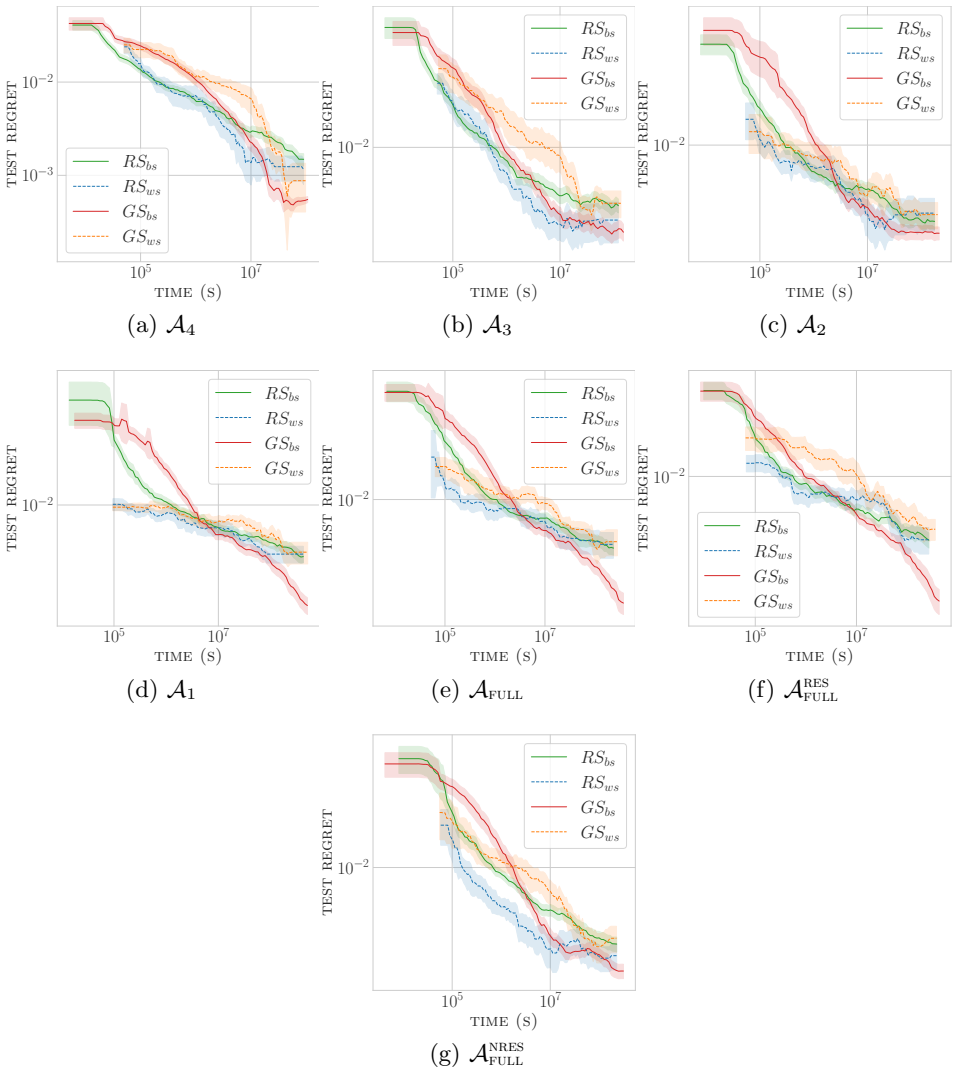
	NO FT	BNS-FT	SINGLE-K	PRO-RATA	AVG-3	S-K + P-R
$\mathcal{A}_4$	0.08 ± 0.17	**0.64 ± 0.03**	0.66 ± 0.01	0.68 ± 0.03	∗0.67 ± 0.02	**0.69 ± 0.02**
$\mathcal{A}_3$	0.12 ± 0.15	**0.59 ± 0.03**	0.62 ± 0.02	0.63 ± 0.02	0.61 ± 0.03	**0.66 ± 0.02**
$\mathcal{A}_2$	0.24 ± 0.03	**0.60 ± 0.04**	0.64 ± 0.02	0.64 ± 0.02	0.61 ± 0.01	**0.65 ± 0.02**
$\mathcal{A}_1$	0.32 ± 0.05	**0.68 ± 0.02**	0.72 ± 0.02	**0.75 ± 0.02**	0.73 ± 0.01	0.67 ± 0.01
$\mathcal{A}_{\text{FULL}}$	0.24 ± 0.05	**0.56 ± 0.04**	∗**0.63 ± 0.02**	0.59 ± 0.03	0.61 ± 0.02	0.58 ± 0.02
$\mathcal{A}_{\text{FULL}}^{\text{NRES}}$	∗0.11 ± 0.05	∗**0.46 ± 0.06**	∗**0.58 ± 0.02**	0.56 ± 0.03	0.52 ± 0.02	0.49 ± 0.02
$\mathcal{A}_{\text{FULL}}^{\text{RES}}$	0.34 ± 0.10	**0.71 ± 0.02**	0.68 ± 0.01	**0.72 ± 0.02**	0.69 ± 0.02	0.66 ± 0.01

Table 4.3: Spearman's Rank Correlation Coefficient Between WS and Standalone Evaluations for Various Search Spaces, WS Variants, and Evaluation Schemes. We report the average over 5 independent runs and the 95% confidence interval for its estimation. On the left, we use the baseline WS approach and we perform either no fine-tuning, or batch-norm statistics fine-tuning. On the right, we test some variants of WS described in Section 4.2.2 and always fine-tune batch-norm statistics during evaluations. Results marked with an asterisk ∗ indicate that one of the super-net failed to converge, and that the reported statistics are computed using only the four others.

	$RS_{bs} - RS_{ws}$	$RS_{bs} - GS_{ws}$
$\mathcal{A}_4$	-0.68 ± 0.80 (p=0.00, d=−0.85)	-0.82 ± 0.83 (p=0.00, d=−0.99)
$\mathcal{A}_3$	-0.20 ± 0.77 (p=0.25, d=−0.26)	-0.62 ± 0.88 (p=0.00, d=−0.71)
$\mathcal{A}_2$	0.36 ± 0.87 (p=0.07, d=+0.42)	0.54 ± 0.85 (p=0.01, d=+0.63)
$\mathcal{A}_1$	1.18 ± 1.08 (p=0.00, d=+1.09)	1.30 ± 1.10 (p=0.00, d=+1.18)
$\mathcal{A}_{\text{FULL}}$	0.53 ± 1.22 (p=0.06, d=+0.44)	0.76 ± 0.96 (p=0.00, d=+0.79)
$\mathcal{A}_{\text{FULL}}^{\text{NRES}}$	0.63 ± 1.30 (p=0.03, d=+0.48)	0.32 ± 1.44 (p=0.32, d=+0.22)
$\mathcal{A}_{\text{FULL}}^{\text{RES}}$	0.33 ± 0.52 (p=0.01, d=+0.63)	-0.02 ± 0.58 (p=0.87, d=−0.04)

Table 4.4: We report the average regret differences between the baseline random search and the WS-based random search ($RS_{bs} - RS_{ws}$), and between the baseline random search and the WS-based greedy search ($RS_{bs} - GS_{ws}$) in the one-shot paradigm. We additionally report the pooled standard deviation, the p-value, as well as the effect size d. **For clarity purposes, regrets are multiplied by 100**. We test for the statistical significance of the difference using an independent t-test and report the resulting p-values. Results highlighted in blue correspond to settings where the considered method performed significantly worse than the random search baseline ($p < 0.05$), whereas results marked in red highlight settings in which the considered method performed significantly better than random search baseline ($p < 0.05$).

of a single WS-picked architecture), WS-based approaches seem to perform better on average than the baseline RS and GS but can be quite unreliable.



(a) $\mathcal{A}_4$ (b) $\mathcal{A}_3$ (c) $\mathcal{A}_2$

(d) $\mathcal{A}_1$ (e) $\mathcal{A}_{\text{FULL}}$ (f) $\mathcal{A}^{\text{RES}}_{\text{FULL}}$

(g) $\mathcal{A}^{\text{NRES}}_{\text{FULL}}$

Figure 4.2: We report for a few search space the test regret as a function of time for the different NAS algorithms considered. Curves are averaged over 100 runs for RS_{bs} and GS_{bs}, and 25 runs for RS_{ws} and GS_{ws}. Visible colored areas correspond to the 95% confidence interval for the estimation of the average. Notice that both axes use a logarithmic scale.

We report in Table 4.4 the average regret difference between the baseline random search and the weight-sharing based random search ($RS_{bs} - RS_{ws}$), and between the baseline random search and the weight-sharing based greedy search ($RS_{bs} - GS_{ws}$), in the one-shot setting. Numerical and statistical results coincide with the visual results of Figure 4.2. Similarly, we report in Table 4.5 the average regret difference between the baseline

	$GS_{bs} - RS_{ws}$	$GS_{bs} - GS_{ws}$
$\mathcal{A}_4$	0.30 ± 1.21 (p=0.27, d=+0.25)	0.19 ± 1.25 (p=0.51, d=+0.15)
$\mathcal{A}_3$	0.72 ± 1.45 (p=0.03, d=+0.50)	0.28 ± 1.50 (p=0.41, d=+0.18)
$\mathcal{A}_2$	1.67 ± 2.92 (p=0.01, d=+0.57)	1.86 ± 2.92 (p=0.01, d=+0.64)
$\mathcal{A}_1$	2.46 ± 1.85 (p=0.00, d=+1.33)	2.50 ± 1.84 (p=0.00, d=+1.36)
$\mathcal{A}_{\text{FULL}}$	1.42 ± 3.03 (p=0.04, d=+0.47)	1.67 ± 2.95 (p=0.01, d=+0.57)
$\mathcal{A}_{\text{FULL}}^{\text{NRES}}$	1.48 ± 1.76 (p=0.00, d=+0.84)	1.14 ± 1.85 (p=0.01, d=+0.62)
$\mathcal{A}_{\text{FULL}}^{\text{RES}}$	0.78 ± 0.87 (p=0.00, d=+0.90)	0.43 ± 0.93 (p=0.04, d=+0.47)

Table 4.5: We report the average regret differences between the baseline greedy search and the WS-based random search ($GS_{bs} - RS_{ws}$), and between the baseline greedy search and the WS-based greedy search ($GS_{bs} - GS_{ws}$) in the one-shot paradigm. We additionally report the pooled standard deviation, the p-value, as well as the effect size d. **For clarity purposes, regrets are multiplied by 100**. We test for the statistical significance of the difference using an independent t-test and report the resulting p-values. Results highlighted in blue correspond to settings where the considered method performed significantly worse than the greedy search baseline ($p < 0.05$), whereas results marked in red highlight settings in which the considered method performed significantly better than the greedy search baseline ($p < 0.05$).

greedy search and the weight-sharing based random search ($GS_{bs} - RS_{ws}$), and between the baseline greedy search and the weight-sharing based greedy search ($GS_{bs} - GS_{ws}$).

Combining WS and random search results in a significant improvement over the baseline random search (left column of Table 4.4) on $\mathcal{A}_1$ ($d = +1.09, p = 0.00$), $\mathcal{A}_{\text{FULL}}^{\text{NRES}}$ ($d = +0.48, p = 0.03$) and $\mathcal{A}_{\text{FULL}}^{\text{RES}}$ ($d = +0.63, p = 0.01$), but grants significant worse results on $\mathcal{A}_4$ ($d = -0.85, p = 0.00$). Results on $\mathcal{A}_3$ ($d = -0.26, p = 0.25$), $\mathcal{A}_2$ ($d = +0.42, p = 0.07$) and $\mathcal{A}_{\text{FULL}}$ ($+0.44, p = 0.06$), are below our effect size threshold and non-significant. Unsurprisingly, the baseline greedy search performs worse than the baseline random search in the one-shot setting, as can be deduced from the left column of Table 4.5, where GS is outperformed on all but one search space. This is to be expected from an algorithm without any direct exploration of the search space. Results in Table 4.4 moreover suggest that combining WS with GS rather than RS results in unexpected behaviors, as it can either improve ($\mathcal{A}_2$, $\mathcal{A}_1$, $\mathcal{A}_{\text{FULL}}$), or be detrimental to ($\mathcal{A}_4$, $\mathcal{A}_3$, $\mathcal{A}_{\text{FULL}}^{\text{NRES}}$, $\mathcal{A}_{\text{FULL}}^{\text{RES}}$) performances depending on the search space.

All in all, results suggest that in a one-shot setting, WS can improve the performance of RS but that its efficiency is inconsistent and on average relatively small. To put the different reported regrets in context, one can consider that with an effect size $d = +0.44$ for the WS based RS over baseline RS on $\mathcal{A}_{\text{FULL}}$, the probability that a random run with WS produces

a smaller regret than the baseline given the same time budget is only around 61% (Magnusson, 2020). On $\mathcal{A}_1$, where WS is somehow very effective and produces a large effect size of $d = +1.09$, this probability reaches a maximum of 78%. On the contrary, on $\mathcal{A}_4$, where WS is least effective, this probability can get as low as 7%. As it has been noted several times in the literature Li et al., 2019; A. Yang et al., 2020; K. Yu et al., 2020b, reporting the results over several runs is thus crucial to NAS research, especially when considering moderate or small effect sizes.

Interestingly, Table 4.4 reveals no clear link between the average level of correlation reached by WS on a search space and its ability to outperform a baseline RS in a one-shot setting: on $\mathcal{A}^{\text{NRES}}_{\text{FULL}}$, where correlations in Table 4.3 are the lowest, WS significantly outperforms RS, whereas it offers terrible results on $\mathcal{A}_4$ despite significantly better correlations. WS offers similar correlations on $\mathcal{A}_2$ and $\mathcal{A}_3$, but the WS-guided greedy search respectively gives smaller and larger regrets than the baseline random search. On $\mathcal{A}^{\text{RES}}_{\text{FULL}}$ and $\mathcal{A}_1$, where WS offers the best correlations, the WS-guided greedy search is respectively equivalent to RS, and much better than RS.

Under the time constraints of one-shot NAS, WS can slightly outperform a baseline RS, although rarely to a significant extent, and can even be worse. Besides, there seems to be no obvious relationship between the level of correlation between proxy and standalone evaluations, and the performances of WS on a search space.

4.4.3 Variations between Search Spaces

From Section 4.4.2, WS-guided NAS seems to often slightly outperform RS, but this depends on the search space.

Coincidentally, we notice from the results of Section 4.4.1 that simply changing the number of nodes connected to the output makes the average correlation vary between 0.59 on $\mathcal{A}_3$ and 0.68 on $\mathcal{A}_1$. Additionally, restricting the search space to architectures presenting a residual connection has a noticeable positive effect on the correlations, as they increase from 0.46 to 0.71 between $\mathcal{A}^{\text{NRES}}_{\text{FULL}}$ and $\mathcal{A}^{\text{RES}}_{\text{FULL}}$. The search space itself has an important impact on the correlations, even more so when using the training enhancements described in Section 4.2.2.

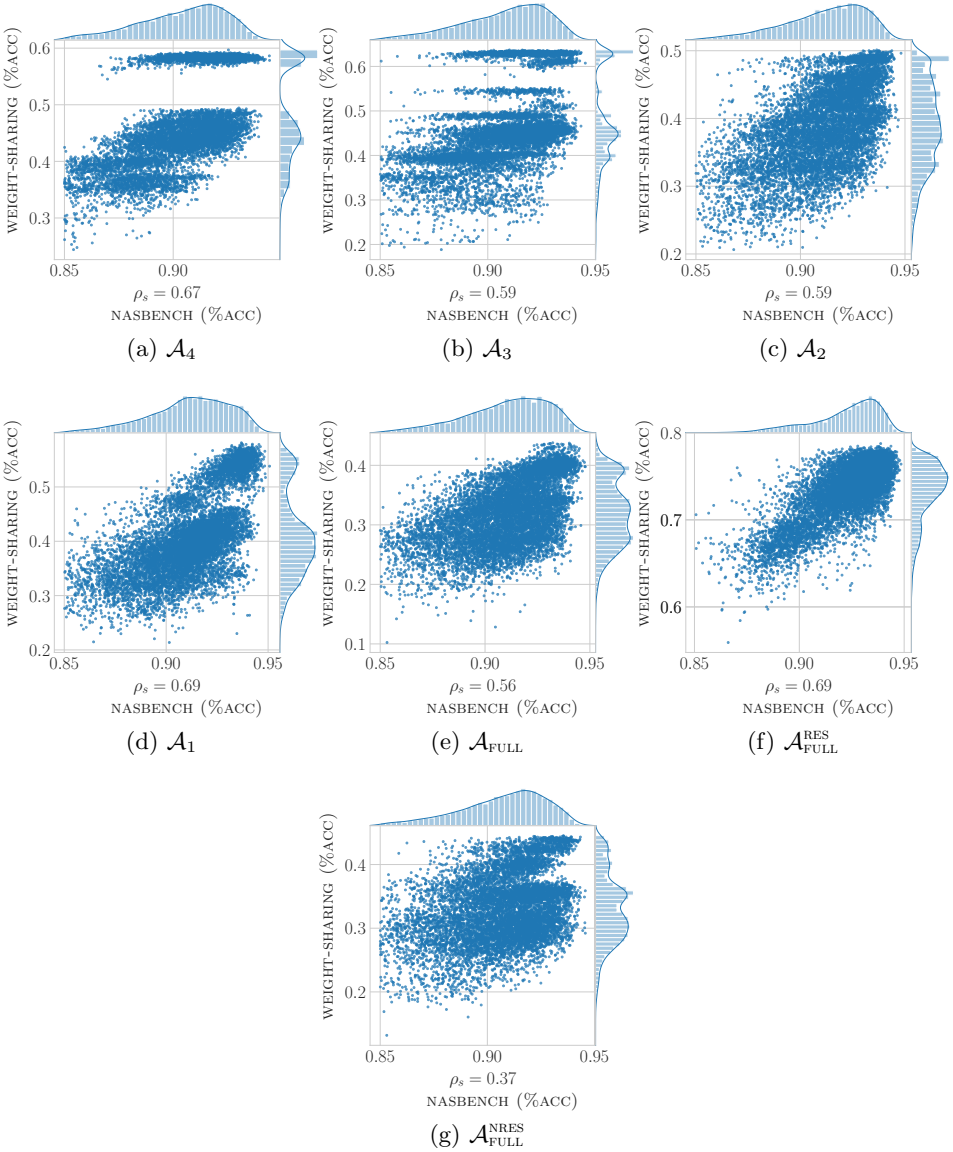
The size of the datasets could explain the varying correlations. It has often been asserted in the literature that, the more architectures there are in the search space, the harder it is to train the super-net. The Spearman rank's correlation between the average correlation obtained with batch-size fine-tuning (BNS-FT) reported in Table 4.3 and the sizes of the dataset reaches -0.71 ($p = 0.07$). The effect hints that larger search-spaces could possibly lead to smaller correlations between proxy and standalone evaluations, but

the relatively low number of search spaces of this study prevents us from positively rejecting the null hypothesis that it does not with great confidence, and further studies are required to conclude on this matter. Besides, results in Section 4.4.1 suggest that it is probably not the only aspect of the search space that is of influence. On $\mathcal{A}_2$ and $\mathcal{A}_3$, WS offers roughly the same level of correlation, despite $\mathcal{A}_2$ being twice larger than $\mathcal{A}_3$. The correlation achieved is 25% smaller in $\mathcal{A}_{\text{FULL}}^{\text{NRES}}$ than in $\mathcal{A}_{\text{FULL}}$, with 23% less architectures. It is also interesting to note that few architectures are actually seen during training: given 432 training epochs of 157 mini-batches of data, less than $67,824$ unique architectures are used to update the super-net. This might be enough to cover $\mathcal{A}_4$ or $\mathcal{A}_{\text{FULL}}^{\text{RES}}$, but represents only a tiny fraction of larger datasets, such as $\mathcal{A}_2$ ($\simeq 200,000$ architectures), or $\mathcal{A}_{\text{FULL}}$ ($\simeq 400,000$ architectures). Further studies are required to clearly establish whether the size of the dataset has a non-negligible impact on the correlation capabilities of WS, but several facts suggest that it cannot entirely explain the discrepancies between the different search spaces.

We report in Figure 4.3 and for each search space a scatter plot between the true validation accuracies and the proxy accuracies resulting from training a super-net. Interestingly, several visible clusters seem to be linked to proxy evaluations. For each scatter-plot, we report the distributions of the true validation and proxy accuracies over sampled architectures. Coincidentally with the different visible architecture clusters, distributions of proxy evaluations are much less regular than their true validation counterparts, often presenting several modes. The clusters of architectures in the scatter-plots visually transcribe existing biases in proxy evaluations.

There is no trivial relation between different biases and particular structural properties of the architectures. Fortunately, some biases are easier to highlight than others. We focus on two such biases in Figure 4.4. On $\mathcal{A}_1$, architectures with a residual connection tend to get better evaluations than those without. On $\mathcal{A}_4$, the presence of a 3×3 convolution on the first node triggers over-evaluation. Such clusters can be seen in the scatter plots of all search spaces except $\mathcal{A}_{\text{FULL}}^{\text{RES}}$. Different search spaces bias the super-nets in different ways, resulting in different structural patterns of over/under-evaluations.

The patterns appearing in the scatter-plots may explain the search results of Section 4.4.2 better than the correlations level reached by WS. On $\mathcal{A}_4$, the over-evaluation bias visible in Figure 4.4 creates a cluster of architectures with excellent proxy accuracies. As a result, in a one-shot paradigm, WS neglects a large number of architectures with equal or better capabilities that random search does not miss. Although the cluster contains a few of the best architectures, its average standalone accuracy is particularly poor. This impedes WS from selecting top models, and makes the early WS-



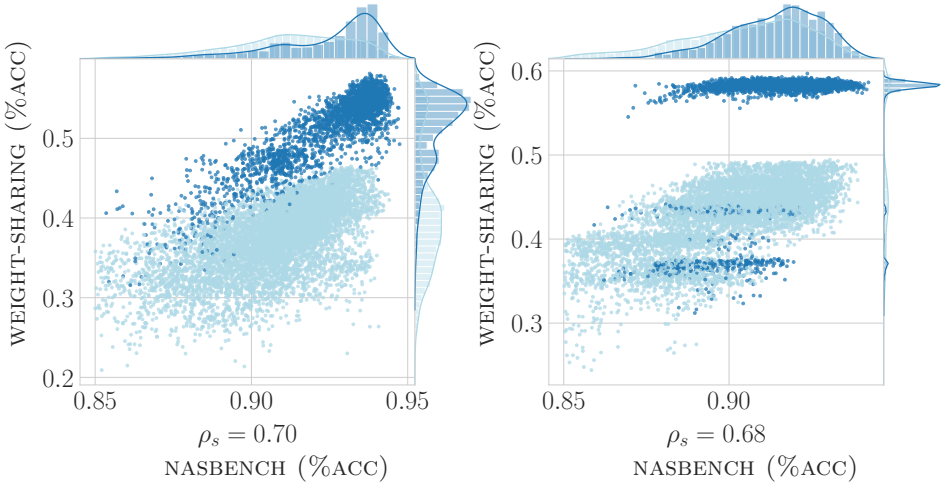
(a) $\mathcal{A}_4$ (b) $\mathcal{A}_3$ (c) $\mathcal{A}_2$

(d) $\mathcal{A}_1$ (e) $\mathcal{A}_{\text{FULL}}$ (f) $\mathcal{A}_{\text{FULL}}^{\text{RES}}$

(g) $\mathcal{A}_{\text{FULL}}^{\text{NRES}}$

Figure 4.3: We report, for a few search spaces, a scatter plot of the proxy accuracy computed using a super-net BNS-FT (y-axis), and the average validation accuracy returned by NB-101(x-axis) for 10,000 architectures.

guided search worse than RS on this particular search space. On $\mathcal{A}_1$, the over-evaluation bias towards residual connections benefits to the search, as architectures with residual connections are better on average and constitute most of the best architectures of the search space. The WS-guided search is in turn quite efficient.



(a) $\mathcal{A}_1$ (b) $\mathcal{A}_4$

Figure 4.4: We report for a super-net trained on $\mathcal{A}_1$ (left) and $\mathcal{A}_4$ (right) the proxy accuracy computed after fine-tuning the batch-norm statistics (y-axis), and the average validation accuracy returned by NB-101 (x-axis) for $1,000$ architectures. We highlight in a darker tone the points corresponding to architectures with residual connections (left) and architectures with a 3×3 convolution on the first node (right). Both examples reveal a clear bias in super-net evaluations. We also report the distributions of the proxy and standalone accuracies of the sampled architectures for the complete population and for the sub-population of interest.

The patterns of over/under-evaluations dictate the search behavior when exploiting WS. If WS is biased towards interesting patterns in the considered search space, then it is likely to perform much better than random search. Otherwise, the difference may not be significant. In the worst scenario, the bias can even be strong enough to undermine the performances of WS.

Chapter 5

Neural Architecture Search for Fracture Classification

Contents

In this chapter, we try to assess the efficacy of tailoring DCNN architectures to traumatic radiographs. As an alternative to the fracture detection problem, we start by introducing a fracture patch classification task on which NAS is tractable. Then, after reaffirming the importance of using transfer learning from natural images, and emphasizing the apparent incompatibility between NAS and pre-training, we introduce an efficient scheme based on weight sharing that makes it possible to exploit both conjointly. Using a plain genetic algorithm and a search space of efficient architectures, we show that it is possible to find architectures that are better suited to our problem than their counterparts selected on ImageNet, both in terms of diagnostic performance, and inference-time computational efficacy.

5.1 Introduction

We have seen in Chapter 2 that the high prevalence of traumatic skeletal injuries and the shortage of radiologists called for the automation of the fracture detection task. We additionally presented several deep learning systems which are capable of increasing the fracture detection diagnostic performance of radiologists, and can even outperform medical experts on their own. Such solutions notably include the work of Lindsey et al., 2018 and the approach developed by Gleamer (Duron et al., 2021; Guermazi et al., 2021).

Those algorithms were built using long established CNN backbones originally designed and tuned to perform well on natural images datasets. Reusing such off-the-shelf models allows authors to rely on transfer learning through fine-tuning, a feature that is key to improve performance when data sources are relatively scarce (Kornblith et al., 2019). Pre-training is however a double-edged sword, as fine-tuning models means making use of architectures which design might be sub-optimal for the task at hands. In opposition, tailoring model architectures to radiographs using the NAS paradigm which we thoroughly described in Chapter 3 is a promising lead for improved performance.

Unfortunately, classical NAS approaches are hardly compatible with pre-training, as they involve the evaluation of many different architectures for which no set of pre-trained weights might be available in advance. Since in practice we find that pre-training models grants a greater performance improvement than manually selecting better architectures, NAS appears to be meaningless in settings where one can only perform one or the other. A naive approach to bridge the gap between the two paradigms would be to train each architecture considered by the NAS algorithm twice: once as pre-training on a dataset of natural images, and a second time for the fine-tuning on the task of interest. But it transpires quite clearly that this scheme would further increase the computational burden of an approach which is already remarkably resource-inefficient.

In this chapter, we first introduce the task of fracture patch classification as a cheaper alternative to the fracture detection task. Using this new problem, we reaffirm the importance of transfer learning. We then introduce a procedure which uses a supernet trained with WS as a source of pre-trained weights for an entire search space and thus reconciles classical NAS approaches with transfer learning. By exploiting our approach on a search space of efficient architectures, we challenge the efficacy of the architecture fitting paradigm, and demonstrate using a plain genetic algorithm that it is possible to create architectures better-suited to radiographs both in terms of clinical performance, and computational efficiency.

5.2 Related Work

As presented in Chapter 2, several deep learning based solutions to the fracture detection problem have been studied in the literature. However, no AI-technology breakthrough has clearly emerged so far. All the authors *de facto* reuse an off-the-shelf architecture designed and pre-trained on datasets of natural images, which they fine-tune on private annotated datasets of fractures. In particular, this work is based on the detection model of Gleamer Duron et al., 2021; Guermazi et al., 2021, which we also presented in Chapter 2. This model was obtained by fine-tuning a Mask-RCNN (He et al., 2017) pre-trained on COCO on a private internal dataset of 60,000 radiographs of patients with trauma gathered from 22 institutions and annotated by medical experts.

Adjusting the design of a DCNN to a particular problem is a time-consuming task, which can be automated using NAS, as previously described in Chapter 3. Several approaches to NAS have been studied in the literature, from using deep reinforcement learning (RL) Zoph et al., 2017a; Zoph et al., 2018b, evolutionary algorithms Real et al., 2017; Real et al., 2019a or even fully differentiable approaches H. Liu et al., 2019a. However the inherent training cost of a single deep learning architecture is a strong limitation of this paradigm, and early NAS approaches Real et al., 2019a; Zoph et al., 2018b; Zoph et al., 2017a suffered from impractical computational costs. A plethora of papers relying on Weight-Sharing (WS), a computational trick allowing to train all the architectures of a search space at once have since emerged. This WS mechanism is however poorly understood, and we have shown in Chapter 4 that its capabilities are quite limited. In this paper, rather than directly conducting NAS using WS, we propose to exploit a super-net trained on natural images as a generator of pre-trained weights used to fine-tune architectures on our fracture classification task.

5.3 Methods

In this section, we first introduce our dataset of fracture patches. We then describe our scheme to combine NAS and transfer learning from natural images. From there, we introduce the search space of the Once For All framework (Cai et al., 2020), and the genetic algorithm with which we perform NAS. Finally, we describe the individual training setup of architectures.

5.3.1 Fracture Patches Dataset

Medical object detection models often take a fairly long time to train, on account of the high resolution images required to detect the complex and

wide variety of patterns that they need to work with. In turn, rather than performing NAS on the fracture detection problem, we propose to consider the easier task of predicting whether an image patch contains a fracture or not. In order to create a diverse yet realistic set of patches, we rely on the detection model of Gleamer. During inference, the detection model produces bounding boxes of interest, each coming with an associated score, ranging from 0 to 100, expressing the confidence of the network in the presence of a fracture within the suggested region. We performed a forward pass on the training images of the private internal dataset of Gleamer, and kept the predicted bounding boxes with a confidence above 10. This procedure generated 500,000 patches, which we split into a training, a validation and a test dataset, respectively containing 60%, 20% and 20% of the samples. Extracting patches solely from the detection training dataset is necessary for fair model comparisons as it ensures that the distribution of prediction localizations is the same during training, validation and test of the classification task. Predicted bounding boxes that had an intersection over union (IoU) with a ground truth bounding box above 0.5 were considered positives. Samples with an IoU below 0.2 were deemed negatives. During training, boxes with intermediate IoUs between 0.2 and 0.5 were ignored so as not to confuse the classification networks, but were considered negatives during testing.

5.3.2 ImageNet pre-training

Fine-tuning architectures pre-trained on large image classification datasets such as ImageNet is a well known strong baseline when working with scarce data sources (Kornblith et al., 2019). Although the extent to which this transfer learning is beneficial ultimately depends on the task at hands (Kornblith et al., 2019), we found that leveraging ImageNet pre-training was essential for our fracture classification task. As will transpire clearly in Section 5.4, the performance increase resulting from ImageNet pre-training is substantially greater than what is usually gained by optimizing an architecture to a problem. Nonetheless, NAS algorithms commonly train architectures from scratch. This is reasonable, as the ImageNet dataset itself is used to evaluate NAS procedures. Besides, naively incorporating this pre-training further increases computational expenses, as each architecture needs to be first pre-trained, then fine-tuned on the task of interest. Instead of pre-training each individual architecture separately, we propose to exploit a pre-trained super-net. Indeed, by design, the weights of all associated architectures can be extracted from the sole training of this super-net. Although the individual performance obtained with shared pre-trained weights are likely worse than those of individual training, they provide a cheap and efficient proxy.

5.3.3 Search Space

Training super-nets is a long and minute process which requires very specific tricks such as the in-place distillation of bigger sub-networks into smaller ones, specific initialization schemes, or the progressive shrinking of the sub-networks considered (Cai et al., 2020; J. Yu et al., 2020). To overcome those difficulties, we rely on publicly available assets. Few of the approaches described in the literature provide the code used for their development. The sole work additionally providing the weights of their trained super-net that we could find was the Once For All (OFA) framework of Cai et al., 2020. The goal of the OFA approach is to quickly find neural architectures adapted to specific inference settings, a task in which WS particularly shines given its ability to very quickly evaluate architectures of interest. Each model of their search space can be divided into five stacks of several convolutional layers. The number of convolution layers in each stack varies between 2, 3 and 4. Individual layers are based on inverted residual blocks Sandler et al., 2018, and each layer has an expansion ratio chosen between 3, 4 and 6, and a kernel size, chosen between 3, 5 and 7. In total, an architecture is described by 45 parameters, each taking 3 possible values. For deeper understanding of the ins and outs of OFA, we refer the reader to the original paper.

5.3.4 NAS Configuration

There is evidence in the literature that local search based NAS algorithms perform quite well (Den Ottelander et al., 2021; White et al., 2020). Using a genetic algorithm to optimize the architecture follows naturally, especially considering the original results of Real et al., 2017; Real et al., 2019a. To perform the optimization, we opt for a plain $(1 + 1)$ evolution strategy which benefits from straight-forward parallelization (Rechenberg, 1978). At any time, to suggest an architecture, the algorithm considers the best model found so far, and creates a child candidate by applying random mutations. During this mutation process, each of the 45 parameters of the current best model is modified with probability p, which varies with the considered heuristic. Typically, p is set to the inverse of the dimension of the problem, such that on average a single parameter is modified. To further accelerate the process, we bootstrap the search from the best model found on ImageNet by the OFA authors, and set it as the first best architecture. Still, to prevent the optimization process from getting stuck around this model, we mimic brain storm optimization (Y. Shi, 2011), by decaying the mutation probability from 1 to $\frac{1}{45}$ over the course of the optimization. Since we indirectly encourage the NAS approach to search for architectures in the neighbourhood of the best OFA model, we do not impose strong computational constraints on the found architectures, and optimize solely for the performance metric described in the next section. The optimization is

performed using the Nevergrad framework (Rapin et al., 2018).

5.3.5 Architecture Training and Evaluation

To reduce the computational cost of search, we follow the path of Tan et al., 2019a; Tan et al., 2019b and evaluate models after five training epochs. We optimize architectures with standard stochastic gradient descent, using an initial learning rate of 0.02 decayed to 0 over the course of the five epochs using a cosine annealing scheme, a momentum of 0.9, batch-size 128 and a weight-decay of 10^{-4}. We additionally use exponential moving average of the weights, with a decay of 0.99. To evaluate the obtained models on the validation dataset, we report the performance of the underlying detection model obtained when replacing the scores of the original detection model with those of the newly trained model. Each patch is associated to the fracture score returned by the classification model, and we compute the area under the FROC curve (AUFROC) (Bandos et al., 2009) as the metric of interest. Images of our datasets are grouped in studies, with each study corresponding to a unique patient examination. The FROC curve is obtained by performing a study-wise average of the recall, as a function of the average number of false positives. A fracture is considered detected if it is accurately pinpointed on at least one of the images of the considered study. The AUFROC is the area under the resulting curve. We report the reader to the analysis of the fracture detection metrics which we performed in Chapter 2 for further details. To avoid having to work with varying intervals of definition, we choose to integrate the AUFROC between 0.02 and 0.5 false positives per image, as those two points cover most of the operating points of radiologists (Duron et al., 2021). During the NAS run, we evaluate 100 architectures. Each individual training is performed on a cluster of four K80 GPUs, with 10 cluster running in parallel at all times.

5.4 Results

We report the evolution of the validation scores of the models selected by the NAS algorithm in Figure 5.1. The total computing time amounts to around 450 hours. The influence of the brain storm optimization process presented in Section 5.3.4 can be deducted from the performance evolution. The top left point, which corresponds to the best ImageNet model found by the OFA authors, provides a strong anchor point to the search. In the beginning of the run, the high mutation rate results in a performance drop, but several candidates of interest are found later on.

Figure 5.2 provides a scatter plot of the number of trainable parameters of the architectures encountered over the run, against their validation score. Interestingly, no architecture was sampled with more parameters than the

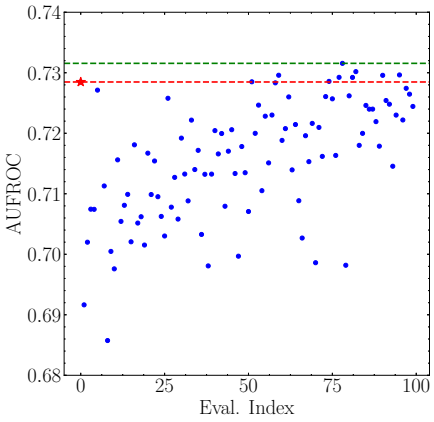


Figure 5.1: Evolution of validation performance over the search. The red point marked by a star on the top left corresponds to the best architecture found on ImageNet by Cai et al., 2020. The red (resp. green) horizontal line indicates the corresponding score, (resp. the best found validation metric).

reference OFA model. Among the sampled parameters, the first five act most on the number of parameters, as they dictate the number of convolutions appearing in each of the five blocks constituting the model. For the original OFA architecture, four out of those five parameters were set to their maximum value. This makes it hard to sample an architecture with more parameters in the early part of the brainstorm process where search is close to random. In the middle of the search, the algorithm could find an architecture with less convolution layers that performed better. This new architecture became the reference around which the next architectures were more closely sampled as the brainstorming faded away, which made sampling architectures with less parameters easier.

From the architectures found by the NAS, we extract the three with the best validation metrics and re-train them with the setting described in Section 5.3.5, but increasing the number of epochs to 20. Out of the three, our final architecture is the one with the best validation performance after this longer training. We refer to it as FractNet. We also perform this re-training for the best ImageNet OFA model. Additionally, we train with the same setting ResNet-152 and DenseNet-161 baselines, and models from the EfficientNet family (Tan et al., 2019b), which are state-of-the-art lightweight models, obtained using NAS on the ImageNet dataset over a different search space than OFA. All of those additional networks were pre-trained on ImageNet. To illustrate the relevance of pre-training, we further gather the performances of the OFA model and our FractNet, when training from scratch under the same setting.

Table 5.1 reports the final test metrics, as well as the number of param-

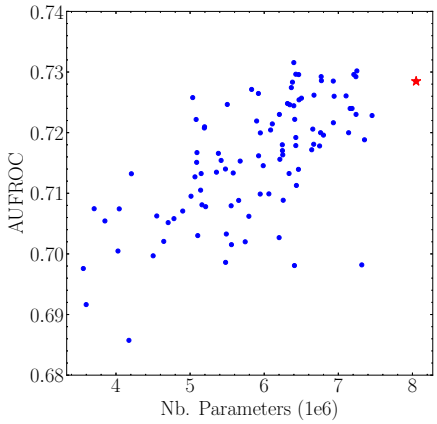


Figure 5.2: Scatter plot of the validation performance against the number of parameters of the architectures seen during NAS. The red star marks the best OFA ImageNet architecture.

eters and the number of multiply-adds for the processing of a single image for each architecture and training of interest. The results first show that transfer learning is key to the problem, and that performing NAS without pre-training would be pointless. Secondly, our final FractNet model performs slightly better than the best ImageNet-selected OFA network, whilst having around 18% less parameters. This indicates that the architecture tailoring process has potential for improving performances in computationally constrained settings. Thirdly, our FractNet model has about the same number of parameters as an EfficientNet-b1, whilst performing slightly better. This strengthens our stand that task-specific architectures can improve performances. Finally, we see that our FractNet model reaches promising results compare to much larger models such as ResNet-151 or DenseNet-161, which require 20 and 13 times as many multiply-adds operations per image processed respectively.

5.5 Conclusion

In this chapter, we have considered tuning DCNN architectures to the problem of fracture patch classification. We have shown that exploiting transfer learning was key to alleviate data sparsity and drastically reduces computing times. To perform NAS without losing the benefits of pre-training, we have proposed to exploit a super-net trained on ImageNet as a generator of pre-trained weights used to fine-tune architectures on our task. From there, we introduced a plain genetic NAS algorithm and performed NAS on the search space of computationally efficient architectures introduced by the authors of the OFA framework. With this approach, we have created the

Network	AUFROC	#Params	#MAdds
DenseNet-161	0.773	26.3M	7.7B
ResNet-152	0.768	58.2M	11.7B
EfficientNet-b0	0.740	4.0M	0.38B
EfficientNet-b1	0.748	6.5M	0.57B
OFA	0.746	7.8M	0.61B
FractNet	0.754	6.4M	0.56B
OFA •	0.676	7.8M	0.61B
FractNet •	0.674	6.4M	0.56B

Table 5.1: For each model of interest, we report the test AUFROC, the number of trainable parameters (#Params), and the number of multipy-adds operations for the inference of a single image (#MAdds). OFA refers to the best model found on ImageNet by the authors of the OFA framework (Cai et al., 2020). FractNet refers to the model selected using our NAS approach. Models marked with a black bullet were trained without ImageNet pre-training.

FractNet model, which obtains better performances on the fracture classification problem whilst reducing computational overheads. This validates both the relevance of the architecture tailoring process, as well as our introduced super-net pre-training protocol. As further work, one could try to replicate this scheme on other search spaces, such as the EfficientNet search space. It would also be interesting to explore whether the performance gap with respect to much larger architectures could be reduced through techniques such as model distillation, or by slightly increasing model capacity using the EfficientNet growing strategy. Still, it is worth noting that the increase in performance resulting from our scheme was only moderate, and was obtained at an important cost of around 450 GPU hours worth of computing.

Chapter 6

Estimating Bone Age With Deep Learning

Contents

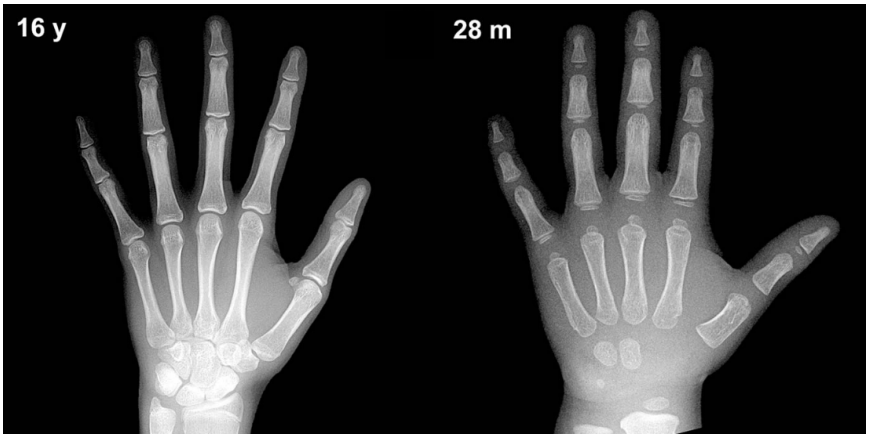
In this chapter, we introduce the bone age assessment (BAA) problem and the different methods used by radiologists to perform its diagnosis. We then carry out a brief review of recent deep learning solutions to BAA, with a particular focus on the results of the RSNA pediatric bone age challenge. To analyze the generalization capabilities of a BAA model trained on the RSNA dataset, we introduce a BAA clinical dataset of our own. Starting from the solution of the winners of the challenge, we modernize the training setup to slightly alleviate the computational resources, and reveal that the resulting model is biased towards certain age and sex categories. We propose a simple weighting mechanism to counteract this bias and display its efficiency on our internal dataset. Making use of another public dataset for which the chronological age of the patients were available, we demonstrate that incorporating this information as an input to the model decreases the global average error, but further biases the model, deteriorating its performance in the detection of pathological patients. We argue that this effect has been ignored in the literature, and show that by correcting the two biases at the same time using the weighting procedure, we are able to significantly improve the performance of our baseline model, which performs substantially better than a reference radiologist tasked to perform BAA on our clinical dataset.

6.1 Introduction

Bone age, as opposed to calendar or chronological age, is a constructed estimation of a patient's skeletal maturity. When performing bone age assessment (BAA), radiologists determine a bone age value by referring to one of several standard BAA methods. The main indications for BAA are the detection of early or late puberty, the detection and monitoring of endocrine disorders and the assessment of adult height (Satoh, 2015). Another controversial indication of BAA is to determine whether juvenile migrants are of legal age (Schumacher et al., 2018).

Out of all BAA methods, the two most widely used are the Greulich and Pyle (G&P) method (Greulich et al., 1959) and the Tanner-Whitehouse (TW) method (James Mourilyan Tanner, 1983). Both rely on a radiograph of the patient's left hand. The predominance of hand radiographs can be explained by several factors. First, the hand is comprised of 27 distinct bones (8 carpal bones and 19 short bones forming the fingers), which can all be clearly observed from a single X-ray. Secondly, hand radiographs are easier to perform, especially on younger patients. Finally, hand radiographs can be obtained with relatively low level of radiations (Mettler Jr et al., 2008). Although it is possible to assess bone age using both hands, it is customary to use the left one, as it is less likely to be hurt considering the natural prevalence of right-handed persons. Other notable anatomical regions explored to estimate bone age are the clavicle (Kreitner et al., 1998), the teeth (Willems, 2001), the iliac crest (Little et al., 1994) and the femoral head (Castriota-Scanderbeg et al., 1995). One significant feature of almost all BAA methods is that male and female patients are diagnosed separately, due to difference in growth patterns.

The G&P method is by far the most prevalent BAA method in practical settings. It consists in comparing a hand radiography of a patient with an atlas of reference cliches. This atlas was collected between 1931 and 1942 from upper middle class Caucasian children living in Cleveland, Ohio, United States. Cliches were separated by sex, and each sex atlas had its patients split in bone age bins of various sizes. We provide examples of such reference images in Figure 6.1. When performing the exam, the radiologist matches a patient radiograph with its closest image from the atlas, and reports the bone age provided by the atlas. The main reasons why this approach has become popular is because it is both easy to learn, and relatively fast compared to other methods. Still, there are some drawbacks to G&P. First, it is known that there is a non-negligible part of subjectivity in the result, and that the measure has a high intra-reader and inter-reader variability. This subjectiveness can be partially explained by the absence of standardization in the contribution of different bones to the final bone age score. Hand bones grow at different rates, and different bones can display



(a) F, 16Y (b) M, 3Y4M

Figure 6.1: Examples of reference images found in the Greulich and Pyle atlas. On the left, a hand radiograph of a female patient with a reference bone age of 16 years. On the right, a male patient with a reference bone age of 28 months. Images are from Gilsanz et al., 2005. Growth is almost complete on the left, whereas the hand on the right is missing most of the carpal bones, and the cartilage at the tip of the finger bones are not yet merged with the bones themselves.

growth characteristics of different bone ages. From there, some radiologists might give more importance to carpal bones or finger bones, resulting in different matches in the G&P atlas, and thus different bone ages. It is also important to note that the atlas itself was designed more than 60 years ago and with a very specific populations, indicating that it might be less relevant to use for current children, and children of a different ethnicity (Alshamrani et al., 2019).

The other common BAA method is the Tanner-Whitehouse (TW) method (James Mourilyan Tanner, 1983), and its updates. Unlike G&P, TW is based on the score of local regions. 20 anatomical regions, corresponding to different bones, are each assigned a maturity level which is translated into a numerical score. Local scores are then summed into a unique global score, which is mapped into the final bone age estimate, based on the sex of the patient. TW offers several advantages over G&P. First, the overall result is more objective and more reproducible than G&P, thanks to the more fine-grained assessment of the bone ages. Additionally, the final mapping between the global maturity score and the final bone age results can be fine-tuned to different populations. Despite TW being objectively more appropriate, it is much more time expensive, which severely limits its application in clinical settings.

In this chapter, we introduce a clinical validation dataset for the BAA task, and study whether a relevant bone age prediction tool can be developed using publicly available bone age assets. We show that solutions introduced in the literature contain hierarchical bias structures that impede their direct application, and introduce simple weighting mechanisms to counteract those effects.

6.2 Related Work

Bone age assessment methods, because of their important inter-reader and intra-reader variance, are well suited for automation. Potential benefits for radiologists are in the detection of minute growth differences, and the standardization of the bone age estimations. Several automated BAA tools were created in the early days of AI (Michael et al., 1989; J. Tanner et al., 1992), and an extensive survey of those methods can be found in the work of Mansourvar et al., 2013. One of their notable aspect is that they tend to focus on the TW paradigm, in which local features and scores are combined, rather than the holistic paradigm of G&P. More recently, Thodberg et al., 2008 introduced a ML-based solution to the BAA task. The architecture of their system, BoneXpert, combines three layers. The first layer's task is to reconstruct the borders of 15 hand bones, and predict whether their shape is valid. The second layer examines each bone independently, and assigns it a bone age score based on its shape, intensity and texture. In the final layer, the individual scores are aggregated to form a final prediction according to either the G&P or TW method. BoneXpert was trained using 1,559 images, and was validated on several clinical test datasets. We focus the rest of this literature review on approaches making use of DL technologies.

Spampinato et al., 2017 were the first to train several DCNNs to directly predict bone age from hand radiographs. For this purpose, the Digital Hand Atlas Database (DHAD) (Gertych et al., 2007), a public dataset of 1,400 hand X-rays equally representing four ethnic backgrounds was exploited. Each image of the DHAD had previously been examined by two paediatric radiologists who independently decided on a bone age using the G&P method. The average of their predictions constituted the final ground truth labels. The authors fine-tuned several architectures such as GoogLeNet (Szegedy et al., 2015), VGG (Simonyan et al., 2014) and OverFeat (Sermanet et al., 2013) on the dataset. Although fine-tuned networks performed great, authors additionally introduced their BoNet architecture, which was composed of pre-trained layers from an OverFeat network, with additional convolutional layers randomly initialized, and a hand localization layer (Jaderberg et al., 2015). Models were trained on crops of resized versions of the original image. The size of the crops varied from 224×224 to 299×299, depending on the utilized backbone. Crops were augmented using standard data

augmentation techniques, including horizontal flips, rotations and translations. Networks were optimised using SGD with momentum for 150 epochs to minimize the Mean Squarred Error (MSE) loss with the ground-truth bone ages. Since the DHAD does not propose a train/validation/test split, models were evaluated using 5-fold cross validation. The final BoNet system had a Mean Average Error (MAE) with the ground-truth annotations of 9.48 months.

To facilitate future researches in the BAA field, Larson et al., 2018, introduced a dataset of 14,035 left hand radiographs coming from two different institutions. Similarly to DHAD, X-rays were examined by pediatric radiologists and the average of their G&P predictions was used as ground-truth. The dataset was then split into a training, validation and test sub-dataset. The authors then fine-tuned a pre-trained ResNet-50 network, to predict a probability score for each sex and each month from 0 to 19 years, resulting in a total of 456 classes. Optimization was performed using the Adam algorithm (Kingma et al., 2014). Input images were resized from their original size to 256×256 and pre-processed with the CLAHE algorithm (Pizer et al., 1987), with different thresholds used to create fake color channels. Standard data augmentations were also performed. On the newly submitted test set, the authors report a MAE of 6 months, whilst on the DHAD test set, their model reached a Root Mean Square Error (RMSE) of 8.76 months. The MAE error on the DHAD test set is not reported.

The popularity of the BAA task greatly increased within the DL community with the organization of the Radiological Society of North America (RSNA) Pediatric Bone Age Machine Learning Challenge, by Halabi et al., 2019. Reusing the dataset introduced by Larson et al., 2018, which from now on we will refer to as the RSNA dataset, contestants were invited to try to predict the bone age associated to full hand radiographs. The authors briefly recap the different solutions proposed by the top five teams.

1. Fifth place, constestants, Chen et al., introduced a U-Net model (Ronneberger et al., 2015), trained to predict segmentation masks for the hands using 400 manually labeled images. New image channels were then built with the resulting masks. Several architectures including ResNet-50, Inception-v3 and Xception were optimized for 200 epochs using images resized to 299×299 and standard data augmentations. The gender information was incorporated to the network through an embedding layer. Using an ensemble of their best models the participants reached a MAE of 4.527 months.

2. The fourth place was obtained by the BoneXpert method of Thodberg et al., 2008 described above. The system reached a MAE of 4.505 months.

3. The third place was attributed to Kitamura et al., who introduced a new *Ice* module, inspired by the *Fire* module of the SqueezeNet architecture (Iandola et al., 2016). In the SqueezeNet architecture, the *Fire* module is used to squeeze and then expand the input features channel information using 1×1 convolutions. In contrast, the *Ice* module squeezes and expands the spatial information using transposed convolutions. As pre-processing, images were resized to 550×550 while keeping their original aspect ratio. The gender was appended to the features extracted by their backbones. Combining the *Ice* module with vanilla convolutional layers and averaging four models trained on different folds of the data, the entrants reached a final MAE of 4.382 months.

4. In second place, Pan et al. manually cropped the original images of the dataset to focus on the hands, and resized the result to 560×560 images, further pre-processing the result with CLAHE. Rather than training their models on the full images, Pan et al. proposed to regularly extract 49 patches of size 224×224 from the image and use those as to fine-tune a ResNet-50 architecture pre-trained on ImageNet. Separate models were trained for male and female patients. During inference, the model predicted bone ages for each of the 49 patches, and a selected percentile (on average the 50th percentile, i.e. the median) of the resulting outputs was chosen as the final predicted bone age. The ensemble of nine models allowed the authors to reach a final MAE of 4.350 months.

5. Bilbily et al. won the first place by training an Inception-v3 network from scratch using images resized to 500×500. The patient sex information was appended to the features extracted from the image. However, the authors argue that a single bit of information might be overlooked by the network when juxtaposed with the several thousand inputs of the backbone. In order to give more weight to the sex encoding, the original binary scalar was mapped to a 32-dimensional vector using a linear mapping. Unlike other competitors, the authors did not introduce any specific scheme, and the resulting approach is rather straightforward. Using an average of several models, the final reached MAE was 4.252 months.

Because of the limited statistical resolution offered by the 200 images of the RSNA test set, the organizers considered that all of those top 5 contestants had won the challenge. Still, we find that the work of Bilbily et al was the most interesting, not only because of its great performance, but also due to its straightforward design. In contrast, Chen et al. had to create segmentation masks, Thodberg et al. used a carefully tuned classical machine learning approach, Kitamura et al. introduced a completely new type of

convolutional operator, and Pan et al. processed image patches rather than full images. In the rest of this chapter, we thus consider the approach of Bilbily et al. as a reference baseline.

After this competition, a different BAA dataset was introduced by Escobar et al., 2019. The Radiological Hand Pose Estimation (RHPE) dataset consists of 6288 radiographs of both hands. Likewise, ground-truth bone ages were estimated as the average G&P diagnosis of two expert radiologists. The overall dataset was split in three, with 5,492 serving as train set, 716 as validation set, and 80 as test set. The distributions of male and female, and the overall distribution of bone ages were similar to those of the RSNA dataset. Additionally, the authors introduced supplementary annotations for both the RSNA and RHPE dataset. For each image, a cohort of annotators created a bounding box around the left hand present in the image, and marked the position of 17 keypoints on the left hand, positioned at locations central to BAA, including the carpal bones, the distal radius and ulna, and the proximal, middle, and distal phalanges of each finger. Using those annotations, Escobar et al., 2019 were able to consequently improve their BAA model. Starting from the approach of the winners of the RSNA challenge, the following modifications were introduced: first images were cropped around the hand bounding box before being resized, limiting the information loss induced by the down-sampling of the original X-rays. Secondly, an additional channel was artificially created for the images, by generating Gaussian distributions around the different keypoints, allowing networks to rapidly focus on regions of interest. To be able to use those complementary informations during validation and testing on new datasets, the authors additionally trained a detection model to output the location of the left hand within the image, a keypoint detection model to pinpoint the 17 anatomical regions of interest. They showed that both aspects of their approach improved performances. They report a final MAE of the RSNA test set of 4.14 months when training only on the RSNA train set, and 3.85 months when using the RHPE dataset conjointly. On the RHPE test set, final MAE were respectively 7.60 months and 6.86 months when respectively using only the RHPE dataset, and both datasets.

Even more recently, González et al., 2020 improved upon the work of Escobar et al., 2019 by introducing slight modifications to the overall network and adding the chronological age of patients as a supplementary input to their networks. The authors argue that similarly to the gender information, the chronological age of the patient is an important information to possess in clinical settings, as it facilitates diagnosis. Indeed, the default G&P process requires radiologists to extensively examine hand atlases. Chronological age can be used as a first bone age approximation in order to quickly target a few reference radiographs of interest. The authors proposed to include this age information in two ways: first, by directly appending the age to

the features extracted from the image by the backbone, and secondly, by training networks to predict, not directly the bone age of the patient, but rather the offset between its chronological and bone ages. Rather than concatenating fabricated high-dimension embeddings of the additional features as proposed by the winners of the RSNA challenge, the authors suggest to simply multiply the age and sex information scalars by a learnable parameter, in order to let the network learn the importance of those features through the magnitude of the learned coefficients. On the RHPE test set, the authors report a final MAE of 5.47 months. Due to the correlations between bone age and chronological age, one could expect that feeding the network with the chronological age could make the network prone to predict bone ages closer to the chronological age. However, the authors show that there is no correlation between the absolute difference between the true and predicted bone age, and the absolute difference between the true bone age and the chronological age of the patients.

6.3 Experiments

In this section, we describe the different approaches that we considered to build and improve our DL solution to the BAA problem. We first introduce the dataset that we created to evaluate the performance of our algorithms on a population of interest. Then, we describe our baseline training scheme, and how we managed to reach the performance level of the winners of the RSNA challenge, without performing model ensembling. From there, we reveal a bias in the distribution of the bone ages of the RSNA dataset, and introduce a procedure to limit its influence on our model. Afterwards, we train a model on the RHPE dataset using the baseline setup, highlighting the performance difference with the models trained using the RSNA dataset, and additionally train a model that used the chronological ages of the patients. Finally, we reveal that whilst using chronological age improves performance on a specific sub-group of the patients, it tends to make the model conservative in its predictions, by strongly anchoring them around the chronological age. We finally introduce and demonstrate the efficacy of another procedure that tackles this problem.

6.3.1 Clinical Validation Dataset

Creating a dataset of bone age annotations sufficiently large to train a DL model is a costly task which we could not afford. The introduction of the RSNA and RHPE benchmarks are thus of significant practical importance. However, given the high inter-reader variability of the G&P method, and the limits inherent to its holistic approach, it is critical to question the generalization capabilities of AI algorithms trained on such datasets to unseen populations. To appraise this capability, we gathered a clinical dataset of

206 bone age exams performed in several French centers. Patients were randomly selected in such a way that the distribution of chronological ages was uniform, between 5 and 17 years old, and were evenly divided between boys and girls. Exams of patients below 5 years old were discarded as BAA is of poor medical relevance for younger children. Patients above 17 were not considered on account of the ethical concerns raised in Section 6.1. Two pediatric radiologists were tasked to predict the bone age of the patients using the G&P method. The final bone age was considered as the average of the two predictions. An additional radiologist with limited experience in BAA was asked to perform the same task. The performance of this radiologist serves as a baseline comparison for the algorithms which we introduce in the sections hereafter. During their screenings, all radiologists had access to the sex and the chronological age of the patients. Exams were flagged as either healthy or pathological based on the difference between the chronological age, and the assessed bone age, as described in Greulich et al., 1959. Of the 206 exams, 56 were deemed pathological.

6.3.2 Setting up a Baseline

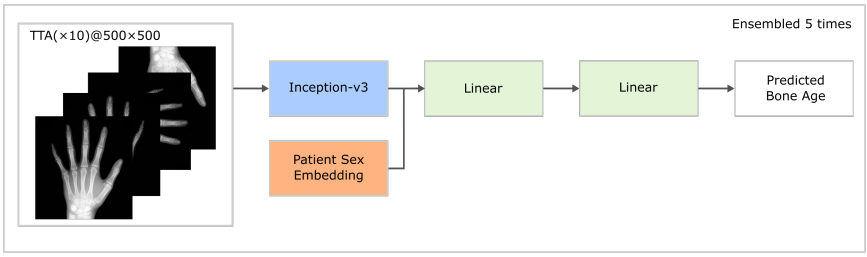
The winning entry of the RSNA challenge was quite compute-intensive for several reasons. First, Bilbily et al. propose to train their networks for 500 epochs, which we suspect is much higher than required. Secondly, the competitors reach their reported performance by employing an ensemble of five models. Thirdly, each image of interest was processed 10 times by each of the five networks using Test Time Augmentations (TTA). Network ensembling and TTA are often an optimal strategy in DL challenges for which no time limit is established for the processing of input images. In practical scenarios however, the memory required to store the parameters of the final models, and the time incurred by the inference of an input image are crucial resources.

The challenge took place in 2017. In the mere 5 years that separate their work from this study, the fast-evolving computer vision literature has tremendously grown. As such, several efficient DCNNs backbones, as well as training tricks and techniques have been introduced. We were therefore convinced that the original approach of Bilbily et al. could be improved without blows and whistles. Accordingly, our first objective was to bring up to date the training scheme of the authors and hopefully improve the overall efficacy of the DL pipeline by doing so. The result of this preliminary work was a baseline setup which we employed in all of our following experiments.

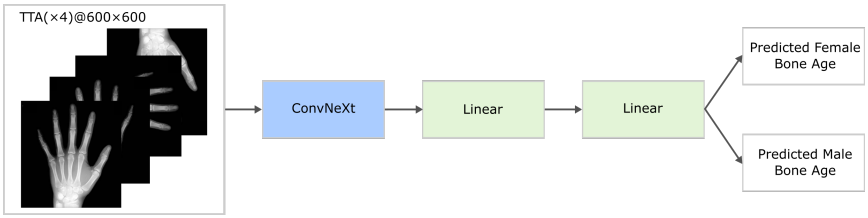
First and foremost, rather than the Inception-v3 backbone, we propose to make use of a recent family of models called ConvNeXt (Zhuang Liu et al., 2022). The ConvNeXt architecture was designed to mimic the macro

structure of the Swin image Transformer of Ze Liu et al., 2021, but replacing self-attention layers by convolutional layers with wide kernel sizes. A particularly relevant aspect of ConvNeXt models is that they do not make use of batch normalization layers (Ioffe et al., 2015) which typically require large batch sizes that are hard to attain when working with high-dimension inputs, as is the case for BAA. All of our models exploit a "Small" variant of ConvNeXt pre-trained on the ImageNet-22K dataset, which we fine-tune for 20 epochs rather than 500. We imitate the schedule of Zhuang Liu et al., 2022, and use the AdamW optimizer (Loshchilov et al., 2017), with a learning rate of 5×10^{-5} and a cosine-annealing schedule, a weight-decay of 10^{-8}, and a batch size of 16. Input images are resized to 600×600, keeping original aspect ratios. We employ exponential moving averaging of the model weights with a decay of 0.999, and perform standard augmentations including random horizontal and vertical flips, random 90 degrees rotations, random rotations, shift and scaling of the image, and random erasing (Zhun Zhong et al., 2020). To regularize the networks, we further apply path dropout (Huang et al., 2016), with probability 0.1. We reduce the original size of the two fully connected layers in the regression head from 1,000 to 256, but rather than embedding the binary sex information in a somewhat randomly sized space, we let our networks predict two bone ages, one for boys and one for girls, and simply select the appropriate output. Networks are optimized to minimize the L1 distance between their outputs and the normalized targets obtained by subtracting the mean and dividing by the standard deviation of all train labels. During inference, we make use of a single model but perform TTA, averaging the predictions obtained for the original image, the image flipped vertically, the image flipped horizontally, and the image flipped both ways. In Figure 6.2, we present an overview of the approach of Bilbily et al, and of our baseline model. Using this setup, we obtain a MAE of 4.26 months on the RSNA test set, which is slightly better than the replications results introduced by Escobar et al., 2019 of 4.45 months, and on par with the original results of 4.25 months.

In Table 6.1, we report the performance of the baseline, as well as the reference radiologist, on our internal dataset. For each entry, we list the MAE, and the sensitivity and specificity in the detection of pathological cases. Altogether, the baseline reaches a MAE of 7.44 months, a significant improvement over the radiologist (8.36 months). The sensitivity of the AI for the detection of pathological cases is 0.76, which is better than that of the radiologist (0.60), but the gain comes with an inferior specificity, with the baseline reaching 0.87, against an excellent specificity of 0.95 for the radiologist. In Table 6.2 and Table 6.3, we detail the MAE on the female and male populations respectively. We additionally provide an analysis on subgroups of patients of different ages by splitting the $[5, 17]$ chronological range in three bins of equivalent population: $[5, 9[$, $[9, 13[$ and $[13, 17]$. Results



(a) Bilbily et al.



(b) Ours

Figure 6.2: Overview of the different models introduced in this chapter: (a) the baseline approach of Bilbily et al, as presented in the RSNA challenge; (b) our baseline introduced in Section 6.3.2.

reveal that the AI is only significantly better than the radiologist on the girl population (5.92 months against 7.58 months). We further notice that on average, both the AI and the radiologist better assess the bone ages of girls than that of boys (5.92 months against 8.98 months for the AI, 7.58 months against 9.06 months for the radiologist). The performance is roughly even across all age groups for girls, although the performance of the AI seems to be relatively better on girls between 5 and 9 years old. For boys, the MAE of the AI is suspiciously smaller on boys from 13 to 17 years old, going as low as 4.47 months, whereas the average MAE across all other groups is roughly twice as high. The radiologist, on the other hand, performs slightly worse on younger patients for both populations, but has an otherwise approximately constant performance across all sub-groups of the same sex. This discrepancy suggests the existence of a bias in the design of our baseline algorithm.

6.3.3 Adjusting for Prevalence Bias

In the previous section, we evidenced that our baseline approach was significantly better on boys between 13 and 17 years old, possibly at the detriment of other age groups. Our first hypothesis is that this discrepancy can be explained by the prevalence of bone ages found in the RSNA dataset. Indeed, whilst we built our dataset to represent all ages evenly, no such concerns were considered for the RSNA datasets, and as such some age categories

	MAE	Sensitivity	Specificity
Radiologist	8.35[7.51,9.22]	0.60[0.48,0.70]	0.95[0.91,0.97]
BSL_{RSNA}	7.51[6.80,8.34]	0.74[0.62,0.82]	0.87[0.84,0.92]
$+\mathcal{W}$(BA)	**6.67[6.01,7.34]**	**0.86[0.78,0.94]**	0.85[0.83,0.92]
BSL_{RHPE}	6.59[6.02,7.21]	0.68[0.58,0.78]	0.90[0.85,0.93]
+CA	6.50[5.79,7.14]	0.40[0.29,0.51]	**0.99[0.97,1.00]**
+CA+$\mathcal{W}$(DA)	6.20[5.67,6.77]	0.60[0.48,0.70]	0.93[0.91,0.97]
+CA+$\mathcal{W}$(BA,DA)	**5.72[5.21,6.30]**	0.65[0.55,0.76]	0.93[0.88,0.96]

Table 6.1: For each entry, we report the MAE over all patients, as well as the sensitivity and the specificity in the detection of pathological cases. Each score is provided with a 90% confidence interval which was estimated using bootstrap. Radiologist is the performance of our reference radiologist. BSL_{RSNA} and BSL_{RHPE} correspond to models trained with the baseline setup on the RSNA and RHPE dataset respectively. +CA indicates that the model was traind with the chronological age as input. $+\mathcal{W}$(BA), $+\mathcal{W}$(DA) and $+\mathcal{W}$(BA,DA), respectively indicate that the model was trained, with the prevalence correction introduced in Section 6.3.3, with the pathological prevalence correction introduced in Section 6.3.5, and with both corrections. Results are highlighted if the corresponding score is better than the baseline of reference, and confidence intervals do not overlap.

were less represented. In Figure 6.3a, we display the distribution of bones ages for boys and girls on the RSNA train set. For boys, a notable bias exists in the bone ages distribution, which is heavily skewed towards patients around 13 years old. This bias could potentially lead to an over-evaluation bias for patients below 13 years old, and an under-evaluation bias for patients above 13 years old, as the model will on average try to center its predictions around 13 years old.

To compensate for the bias in bone ages, we propose to modify the training procedure by incorporating weights for each samples of the train set. Ideally, such weights would put more emphasis on poorly-represented samples. Suppose than the train bone age labels (y_i) are the outcome of a random variable which admits a density $f_{BA}(y)$. A natural candidate for the weighting scheme is to consider the inverse of the density $\omega(y) = \frac{1}{f_{BA}(y)}$. In practice, we estimate this density using the gaussian_kde implementation of the SciPy library (Virtanen et al., 2020). Since the loss associated to a ground-truth bone age y_i is multiplied by $\omega(y_i)$, on average the loss computed for a batch of samples is multiplied by $\frac{1}{N}\sum_i \omega(y_i)$, where N is the number of training sample, and in turn the overall learning rate seen by the optimization process is multiplied by the same amount. To avoid modifying the learning rate specifically for this new training, we additionally divide the weights by the average weights encountered in the training dataset, and

	F[5,9[(N=32)	F[9,13[(N=32)	F[13,17[(N=33)	F (N=97)
Radiologist	8.73[6.16,12.0]	6.44[4.59,8.53]	7.57[5.84,9.27]	7.58[6.23,8.84]
BSL_{RSNA}	4.64[3.62,5.69]	6.18[4.65,7.69]	6.92[5.11,8.74]	5.92[5.09,6.76]
$+\mathcal{W}$(BA)	4.37[3.42,5.45]	6.76[5.29,8.31]	6.09[4.57,7.58]	5.75[4.86,6.55]
BSL_{RHPE}	6.14[4.96,7.39]	5.19[4.15,6.45]	7.50[5.65,9.54]	6.29[5.43,7.20]
+CA	6.41[4.84,8.06]	5.25[4.09,6.46]	7.50[5.54,9.73]	6.40[5.50,7.37]
+CA+$\mathcal{W}$(DA)	5.90[4.69,7.18]	5.30[4.29,6.45]	6.77[5.05,8.87]	6.00[5.11,6.77]
+CA+$\mathcal{W}$(BA,DA)	6.02[4.68,7.33]	4.97[3.89,6.00]	**5.57[4.02,7.27]**	**5.52[4.71,6.29]**

Table 6.2: For each entry, we report the MAE across all girls (F), and across several girl sub-groups F[X,Y[, indicating that the age of the patient is comprised between X, included and Y, excluded. Each score is provided with a 90% confidence interval which was estimated using bootstrap. Details about the different labels on the left can be found in Table 6.1.

	M[5,9[(N=32)	M[9,13[(N=32)	M[13,17[(N=32)	M (N=96)
Radiologist	9.81[7.59,11.9]	8.75[6.75,11.0]	8.62[6.27,11.2]	9.06[7.78,10.4]
BSL_{RSNA}	12.0[10.0,14.1]	10.5[8.42,12.7]	4.47[3.34,5.75]	8.98[7.83,10.3]
$+\mathcal{W}$(BA)	**9.79[7.65,11.9]**	**9.06[7.22,10.8]**	4.65[3.67,5.69]	**7.83[6.82,8.79]**
BSL_{RHPE}	7.03[5.49,8.76]	6.85[5.25,8.71]	5.00[3.90,6.24]	6.29[5.41,7.19]
+CA	7.12[5.12,9.55]	**5.21[4.15,6.48]**	6.24[4.79,7.59]	6.19[5.26,7.32]
+CA+$\mathcal{W}$(DA)	6.49[4.92,8.26]	**5.30[4.10,6.41]**	6.01[4.80,7.45]	5.93[5.18,6.85]
+CA+$\mathcal{W}$(BA,DA)	7.07[5.42,8.93]	**5.48[4.53,6.52]**	5.00[3.90,6.26]	5.85[5.08,6.67]

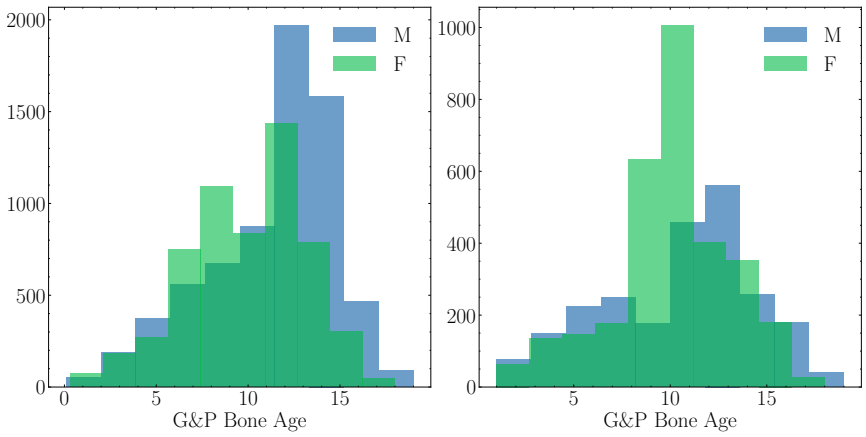
Table 6.3: For each entry, we report the MAE across all girls (M), and across several boy sub-groups M[X,Y[, indicating that the age of the patient is comprised between X, included and Y, excluded. Each score is provided with a 90% confidence interval which was estimated using bootstrap. Details about the different labels on the left can be found in Table 6.1.

consider instead weighting the samples by $\tilde{\omega}(y_i) = \frac{N\omega(y_i)}{\sum_i \omega(y_i)}$.

The performance of the resulting method is displayed next to the Baseline in Tables 6.1, 6.2 and 6.3. On girls of all ages, this new modification did not result in any significant performance difference with the baseline approach. On male patients, the procedure produced significant positive effects on the two subgroups of interests. For patients with chronological age between 5 and 9 years old, the MAE was reduced from 12.0 months to 9.79 months. For patients between 9 and 13 years old, the MAE was reduced from 10.5 months to 9.08 months. Still, the performance on the last category, including patients from 13 to 17 years old was still twice as good.

6.3.4 Exploiting the RHPE Dataset

Using our baseline training setup, we similarly train a model on the RHPE dataset of Escobar et al., 2019. Unlike the RSNA dataset, the radiographs



(a) RSNA (b) RHPE

Figure 6.3: Distribution of the G&P bone ages annotations for the RSNA (left) and RHPE (right) training datasets for male (M) and female (F) patients. On the RSNA dataset the distribution of boy bone ages is skewed towards 13 year old patients. On the RHPE dataset, the distribution of girl bone ages is skewed towards 10 year old patients.

of the RHPE dataset contain both hands of the patient. Rather than processing both hands at the same time, we artificially multiply the size of the dataset by two, by splitting the radiographs so that each hand makes up a single image. We obtain a MAE on the RHPE validation set of 6.74 months, which is far better than the performance of the BoNet approach (Escobar et al., 2019) which is reported by González et al., 2020, even though our model does not exploit any of the additional bounding box and keypoints annotations provided.

The resulting performance on our validation dataset can be observed in Tables 6.1, 6.2 and 6.3. Overall, switching from the RSNA to the RHPE dataset reduces the MAE from 7.51 months to 6.59 months, slightly decreases the sensitivity from 0.74 to 0.68 but slightly increases the specificity from 0.87 to 0.90. The benefits in terms of MAE are thus quite noticeable. Looking at the different sub-group analysis reveals that the decrease in MAE mostly comes from male patients, and in particular from the [5,9[and [9,13[sub-groups for which we introduced a bias correction mechanism in the previous section. Looking at the distribution of bone ages for male patients in Figure 6.3, we see that it is much less skewed than that of the RSNA, which seems to confirm the intuition of the previous section. However, the resulting model is still much better on boys between 13 and 17 years old, indicating that other factors might be at play. On girl patients, this model performs slightly worse than the RSNA-trained model, the difference being

the most important for younger patients between 5 and 9 years old. The skewdness of the bone ages distributions for girls suggest that another bias might be at play.

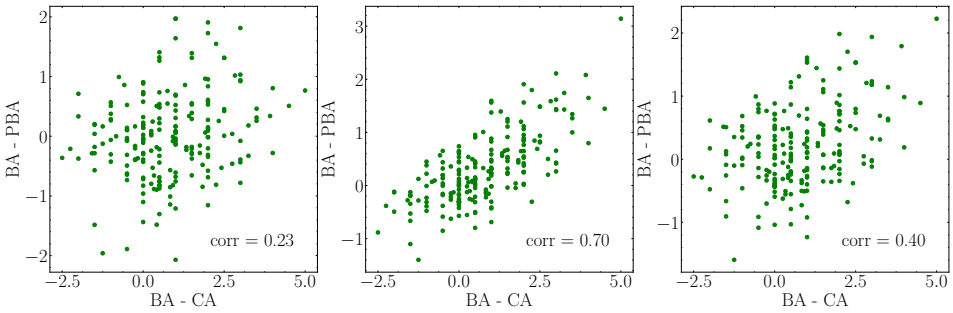
Unlike the RSNA dataset however, the authors of the RHPE dataset give access to the chronological age of the patients. Using those chronological ages can be used to refine the predictions of the AI. Indeed, in practice, when a radiologist is tasked to perform BAA using the G&P method, they need to search for the hand that resembles the most that of the patient. Using the chronological age of the patient grants an important time save by providing the radiologists with a first approximation of bone age. In turn, rather than browsing the whole atlas, they can quickly navigate around a few pages of interest. Likewise, introducing the chronological age of the patient to the input of the bone age regression head significantly grants the network with a strong proxy to the looked after bone age. To introduce the chronological age to our network, we employ the scheme introduced by González et al., 2020, and append to the features extracted by the backbone a linear mapping of chronological age. Unlike the authors however, we first normalize the chronological ages by subtracting the mean and dividing by the variance of all the chronological ages of the train set. With this scheme, our model reaches an MAE of 6.27 months, a result which again, is on par with the approach of the authors (6.34 months), without making use of additional annotations. González et al., 2020 propose to further train the model not to directly predict the bone age, but rather predict the offset from the chronological age to the bone age. However, unlike González et al., 2020, we do not observe that this leads to significant gain in MAE and settle with our baseline approach.

Again, looking at Tables 6.1, 6.2 and 6.3 grants us interesting insights. Unlike our belief, we do not observe a significant overall benefit to including chronological age as an input to the model. The global MAE only very slightly decreased from 6.59 months to 6.50 months. The specificity reaches an amazing 0.99, but at the cost of a terrible sensitivity of 0.40. On the different sub-groups, a significant decrease in MAE is only observed for the boys between 9 and 13 years old. This comes in contrast with the overall positive effect of the inclusion of the chronological age on the validation set of RHPE. We are unsure what causes this discrepancy. A possible explanation is that in the RHPE dataset, chronological ages were available up to the month, whereas in our dataset, only the calendar year of the patient was available. Unfortunately, we do not have the data required to test this hypothesis.

6.3.5 Adjusting for Chronological Age Bias

Benchmarks and challenges built on the RSNA and RHPE datasets only report MAE as a measure of interest. This focus fails to measure the propensity of the DL solutions to accurately diagnose patients with actual skeletal growth disorders, which is one of the main indications of BAA, and instead reward approaches which decrease the overall MAE by a fraction of a month. In the previous section, we demonstrated that, whilst the model utilizing chronological age is slightly better in terms for MAE on a specific population, an particularly adverse effect was observed on the sensitivity of the model. A possible effect that could explain this behavior is the over-reliance of the AI on the provided chronological ages. Indeed, chronological age and bone age are inherently correlated, as ideally, for a healthy patient, the bone age should be close to the chronological age. In turn, if the dataset is constituted of a majority of healthy patients, it is possible that the model might simply output the chronological age given input. To study this phenomenon, we display in Figure 6.4a and 6.4b, a scatter plot of the difference between the chronological age and the bone age, against the difference between the bone age and the predicted bone age for the model obtained from training respectively with and without the chronological ages. It transpires clearly from the resulting graph that an important correlation exists between the two when incorporating the chronological age, suggesting that the resulting model is quite conservative, and has a tendency to predict a bone age that is close to the chronological age, even in cases where the chronological age is in fact unrepresentative of the actual bone age of the patients, i.e. for pathological cases, the model has a tendency to predict a bone age value that is close to the patient chronological age. This explains the propensity of the model to commit very few false positives and thus have a near perfect specificity, but also miss 60% of the actually pathological patients.

To reduce the influence of this bias, we propose, as done in Section 6.3.3, to re-weight samples during training based on the distribution of the difference between the bone age and the chronological age. This time, we do not estimate the distribution of the bone ages, but the distribution of the difference between bone ages and chronological ages. We find however that the resulting distribution is too narrowed around a few points. To dampen the result, we consider weighting not by the inverse, but by a negative power of the distribution, i.e. $\omega(y) = f_{DA}(y)^{-\alpha}$. setting $\alpha = 1$ reverts to the scheme introduced in 6.3.3. We find that 0.5 works well in our setup, but do not try to tune α to avoid over-fitting on our clinical dataset. We report the performance of the resulting model in Tables 6.1, 6.2 and 6.3. Using the weighting scheme leads to a slight decrease in global MAE from 6.50 months to 6.20 months. A small MAE improvement is observed for both girls and boys. Most importantly, the significant gain for boys between 9



(a) BSL_{RHPE} (b) BSL_{RHPE}+CA (c) BSL_{RHPE}+CA+$\mathcal{W}$(DA)

Figure 6.4: Scatter plot of the difference between the bone age and the predicted bone age (BA - PBA) against the difference between bone age and chronological age (BA - CA) for three different networks. For each figure, we additionally report the Pearson correlation coefficient between the two variables. The left most plot corresponds to the model trained with the baseline seutp on the RHPE dataset. In the middle, the model trained with the additional chronological information. On the right, the model trained with the chronological information, and the weighting procedure described in Section 6.3.5.

and 13 years old that resulted from the introduction of the bone age is still present. A significant increase in sensitivity is observed from 0.40 to 0.60, accompanied by a reduction of specificity from 0.99 to 0.93, suggesting that the procedure had the intended result. We additionally report in Figure 6.4c the same correlations plots described above, for the new model. The correlations between the two variables noticeably reduced, as suggested by the evolution of the Pearson correlation coefficient, which dropped from 0.70 to 0.40.

We train a final model, with the aim of encompassing both of the weighting strategies introduced in this Section, in order to correct the bone age distribution bias, and the chronological age anchoring bias. A possible approach could have been to simply multiply the weights obtained with each procedure and normalize the result. This would be equivalent to considering that the two variables are independent, which is unlikely, but would offer the advantage of applying different dampening factors α to each set of weights. Instead we directly learn the joint distribution $f_{BA,DA}$ with a Gaussian kernel density estimator, and use a dampening factor α of 0.5. We report the results in Tables 6.1, 6.2 and 6.3. This last model reaches a global MAE of 5.72 months, a significant improvement over the baseline training scheme (6.59 months), with a roughly equivalent sensitivity and specificity. Delving into sub-groups, the additional weighting by bone age prevalence improved

over the weighting by the age difference prevalence in girls between 13 and 17 years old. The combination of the two results in a significant improvements over all girls from 6.29 months to 5.52 months. For boys, a similar global improvement was observed, but was only substantial for boys between 9 and 13 years old. All in all, the different modifications that we introduced led us to close the gap between male and female patients, as the MAE for boys greatly decreased from 7.58 months to 5.52 months.

6.4 Conclusion and Perspectives

In this chapter, we introduced a strong DL baseline for BAA that is able to perform on par with state of the art approaches without blows and whistles. The meticulous analysis of the performance of the algorithm on a clinical validation set allowed us to spot several problems with the available public datasets, which we attempted to resolve using several weighting mechanisms. The resulting model reaches an excellent MAE across all patients of 5.72 months, a fairly significant improvement over the performance of the reference non-expert radiologist. We detail below some of the limitations of this study, and several perspectives for future work.

Although we did find benefits to training models on both the RSNA and the RHPE datasets, leveraging chronological age significantly decreased MAE on boys between 9 and 13 years old. We thus chose to drop the RSNA dataset, because of the absence of reported chronological ages. Still, even without making use of chronological age, models trained on the RSNA dataset seemed to perform better on girls, whilst models trained on the RHPE dataset performed much better in boys. Although the distribution of the labels can partially explain the discrepancy, the weighting procedure introduced did not completely negate the biases. Albeit building a dataset with uniform distribution seems to be unrealistic given the very low prevalence of some bone age classes, perhaps we could have considered building an overall dataset with less skewed distributions simply by removing some of the over-represented samples from both training datasets, or by modifying the sampling strategy of the samples considered during training. Additional mechanisms could have also been introduced to allow training on samples both with and without an available chronological age.

Because classical BAA methods rely on localized information in the image, using the keypoints and bounding box annotations of the RHPE dataset is completely relevant. However, their incorporation into the prediction pipeline, as done by Escobar et al., 2019, requires to train and perform inference with two additional models, one to learn the bounding boxes in an object detection fashion, and the other to learn to predict the different keypoints. The additional costs incurred prevented their usage in our scenario.

However, several approaches in the literature have considered learning to predict both the bone age, and the additional spatial information at once (C. Chen et al., 2021; D. Wang et al., 2020). Taking advantage of such methods, and considering how they could be improved is a thrilling perspective to further improve the performances of our BAA algorithm. Unfortunately we did not find the time to thoroughly explore those methods during the extent of this PhD.

Although the different bias compensation steps introduced significantly improve the performance of our method on the clinical validation set, it is to easy to assess which final model is truly better. The model tuned to our internal model seems to display more fair performance, as it performs roughly the same on all our introduced sub-groups. However, the uniform distribution across sub-groups that we introduced in the dataset is artificial. For instance, the natural prevalence of chronological age for exams is far from uniform. Besides, pathological cases were most likely over-represented in our dataset. It is entirely plausible that on a dataset built to mimic a natural prevalence of those data characteristics, the modifications that we brought forward in this chapter could be detrimental. On top of this, we were limited in our stratified analysis by the available external metadata offered by the different datasets, but it is very likely that other data strates which cannot be represented exist in the dataset, for which our model is significantly worse or better. A natural candidate for such a strate would be the ethnical background of the patient, as there is already evidence in the literature that the holistic approach of the G&P atlas is not adapted to other populations. The TW atlas was corrected for several populations.

A strong limitation of our study is that we were only able to build a single clinical dataset, which size is relatively limited. In turn, it was difficult to tune some aspects of our algorithms, as doing so would have invariably led to over-fitting. This suggests that there is still a significant room for improvement, that could be unlocked by gathering and annotating more validation data. Besides, the overall clinical proof provided by this work is quite limited for two major reasons. First, the performance of a single radiologist was considered as reference, which is evidently non-representative and provides very limited proof of the advantages of the AI over the radiologist. Secondly, as pointed in other parts of this manuscript, it would be much more interesting to consider the gain in performance obtained by radiologists when assisted with such a tool.

Chapter 7

General Conclusion

In this ultimate chapter, we condense and put into perspective the different contributions of this PhD. We then discuss potential improvements, and plant the seeds of what we hope will grow into a flourishing future work.

7.1 Contributions

The course of this PhD has led us to scrutinize two instances of medical problems typically encountered by radiologists. The first one, fracture detection, deals with the precise localization of bone fractures, a frequent outcome of traumatic events. The second one, bone age assessment, is a meticulous exercise which purpose is to assess the skeletal development of children. Accuracy in both tasks is clinically crucial, as the consequences of errors in the former are costly and frequently debilitating, whilst a careful conduction of the latter is necessary for the detection of hormonal disorders. The ever increasing need for radiographic interpretation coupled with the difficulty inherent to the diagnosis of one or the other has led both problems to be candidate for computer-aided diagnosis tools for quite some time.

The advanced medical knowledge required to design classical feature-based machine learning approaches has made medical imaging arguably one of the great winners of the ubiquitous success of deep learning. In this work, we have presented two commonplace deep learning models which were almost directly transferred from natural to medical images. In chapter 2, we saw how the Mask R-CNN approach was at the core of the fracture detection tool of Gleamer. In chapter 6, we described a baseline approach for bone age assessment reusing a ConvNeXt backbone pre-trained on the ImageNet dataset. The core essence of this thesis was to explore ways to improve those baselines, and several strategies have been considered in this matter.

Chronologically, the initial gamble of this PhD was that tuning the architecture of deep learning models to radiographs using NAS would significantly improve their performance. In turn, NAS holds a special place in this work. The rationale behind our stance was that classical deep convolutional architectures had been specifically designed and tuned for natural images, and that a significant domain gap exists between natural images and X-rays. Bolstered by the evidence provided by early NAS approaches, we started reviewing new algorithms as they were released, with a particular focus on their optimization processes. We tried to capture the essence of the accumulated knowledge in Chapter 3. This review suggested that traditional NAS methods were out of our reach due to their expensive computational requirements. Yet a glimpse of hope seemingly appeared along the emergence of efficient NAS approaches making use of weight-sharing.

Unfortunately, despite its blatant computational efficacy, we had little success with WS, and realized that its adequacy as a proxy evaluation tool was subject to debate. On the one hand, several works still emerged successfully applying WS to new problems. On the other, several papers had appeared that emphasized its uselessness. As we found that the evidence supporting either side was lacking, we strove to study the matter ourselves. This is the gist of Chapter 4, in which we made use of a benchmark of architecture evaluations to systematically analyse WS on a wide variety of search spaces. A notable question that puzzled us was whether a standard NAS approach using WS could systematically outperform a baseline random search. Our conclusion for WS were grim, as we revealed that WS is particularly unreliable and search space dependent, sometimes performing slightly better than random search, most of the time being on par with random search. We used the work in this chapter as an opportunity to introduce good practices in the evaluation of WS based approaches, which we hope will foster future research in the field.

This bleak result did not completely falter our ambition to adapt neural architectures to radiographs, as we imputed the failure of NAS with WS to the untrustworthiness of the latter. Noticing that ImageNet pre-training was crucial to reach correct performance in our different practical setups, we were struck by the realization that any successful NAS application would require to support ImageNet pre-training, for the gain resulting from architecture tuning would most likely be smaller than the gain achieved from said pre-training. This additional condition further increased the resources required to perform NAS. To alleviate this pre-training burden, we suggested in Chapter 5 to use WS, not as an evaluation tool, but as a source of pre-trained weights. By combining this pre-training with a tight training schedule, we were able to perform standard NAS on a dataset of fracture patches, and improved the design of a baseline architecture tuned for natural images. Still, we realized that the resources required for a marginal

improvement resulted in a poor return over investment, and thus ended our NAS experiments with this result.

The strategy that unveiled from the refinement of our bone age model, which we detailed in Chapter 6, was utterly different. Rather than blindly optimizing an architecture to this new task, our take was to reuse a deep convolutional network designed for natural images and spend minimal time tuning its design. In turn, the few modifications that we introduced over prior approaches were simply to update the training setup with contemporary computer vision tips and tricks. Instead, we invested our time and expertise in the in-depth examination of the model's results across sub-populations of interest. This data-centric approach helped us uncover several biases created either by the dataset used to train our model, or by the modifications that we introduced on the model itself. Adjusting the training procedure for those biases using plain sample weighting schemes resulted in a significant improvement of the resulting model on the sub-populations of interest.

Finally, throughout this PhD, a particular care was given to provide clinically relevant insights into the different AI solutions introduced. The foundation of Chapter 2 lies with the introduction of the clinical study used to assess the quality of Gleamer's fracture detection tool. We presented the clinical protocol used to monitor the evolution of the performances of the different medical experts with and without the assistance of the AI. Most notably, we tried to explain with great care the different introduced measures of reader performance, and why previously used performance indicators were inadequate and not relevant. In Chapter 6, albeit with more limited resources, we tried to compare our bone age assessment model with a reference radiologist. More importantly, we raised concerns towards the literature's quest for the lowest possible mean average error, which completely conceals the problem of interest that is the detection of pathological patients.

7.2 Limits and Perspectives

Our different contributions to the NAS literature are not without limits. In the extensive analysis of WS that we provide in Chapter 4, the sole usage of the NAS-Bench-101 dataset coupled with the simplicity of the baseline NAS approaches considered non-negligibly weakens our different claims. This could readily be fixed by extending our work with the analysis of distinct evaluation datasets according to our introduced guidelines. As for the pre-training scheme introduced in Chapter 5, the prerequisite of having access to a pre-trained supernet hinders its practical applications for at least two reasons: (i) Training a super-net is costly, and requires several tricks that

are poorly documented in the literature; (ii) Supernets come with built-in search spaces, which are typically made up of architectures with limited sizes. In turn, looking for state-of-the-art architectures using this method is unrealistic. The architecture that we found using our method was only of interest in a scenario where the inference-time computational requirements of the models needed to be considered, which has hardly ever been a problem in our different medical applications. In turn the performance of the resulting architecture was quite underwhelming compared to that of baseline ResNet and DenseNet models. A possible way to fix this would be to increase the size of the found architecture a posteriori, as performed by Tan et al., 2019b at the cost of further computational expenses.

Our global experience with NAS has been rather negative, as even in the most favorable scenario and with decent computation resources, we have only observed marginal gains. This poor return over investment has made us question the utility of this paradigm, even more so given the environmental concerns of such resource-hungry approaches (Dhar, 2020). Interestingly, companies which led NAS research a few years ago have also displayed a complete loss of interest in the field, suggesting that even when tackled with the largest pool of resources practically conceivable the NAS paradigm is still lacking. As of today, the focus of such companies as instead shifted towards designing rather plain model and making use of extremely large datasets of labeled images, or making use of self-supervised techniques (Jaiswal et al., 2020; He et al., 2022) as pre-training. In particular the vision transformer (ViT) architectures (Dosovitskiy et al., 2020) and other transformer-based models (Ze Liu et al., 2021), which design are almost directly copied from plain text transformer models (Vaswani et al., 2017), have become a growing trend in ai research, hinting at a more general family of models capable of learning over various sources of data, further questionning the benefits provided by NAS. In turn, although the answer to the concerns raised in the previous paragraph would be of interest to us out of pure curiosity, this PhD has made it clear that Gleamer tools are unlikely to ever directly benefit from NAS.

On the contrary, our study of BAA was quite successful. Although some aspects still need to be elucidated, including why the different public datasets induce different performances on boys and girls, and how model performance could be improved using the additional hand keypoint annotations, the data and results centric approach was much more favorable to the problem. Our biggest regret with the study of Chapter 6 is the relatively limited clinical evidence provided in the analysis of our BAA tool. First, our study falls into one of the shortcomings described in Chapter 2 since it only considered the standalone performance of the AI approach and did not address the potential increase of performance for radiologists. Secondly, a single medical expert was considered to be compared with our AI model,

which impedes any form of statistical evidence. The main reasons behind those decisions was that the work presented in Chapter 6 was incremental and quite exploratory. Overall, its goal was to study the feasibility of transferring a model learned using public datasets to a new population of interest. Given that recruiting medical experts incurs significant costs and that performing a clinical study takes considerable time, we decided to limit our investigation to this minimal setting. However, now that we are satisfied with the resulting model design and performance, we hope that in the near future we will be able to truly explore whether the AI genuinely helps radiologists.

Bibliography

Adams, Matthew et al. (2019). "Computer vs human: deep learning versus perceptual training for the detection of neck of femur fractures". In: *Journal of medical imaging and radiation oncology* 63.1, pp. 27–32.

Ahmed, Karim and Lorenzo Torresani (2018). "Maskconnect: Connectivity learning by gradient descent". In: *Proceedings of ECCV)*, pp. 349–365.

Akimoto, Youhei et al. (2019). "Adaptive Stochastic Natural Gradient Method for One-Shot Neural Architecture Search". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 171–180. URL: http://proceedings.mlr.press/v97/akimoto19a.html.

Alshamrani, Khalaf, Fabrizio Messina, and Amaka C Offiah (2019). "Is the Greulich and Pyle atlas applicable to all ethnicities? A systematic review and meta-analysis". In: *European radiology* 29.6, pp. 2910–2923.

Angeline, Peter J, Gregory M Saunders, and Jordan B Pollack (1994). "An evolutionary algorithm that constructs recurrent neural networks". In: *IEEE transactions on Neural Networks* 5.1, pp. 54–65.

Arasu, Vignesh A et al. (2015). "Diagnostic emergency imaging utilization at an academic trauma center from 1996 to 2012". In: *Journal of the American College of Radiology* 12.5, pp. 467–474.

Arulkumaran, Kai et al. (2017). "Deep reinforcement learning: A brief survey". In: *IEEE Signal Processing Magazine* 34.6, pp. 26–38.

Baker, Bowen et al. (2017). "Accelerating neural architecture search using performance prediction". In: *arXiv preprint arXiv:1705.10823*.

Bandos, Andriy I et al. (2009). "Area under the free-response ROC curve (FROC) and a related summary index". In: *Biometrics* 65.1, pp. 247–256.

Bender, Gabriel et al. (2018). "Understanding and Simplifying One-Shot Architecture Search". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by Jennifer G. Dy and Andreas Krause.

Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 549–558. URL: http://proceedings.mlr.press/v80/bender18a.html.

Beyer, Hans-Georg and Hans-Paul Schwefel (2002). "Evolution strategies–A comprehensive introduction". In: *Natural computing* 1.1, pp. 3–52.

Brock, Andrew et al. (2018). "SMASH: One-Shot Model Architecture Search through HyperNetworks". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. URL: https://openreview.net/forum?id=rydeCEhs-.

Busby, Lindsay P, Jesse L Courtier, and Christine M Glastonbury (2018). "Bias in radiology: the how and why of misses and misinterpretations". In: *Radiographics* 38.1, pp. 236–247.

Cai, Han, Ligeng Zhu, and Song Han (2018). "Proxylessnas: Direct neural architecture search on target task and hardware". In: *arXiv preprint arXiv:1812.00332*.

Cai, Han et al. (2020). "Once-for-All: Train One Network and Specialize it for Efficient Deployment". In: *Proc. of ICLR*. OpenReview.net. URL: https://openreview.net/forum?id=HylxE1HKwS.

Cao, Yu et al. (2015). "Fracture detection in x-ray images through stacked random forests feature fusion". In: *2015 IEEE 12th international symposium on biomedical imaging (ISBI)*. IEEE, pp. 801–805.

Casale, Francesco Paolo, Jonathan Gordon, and Nicolo Fusi (2019). "Probabilistic Neural Architecture Search". In: *arXiv e-prints*, arXiv:1902.05116, arXiv:1902.05116. arXiv: 1902.05116 [stat.ML].

Castellino, Ronald A (2005). "Computer aided detection (CAD): an overview". In: *Cancer Imaging* 5.1, p. 17.

Castriota-Scanderbeg, ADMV and V De Micheli (1995). "Ultrasound of femoral head cartilage: a new method of assessing bone age". In: *Skeletal radiology* 24.3, pp. 197–200.

Chellapilla, Kumar, Sidd Puri, and Patrice Simard (2006). "High performance convolutional neural networks for document processing". In: *Tenth international workshop on frontiers in handwriting recognition*. Suvisoft.

Chen, Chao et al. (2021). "Attention-guided discriminative region localization and label distribution learning for bone age assessment". In: *IEEE Journal of Biomedical and Health Informatics*.

Chen, Wei et al. (2017). "National incidence of traumatic fractures in China: a retrospective survey of 512 187 individuals". In: *The Lancet Global Health* 5.8, e807–e817.

Cheng, Chi-Tung et al. (2019). "Application of a deep learning algorithm for detection and visualization of hip fractures on plain pelvic radiographs". In: *European radiology* 29.10, pp. 5469–5477.

Chu, Xiangxiang et al. (2019). "FairNAS: Rethinking Evaluation Fairness of Weight Sharing Neural Architecture Search". In: *arXiv e-prints*, arXiv:1907.01845, arXiv:1907.01845. arXiv: 1907.01845 [cs.LG].

Chung, Seok Won et al. (2018). "Automated detection and classification of the proximal humerus fracture by using deep learning algorithm". In: *Acta orthopaedica* 89.4, pp. 468–473.

Ciresan, Dan Claudiu et al. (2011). "Flexible, high performance convolutional neural networks for image classification". In: *Twenty-second international joint conference on artificial intelligence*.

Cohen, Jacob (1988). "Statistical power analysis for the behavioral sciences. Abingdon". In: *England: Routledge*.

Colas, Cédric, Olivier Sigaud, and Pierre-Yves Oudeyer (2018). "How many random seeds? statistical power analysis in deep reinforcement learning experiments". In: *arXiv preprint arXiv:1806.08295*.

Colson, Benoît, Patrice Marcotte, and Gilles Savard (2007). "An overview of bilevel optimization". In: *Annals of operations research* 153.1, pp. 235–256.

Den Ottelander, Tom, Arkadiy Dushatskiy, Virgolin, et al. (2021). "Local search is a remarkably strong baseline for neural architecture search". In: *EMO*. Springer, pp. 465–479.

Dhar, Payal (2020). "The carbon impact of artificial intelligence". In: *Nature Machine Intelligence* 2.8, pp. 423–425.

DiMaggio, Charles J et al. (2017). "The epidemiology of emergency department trauma discharges in the United States". In: *Academic emergency medicine* 24.10, pp. 1244–1256.

Domhan, Tobias, Jost Tobias Springenberg, and Frank Hutter (2015). "Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves". In: *Twenty-fourth international joint conference on artificial intelligence*.

Dong, Xuanyi and Yi Yang (2020). "NAS-Bench-201: Extending the Scope of Reproducible Neural Architecture Search". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. URL: https://openreview.net/forum?id=HJxyZkBKDr.

Donnelley, Martin, Greg Knowles, and Trevor Hearn (2008). "A CAD system for long-bone segmentation and fracture detection". In: *International Conference on Image and Signal Processing*. Springer, pp. 153–162.

Dosovitskiy, Alexey et al. (2020). "An image is worth 16x16 words: Transformers for image recognition at scale". In: *arXiv preprint arXiv:2010.11929*.

Dudziak, Lukasz et al. (2020). "Brp-nas: Prediction-based nas using gcns". In: *Advances in Neural Information Processing Systems* 33, pp. 10480–10490.

Duron, Loïc, Alexis Ducarouge, André Gillibert, et al. (2021). "Assessment of an AI aid in detection of adult appendicular skeletal fractures by emergency physicians and radiologists: a multicenter cross-sectional diagnostic study". In: *Radiology* 300.1, pp. 120–129.

Eberhart, Russell C and Yuhui Shi (1998). "Comparison between genetic algorithms and particle swarm optimization". In: *International conference on evolutionary programming*. Springer, pp. 611–616.

Elsken, Thomas, Jan Hendrik Metzen, and Frank Hutter (2019a). "Efficient Multi-Objective Neural Architecture Search via Lamarckian Evolution". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. URL: https://openreview.net/forum?id=ByME42AqK7.

— (2019b). "Neural architecture search: A survey". In: *The Journal of Machine Learning Research* 20.1, pp. 1997–2017.

Escobar, María et al. (2019). "Hand pose estimation for pediatric bone age assessment". In: *International conference on medical image computing and computer-assisted intervention*. Springer, pp. 531–539.

Falkner, Stefan, Aaron Klein, and Frank Hutter (2018). "BOHB: Robust and Efficient Hyperparameter Optimization at Scale". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by Jennifer G. Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 1436–1445. URL: http://proceedings.mlr.press/v80/falkner18a.html.

Fenton, Joshua J et al. (2007). "Influence of computer-aided detection on performance of screening mammography". In: *New England Journal of Medicine* 356.14, pp. 1399–1409.

Feurer, Matthias and Frank Hutter (2019). "Hyperparameter optimization". In: *Automated machine learning*. Springer, Cham, pp. 3–33.

Franceschi, Luca et al. (2018). "Bilevel Programming for Hyperparameter Optimization and Meta-Learning". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by Jennifer G. Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 1563–1572. URL: http://proceedings.mlr.press/v80/franceschi18a.html.

Fukushima, Kunihiko and Sei Miyake (1982). "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition". In: *Competition and cooperation in neural nets*. Springer, pp. 267–285.

Gale, William et al. (2017). "Detecting hip fractures with radiologist-level performance using deep neural networks". In: *arXiv preprint arXiv:1711.06504*.

Gertych, Arkadiusz et al. (2007). "Bone age assessment of children using a digital hand atlas". In: *Computerized medical imaging and graphics* 31.4-5, pp. 322–331.

Ghafoorian, Mohsen et al. (2016). "Non-uniform patch sampling with deep convolutional neural networks for white matter hyperintensity segmentation". In: *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*. IEEE, pp. 1414–1417.

Gilsanz, Vicente and Osman Ratib (2005). *Hand bone age: a digital atlas of skeletal maturity*. Springer.

Girshick, Ross (2015). "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448.

Girshick, Ross et al. (2014). "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587.

Goldberg, David E and Kalyanmoy Deb (1991). "A comparative analysis of selection schemes used in genetic algorithms". In: *Foundations of genetic algorithms*. Vol. 1. Elsevier, pp. 69–93.

González, Cristina et al. (2020). "SIMBA: Specific identity markers for bone age assessment". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, pp. 753–763.

Greulich, William Walter and Sarah Idell Pyle (1959). *Radiographic atlas of skeletal development of the hand and wrist*. Stanford university press.

Grigorescu, Sorin et al. (2020). "A survey of deep learning techniques for autonomous driving". In: *Journal of Field Robotics* 37.3, pp. 362–386.

Gruau, Frederic (1994). "Neural network synthesis using cellular encoding and the genetic algorithm". In.

Guermazi, Ali, Chadi Tannoury, Andrew J Kompel, et al. (2021). "Improving Radiographic Fracture Recognition Performance and Efficiency Using Artificial Intelligence". In: *Radiology*, p. 210937.

Guly, HR (2001). "Diagnostic errors in an accident and emergency department". In: *Emergency Medicine Journal* 18.4, pp. 263–269.

Guo, Zichao et al. (2020). "Single path one-shot neural architecture search with uniform sampling". In: *European Conference on Computer Vision*. Springer, pp. 544–560.

Ha, David, Andrew M. Dai, and Quoc V. Le (2017). "HyperNetworks". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. URL: https://openreview.net/forum?id=rkpACe1lx.

Halabi, Safwan S et al. (2019). "The RSNA pediatric bone age machine learning challenge". In: *Radiology* 290.2, pp. 498–503.

He, Kaiming et al. (2016). "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, pp. 770–778. DOI: 10.1109/CVPR.2016.90. URL: https://doi.org/10.1109/CVPR.2016.90.

He, Kaiming et al. (2017). "Mask R-CNN". In: *Proc. of ICCV*. IEEE Computer Society, pp. 2980–2988. DOI: 10.1109/ICCV.2017.322. URL: https://doi.org/10.1109/ICCV.2017.322.

He, Kaiming et al. (2022). "Masked autoencoders are scalable vision learners". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009.

Howard, Andrew et al. (2019). "Searching for mobilenetv3". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1314–1324.

Huang, Gao et al. (2016). "Deep networks with stochastic depth". In: *European conference on computer vision*. Springer, pp. 646–661.

Hubel, David H and Torsten N Wiesel (1968). "Receptive fields and functional architecture of monkey striate cortex". In: *The Journal of physiology* 195.1, pp. 215–243.

Iandola, Forrest N et al. (2016). "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and¡ 0.5 MB model size". In: *arXiv preprint arXiv:1602.07360*.

Ioffe, Sergey and Christian Szegedy (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. PMLR, pp. 448–456.

Jaderberg, Max, Karen Simonyan, Andrew Zisserman, et al. (2015). "Spatial transformer networks". In: *Advances in neural information processing systems* 28.

Jaeger, Paul F et al. (2020). "Retina U-Net: Embarrassingly simple exploitation of segmentation supervision for medical object detection". In: *Machine Learning for Health Workshop*. PMLR, pp. 171–183.

Jaiswal, Ashish et al. (2020). "A survey on contrastive self-supervised learning". In: *Technologies* 9.1, p. 2.

James, John T (2013). "A new, evidence-based estimate of patient harms associated with hospital care". In: *Journal of patient safety* 9.3, pp. 122–128.

Jones, Donald R, Matthias Schonlau, and William J Welch (1998). "Efficient global optimization of expensive black-box functions". In: *Journal of Global optimization* 13.4, pp. 455–492.

Kim, DH and T MacKinnon (2018). "Artificial intelligence in fracture detection: transfer learning from deep convolutional neural networks". In: *Clinical radiology* 73.5, pp. 439–445.

Kim, Young W and Liem T Mansfield (2014). "Fool me twice: delayed diagnoses in radiology with emphasis on perpetuated errors". In: *AJR Am J Roentgenol* 202.3, pp. 465–470.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Kitamura, Gene, Chul Y Chung, and Barry E Moore (2019). "Ankle fracture detection utilizing a convolutional neural network ensemble implemented with a small sample, de novo training, and multiview incorporation". In: *Journal of digital imaging* 32.4, pp. 672–677.

Klein, Aaron et al. (2016). "Learning curve prediction with Bayesian neural networks". In.

Klein, Aaron et al. (2017). "Fast bayesian optimization of machine learning hyperparameters on large datasets". In: *Artificial intelligence and statistics*. PMLR, pp. 528–536.

Kornblith, Simon, Jonathon Shlens, and Quoc V. Le (2019). "Do Better ImageNet Models Transfer Better?" In: *Proc. of CVPR*. Computer Vision Foundation / IEEE, pp. 2661–2671. DOI: 10.1109/CVPR.2019.00277. URL: http://openaccess.thecvf.com/content%5C_CVPR%5C_2019/html/Kornblith%5C_Do%5C_Better%5C_ImageNet%5C_Models%5C_Transfer%5C_Better%5C_CVPR%5C_2019%5C_paper.html.

Kreitner, K-F et al. (1998). "Bone age determination based on the study of the medial extremity of the clavicle". In: *European radiology* 8.7, pp. 1116–1122.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25.

Kuo, Rachel YL et al. (2022). "Artificial intelligence in fracture detection: a systematic review and meta-analysis". In: *Radiology*, p. 211785.

Larson, David B et al. (2018). "Performance of a deep-learning neural network model in assessing skeletal maturity on pediatric hand radiographs". In: *Radiology* 287.1, pp. 313–322.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). "Deep learning". In: *nature* 521.7553, pp. 436–444.

LeCun, Yann et al. (1989). "Handwritten digit recognition with a back-propagation network". In: *Advances in neural information processing systems* 2.

Lee, June-Goo et al. (2017). "Deep learning in medical imaging: general overview". In: *Korean journal of radiology* 18.4, pp. 570–584.

Li, Liam and Ameet Talwalkar (2019). "Random Search and Reproducibility for Neural Architecture Search". In: *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019, Tel Aviv, Israel, July 22-25, 2019*. Ed. by Amir Globerson and Ricardo Silva. Vol. 115. Proceedings of Machine Learning Research. AUAI Press, pp. 367–377. URL: http://proceedings.mlr.press/v115/li20c.html.

Lin, Tsung-Yi et al. (2017). "Feature pyramid networks for object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125.

Lindsey, Robert, Aaron Daluiski, Sumit Chopra, et al. (2018). "Deep neural network improves fracture detection by clinicians". In: *Proceedings of the National Academy of Sciences* 115.45, pp. 11591–11596.

Litjens, Geert et al. (2017). "A survey on deep learning in medical image analysis". In: *Medical image analysis* 42, pp. 60–88.

Little, David G and Michael D Sussman (1994). "The Risser sign: a critical analysis." In: *Journal of pediatric orthopedics* 14.5, pp. 569–575.

Liu, Chenxi et al. (2018). "Progressive neural architecture search". In: *Proceedings of the European conference on computer vision (ECCV)*, pp. 19–34.

Liu, Hanxiao, Karen Simonyan, and Yiming Yang (2019a). "DARTS: Differentiable Architecture Search". In: *Proc. of ICLR*. OpenReview.net. URL: https://openreview.net/forum?id=S1eYHoC5FX.

— (2019b). "DARTS: Differentiable Architecture Search". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. URL: https://openreview.net/forum?id=S1eYHoC5FX.

Liu, Hanxiao et al. (2017). "Hierarchical representations for efficient architecture search". In: *arXiv preprint arXiv:1711.00436*.

Liu, Li et al. (2020). "Deep learning for generic object detection: A survey". In: *International journal of computer vision* 128.2, pp. 261–318.

Liu, Ze et al. (2021). "Swin transformer: Hierarchical vision transformer using shifted windows". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022.

Liu, Zhuang et al. (2022). "A ConvNet for the 2020s". In: *arXiv preprint arXiv:2201.03545*.

Loshchilov, Ilya and Frank Hutter (2017). "Decoupled weight decay regularization". In: *arXiv preprint arXiv:1711.05101*.

Loy, Clement T and Les Irwig (2004). "Accuracy of diagnostic tests read with and without clinical information: a systematic review". In: *Jama* 292.13, pp. 1602–1609.

Luo, Renqian, Tao Qin, and Enhong Chen (2019). "Balanced One-shot Neural Architecture Optimization". In: *arXiv preprint arXiv:1909.10815*.

Luo, Renqian et al. (2018). "Neural Architecture Optimization". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by Samy Bengio et al., pp. 7827–7838. URL: https://proceedings.neurips.cc/paper/2018/hash/933670f1ac8ba969f32989c312faba75-Abstract.html.

Magnusson, Kristoffer (2020). *Interpreting Cohen's d Effect Size: An Interactive Visualization*. Version 2.1.1. URL: https://rpsychologist.com/d3/cohend/.

Mansourvar, Marjan et al. (2013). "Automated bone age assessment: motivation, taxonomies, and challenges". In: *Computational and mathematical methods in medicine* 2013.

Mettler Jr, Fred A et al. (2008). "Effective doses in radiology and diagnostic nuclear medicine: a catalog". In: *Radiology* 248.1, pp. 254–263.

Michael, David J and Alan C Nelson (1989). "HANDX: a model-based system for automatic segmentation of bones from digital hand radiographs". In: *IEEE transactions on medical imaging* 8.1, pp. 64–69.

Miikkulainen, Risto et al. (2019). "Evolving deep neural networks". In: *Artificial intelligence in the age of neural networks and brain computing*. Elsevier, pp. 293–312.

Negrinho, Renato and Geoff Gordon (2017). "Deeparchitect: Automatically designing and training deep architectures". In: *arXiv preprint arXiv:1704.08792*.

Ning, Xuefei et al. (2020). "A generic graph-based neural architecture encoding scheme for predictor-based nas". In: *European Conference on Computer Vision*. Springer, pp. 189–204.

Ottelander, T Den et al. (2020). "Local Search is a Remarkably Strong Baseline for Neural Architecture Search". In: *arXiv preprint arXiv:2004.08996*.

Pan, Zhaoqing et al. (2019). "Recent progress on generative adversarial networks (GANs): A survey". In: *IEEE Access* 7, pp. 36322–36333.

Pham, Hieu et al. (2018). "Efficient Neural Architecture Search via Parameter Sharing". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by Jennifer G. Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 4092–4101. URL: http://proceedings.mlr.press/v80/pham18a.html.

Pizer, Stephen M et al. (1987). "Adaptive histogram equalization and its variations". In: *Computer vision, graphics, and image processing* 39.3, pp. 355–368.

Rapin, J. and O. Teytaud (2018). *Nevergrad*. https://github.com/facebookresearch/nevergrad.

Real, Esteban, Sherry Moore, Andrew Selle, et al. (2017). "Large-Scale Evolution of Image Classifiers". In: *Proc.* Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 2902–2911. URL: http://proceedings.mlr.press/v70/real17a.html.

Real, Esteban et al. (2019a). "Regularized Evolution for Image Classifier Architecture Search". In: *Proc. of AAAI*. AAAI Press, pp. 4780–4789. DOI: 10.1609/aaai.v33i01.33014780. URL: https://doi.org/10.1609/aaai.v33i01.33014780.

— (2019b). "Regularized Evolution for Image Classifier Architecture Search". In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, pp. 4780–4789. DOI: 10.1609/aaai.v33i01.33014780. URL: https://doi.org/10.1609/aaai.v33i01.33014780.

Rechenberg, Ingo (1978). "Evolutionsstrategien". In: *Simulationsmethoden in der Medizin und Biologie*. Springer, pp. 83–114.

Ren, Pengzhen et al. (2020). "A comprehensive survey of neural architecture search: Challenges and solutions". In: *arXiv preprint arXiv:2006.02903*.

Ren, Shaoqing et al. (2015). "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems* 28.

Rimmer, Abi (2017). "Radiologist shortage leaves patient care at risk, warns royal college". In: *BMJ: British Medical Journal (Online)* 359.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer, pp. 234–241.

Sandler, Mark et al. (2018). "MobileNetV2: Inverted Residuals and Linear Bottlenecks". In: *Proc. of CVPR*. IEEE Computer Society, pp. 4510–4520. DOI: 10.1109/CVPR.2018.00474. URL: http://openaccess.thecvf.com/content%5C_cvpr%5C_2018/html/Sandler%5C_MobileNetV2%5C_Inverted%5C_Residuals%5C_CVPR%5C_2018%5C_paper.html.

Satoh, Mari (2015). "Bone age: assessment methods and clinical applications". In: *Clinical Pediatric Endocrinology* 24.4, pp. 143–152.

Scepi, Michel et al. (2005). "Discordant results in x-ray interpretations between ED physicians and radiologists. A prospective investigation of 30000 trauma patients". In: *The American journal of emergency medicine* 23.7, pp. 918–920.

Schumacher, Günter, Andreas Schmeling, and Ernst Rudolf (2018). "Medical age assessment of juvenile migrants: an analysis of age marker-based assessment criteria". In: *Joint Research Centre (JRC) science for policy report, European Union, Luxembourg*.

Seabold, Skipper and Josef Perktold (2010). "statsmodels: Econometric and statistical modeling with python". In: 9^{th} *Python in Science Conference*.

Sermanet, Pierre et al. (2013). "Overfeat: Integrated recognition, localization and detection using convolutional networks". In: *arXiv preprint arXiv:1312.6229*.

Shaukat, Furqan, Gulistan Raja, and Alejandro F Frangi (2019). "Computer-aided detection of lung nodules: a review". In: *Journal of Medical Imaging* 6.2, p. 020901.

Shen, Li et al. (2019). "Deep learning to improve breast cancer detection on screening mammography". In: *Scientific reports* 9.1, pp. 1–12.

Shi, Han et al. (2020). "Bridging the gap between sample-based and one-shot neural architecture search with bonas". In: *Advances in Neural Information Processing Systems* 33, pp. 1808–1819.

Shi, Yuhui (2011). "Brain storm optimization algorithm". In: *International conference in swarm intelligence*. Springer, pp. 303–309.

Shin, Richard, Charles Packer, and Dawn Song (2018). "Differentiable neural network architecture search". In.

Shirakawa, Shinichi, Yasushi Iwata, and Youhei Akimoto (2018). "Dynamic optimization of neural network structures using probabilistic modeling". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1.

Simon, Dan (2013). *Evolutionary optimization algorithms*. John Wiley & Sons.

Simonyan, Karen and Andrew Zisserman (2014). "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556*.

Spampinato, Concetto et al. (2017). "Deep learning for automated skeletal bone age assessment in X-ray images". In: *Medical image analysis* 36, pp. 41–51.

Stamoulis, Dimitrios et al. (2019). "Single-Path NAS: Designing Hardware-Efficient ConvNets in less than 4 Hours". In: *arXiv e-prints*, arXiv:1904.02877, arXiv:1904.02877. arXiv: 1904.02877 [cs.LG].

Stanley, Kenneth O and Risto Miikkulainen (2002). "Evolving neural networks through augmenting topologies". In: *Evolutionary computation* 10.2, pp. 99–127.

Suganuma, Masanori, Shinichi Shirakawa, and Tomoharu Nagao (2017). "A genetic programming approach to designing convolutional neural network architectures". In: *Proceedings of the genetic and evolutionary computation conference*, pp. 497–504.

Sutton, David (1987). "A textbook of radiology and imaging". In.

Szegedy, Christian et al. (2015). "Going deeper with convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9.

Szegedy, Christian et al. (2016). "Rethinking the inception architecture for computer vision". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826.

Tan, Mingxing, Bo Chen, Ruoming Pang, et al. (2019a). "MnasNet: Platform-Aware Neural Architecture Search for Mobile". In: *Proc. of CVPR*. Computer Vision Foundation / IEEE, pp. 2820–2828. DOI: 10.1109/CVPR.2019.00293. URL: http://openaccess.thecvf.com/content%5C_CVPR%5C_2019/html/Tan%5C_MnasNet%5C_Platform-Aware%5C_Neural%5C_Architecture%5C_Search%5C_for%5C_Mobile%5C_CVPR%5C_2019%5C_paper.html.

Tan, Mingxing and Quoc V. Le (2019b). "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: *Proc. of ICML*. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 6105–6114. URL: http://proceedings.mlr.press/v97/tan19a.html.

Tanner, James Mourilyan (1983). "Assessment of skeletal maturity and prediction of adult height". In: *TW 2 Method*, pp. 50–106.

Tanner, JM, RD Gibbons, and RD Bock (1992). "An image analysis system for TW skeletal maturity". In: *Hormone research* 37.supplement 3, pp. 11–15.

Thodberg, Hans Henrik et al. (2008). "The BoneXpert method for automated determination of skeletal maturity". In: *IEEE transactions on medical imaging* 28.1, pp. 52–66.

Uijlings, Jasper RR et al. (2013). "Selective search for object recognition". In: *International journal of computer vision* 104.2, pp. 154–171.

Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems* 30.

Virtanen, Pauli et al. (2020). "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17, pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

Wang, Dong et al. (2020). "Improve bone age assessment by learning from anatomical local regions". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, pp. 631–640.

Wang, Linnan, Yiyang Zhao, and Yuu Jinnai (2018). "AlphaX: eXploring Neural Architectures with Deep Neural Networks and Monte Carlo Tree Search". In: *CoRR* abs/1805.07440. arXiv: 1805.07440. URL: http://arxiv.org/abs/1805.07440.

Wang, Linnan et al. (2021). "Sample-efficient neural architecture search by learning actions for monte carlo tree search". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

White, Colin, Willie Neiswanger, and Yash Savani (2019). "Bananas: Bayesian optimization with neural architectures for neural architecture search". In: *arXiv preprint arXiv:1910.11858* 1.2, p. 4.

White, Colin, Sam Nolen, and Yash Savani (2020). "Local Search is State of the Art for NAS Benchmarks". In: *arXiv preprint arXiv:2005.02960*.

White, Colin et al. (2021). "How Powerful are Performance Predictors in Neural Architecture Search?" In: *Advances in Neural Information Processing Systems* 34.

Willems, Guy (2001). "A review of the most commonly used dental age estimation techniques." In: *The Journal of forensic odonto-stomatology* 19.1, pp. 9–17.

Willett, Jessica K (2019). "Imaging in trauma in limited-resource settings: a literature review". In: *African Journal of Emergency Medicine* 9, S21–S27.

Williams, Ronald J. (1992). "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Machine Learning* 8.3, pp. 229–256. ISSN: 1573-0565. DOI: 10.1007/BF00992696. URL: https://doi.org/10.1007/BF00992696.

Wistuba, Martin (2017). "Finding competitive network architectures within a day using uct". In: *arXiv preprint arXiv:1712.07420*.

Wu, Yuxin, Alexander Kirillov, Francisco Massa, et al. (2019). *Detectron2*. https://github.com/facebookresearch/detectron2.

Wu, Zonghan et al. (2020). "A comprehensive survey on graph neural networks". In: *IEEE transactions on neural networks and learning systems* 32.1, pp. 4–24.

Xie, Lingxi et al. (2020). "Weight-Sharing Neural Architecture Search:A Battle to Shrink the Optimization Gap". In: *arXiv preprint arXiv:2008.01475*.

Xie, Saining et al. (2017). "Aggregated residual transformations for deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500.

Xie, Sirui et al. (2019). "SNAS: stochastic neural architecture search". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. URL: https://openreview.net/forum?id=rylqooRqK7.

Yang, Antoine, Pedro M. Esperança, and Fabio Maria Carlucci (2020). "NAS evaluation is frustratingly hard". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. URL: https://openreview.net/forum?id=HygrdpVKvr.

Yao, Xin (1999). "Evolving artificial neural networks". In: *Proceedings of the IEEE* 87.9, pp. 1423–1447.

Ying, Chris et al. (2019). "NAS-Bench-101: Towards Reproducible Neural Architecture Search". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 7105–7114. URL: http://proceedings.mlr.press/v97/ying19a.html.

Yu, Jiahui, Pengchong Jin, Hanxiao Liu, et al. (2020). "Bignas: Scaling up neural architecture search with big single-stage models". In: *Proc. of ECCV*. Springer, pp. 702–717.

Yu, Kaicheng, Rene Ranftl, and Mathieu Salzmann (2020a). "How to Train Your Super-Net: An Analysis of Training Heuristics in Weight-Sharing NAS". In: *arXiv preprint arXiv:2003.04276*.

Yu, Kaicheng et al. (2020b). "Evaluating The Search Phase of Neural Architecture Search". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. URL: https://openreview.net/forum?id=H1loF2NFwr.

Zela, Arber, Julien Siems, and Frank Hutter (2020). "NAS-Bench-1Shot1: Benchmarking and Dissecting One-shot Neural Architecture Search". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. URL: https://openreview.net/forum?id=SJx9ngStPH.

Zela, Arber et al. (2018). "Towards automated deep learning: Efficient joint neural architecture and hyperparameter search". In: *arXiv preprint arXiv:1807.06906*.

Zhang, Chris, Mengye Ren, and Raquel Urtasun (2018). "Graph hypernetworks for neural architecture search". In: *arXiv preprint arXiv:1810.05749*.

Zhang, Yuge, Quanlu Zhang, and Yaming Yang (2020a). "How Does Supernet Help in Neural Architecture Search?" In: *arXiv e-prints*, arXiv:2010.08219, arXiv:2010.08219. arXiv: 2010.08219 [cs.LG].

Zhang, Yuge et al. (2020b). "Deeper Insights into Weight Sharing in Neural Architecture Search". In: *Submitted to International Conference on*

Learning Representations. rejected. URL: https://openreview.net/forum?id=ryxmrpNtvH.

Zheng, Qiuhan et al. (2021). "Artificial intelligence performance in detecting tumor metastasis from medical radiology imaging: A systematic review and meta-analysis". In: *EClinicalMedicine* 31, p. 100669.

Zhong, Zhao et al. (2020). "Blockqnn: Efficient block-wise neural network architecture generation". In: *IEEE transactions on pattern analysis and machine intelligence* 43.7, pp. 2314–2328.

Zhong, Zhun et al. (2020). "Random erasing data augmentation". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 07, pp. 13001–13008.

Zhou, Dongzhan et al. (2020). "Econas: Finding proxies for economical neural architecture search". In: *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pp. 11396–11404.

Zoph, Barret and Quoc V. Le (2017a). "Neural Architecture Search with Reinforcement Learning". In: *Proc. of ICLR*. OpenReview.net. URL: https://openreview.net/forum?id=r1Ue8Hcxg.

— (2017b). "Neural Architecture Search with Reinforcement Learning". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. URL: https://openreview.net/forum?id=r1Ue8Hcxg.

Zoph, Barret et al. (2018a). "Learning Transferable Architectures for Scalable Image Recognition". In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, pp. 8697–8710. DOI: 10.1109/CVPR.2018.00907. URL: http://openaccess.thecvf.com/content%5C_cvpr%5C_2018/html/Zoph%5C_Learning%5C_Transferable%5C_Architectures%5C_CVPR%5C_2018%5C_paper.html.

— (2018b). "Learning Transferable Architectures for Scalable Image Recognition". In: *Proc. of CVPR*. IEEE Computer Society, pp. 8697–8710. DOI: 10.1109/CVPR.2018.00907. URL: http://openaccess.thecvf.com/content%5C_cvpr%5C_2018/html/Zoph%5C_Learning%5C_Transferable%5C_Architectures%5C_CVPR%5C_2018%5C_paper.html.