Politecnico di Milano
Corso di laurea Magistrale in Computer Science and Engineering
Dipartimento di Elettronica, Informazione e Bioingegneria

POLITECNICO
MILANO 1863

Multimodal Deep Learning for Prediction of Postoperative Complications in Cardiac Surgery

Relatore:
Prof. Pier Luca LANZI

Tesi di Laurea di:
Giada CONFORTOLA
Matr. 898540

Anno Accademico 2018-2019

Acknowledgements

First of all, I would like to thank Prof. Pier Luca Lanzi for supervising me throughout the work.

Moreover, I would like to thank the German Heart Center Berlin (DHZB) and the Fraunhofer MEVIS for allowing me to work on this project. In particular my thankfulness goes to Prof. Dr.-Ing. Anja Hennemuth and Dr. Alexander Meyer, and for advising and guiding me during the project development.

In addition, I would like to express my gratitude to Boris Pfahringer, that always found time to help me and answer my questions.

I would also like to thank EIT Digital, the Polytechnic University of Milan and the Technical University of Berlin for giving me the opportunity to study and live in two different countries and connect with people from all over the world.

I would like to thank all my friends and colleagues that I met during the studies, that made this path more pleasant. In particular, my gratefulness goes to Luca Comoretto, for his constant support and his precious theoretical explanations over the phone.

In conclusion, I must express my thankfulness to my family, especially to my parents and grandparents, for their constant and unconditional support throughout these years.

This accomplishment would not have been possible without all of them.

Thank you.

Milan, April 2020 G. C.

To my brother

Abstract

Chest radiographs are the most common imaging exam and are used for both diagnosis and monitoring of different diseases. When a patient undergoes a cardiac surgery, it is routine practice to obtain a chest X-ray right after the operation, that can support physicians in the detection of possible complications due to the surgery or the anesthesia. Together with the radiograph, vital parameters, that are collected at regular intervals in the intensive care unit, are checked by physicians to determine whether the patient's state changes over time.

Deep learning methods can be exploited to support specialists in interpreting chest X-rays at the point of care. Similarly, they can also support them in understanding the patient's state from the monitored parameters, which are produced at a fast rate and can be difficult to process by the human brain, that is inclined to miss part of the information.

A recurrent neural network was used in previous work to predict complications in the 24 hours following a cardiac surgery, by considering vital parameters, such as blood pressure, monitored at a time interval of 30 minutes [1]. The goal of this thesis is to investigate whether a multimodal approach, considering both the monitored data used in [1] and a chest radiograph that is taken after the operation can improve the prediction of complications. In particular, this work focuses on postoperative bleeding, a complication that requires a re-exploration surgery, and should be detected as early as possible.

Different integration strategies are examined. The results of each experiment are compared with the results obtained by the model considering only the monitored temporal parameters to verify whether images could actually bring additional information to the model.

The outcomes show that the multimodal models, that include the chest radiographs, performs better than the baseline, which is the model considering only the monitored parameters. Overall, the multimodal model outperforms the simple RNN, with an accuracy of 83.86 % and a ROC AUC score of 88.88%, which represents an absolute improvement of respectively 5.91 % and 3.19%.

Keywords: Chest Radiograph, Critical Care, Deep Learning , Multimodality

Sommario

Le radiografie toraciche sono l'esame visuale più comune per la diagnosi ed il monitoraggio di malattie di vario tipo. Quando un paziente subisce un intervento chirurgico, una x-ray del torace viene fatta di routine dopo l'operazione; essa può essere di supporto ai medici nel rilevamento di possibili complicanze dovute all'operazione o all'anestesia. Insieme alla radiografia, i parametri vitali, monitorati ad intervalli regolari in terapia intensiva, sono controllati dai dottori per determinare se lo stato del paziente cambia nel tempo.

I metodi di apprendimento profondo (deep learning) possono essere sfruttati per supportare gli specialisti nell'interpretazione delle radiografie del torace. Allo stesso modo, tali metodi possono aiutare gli esperti nel comprendere lo stato del paziente attraverso i parametri monitorati, che sono spesso prodotti ad un elevato tasso di velocità e possono essere difficili da processare dall' uomo, che è incline a perdere parte dell'informazione.

Una rete neurale ricorrente è stata utilizzata in uno studio precedente per predire le complicanze nelle 24 ore che seguono un intervento cardiaco, considerando parametri vitali (es. pressione sanguigna) monitorati ogni 30 minuti [1]. L' obiettivo di questa tesi è quello di investigare se un approccio multimodale, che considera i dati usati in [1], uniti ad una lastra toracica del paziente fatta dopo l'operazione, possa migliorare la predizione di complicanze. In particolare, questo lavoro si focalizza sul sanguinamento postoperatorio, una complicanza che richiede un'operazione riesplorativa, e che va diagnosticata il prima possibile.

Sono state considerate diverse strategie di integrazione. I risultati di ciascun esperimento sono stati comparati con quelli ottenuti dal modello che considera soltanto i parametro temporali monitorati, per verificare se le immagini potessero portare informazioni aggiuntive al modello.

I risultati mostrano che il modello multimodale, che include le lastre toraciche, è più performante di quello originale. Nel complesso, il modello multimodale supera la rete neurale ricorrente, con una accuratezza dell'83,86% ed un punteggio ROC AUC dell'88,88%, che rappresenta un miglioramento assoluto rispettivamente del 5,91% e del 3,19%.

Parole Chiave: Radiografia Toracica, Terapia Intensiva, Deep Learning, Multimodalità

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Patients that undergo major open-heart surgery, such as aortic surgery, coronary artery bypass grafting, and heart or lung transplants, are subject to different complications in the hours following the surgery. When a complication occurs, the time between the discovery and the treatment can be crucial for the patient's outcome. For this reason, patients are transferred to the intensive care unit (ICU) right after the surgery, where their vital parameters, such as blood pressure and O_2 saturation, are monitored constantly. Together with them, laboratory values and other patient's information are available to nurses and doctors to monitor the patient's status and eventually diagnose a complication. However, the overwhelming amount of data that is produced for each patient can be difficult to interpret for the human brain, that may miss part of the available information when evaluating the patient's state.

Machine learning (ML) methodologies are particularly tailored for tasks in which data are produced at a fast rate. Neural networks (NN) are a category of ML models that have become very popular in recent years thanks to the demonstrated success on image and sequential data (such as text) processing. In particular, a recurrent neural network (RNN) is a model able to process sequences of data, by taking into account for each time step, not only the current values but also information from the past. This characteristic makes RNNs perfect for processing patient's data, which are collected at different time steps.

The clinical parameters recorded in the electronic health record at a time interval of 30 minutes, together with some patients' characteristics such as age and sex, were considered by Meyer et al. to build a recurrent neural network for real-time prediction of complications in critical care [1]. This work is described in detail in section 4.2.

Together with periodically monitored parameters, it is routine clinical practice

to obtain a chest radiograph of the patient right after the surgery, which may give additional information to physicians regarding the patient's status.

Chest X-rays are the most common medical imaging exam worldwide because they can be highly informative and can be used by doctors both to diagnose and to monitor the patient's status in different medical scenarios. This characteristic of chest radiographs makes it easy to collect big datasets of samples.

Having datasets that are big enough to train a neural network is crucial to obtaining robust models and is one of the most common issues when ML is applied to medical data, as large datasets are usually complex to retrieve due to heterogeneity of medical systems and privacy regulations protecting them.

Chest radiographs, used by physicians to support their diagnosis on patients treated in the ICU, could potentially bring additional information to the machine learning model that processes the sequential vital parameters for prediction of postoperative complications.

1.2 Objective

This work aimed to develop a multimodal deep learning model that processes temporal clinical parameters recorded at a time interval of 30 minutes on patients that have undergone a cardiac surgery, together with the chest radiograph obtained after the procedure, to predict possible complications in the 24 hours following the surgery. Different integration strategies were adopted and compared.

Furthermore, a comparison between the integrated models and the original unimodal model was carried out to investigate whether integrating images into the original RNN model could bring more information to it and increase performance, in terms of accuracy of the model and area under receiver operating characteristic curve (AUC).

A particular focus was also placed on the solutions adopted to overcome the difficulties of implementing an image processing model with a relatively small dataset, by exploiting transfer learning, a techniques that takes advantage of big available sets of data to train a model and then translates the information to another model built on smaller datasets.

The previous work [1] considered three complications as targets: postoperative bleeding, renal failure and mortality. This thesis focused on one complication in detail: postoperative bleeding.

1.3 Thesis Outline

Chapter 2 is meant to give the reader the background needed to understand the work carried on, by introducing the main deep learning concepts. Moreover, the concept of multimodality is presented. Finally, an overview of evaluation metrics for classification models is given.

Chapter 3 gives an overview of traditional computer techniques for chest radiographs processing. The chapter then continues by presenting the current state of the art for chest x-rays analysis, focusing on x-ray classification methodologies.

Chapter 4, contextualizes the work done in this thesis in the medical domain, by explaining the main concepts and presenting in detail the work done previously.

In Chapter 5, the image classification model, developed to process chest radiographs is presented.

Chapter 6 presents the techniques used to integrate the models, the results obtained and the comparison with the original RNN model.

Finally, Chapter 7 gives a conclusion and proposes ideas for potential future work.

Chapter 2

Fundamentals and Theoretical Background

This chapter gives an overview of the necessary theory and the state of the art techniques used to tackle the task under consideration.

In particular, section 2.1 introduces the main concepts of deep learning. In section 2.2, convolutional neural networks, which are considered the state of the art for medical imaging classification, are described in detail.

Section 2.3 presents the evaluation process and the most common metrics used for image classification models.

Section 2.4 explains the concept of transfer learning, while section 2.5 illustrates techniques commonly used to integrate models that process multiple inputs.

2.1 Deep Learning

Deep learning (DL) is a sub-field of machine learning that comprises a set of methods based on artificial neural networks and used in different applications, such as natural language processing and computer vision.

2.1.1 Artificial Neural Networks

The concept of artificial neural network (ANN), also known as simply neural network (NN), comes from the idea of modeling biological neural systems. A feed-forward neural network, which is the simplest ANN, is a directed acyclic graph, whose arcs, which are vaguely inspired by the synapses in human brains, connect simple computational units, the neurons, which are distributed in layers. A simple fully connected neural network is represented in Figure 2.1.

A neuron (Figure 2.2) is a computational unit that takes multiple inputs, coming

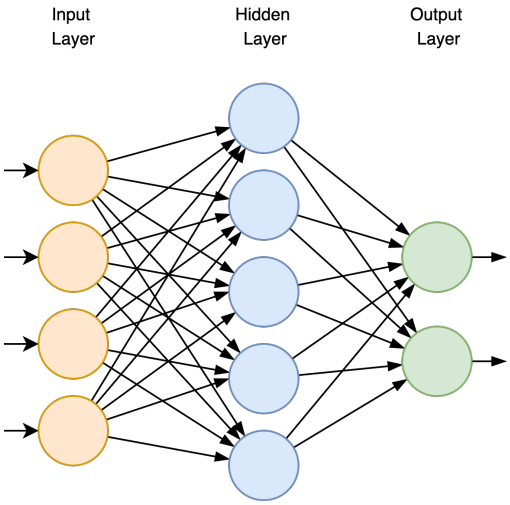


Figure 2.1: Structural representation of a simple fully connected artificial neural network with the input layer, one hidden layer and the output layer

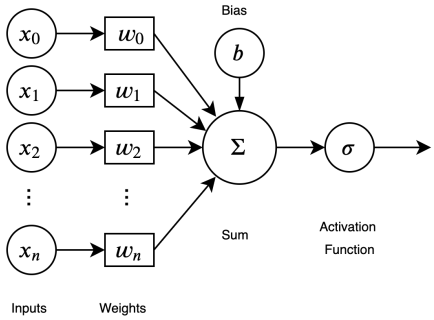


Figure 2.2: Schematic overview of a neuron model, the basic component of an artificial neural network

from the units in the previous layer, each of them with a different weight, and returns one output. The typical function used to produce the output is the following:

$$\hat{y} = \sigma(\sum_i w_i x_i + b) \tag{2.1}$$

Where σ is a nonlinear activation function, w_i is the parameter of the input x_i and b is the bias of the neuron.

Activation Functions

Activation functions are nonlinear functions applied to the result of the linear equation at each neuron of the network. They are essential, as without them the network would just be a sequence of linear models, thus still a linear model.

Common activation functions are:

- **Sigmoid**: the sigmoid non-linear function takes a real value as input and

squishes it in the range (0,1). In particular, large negative numbers will assume value 0, while large positive numbers in input will return 1. The sigmoid function is the following:

$$\sigma(x) = \frac{1}{(1 + e^{-x})} \tag{2.2}$$

One undesirable property of the sigmoid function is that when the neuron's activation saturates at either 0 or 1, the value of its gradient is almost 0, making almost no information flowing through the neuron. For instance, if very high weights are set at the beginning, most of the neurons will saturate and the network will barely learn.

- **Hyperbolic tangent**:

$$\sigma(x) = tanh(x) \tag{2.3}$$

it is similar to the sigmoid function but its outputs are zero-centered. This function has again the saturation problem.

- **Rectified linear unit (ReLU)**: it is probably the most used activation function. It thresholds the activation at 0:

$$\sigma(x) = max(0, x) \tag{2.4}$$

As opposed to the first two activation functions, the ReLU does not saturate. The main problem with ReLU is that the output of the function is set to 0 for each negative input, thus ReLU units can irreversibly "die" during the training. A variant of the original ReLU, called Leaky ReLU was proposed to solve the dying ReLU problem. In this alternative form, instead of the function being zero for x<0, there is a small negative slope.

- **Softmax**: the softmax function is the generalization of the sigmoid function in case of multi-class problem. It is typically used as activation function of the last layer in multi-class or multi-label problems, to map the inputs with the labels to predict. The softmax function returns for each input a vector of values between 0 and 1, representing the probabilities that the input belongs to each class. The function is the following:

$$\sigma(x) = \frac{e^{x}}{\sum_{k=1}^{K} e^{x_k}} \tag{2.5}$$

where K is the number of classes (or labels) considered, x_K represents the input value for class K, while x represents the input value for the class under consideration.

Figure 2.3 shows the graphical representation of three sigmoid functions presented above.

Optimization Algorithms

The training process of a neural network consists in finding the best values for the network's parameters. In a supervised learning scenario, for which the true labels for each input are available, the goal of the training process is to estimate the network's weights so that the outputs are the closest possible to the original targets.

To measure the difference between true and predicted values, an error function $L(\boldsymbol{w})$ is considered. Different loss functions can be considered, depending on the task.

In image classification, the cross-entropy loss is a common choice. In the binary case, the cross-entropy loss is defined as follows:

$$L(\boldsymbol{w}) = -\sum_{n=1}^{N}(y_n(ln(\hat{y}_n)) + (1 - y_n)ln(1 - \hat{y}_n)) \tag{2.6}$$

where y_n and $\hat{y}_n$ are respectively the target value and the predicted probability for a sample n.

The training process starts with the input flowing through the network until the target output is generated. This phase is called *forward propagation*. At this point, the error is computed.

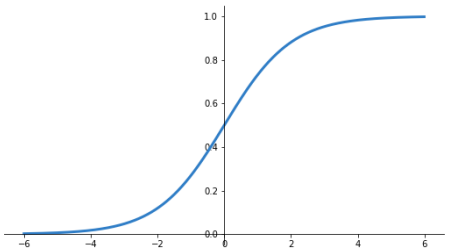
The second phase of the training process is called *backward propagation*. During it, the weights of each neuron are updated by performing a backward pass in the network. The update rule adopted varies on the optimization strategy considered. The most common optimization algorithms are:

- **Gradient descent (GD)**: considers the gradient of the error function to move the weight vector in the direction of the greatest rate of decrease of the error function. At each step, the weight vector $\boldsymbol{w}$ is updated as follow:

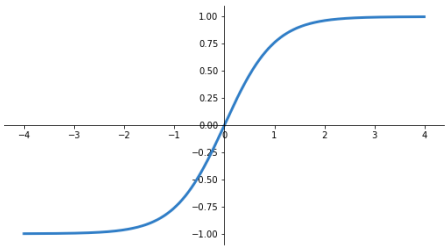
$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \alpha\nabla L(\boldsymbol{w}) \tag{2.7}$$

  where the parameter $\alpha > 0$ is called learning rate (LR) and represents the size of the step taken by the algorithm.

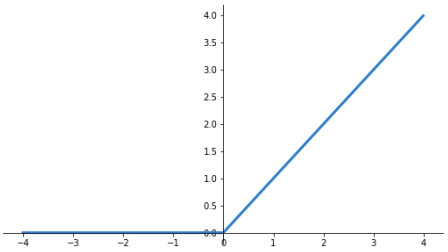
  The traditional algorithm, also called Batch gradient descent, considers the whole train data to compute the gradient.



Sigmoid



Hyperbolic Tangent



ReLU

Figure 2.3: Graphical representation of three popular activation functions

- **Stochastic gradient descent (SGD)**: is a variant of the GD algorithm that considers a single data point to compute the update, instead of the whole dataset. It is much faster than computing the gradient on the whole dataset, but performs frequent updates with high variance, resulting in the objective function to highly fluctuate. For this reason, it shows complications in converging to the exact minimum.

- **Mini-batch gradient descent**: Combines the advantages of the two previous approaches by considering mini-batches of data (commonly between 32 and 256), so that the algorithm leads to a more stable convergence, without being as slow as traditional GD. This is the approach usually adopted when training a NN.

- **SGD with momentum**: One of the main issues encountered when using GD is that the algorithm may get trapped in a sub-optimal local minima; this happens especially when dealing with highly non-convex error functions, that sometimes present areas around a local optima, where the surface curves are much steeper in one direction than in another. In this scenario, GD oscillates across the slopes while only making small progresses towards the minimum.

  Momentum is a method that helps accelerate SGD in the relevant direction by adding a velocity term to the original equation, as follows:

$$\boldsymbol{v}_{t+1} = \rho \boldsymbol{v}_t + \nabla L(\boldsymbol{w}_t) \tag{2.8}$$

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \alpha \boldsymbol{v}_{t+1} \tag{2.9}$$

The momentum term increases for dimensions whose gradient point in the same direction and reduces updates for dimensions whose gradient change directions, thus allowing to achieve faster convergence and reduced oscillation.

- **Adagrad**: adapts the learning to each parameter, performing larger updates for infrequent parameters and smaller for frequent ones. The main weakness of Adagrad is that at each update the learning rate becomes smaller and smaller until it is actually annealed. For this reason, convergence to the minima is not guaranteed. Adadelta and RMSProp are variants of the algorithm that solve the issue by considering the running average of past squared gradients.

- **Adam**: the adaptive moment estimation (Adam) algorithm is another method that computes adaptive learning rates for each parameter. It stores both an exponentially decaying average of past squared gradients v_t (as in RMSProp and Adadelta) and an exponentially decaying average of past gradients $\boldsymbol{m_t}$, (similar to momentum):

$$\boldsymbol{m}_t = \beta_1 \boldsymbol{m}_{t-1} + (1 - \beta_1)\boldsymbol{g}_t \tag{2.10}$$

$$\boldsymbol{v}_t = \beta_2 \boldsymbol{v}_{t-1} + (1 - \beta_2)\boldsymbol{g}_t^2 \tag{2.11}$$

where $\boldsymbol{g_t}$ is the gradient of the loss function $L(\mathbf{w})$, and β_1 and β_2 are hyperparameters that weight the importance of previous steps in the running averages. Since $\boldsymbol{m_t}$ and $\boldsymbol{v_t}$ are initialized as vectors of 0s, they are biased towards 0, especially in the first iterations and when the decay parameters are small. For this reason, a bias-corrected version of them is considered in the weights update:

$$\hat{\boldsymbol{m}}_t = \frac{\boldsymbol{m}_t}{1 - \beta_1^t} \tag{2.12}$$

$$\hat{\boldsymbol{v}}_t = \frac{\boldsymbol{v}_t}{1 - \beta_2^t} \tag{2.13}$$

The operation done to update the parameters is the following:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \frac{\alpha}{\sqrt{\hat{\boldsymbol{v}}_t} + \varepsilon}\hat{\boldsymbol{m}}_t \tag{2.14}$$

The authors of the paper propose default values of 0.9 for β_1, 0.99 for β_2 and 10^{-8} for ε.

2.1.2 Recurrent Neural Networks

Recurrent neural networks (or RNNs) are a class of ANNs widely used in Natural Language Processing applications and when dealing with temporal data, as they are specialized for processing sequences of values. They are based on the idea of sharing parameters across different parts of the model. To do so, they introduce directed cycles in the network, which allow taking into account the dynamic temporal behavior of sequences.

The typical input of an RNN is a sequence of feature vectors. To process the sequence, the network is unfolded into t timesteps, where t represents the length of the sequence.

Considering for simplicity a single recurrent layer, that is represented by 2 matrices $\boldsymbol{U}$ and $\boldsymbol{W}$, the status vector $\boldsymbol{h}_t$ is computed by considering both the current input vector $\boldsymbol{x}_t$ and the vector state at the previous timestep $\boldsymbol{h}_{t-1}$ as follows:

$$\boldsymbol{h}_t = f(\boldsymbol{U}^T\boldsymbol{x}_t + \boldsymbol{W}^T\boldsymbol{h}_{t-1} + \boldsymbol{b}) \tag{2.15}$$

where f is a non-linear function, such as ReLU, $\boldsymbol{b}$ is the bias vector and the first state $\boldsymbol{h}_0$ is initialized with zeros.

Figure 2.4 shows the unfolded structure of a basic RNN.

Long Short Term Memory (LSTM)

Traditional RNNs have problems when dealing with long-term dependencies. When the gradient is propagated through many stages, it usually either vanishes or (more rarely) explodes. Gated RNNs were proposed to solve the issue. The key idea behind them is to create paths through time with derivatives that don't vanish or explode.

Long short term memory networks (shortened as LSTMs), are a variation of traditional RNNs that substitute the traditional cell with gated cells, which are capable of learning long-term dependencies.

Other than the hidden state $\boldsymbol{h}_t$, LSTMs introduce the cell state $\boldsymbol{C}_t$, represented by the horizontal line on top of the diagram in Figure 2.5. The cell state runs through the entire chain with only some minor linear interactions.

Information carried out by the cell state is regulated by structures called gates. LSTM cells are usually composed of three gates:

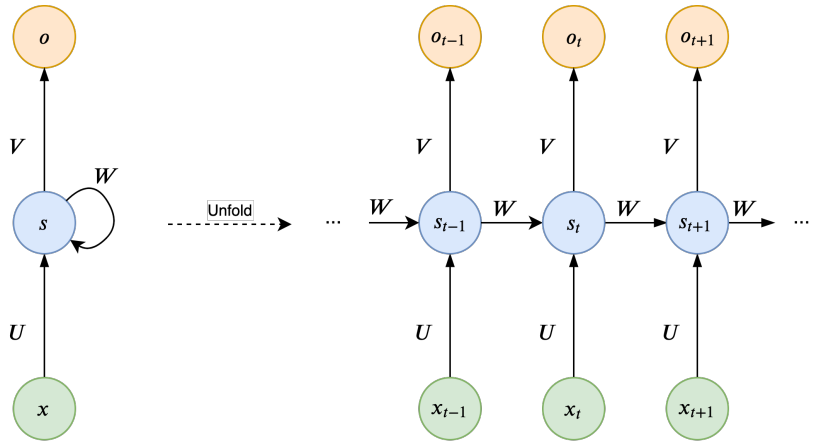


Figure 2.4: Unfolded structure of a simple recurrent neural network

- **Forget gate**: responsible to decide what information should be kept from the cell state coming from the previous cell. To determine what to preserve from the previous cell state, a sigmoid layer is applied to the hidden state $\boldsymbol{h}_{t-1}$ and the new input $\boldsymbol{x}_t$. The result of this operation is multiplied by the cell state $\boldsymbol{C}_{t-1}$ to decide what can be forgotten and what not.

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_f) \tag{2.16}$$

- **Input gate**: determines what new information to store in the cell state. This gate is composed of two parts. First, sigmoid is applied to the input and previous hidden state. Next, the tanh layer is applied to the same inputs to create a candidate value $\widetilde{\boldsymbol{C}}_t$. The two layers are combined by pointwise multiplication and summed to the Cell state:

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_i) \tag{2.17}$$

$$\widetilde{\boldsymbol{C}}_t = tanh(\boldsymbol{W}_C[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_C) \tag{2.18}$$

$$\boldsymbol{C}_t = \boldsymbol{f}_t * \boldsymbol{C}_{t-1} + \boldsymbol{i}_t * \widetilde{\boldsymbol{C}}_t \tag{2.19}$$

- **Output gate**: returns a filtered version of the cell state, that defines the output of the network at the considered timestep. Again, a sigmoid layer is applied to the input and previous hidden state to define what part of the cell state to output. Values of the cell state $\mathbf{C_t}$ are pushed between -1 and +1 using tanh and then filtered by multiplying it with the output of the sigmoid.

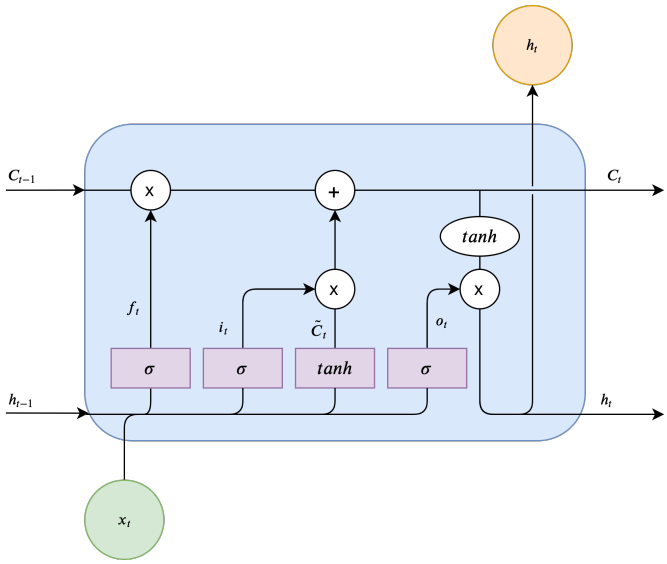


Figure 2.5: Long short term memory cell structure

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_o) \tag{2.20}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t * tanh(\boldsymbol{C}_t) \tag{2.21}$$

Gated Recurrent Unit (GRU)

Gated recurrent units (Figure 2.6), also known as GRUs, are an alternative to LSTMs. They only implement two gates inside cells, thus decreasing the number of parameters of the model.

The gates are:

- **Update gate**: sigmoid layer is applied to the previous hidden state and current input:

$$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_z[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_z) \tag{2.22}$$

- **Reset gate**: determines the contribution of previous hidden state $\boldsymbol{h_{t-1}}$ at time t. It is defined by:

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_r) \tag{2.23}$$

From the two gates, the candidate input state is obtained by:

$$\boldsymbol{c}_t = tanh(\boldsymbol{W}[\boldsymbol{r}_t * \boldsymbol{h}_{t-1}, \boldsymbol{x}_t]) \tag{2.24}$$

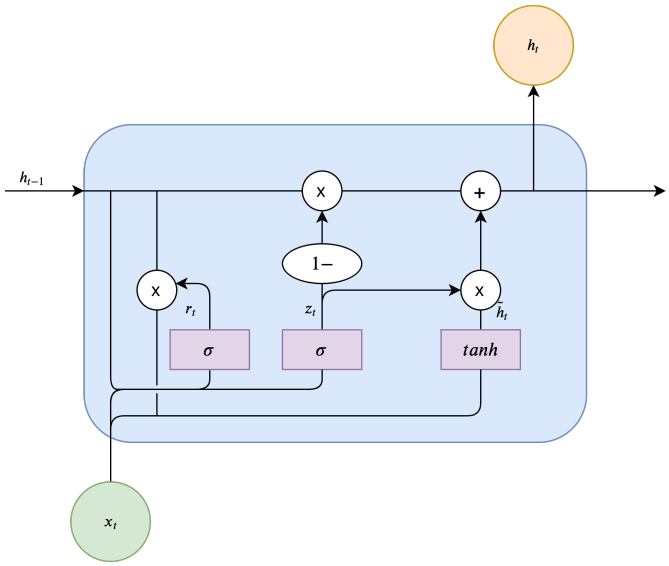


Figure 2.6: Gated recurrent unit cell structure

and the hidden state is the following:

$$\boldsymbol{h}_t = (1 - \boldsymbol{z}_t) * \boldsymbol{h}_{t-1} + \boldsymbol{z}_t * \boldsymbol{c}_t \tag{2.25}$$

2.2 Convolutional Neural Networks

A convolutional neural network (CNN) is a particular NN used to process data that have a grid-like topology (Figure 2.7). They are typically used in computer vision tasks, such as image classification and segmentation.

When considering an image classification task, one simple approach would be to treat images as sequences of features and treat them as inputs of a fully connected network. This approach, however, brings two main issues:

- When processing images, the number of features in input is generally much higher as compared to when dealing with tabular data. For this reason, unless the input size is reduced, the number of parameters in the model is too high and the model may be impossible to train.

- Considering the image as a sequence of pixels would mean ignoring the high correlation between nearby pixels and potentially lose important information.

CNNs introduce two new operations, convolution and pooling, that are specifically designed to deal with grid-structured data. Fully connected layers are typically added at the end of the network to get the final probabilities.

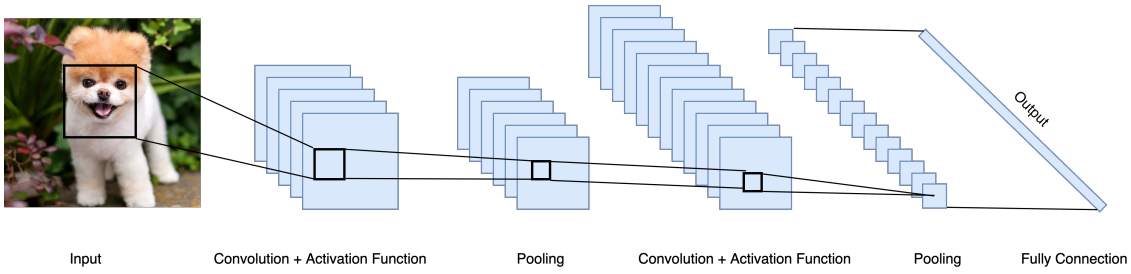


Figure 2.7: Schematic overview of a convolutional neural network

2.2.1 Convolution Layer

The convolution operation is the core building block of the CNN architecture. The input image is represented by a tensor (a multi-dimensional matrix) of size *width*height*depth*. At the first layer, the depth of the tensor corresponds to the number of channels of the images. RGB images have 3 channels, while black and white images 1.

A second tensor, the kernel (or filter) contains the learnable parameters of the model. The filter has a smaller width and depth as compared to the input image but extends to the full depth. A typical kernel size is *3*3*depth*.

During the forward pass, the filter is convolved (slided) across the width and height of the input, and the dot product between the entries of the filter and the values of the input under consideration is computed. The results obtained by the convolution operation, for a certain number of filters n, are stacked to produce the output volume. Figure 2.8 shows how the convolution operation works in a 2D scenario.

The dimension of the output feature map depends on three parameters:

- **Number of filters** : defines the depth of the output. It is the number of filters that will be applied to the input in the convolutional layer.
- **Stride**: represents the number of steps that the kernel will perform from a convolution to the next one. The lower it is, the higher the overlapping between the receptive fields of the convolutions.
- **Zero-padding**: allows to control the size of the feature map, by padding the original image with 0s, to obtain an output with the same width and height of the input.

The output size can be computed as follows:

$$O = \frac{N + 2P - D}{S + 1} \tag{2.26}$$

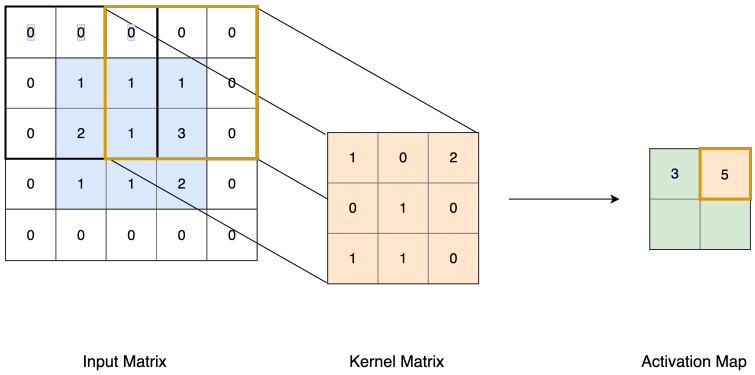


Figure 2.8: Example of the convolution operation on a 3x3 input with zero-padding and stride=2

where N is the input size, P is the size of padding, D is the size of the receptive field and S is the stride.

Activation functions, such as ReLUs, are placed after convolutional layers to add non-linearity to the model.

2.2.2 Pooling Layer

Pooling layers are usually stacked after convolutional layers to reduce the size of the feature maps and thus the number of parameters in the model.

Reducing the number of parameters in the network is necessary both to be able to train the model and to prevent overfitting, which is the situation in which the model built is performing particularly good on the training instances, but it is not able to generalize on unseen samples, because it is excessively adapted to the original data.

Pooling layers apply filters to the tensor they get as inputs and summarize the values considered by the filter in some way. Different operations can be considered to summarize the values; the most common ones are average and max operations. An example of max-pooling operation is shown in Figure 2.9.

2.2.3 Batch Normalization

Batch normalization (BN) layers can be placed after convolutional layers to reduce the internal covariate shift of the network, which is the change in the distribution of network activations due to the change in network parameters during training. The input in a BN layer is normalized by subtracting the batch mean and dividing by its standard deviation.

Adding BN layers to the network brings different advantages:

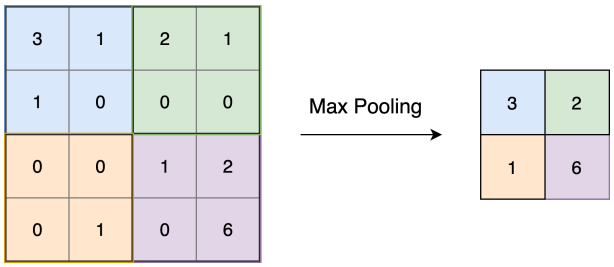


Figure 2.9: Example of the max-pooling operation on a 4x4 input and a 2x2 filter

- **Train speed-up**: such kind of normalization usually speeds up the convergence to the optimum.
- **Higher learning rates**: normalization of the activations throughout the network prevents small changes in the parameters from amplifying into sub-optimal changes in activations in gradients.
- **Model regularization**: BN better generalizes the network, as a single training example is seen in conjunction with other samples, thus avoiding predicting deterministic values for a given training point.

2.2.4 DenseNet

The first CNN architectures used for image processing tasks were composed of a few layers. In general, adding layers to the networks leads to some issues, since when information passes through many of them, it can vanish before reaching the end of the network. All the approaches presented to deal with this problem share the same key concept: create short paths from early layers to later ones to avoid information to vanish.

The core idea behind DenseNet is to concatenate layers within blocks. Each layer receives the feature maps of the previous layers and passes its own to the subsequent ones. Features are never summed (in contrast to other architectures such as ResNet [2]). For this reason, DenseNet requires fewer parameters, as there is no need to relearn feature maps coming from previous layers. DenseNet architecture also differentiates between information that is added to the network and information that is preserved. Figure 2.10 presents the general architecture of a DenseNet.

DenseNet presents two main building blocks (Figure 2.11) :

- **Dense block**: composed of multiple Dense layers. A direct connection from any layer to all subsequent layers is introduced to allow the information to flow. The single dense layers present a 1x1 convolution, also called the bottleneck layer, that precedes the 3x3 convolution. The bottleneck layer is introduced to

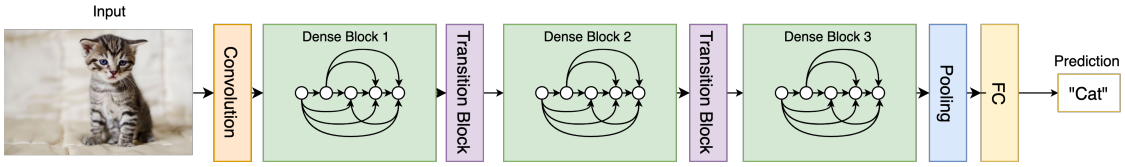


Figure 2.10: Example of a deep DenseNet with 3 dense blocks

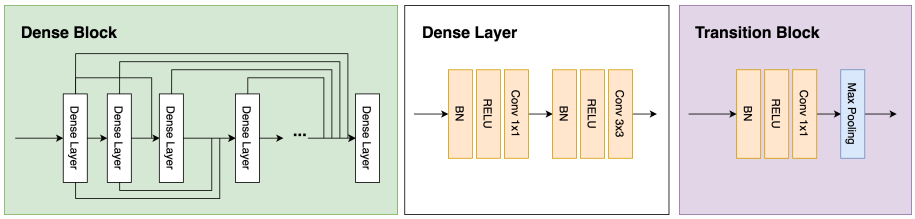


Figure 2.11: Details of the blocks composing the DenseNet architecture

reduce the number of input features and thus to improve the computational efficiency.

- **Transition block**: it is placed between two dense blocks for down-sampling. For this reason, it is composed of a convolution, followed by a pooling layer.

2.3 Classification Models Evaluation

The objective of this section is to illustrate how the performances of an image classification model are evaluated.

Dataset Split

Given a set of data, it is easy to build a model able to perform well on that set. However, the final goal of an image classification task is to be able to build a model that generalizes on unseen instances. For this reason, the initial data available is usually split into three distinct datasets: the training, the validation, and the test set.

The first set is the one that is used to train the network and optimize its parameters via backpropagation. A second set, the validation set, is kept to obtain an unbiased evaluation of the model fitted on the training set while tuning the model's hyperparameters, which are parameters which value is set before the training process (for instance the number of layers of the network or the learning rate).

When the model is a neural network, which is trained for a certain number of epochs, the validation dataset can also be used to determine at what epoch to stop the training process (Early stopping). This technique is useful to detect the point in

time in which the model stops learning and starts overfitting the training set and consists of stopping the model training when the validation loss reaches its minimum before starting to increase.

Finally, the test set is used to compute the metrics and evaluate the final model on unseen data. Having a dataset that was not used during the model fitting is essential, as it allows to obtain an unbiased evaluation of the model.

The percentage of instances that end up into each set may vary, depending on the problem under consideration. Generally, the training set is the one that contains the majority of the samples, since the robustness of the model highly depends on the amount of data used for training it. The remaining two sets usually contain only a small portion of them. A common split is 80% for the training set and 10% for each of the remaining sets.

Evaluation Metrics

Evaluation metrics are a set of measures used to assess the model's performances on an unseen set of instances.

The confusion matrix, represented in Figure 2.12, is a table that summarizes the performance of the classification task. Four categories can be identified in the matrix. True positive (TP) refers to those images that were labeled as positive samples by the model and are actually positive. Similarly true negatives (TN) are images correctly labeled as negative.

The other two categories, false positives (FP) and false negatives (FN) represent images wrongly labeled. FP are negative samples classified as positive by the model, while FN are positive images classified as negative.

A binary classification model outputs the probability of a certain sample to belong to the positive class; a threshold is then used to decide whether to label the instance as positive or negative. A common threshold is 0.5, but it can be set to higher or lower values, based on the importance to avoid FP or FN according to the task.

For instance, a higher threshold can be chosen when it is desirable to label an instance as positive with a higher confidence (thus diminishing the false positives).

From the confusion matrix, a variety of metrics that help understanding how good is the model considered, can be computed:

- **Accuracy**: Most widely used metric, it measures the percentage of correctly classified instances:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.27}$$

  Although it is the simplest and the most explainable metric, it may be misleading and is not suitable when the dataset considered is highly unbalanced

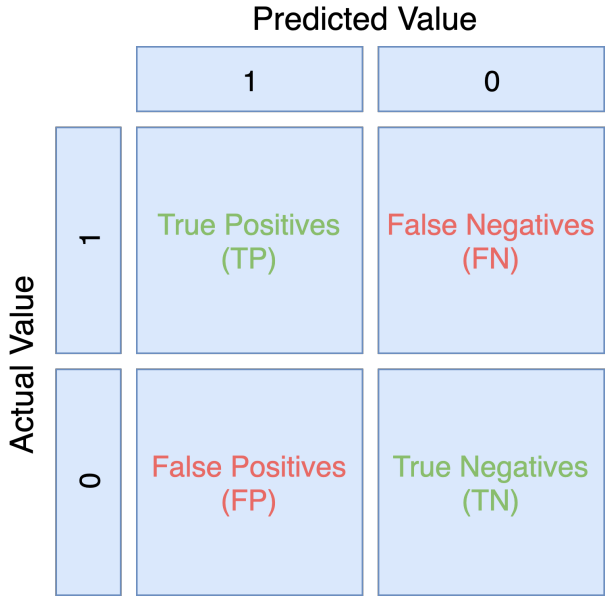


Figure 2.12: Confusion matrix

toward one class, as a model predicting only the class that is most present would have a high accuracy even if the performances would clearly not be good.

- **Precision**: measures the percentage of samples, among the ones that are classified as positives, that are actually positive:

$$Precision = \frac{TP}{TP + FP} \tag{2.28}$$

- **Recall**: represents the fraction of correctly classified as positive examples among the total number of positive samples in the dataset:

$$Recall = \frac{TP}{TP + FN} \tag{2.29}$$

- **F1 Score**: can be considered as a summary of precision and recall. It is computed as the harmonic mean of precision (p) and recall (r):

$$F1 - Score = \frac{2pr}{r + p} \tag{2.30}$$

The harmonic mean is considered instead of the arithmetic mean to penalize results with high precision and low recall and vice-versa.

The previous metrics all require a threshold to be set. The following ones, on the other hand, give a more general evaluation by considering the threshold varying from 0 to 1:

- **Area under curve (AUC)**: It measures the area under the receiver operating characteristic Curve (ROC). The ROC curve plots the true positive rate (TPR), which is the recall, against the false positive rate (FPR), that is computed as $FPR = \frac{FP}{TN+FP}$, while threshold varies between 0 and 1.

- **Precision-recall curve (PR-curve)**: The precision is plotted as a function of the recall, for varying threshold values. The optimal classifier would have always precision 1, independently from the recall value. The area under the PR-curve (AUPRC) can be considered as a metric.

Both curves can be helpful to visualize the results obtained; however, the ROC curve is more suitable when the classes are balanced, while the PR-curve can be highly informative also in case of imbalanced datasets.

An example of ROC and precision-recall curves can be seen in Figure 2.13.

All metrics described above can assume values between 0 and 1.

2.4 Transfer Learning

The sensitivity of clinical data makes it challenging to collect big labeled datasets, which are crucial to obtain machine learning models able to generalize on unseen instances. When the dataset used to train the NN is not big enough, it is likely to incur in overfitting, thus building a model that performs almost perfectly on the training set, but whose performances are really bad on the unseen samples in the test set.

Transfer learning is a widely used technique to prevent overfitting, especially when dealing with image models, which have millions of parameters and are more likely to overfit small datasets.

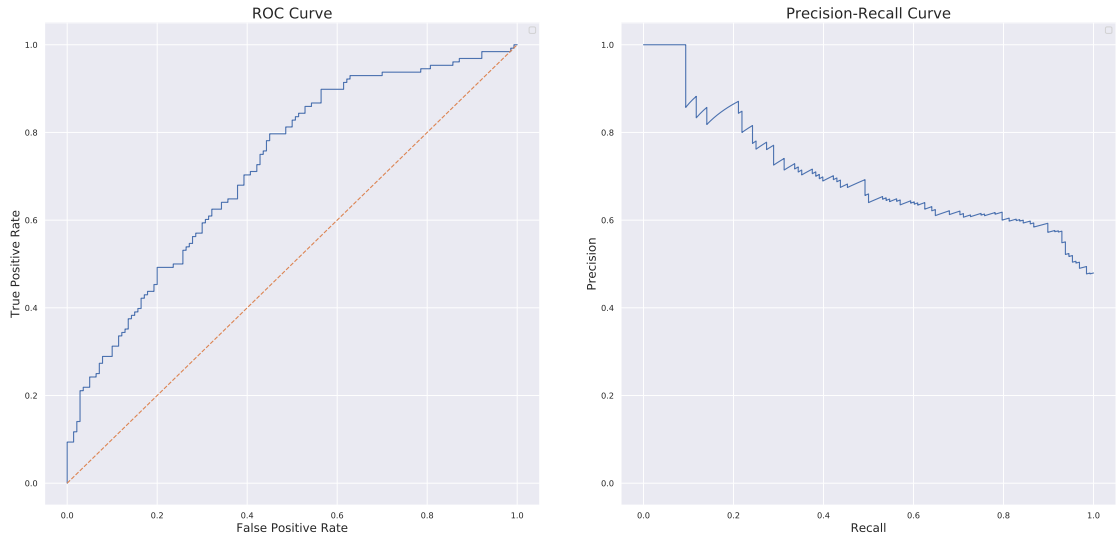
The main idea behind transfer learning is to use a different, very large dataset to train the initial network and then transfer the learned weights for the prediction on the smaller labeled dataset.

Many applications use networks that are pre-trained on ImageNet, which is a large collection of images (1.2 million pictures belonging to 1000 categories).

Figure 2.14 shows schematically how transfer learning works.

Two main transfer learning scenarios can be outlined:

- **Fixed feature extractor**: the last layer of the pre-trained network is replaced with a fully connected layer of different shape, based on the number of targets considered for the task. The parameters of the other layers are frozen and only the last linear layer is trained to predict the final task on the smaller dataset.



(a) **(b)**

Figure 2.13: An example of receiver operating characteristic curve (a) and precision-recall curve (b)

- **Fine-tuning**: In this case, not only the last layer is trained, but also the parameters belonging to the previous layers are fine-tuned by continuing the back-propagation using the data from the small dataset. It is possible to keep training all the layers or to decide to keep the earlier layers fixed to prevent overfitting.

2.5 Multimodality

The term modality is usually associated with sensory modalities, which refers to the primary channels of communications (such as vision and touch). In machine learning, in which information can be available in different forms, such as images, temporal data, and tabular data, multimodal fusion is the concept of integrating information coming from different modalities to predict an outcome. Having access to multiple modalities might allow to capture complementary information and potentially build a more robust model.

Disregarding the typology of the model adopted, fusion techniques can be divided into three types: early, late and intermediate fusion. A schematic view of the three approaches can be seen in Figure 2.15.

Early Fusion

Early fusion refers to multimodal fusion at the input level. Features are integrated after they are extracted (or lightly processed), usually by simply concatenating them.

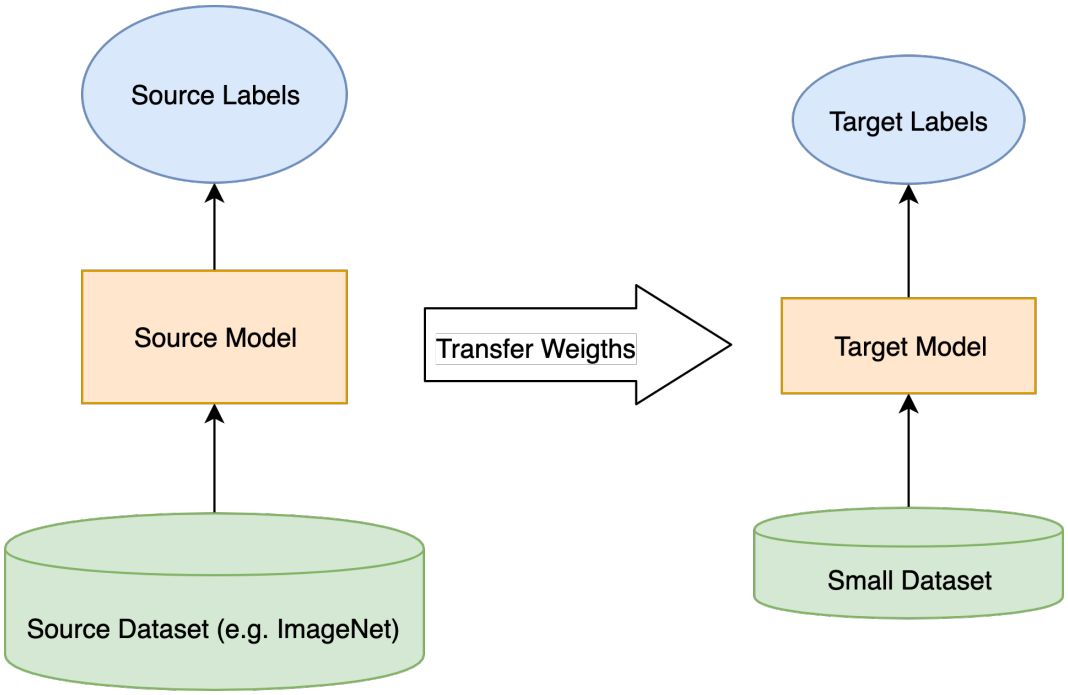


Figure 2.14: General schema of the transfer learning process

This kind of fusion can learn to exploit the interaction between low-level features for each modality. On the other hand, integrating features in the beginning may limit the model choice to a more generic architecture.

In particular, in a deep learning scenario, image inputs can be concatenated to text inputs, by transforming the images into flatten tensors and concatenating them to the tensor containing textual information. The final feature vector can then be processed by a fully connected NN. This way, however, important information may be lost since the correlation between close pixels in the original image is not taken into account. Moreover, processing an image with an FC-NN usually leads to having a number of parameters that is too high to be handled, and thus it is not recommended.

Late Fusion

In this scenario, integration is done after the decision level. As opposed to an early level integration, it ignores the low-level interaction between the features but allows to adopt a specific model for each modality considered and integrates the final results. The final outcomes of each model considered can be combined in different ways, for instance by averaging the single results or by majority voting. A weighted average can be also considered when the models have different importance. Moreover, this approach makes it easier to predict an outcome for samples for which one or more modalities is missing, by simply not considering it in the averaging scheme.

Late fusion can be considered when the models adopted are NN (of any kind). However, as mentioned above, the interaction between features coming from different modalities is ignored when outcomes are combined at the end.

Intermediate Fusion

Intermediate methods attempt to exploit the advantages of both the previous approaches by combining them.

In particular, hybrid fusion defines the approach used when more than two modalities are available and for the integration both early and late fusions are used independently on different sources. For instance, when three modalities are accessible, the first inputs can be combined by early fusion, and the resulting model can then be merged with the outcomes of a model built on the third modality.

Another intermediate technique, more specific to neural networks, consists in processing inputs with separate specialized networks at the beginning (i.e. CNN for visual inputs, RNNs for sequences), combine their parameters in some way (i.e. by concatenating them) and then feed the output to new subsequent FC layers for further joint processing. An example of this intermediate approach can be seen in Figure 2.15 (d).

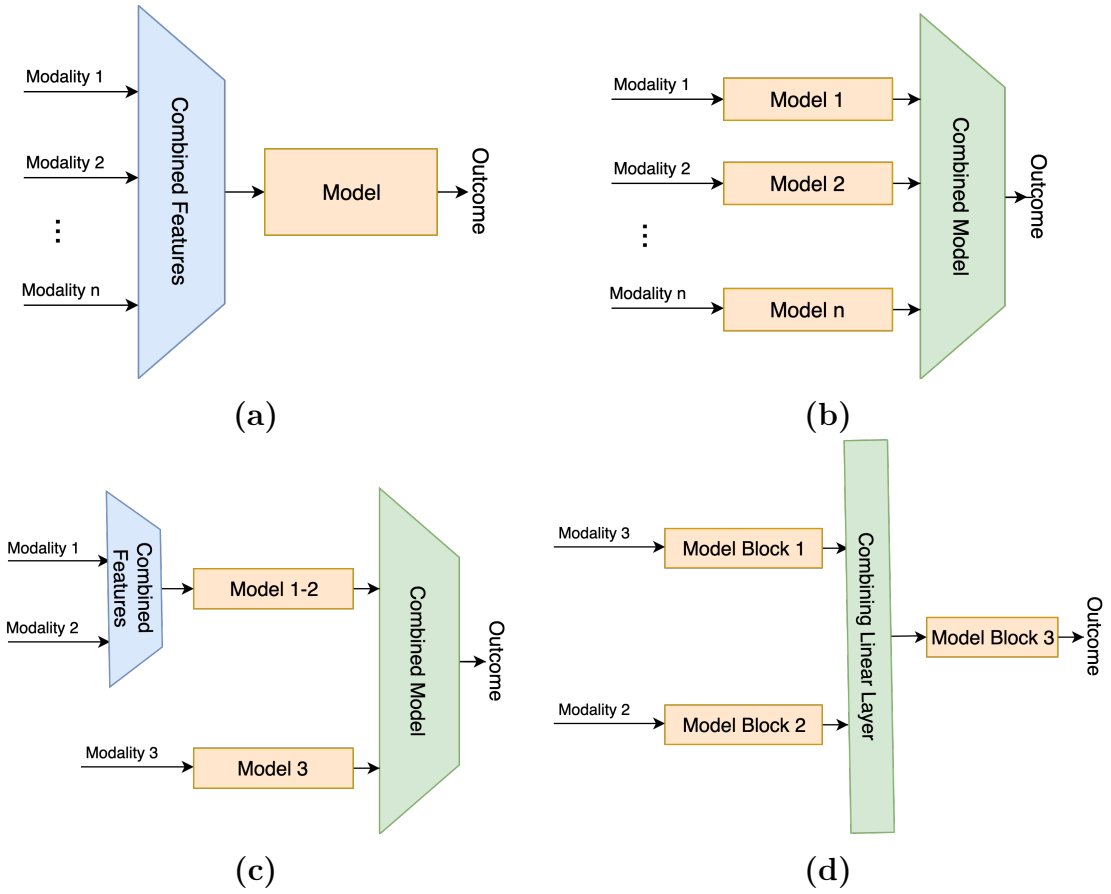


Figure 2.15: Schematic view of multimodal fusion techniques. Early fusion (a), late fusion (b), hybrid fusion (c) and neural network intermediate fusion (d).

Chapter 3

State of the Art

This chapter aims at giving an overview of the current state of the art in chest radiographs processing and classification. First, the main techniques for computer-aided diagnoses described in the literature are illustrated in Section 3.1. Section 3.2 then focuses on the main deep learning approaches for chest X-ray classification, which became the state of the art in recent years.

3.1 Traditional methods for Computer Aided Diagnose

The discovery of X-rays in 1895 has revolutionized the field of diagnostic imaging. Still up to these days, radiographs are the most common imaging exam for the chest area. Interpreting a chest X-ray can be extremely challenging also for experienced radiologists. For this reason, soon after the invention of modern computers, research in terms of computer analysis of X-ray images began [3]. The techniques developed during the years aimed at supporting radiologists by making useful information, rather than trying to develop tools that act like a diagnostician.

The main areas that can be identified in the literature of computer analysis of radiographs are general processing, segmentation and image analysis.

3.1.1 General Processing

This category comprises all the techniques developed to process the image in order to increase its readability. In particular two categories can be identified:

- **Enhancement**: comprises a series of methods proposed to optimize the display of the images and increase readability for diagnosticians. Different preprocessing procedures, such as the use of local histogram equalization techniques and the enhancement of high-frequency details (sharpening) belong to this category [4].

- **Subtracting techniques**: attempt to remove the normal structures present in a chest radiograph in order to show eventual abnormalities more clearly. Two main approaches are available: temporal and dual-energy subtractions [5].

  Temporal subtraction considers a radiograph of the patient taken in a previous point in time to enhance interval change, suppress the unchanged parts of the image (such as the ribs) and highlight eventual abnormalities. An example of temporal subtraction can be seen in Figure 3.1.

  The dual-energy approach produces separate images of the bones and the soft tissues by exploiting the differential attenuation of low-energy X-ray photons by calcium. Figure 3.2 shows the dual-energy technique applied to a chest radiograph.

3.1.2 Segmentation

Image segmentation is the process of partitioning an image into multiple segments, by assigning a label to each pixel of the image. The final goal is to define distinct sections of the image within which pixels share some characteristics [6].

Segmentation on chest radiographs usually aims at delineating the lung fields or the rib cage. Some systems were also developed for a full segmentation of the image, delineating, the rib cage, the lung fields, the heart, clavicles and blood vessels [7].

Segmentation techniques include:

- **Rule-based approaches**: consist of a sequence of steps and rules followed by the algorithm to segment the input image.

- **Classification**: different kinds of classifiers can be used to assign a class to each pixel of the input. Models are trained with a variety of input features such as intensity and location.

3.1.3 Analysis

This section comprises all the methods described in the literature that focused on analyzing a particular aspect of the chest radiograph.

The most common tasks described in the literature are:

- **Size Measurements**: to estimate conditions such as cardiomegaly, which can be diagnosed by evaluating the proportion of heart compared to the dimension of the rib cage.

- **Nodule Detection**: detect nodules in a chest radiograph can be difficult, especially in the beginning. Automated methods are highly useful for this task, and can increase the patient's chance of survival.

Figure 3.1: Example of temporal subtraction from paper [5]. (A) is the current radiograph, (B) is the X-ray of the patient taken 17 months before, and (C) shows the results of the subtraction

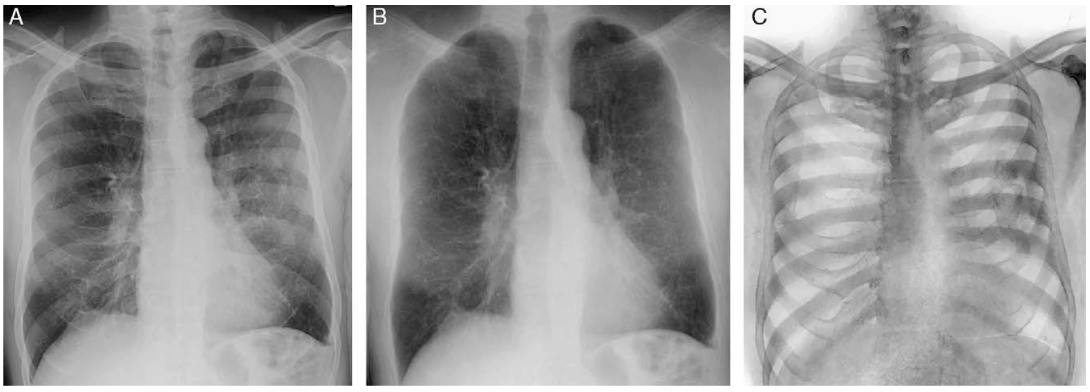


Figure 3.2: Example of dual energy subtraction from paper [5]. (A) shows the original image, (B) shows the soft tissues and (C) is the bone image

- **Texture Analysis**: is done to diagnose diffuse lung (interstitial) diseases, another complex task for a human being.

Other analyses that do not fall into the previous categories comprise the detection of other conditions, such as pneumothorax, and the presence of support devices or catheters.

3.2 Deep Learning

Although the concepts of deep learning and CNN are not new, their popularity has increased only in the last years, due to the previous lack of computational power, which is essential to process huge amount of data. Moreover, the outcomes of a neural network highly depend on the amount of data available to train it, and it is particularly challenging in the medical context, in which data are subject to strict privacy regulations, and often stored in heterogeneous ways, to collect datasets that are big enough to build robust networks.

For these reasons, deep learning for chest radiographs processing has only lately

become mainstream. The plot in Figure 3.3 shows the number of papers found on deep learning for chest X-rays per year, from 2011 to 2019 on the portal "dimensions.ai", a platform that allows users to search publications and analyze research trends. A total of 598 publications were found in the considered period, by using the following query:

((chest OR thorax) AND (radiograph OR x-ray)) AND ((deep learning) OR (artificial intelligence) OR (machine learning) OR (neural network) OR (CNN))

The exponential increase that the research in this field has seen can be related to the publication of some freely available chest radiographs datasets in 2017 and 2019.

Deep learning has now become the state of the art for chest radiographs processing. It can be used for multiple tasks, including the ones described in Section 3.1, such as segmentation or disease classification. Once again, the goal of the methodologies implemented by exploiting deep learning does not aim at building automatic tools that can substitute diagnosticians, but rather supporting their decisions by giving additional insights.

3.2.1 Datasets

Four main public datasets can be identified and are nowadays widely in use by researchers:

- **ChestX-ray8**: published in 2017, comprises 108,948 frontal view X-ray images of 32,717 unique patients and 8 labels [8]. Images were labeled by using text mining on radiologists' reports. The same technique was then used to label 125,000 images with 14 labels (ChestX-ray14).

- **PadChest**: from 2019, contains more than 160,000 images belonging to 67,000 patients from the San Juan Hospital (Spain). [9] Differently from the other datasets, labels for PadChest were extracted by processing reports written in Spanish.

- **CheXPert**: includes 224.316 images of over 65 thousand patients from the Stanford hospital (taken between October 2002 and July 2017) and 14 labels. The dataset was published in 2019 [10].

- **MIMIC-CXR**: contains 371.920 chest X-ray images associated with 227,943, of patients of the Beth Israel Deaconess Medical Center between 2011 and 2016 [11]. The labels of this dataset are the same as the CheXpert dataset,

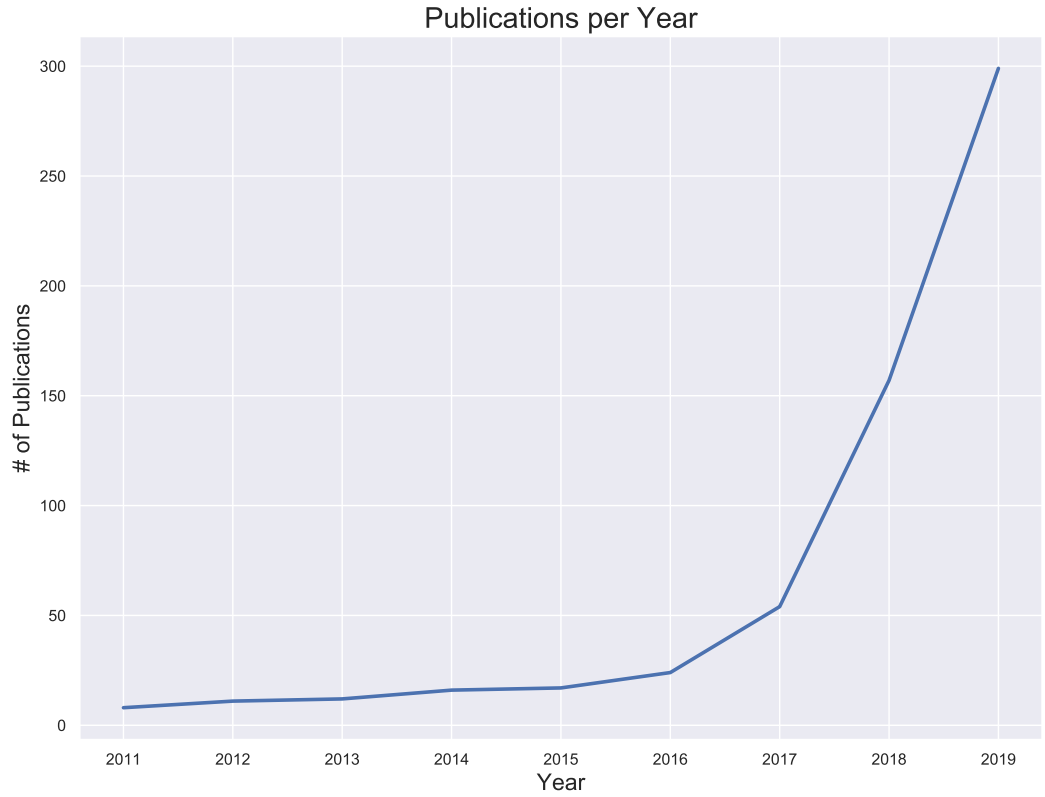


Figure 3.3: Number of publications found in the portal "dimensions.ai" from 2011 to 2019 regarding deep learning on chest radiographs

allowing to combine them and increase the available images to train a model. The dataset is available since January 2019.

In all the datasets mentioned, labels were extracted by automatic text processing methods that examined the radiologists' report.

Before 2017, only a few much smaller datasets were available: in particular, the biggest was OpenI, which contains 7470 images from patients of the Indiana Network for Patient Care [12].

3.2.2 Methods

From 2017 on, a variety of researches that applied deep learning on X-rays was published. Deep learning methods can be exploited for multiple tasks and are substituting traditional methods in image processing, segmentation, and classification tasks.

For instance, Wei et al. developed a deep learning framework for bone suppression on chest images [13].

Different examples of convolutional neural networks used for chest radiographs segmentation can also be found. As an example, Novikov et al. present a CNN

architecture for multiclass segmentation [14], while Arbabshirani et al. focus on segmentation of the lung field [15]. It should be noticed that the mentioned works do not train their models on any of the datasets described in 3.2.1, but rather on smaller ones, such as the Japanese Society of Radiological Technology database, which contains 247 X-ray images [16]. The reason is that segmentation labels, in which each pixel of the image is classified, are not available for the above-mentioned datasets and manual labeling is not feasible, considering the dimension of the datasets.

On the other hand, datasets of subsection 3.2.1, have been widely used for the classification of diseases that can be diagnosed from a radiograph. A few studies pose the focus on one particular disease, such as pneumonia [17], but most of them present a multi-label classifier, thus considering multiple diseases (that can coexist).

The two architectures predominantly used for classification are DenseNet, which is described in detail in 2.2.4, and ResNet. Both architectures are based on the idea of creating short paths from early to later layers in order to increase the number of layers of the network and avoid vanishing gradient problems. The most adopted optimizer in recent researches is ADAM.

The CheXpert dataset manuscript [10] reports the results of a DenseNet121 on the data, stating that the architecture was the one performing the best among the ones tested. Other models considered were ResNet152, Inception-v4, and SEResNeXt101.

A research carried out by Baltruschat et al. analyzes different aspects of deep learning approaches for multi-label chest classification [18]. The model used in the manuscript is ResNet50 and the dataset considered is ChestX-ray14.

The first aspect examined is the impact of weight initialization and pretraining on the final results. Pretraining (transfer learning) is used in various researches, especially when the available set of images is not particularly big. The paper compared the results obtained with a network with random weight initialization vs. two transfer learning approaches: off-the-shelf (OTS), which implies an adaptation of only the weights of the last linear layer to the new task, and fine-tuning (FT), in which one or more layers of the network are retrained with the samples from the target dataset. Weights were transferred from a ResNet50 architecture trained on the ImageNet dataset [19]. The results show that the OTS approach is performing much worse than the others on all the targets considered, with an average AUC that is 8.7% lower than the FT model. The results obtained by the model randomly initialized and the fine-tuned one are comparable, with the first one outperforming the latter one by 0.002 on average AUC. The results are also confirmed by other studies, leading to believe that transfer learning from general images datasets does not bring important advantages on medical imaging tasks, due to the difference in terms of size of the datasets, features, and number of classes, and the high specificity of images in medical datasets [20].

Secondly, the impact of the image size was considered. The model randomly initialized was compared with a model taking images double the size as inputs. The latter architecture was adapted by adding a max-pooling layer after the first three resnet blocks. This was done in order to obtain the same input size at the last averaging pooling layer. The high-resolution variant obtained an average AUC of 0.821, which is 2% higher than the lower quality alternative. Increasing the image quality resulted particularly advantageous for smaller pathologies, such as nodules and masses.

Finally, the research evaluated the advantage of including non-image features, such as age and gender of the patients, as an input to the model. All the previously described models were retrained by considering the new features and each one obtained a higher AUC as compared to its counterpart. The improvement obtained by including non-image features was however really low. The reason is probably that the information added to the model by including those features is somehow already encoded in the images, thus no new information is actually added to the model. In order to verify this, a model to predict each non-image feature from the radiographs was developed, showing how easy it is to predict such features from the X-rays.

Some works on chest X-rays classification used data augmentation. Data augmentation techniques are widely adopted in deep learning to increase the dataset diversity and make the model more robust. Many different transformations can be applied to the input data. However, not all the transformations make sense in all settings, as some of them would change the original input in a way that would never be seen in reality, thus making it not useful. Common augmentation techniques for chest radiographs datasets include random horizontal flipping, random rotation, and random sampling of various size patches of the training images or random cropping of portions of the image.

The last important aspect to examine is model interpretation. As mentioned multiple times, chest radiographs models aim at helping radiologists in the diagnosis. For this reason, a visualization tool, that highlights areas of the image that were considered important by the model when outputting a probability is really important. Moreover, model interpretation tools may help examine whether the algorithm has delineated any non-conventional pattern and may be exploited by the developers to examine false positives and false negatives.

Multiple model interpretation techniques are present in the literature. A method that is widely used in research on chest images is called class activation mapping (CAM) [21]. The method is particularly suited in this context, as it is highly class-discriminant and thus permits to highlight the areas that are important for one single target. In particular, gradient weighted class activation mapping (Grad-CAM), is a generalization of the CAM approach that can be applied to a broader range of

CNNs, such as models with fully connected layers [22].

An example of Grad-CAM applied to a chest radiograph, taken from the CheXpert documentation [10], can be seen in Figure 3.4.

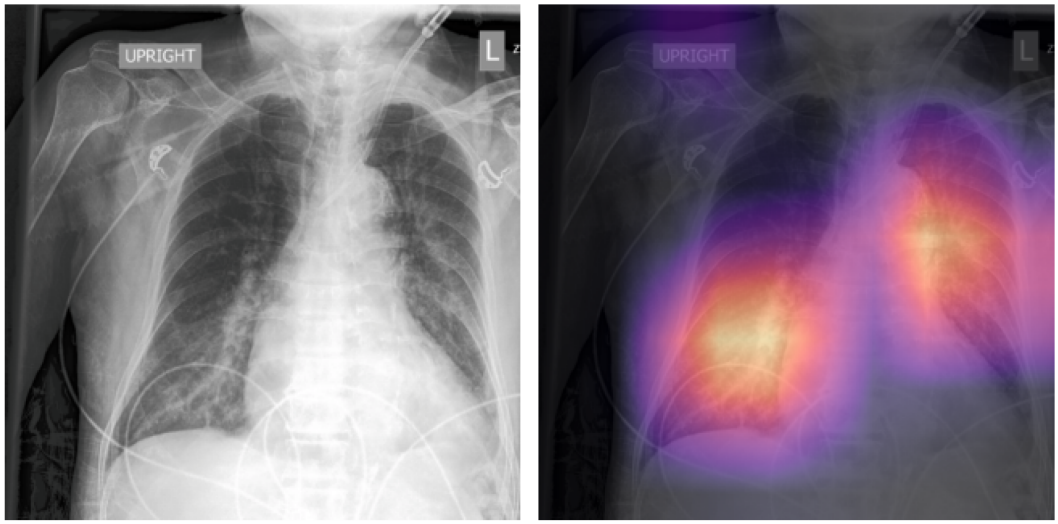


Figure 3.4: Grad-CAM method applied to a frontal radiograph of a patient with pulmonary edema, which is currently localized by the model. Image taken from the CheXpert manuscript [10]

Chapter 4

Medical Contextualization

The goal of this chapter is to give a general overview of the medical concepts encountered in this work. First, postoperative bleeding, which is the complication which this thesis focuses on, is explained in section 4.1. In section 4.2, the paper "Machine learning for real-time prediction of complications in critical care: a retrospective study", whose work is extended by this thesis, is summarized. Finally, some information regarding chest radiographs, is given in section 4.3.

4.1 Postoperative Bleeding

Postoperative bleeding is one of the complications that may occur after cardiac surgery. Two typologies of postoperative hemorrhage can be identified: coagulopathic bleeding, which is related to a coagulation problem (blood or hematoma without ongoing bleeding), and is usually treated by giving the patient coagulation factors, and surgical bleeding, that requires a surgical re-exploration [23].

According to the literature, postoperative bleeding, followed by surgical re-exploration occurs in 2% to 6% of cardiac surgical patients. These patients also have a higher mortality risk, an increased length of hospitalization, use of resources and blood products [24]. At the German Heart Center Berlin, surgical bleeding affects more than 100 adult patients per year.

Discovering postoperative hemorrhages in their early stage is crucial because if a patient bleeds for a long time, he/she may go into hemorrhagic shock, a condition which may lead to hemodynamic instability, decreases in oxygen delivery, decreased tissue perfusion, cellular hypoxia, organ damage, and death [25]. A study conducted on 99 patients that underwent surgical re-exploration due to postoperative bleeding shows that patients that survived were re-explored on average 155 minutes before the others [23]. Moreover, early discovery may help to decrease the use of resource usage and the hospitalization periods.

Postoperative bleeding is hard to detect, especially in the early stage, as there is no clear indication of it in the patient's parameters. A commonly known rule-based guideline proposed in the Bojar algorithm is generally used to decide whether a second surgery is necessary. The rule states that surgical re-exploration due to bleeding should be performed if the bleeding rate of a patient is:

- $>$ 400 mL/h for 1 hour
- $>$ 300 mL/h for 3 hours
- $>$ 200 mL/h for 4 hours

On the other hand, diagnosis of postoperative bleeding, and the consequent decision to perform a surgical re-exploration should be as correct as possible, since a second surgery means additional surgical trauma for the patient, and an increased probability of mortality, to be avoided on patients for which is not necessary.

An automatic method that predicts whether surgical re-exploration is needed would be of great support for physicians monitoring the patient's status in ICUs.

4.2 Real-time Prediction of Complications in Critical Care

The goal of this section is to give an overview of the work presented in the paper "Machine learning for real-time prediction of complications in critical care: a retrospective study" [1], which describes the work done previously and that this thesis extends.

A large number of clinical signals are recorded in intensive care units in the hours following surgery. Health-care personnel can be easily overwhelmed by the amount of data, leading to delays and clinical errors.

Machine learning methods can be particularly useful in this kind of setting when signals are produced at a fast rate, and a human being may have difficulties in processing them. However, the application of ML methods often encounters issues, as data may be difficult to access, may be stored and organized on different heterogeneous systems and usually contains many missing values and errors.

The retrospective study carried on analyzed the electronic health record data from a German tertiary care center for cardiovascular diseases (Deutsches Herzzentrum Berlin, DHZB) of adult patients that underwent open-heart surgery between 2000 and 2016. The study considered three possible outcomes: postoperative bleeding, postoperative renal failure, and postoperative in-hospital mortality. The focus of this thesis is the prediction of postoperative bleeding after open-heart surgery.

A Recurrent Neural Network was used to model the likelihood that a complication after surgery occurred.

Features considered in the model are summarized in Table 4.1. Two types of variables were considered:

- **Static Variables**: features that are not subject to change during the hospitalization period (such as sex and age of the patient). They are replicated at each timepoint. Static features are the ones related to patient and initial surgery information in Table 4.1.

- **Dynamic Variables**: Clinical markers were collected every 30 minutes for the 24 hours following the surgery. Blood gas analysis, laboratory results and output features were tracked on an interval basis. Dynamic features are the ones in the categories "Vital signs", "Arterial blood gas", "Laboratory results" and "Balanced output" in Table 4.1.

Figure 4.1 shows the data extraction and preprocessing steps done to create a uniform input for the recurrent neural network, containing both the static and dynamic features.

Missing values were imputed with the last available value for that feature, or with a default value, chosen by a clinical expert when no previous measurements were available. The outcomes were compared to the standard methods commonly used to recognize the three outcomes under study.

The RNN model used consisted of one GRU layer with 40 hidden nodes and sigmoid activation function. 10% of the data was kept as a test set, and ten-fold cross-validation was done in the training phase. At each time step considered, the model outputted the probability that the event occurred.

For all the tasks considered, the RNN approach provided significantly better accuracy levels than the corresponding reference tool (the Bojar rule for the postoperative bleeding case).

A balanced dataset of 4644 patients was considered. 464 of them were kept as a test set, while the remaining were used in the training process.

The time windows considered was set to 24 hours but was in some cases shorter, since surgical re-exploration may have been performed before the end of the window on bleeding patients. To avoid any influence of the window length on the prediction, for each patient in the control group (non-bleeding instances), a time window duration of some bleeding patient was assigned, so that the distribution of lengths for the two targets was the same and could not affect the RNN results.

The average results on the test set, in terms of accuracy, AUC, precision, recall, and F1-score, for both the proposed RNN model and the clinical baseline, are reported in Table 4.2. The overall performances are computed by micro-averaging the results

	Features
Patient information	Age, sex, height, weight
Initial surgery information	Anaesthesia type, American Society of Anesthesiologists Score, cardioplegic solution, aortic cross-clamp time, cardiopulmonary bypass time, anesthetic monitoring time, surgery duration, surgery type, urgency
Vital signs	Systolic, mean, and diastolic arterial pressure; systolic, mean, and diastolic pulmonary artery pressure; central venous pressure; ventilator FiO2 setting; heart and respiratory frequency; body temperature
Arterial blood gas	Bicarbonate, glucose, hemoglobin, oxygen saturation, partial pressure of carbon dioxide and oxygen, pH level, potassium, sodium
Laboratory results	Albumin, bilirubin, urea, C-reactive protein, creatine kinase, γ-glutamyltransferase, glutamic oxaloacetic transaminase, hemoglobin, hematocrit, international normalized ratio, creatinine, white blood cell count, lactate dehydrogenase, magnesium, partial thromboplastin time, platelets, prothrombin time
Balance output	Bleeding rate, urine flow rate

Table 4.1: RNN model features overview

for each time-step considered. Macro averaging. which would mean averaging the mean performances of the 49 time-steps considered, is not suitable in this case, due to the different lengths of the monitoring time.

Results for the postoperative bleeding prediction (in terms of accuracy), and their comparison with the clinical reference can be seen in Figure 4.2.

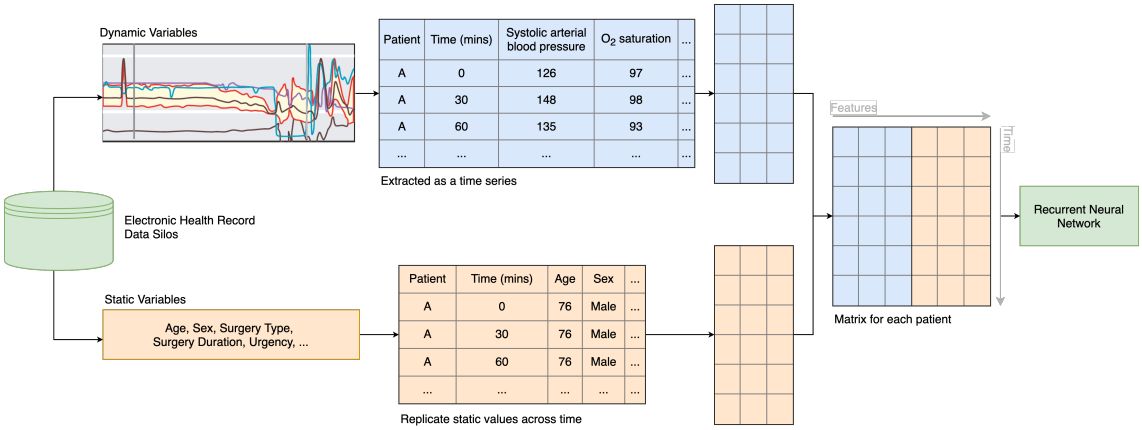


Figure 4.1: Architectural overview of the data extraction and representation processes

	Accuracy	AUC	Precision	Recall	F1-score
RNN	0.80	0.87	0.84	0.74	0.79
Baseline	0.58	0.58	0.81	0.21	0.33

Table 4.2: RNN and clinical baseline overall performance on the test set

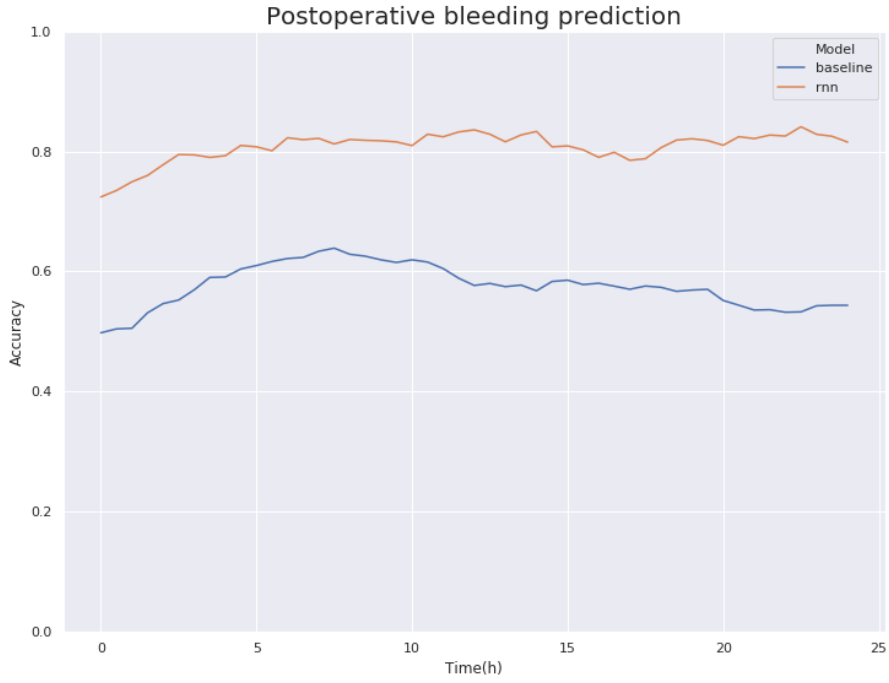


Figure 4.2: Accuracy over time of RNN and baseline models on the test set

4.3 Chest Radiographs

Conventional radiology, which refers to radiographs generated by exposing an X-ray film to ionizing radiations, is the most common image modality adopted in the evaluation of the thorax [26]. It can be used to diagnose different acute and chronic cardiopulmonary conditions, such as pneumonia and pneumothorax, and to verify whether support devices, such as pacemakers, are correctly positioned.

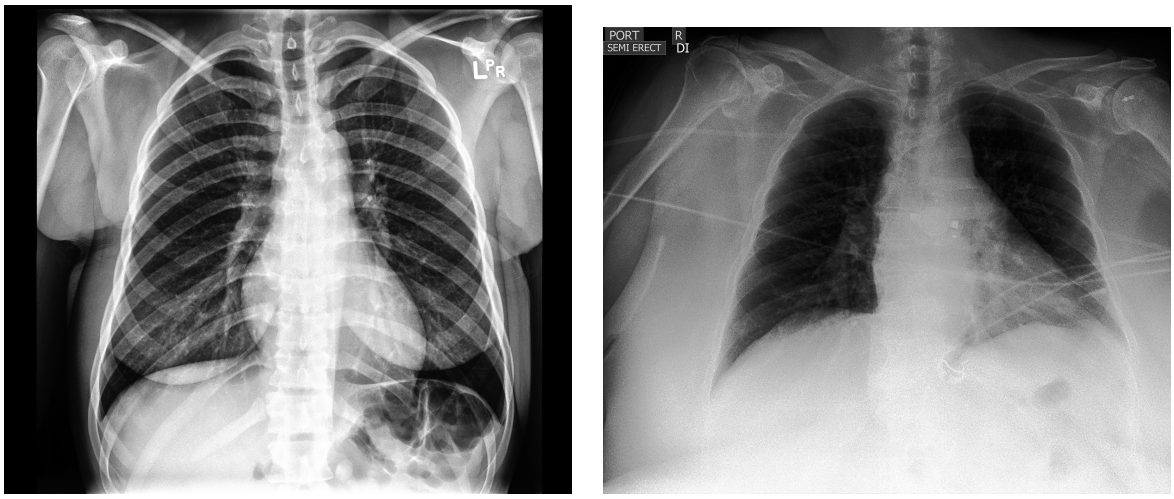
Chest radiographs can be frontal or lateral. Moreover, frontal images can be captured using two different modalities, which differ on how the beam passes through the body of the patient. If the X-rays pass from the front of the patient to the back, the modality is called anteroposterior (AP); vice-versa when the X-rays pass from the back to the front of the patient, the modality is called posterior-anterior (PA).

Generally. the patients are standing in front of the receiving film and the X-ray beam passes through the body before reaching the radiographic cassette (PA modality). To minimize magnification and distortion, there should be a distance of at least around 2 meters between the X-ray tube source and the cassette.

On severely ill patients, that cannot be moved or stand, a portable machine can be used. In this case, the cassette is placed below the patient's bed and the source is positioned above the patient, therefore the image is taken in AP modality. No lateral image is available for patients in this condition.

Images taken in AP modality are usually harder to read, due to magnification effects, which may, for instance, make the heart look larger. Moreover, an ideal radiograph is taken while the patient is at maximal inspiration, which may not be possible in the case of bedridden patients, especially when they are affected by dyspnea.

The difference between thorax radiographs taken in PA and AP modalities can be seen in Figure 4.3.



(a) **(b)**

Figure 4.3: An example of chest radiographs taken in PA modality (a) and AP modality (b). (Images taken from the CheXpert dataset [10])

Chapter 5

Chest Radiographs Classifier

The first phase of the thesis consisted in building a deep learning model to process frontal chest X-ray images. The goal of the first section of this chapter is to describe the steps taken to develop such a model, the datasets considered and the architecture used. Section 5.2 illustrates the technique used to transfer the trained model to the prediction of postoperative bleeding.

5.1 Model Development

5.1.1 Dataset

The CNN implemented was trained considering images coming from two public datasets. The choice to use external datasets was made since the dataset collected at the DHZB was considered too small to train a network able to generalize the task considered.

The public datasets considered were:

- CheXpert [10].
- MIMIC-CXR [11].

As already mentioned in Section 3.2, the two datasets are structured in the same way, allowing to easily merge them.

The datasets contain both frontal and lateral images. Frontal images were taken both in either AP or PA mode.

The final goal of the thesis was to integrate chest radiographs taken right after surgery with static and temporal features measured after the surgery, to predict complications that may occur in the first 24h after it. In this context, only frontal images are taken, and the modality used is always AP, as the patients are bedridden. For this reason. lateral pictures were dropped from the datasets used for training.

On the contrary, both AP and PA images were kept, even if images available for patients considered in the integration part are always taken in AP mode. This choice was done to avoid losing a big portion of data, and since images that are taken in PA modality do not differ too much from those taken in AP modality, except for their quality.

Removing lateral images led to losing around 33% of the images in the MIMIC-CXR dataset and 15% in the CheXpert.

Images in both datasets are in jpeg format, they are organized in directories, by patient ID and study. The same patient may have multiple studies. In general, a study may contain more than one image, when both frontal and lateral views are available. In this context, however, since lateral images were removed, each study contains exactly one frontal image.

Together with the radiographs, a comma-separated-value (CSV) file is available. In the file, the targets for each image are reported, together with information regarding the type of image (frontal or lateral) and the modality used to take it (AP or PA). In the CheXpert dataset, the sex and age of the patients are also available.

Data Split

It is common practice, when training a machine learning model, to split the available data into three distinct sets: the training set, which contains the majority of the available data, and is used to train the chosen model; the validation set, used to tune the hyperparameters of the model, and the test set, which is used only at the end, to evaluate the trained model on unseen instances and verify that the model did not overfit the training set.

Both the available datasets were already split into training, validation and test sets but only training and validation sets are publicly available. Studies in the training sets were labeled using an automatic labeler tool that analyses the associated report written by a physician. Validation sets, which contain 500 and 200 studies respectively for the MIMIC-CXR and the CheXpert dataset, were instead labeled manually by a committee of 3 board-certified radiologists, to provide higher reliability on the labels assigned.

The original split presented however two main problems: first, a test set was not available and second, the validation set was way smaller as compared to the training set.

For this reason, the available validation set was used as a test set in the experiments, while the training set was split into training and validation sets (90% - 10%).

Labels

The original datasets contain 14 labels. A clinical expert selected 9 of them that were considered to be relevant to the bleeding problem. The targets considered are:

- **No finding**: no abnormality found in the radiography.
- **Cardiomegaly**: a medical issue in which the heart is enlarged.
- **Edema**: fluid accumulation in the tissue and air space of the lungs.
- **Consolidation**: occurs when the air that usually fills the small airways in the lungs is replaced with something else, such as a fluid.
- **Pneumonia**: Infection of one or both lungs.
- **Atelectasis**: a complete or partial collapse of the entire lung or area (lobe) of the lung.
- **Pneumothorax**: when air enters the space around the lung.
- **Pleural effusion**: excessive quantity of fluid between the lungs and the chest cavity.
- **Support devices**: the presence of devices such as catheters or pacemakers.

The remaining 5 labels were dropped.

Missing and Unknown Labels

For each study, most of the labels related to it were missing. This is the sign that they were not mentioned in the annotation written by the physician, thus they could be considered as not present.

Moreover, other than 0 (not present) and 1 (present), some instances were labeled with a third target: -1, that represented uncertainty, either in the presence of the condition on the patient, or ambiguity of the automatic labeler in interpreting the report.

The distribution of labels in the final merged dataset for each target can be seen in Table 5.1.

The Stanford paper [10] proposes 5 different approaches to deal with -1 labels:

- **U-Zeros**: set all the uncertain labels to 0 (meaning the label is not present)
- **U-Ones**: assume that the uncertain labels represent a present condition on the patient (and set them to 1)

Label	Present(1)	Absent(0)	Unknown(-1)
No Finding	99642 (22.68%)	339621 (77.32%)	0 (0%)
Cardiomegaly	72279 (16.45%)	353557 (80.49%)	13427 (3.06%)
Edema	79060 (18.00%)	334211 (76.08%)	25992 (5.92%)
Consolidation	24716 (5.63%)	385408 (87.74%)	29139 (6.63%)
Pneumonia	23002 (5.24%)	380599 (86.64%)	35662 (8.12%)
Atelectasis	79346 (18.06%)	318999 (72.62%)	40918 (9.32%)
Pneumothorax	29303 (6.67%)	405980 (92.42%)	3980 (0.91%)
Pleural Effusion	135630 (30.88%)	287777 (65.51%)	15856 (3.61%)
Support Devices	181391 (41.29%)	256688 (58.44%)	1184 (0.27%)

Table 5.1: Label distribution on the merged train dataset

- **U-SelfTrained**: consider them as unlabeled examples, transforming the problem to semi-supervised learning (Multi-label Learning with Missing Values, MLML)

- **U-Multiclass**: consider them as a separate class and have three classes for each layer. It is arguable, however, whether a prediction of uncertainty makes sense.

- **U-Ignore**: set the loss to 0 when the label is -1 so that it does not influence the gradient and does not consider the values when computing the metric. Removing uncertain values from the training data means avoiding making assumptions that may be wrong and that may badly influence the model. On the other hand, this approach leads to a reduction of samples that varies between 0 and 15%, depending on the label, in the CheXpert dataset. The MIMIC-CXR dataset has less uncertain values (up to 7.7%).

After a discussion with a domain expert, the U-Ignore strategy was adopted, thus reducing slightly the dataset (in the merged dataset, up to 9.32% for Atelectasis). This was done both for simplicity and to avoid making wrong assumptions (e.g. consider an unknown as true).

Data Preprocessing and Augmentation

The raw datasets contain both horizontal and vertical images, with slightly different sizes one from the other. For this reason, images needed some preprocessing before feeding them into the network, to obtain inputs of the same size.

The preprocessing steps performed are the following:

- **Resize**: Images were first resized and then cropped to the final size desired. Resizing was necessary to avoid cropping a too small part of the picture and thus cut out relevant parts. Images were resized by setting the smallest side to 900 pixels.
- **Random crop (or center crop)**: this was done to obtain squared images of the same size, as the dataset contains both vertical and horizontal pictures. In the validation and test sets, the center of the image was considered, while on the train set, images were cropped randomly, to augment the data available. Inputs were cropped to a final size of 896x896 pixels.
- **To tensor**: images were then converted into tensors to feed them to the network.
- **Normalize**: the mean and the standard deviation of the original input dataset were computed and used to standardize the pictures before processing them.

Other than this, a few data augmentation techniques, which may be able to better generalize the model, were taken into account.

The augmentations applied to the original training data are the following:

- **Random horizontal flipping**: images were horizontally flipped with a probability p (set to 0.5).
- **Random rotation**: consists in randomly rotating the inputs, up to a certain degree. Rotations considered were up to 10 or 20 degrees, to simulate the case in which the patient is not correctly positioned. The final value chosen was 10 degrees.
- **Cutout**: this technique randomly obscures a portion of the input image. It can also act as a regularization technique, thus decreasing the likelihood of overfitting the training set.

  The cutout method has 2 hyperparameters to set: the size of the area to cut, which was set to 300x300 pixels for this task, and the probability to cut an area on the image (set to 1, thus having an obscured region on each training sample).

  The way cutout affects the input image highly depends on the area that is obscured. If it is on the border of the input, then the cut area is smaller than the size set, and the original image is only marginally modified.
- **Color jitter**: applies a random variation of saturation, brightness, and contrast of the image. Only contrast and brightness were modified in the chosen settings, considering values that only slightly affect them.

Figure 5.1 shows the effect of the adopted data augmentation techniques on one radiograph.

5.1.2 Tools

All code was written in Python 3.7.3, one of the most common languages for deep learning projects. The Pandas (v. 0.24.2) and Numpy (v. 1.17.1) libraries were used to model the data, while the deep learning framework used to build the networks was Pytorch (v. 1.2.0) [27], a low-level framework that is suitable for academic projects, as it allows to customize the architecture and reach high performances. Other libraries, such as Scikit-learn (v. 0.20.3) and Seaborn (v. 0.9.0) were used to compute the metrics, split the datasets and visualize the results.

5.1.3 CNN Model

After the preprocessing phase, training data were fed to the defined Convolutional Neural Network. A few different architectures were taken into account in the developing phase, both simple models with a few layers written from scratch and more complex architectures described in the literature.

Finally, the architecture chosen is a variation of DenseNet. The original DenseNet121 architecture is defined to process images coming from ImageNet, a well-known dataset widely used in deep learning. Images in ImageNet have size 224x224 pixels, while images in the X-ray dataset were shaped to a final size of 896x896. For this reason, to have the same dimension before the final average pooling layer, the DenseNet structure was modified by adding two DenseBlocks, as can be seen in Figure 5.2.

Moreover, the DenseNet architecture usually deals with colored images, with 3 channels, while radiographs are black and white, thus have only one channel. To solve this issue, the first convolutional layer was modified to deal with greyscale inputs.

The chosen optimization technique was Adam, with a learning rate set to 0.001. binary cross-entropy (BCE) Loss was used, modified to ignore losses coming from images with label uncertain (-1).

The model was trained for 30 epochs, on 4 graphics processing units (GPU) and a batch size set to 32. The validation set was used for early stopping, a technique used to choose at what epoch to stop the training. The model chosen for the evaluation on the test set was, therefore, the one for which the lowest loss was measured on the validation set.

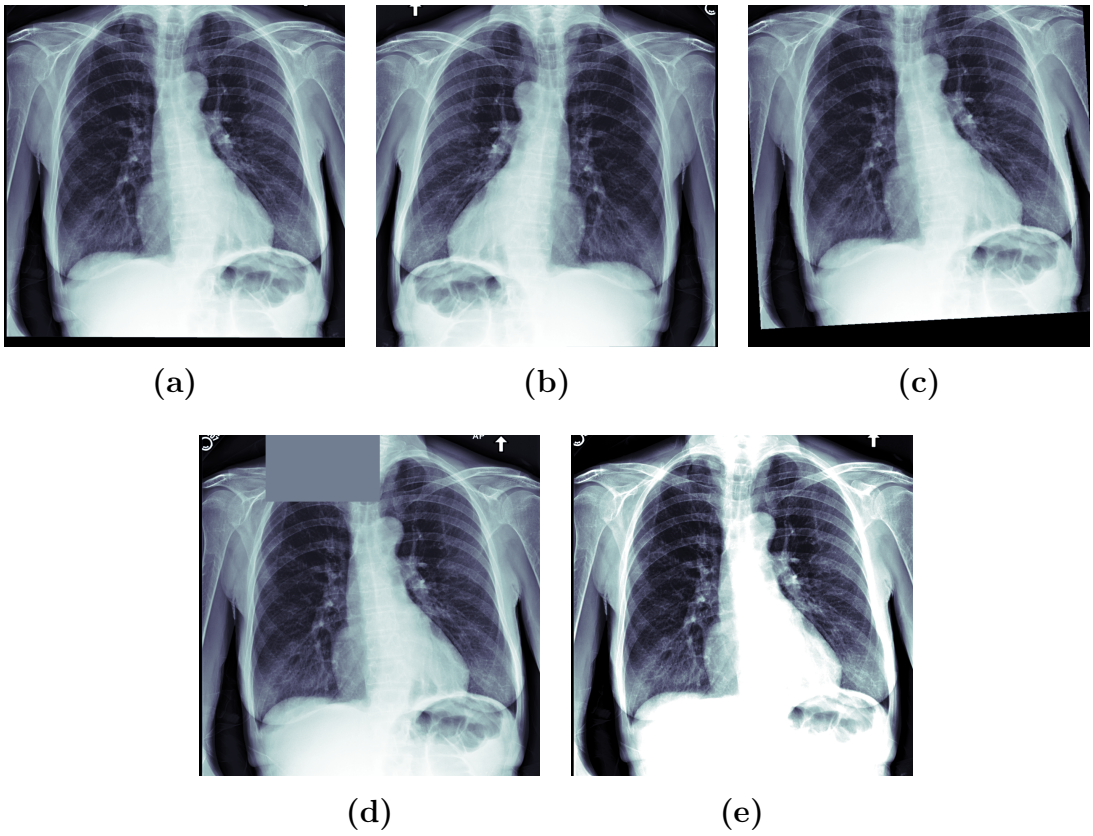


Figure 5.1: Data augmentation techniques applied to the same image. (a) shows the radiograph pre-processed, without data augmentations, on (b) random flipping is applied, (c) is slightly rotated, on (d) cutout has been applied, on (e) brightness and contrast have been modified

5.1.4 Results

This subsection reports the results obtained on the classification of chest radiographs on the images kept as the test set from the public datasets. As mentioned, nine labels were considered in this phase, as they were the ones that are considered to be the most related to postoperative bleeding, which is the final task to predict. Results are reported in terms of Area Under Receiver Operating Characteristic Curve. Accuracy was measured but not considered to be particularly relevant, as for most of the targets considered, there is a strong presence of instances labeled as 0, as can be seen in Table 5.1.

The AUCs were compared to the results reported in the CheXpert paper, for the 5 labels for which they are available (cardiomegaly, edema, consolidation, atelectasis, pleural effusion). Results can be seen in Table 5.2. The comparison is done only to ensure that the model developed can perform similarly, but cannot be considered as absolute results, as the datasets on which the metrics are computed are different

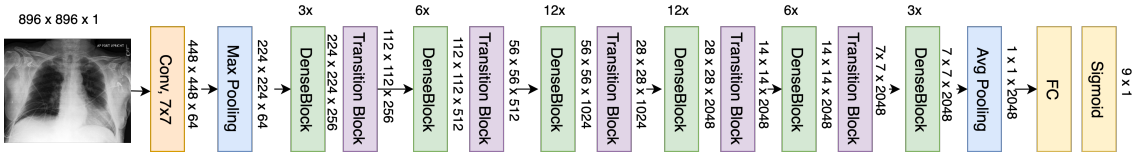


Figure 5.2: Structure of the DenseNet architecture adapted to process 1-channel 896x896 pixel images

Label	Model Results	CheXpert Results
No Finding	0.870	-
Cardiomegaly	0.817	**0.828**
Edema	0.923	**0.934**
Consolidation	0.874	**0.938**
Pneumonia	0.817	-
Atelectasis	**0.847**	0.818
Pneumothorax	0.918	-
Pleural Effusion	**0.935**	0.928
Support Devices	0.939	-

Table 5.2: Comparison between the results of the developed model on the test set and results reported in the CheXpert paper [10] in terms of AUC

(the official test-set is not publicly available) and since the model developed takes into account less information because lateral images were not considered.

ROC curves for each target considered are shown in Figure 5.3.

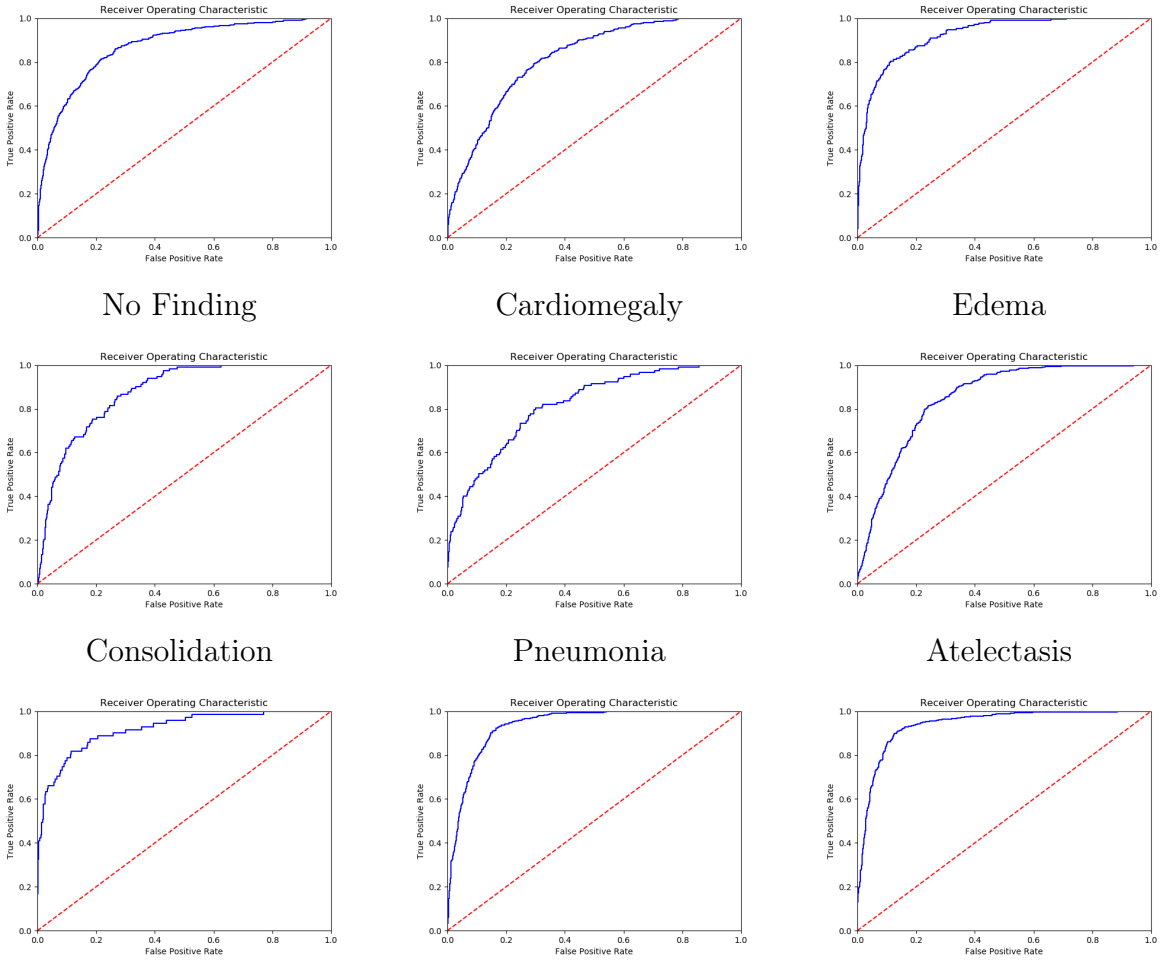


Figure 5.3: ROC curves for each target considered

5.2 Transfer Learning on Chest Radiographs

5.2.1 Bleeding Dataset

The ultimate goal of the research was to extend the work presented in section 4.2 by including chest radiographs taken after the surgery to examine whether X-ray images can bring additional information to the current model. As stated earlier, the previous study considered a balanced dataset of 4644 patients that went under surgery at the German Heart Center Berlin between 2000 and 2016.

Images related to those patients were retrieved from the hospital's picture archiving and communication system (PACS), by querying the server considering the patient's ID and a time interval of 24 hours after the beginning of the monitoring time. For some patients, more than one radiography was taken for the time period considered: in this case, the first one obtained after the surgery was considered. Pydicom (v. 1.3.0) and pynetdicom (v. 1.4.1), which are two python libraries, were used to write the querying script.

The PACS is daily used by physicians at the hospital to retrieve and visualize the images regarding their patients. To be sure not to compromise the hospital's system, by sending too many consecutive requests, risking to postpone those coming from doctors, images were retrieved only after 5 PM, since most of the daily requests to the PACS are sent in the morning. Moreover, a sleeping time of 30 seconds between each request was set, to allow the system to process eventual other queries in the meantime.

Unfortunately, the DHZB stores chest radiographs in a digital format only since May 2005. Moreover, for some patients no image was found, while for others the first radiograph taken after the surgery was obtained after the monitoring period, making it not suitable.

For these reasons, the available labeled dataset was reduced to a final size of 2812 patients, of which 268 were kept as test set. The dataset was still roughly balanced (1388 bleeding cases, 1455 nonbleeding); the test dataset was similarly distributed (140 bleeding cases, 128 nonbleeding). For these patients, both the temporal parameters and the chest radiograph were available.

Images in the PACS are stored in DICOM format. DICOM (Digital Imaging and COmmunications in Medicine) is an international standard used to store medical images. A DICOM file is not only composed of the image, but incorporates the related information as well, to avoid it to ever be separated from the illustration. Different attributes are included in the DICOM object, including the patient's ID, its name, the date and time the image was taken etc.

5.2.2 Experiments

Building a model on such a small dataset is challenging and often leads to overfitting. For this reason, transfer learning was adopted, to exploit a model trained on a similar task to predict postoperative bleeding on the available images.

Using transfer learning when dealing with small datasets is common, but often the models are pre-trained on general datasets, such as ImageNet. When dealing with medical image classification, fundamental differences between the ImageNet dataset and the medical datasets are usually present. Medical datasets usually differ from more general sets in terms of size, features, and number of classes. For this reason, models pre-trained on ImageNet usually offer little benefits to the performance of medical image classifiers [20].

For this reason, instead of considering a model trained on a natural image dataset , the model was pre-trained on a dataset containing the same kind of images, chest radiographs, but with different targets.

Different scenarios were considered and compared:

- **Train model from scratch**: initialize the weights randomly and train the network with the bleeding set, thus not considering the pretrained model.
- **Transfer Learning and train of last FC layer**: import weights from the model trained on the public dataset and train only the final fully connected layer with the bleeding data.
- **Transfer Learning and fine tuning**: import weights from the model trained on the public dataset and keep training the whole model on the bleeding train set to fine tune the weights.

In all scenarios, the model was trained under a ten fold cross-validation scheme. K-fold Cross Validation (CV) is an alternative to the classical train-validation-test split, that can be adopted to make the final model more robust. In this scenario, portion of the data is still kept as a final test set; the remaining training set is splitted into k distinct folds. Each fold is used exactly once as validation set, for a model built on the remaining folds. This way hyperparameters tuning can be done being sure that all the available train instances are used as validation exactly once.

Usually, after the best parameters have been selected, the model is retrained on the whole training dataset with the chosen hyperparameters, and then tested. In this case instead, a different approach was adopted; the k distinct models available, each one with a different best epoch, were used to make ten distinct predictions on the test set. The final score was then obtained by averaging the single outcomes. This way, the final prediction is more robust, since the variance of the ensemble model is reduced by a factor k.

The main downside of training a model using CV is that the computational time increases, since k individual models need to be trained. For this reason, CV is usually adopted when the dataset has a modest size.

The original model, trained on the public datasets, was not trained using CV, since the dataset was more than 100 times bigger than the bleeding dataset, make it unfeasible to train k independent models in a reasonable time.

Area under the receiver operating characteristic curve (AUC), accuracy precision and recall were measured on the test set for comparison between the alternatives. In this case, accuracy could be considered as a meaningful metric, since the classes in the test set were roughly balanced.

5.2.3 Results

Table 5.3 reports the metrics on the test set for each setup considered.

At a first glance, it can be noticed that freezing all the network layers with the exception of the fully connected one, leads to have below average performances. This is supposedly because the 9 labels initially considered when training the source network are too different from the bleeding target, thus weights need to be fine-tuned for the final task.

The network trained from scratch needed a major number of epochs than the pretrained ones, as the weights were randomly initialized. In the fine tuning setting, only 2 to 3 additional epochs were needed, after which the validation loss started to increase, sign of overfitting.

The pretrained model showed an average improvement of 2.99% in accuracy and 1.39% in AUC compared to the same model with random initialization. Comparison between the ROC curves and the PR curves of the three settings considered can be seen in Figures 5.4 and 5.5 respectively.

It can be noticed in Figure 5.4 that, for low values of FPR, the fine-tuning algorithm performed better than the model trained without imported weights. With the increase of the FPR, the importance of using transfer learning diminished, and the model trained from scratch reached higher TPR. as compared to the fine-tuned model.

In specific domains, such as the healthcare one, it is advisable to have algorithms with low FPR. Further analysis could be done by considering the partial AUC, instead of the full AUC, setting a maximum percentage of FPR that can be tolerated [28, 29].

	Accuracy	AUC	Precision	Recall	F1-Score
Train from scratch	0.6455	0.7197	0.6410	0.5860	0.6123
Train last layer	0.6194	0.6682	0.5956	0.6328	0.6136
Fine tuning	**0.6754**	**0.7336**	**0.6667**	**0.6406**	**0.6434**

Table 5.3: Performance on the test set of the considered settings for the bleeding task

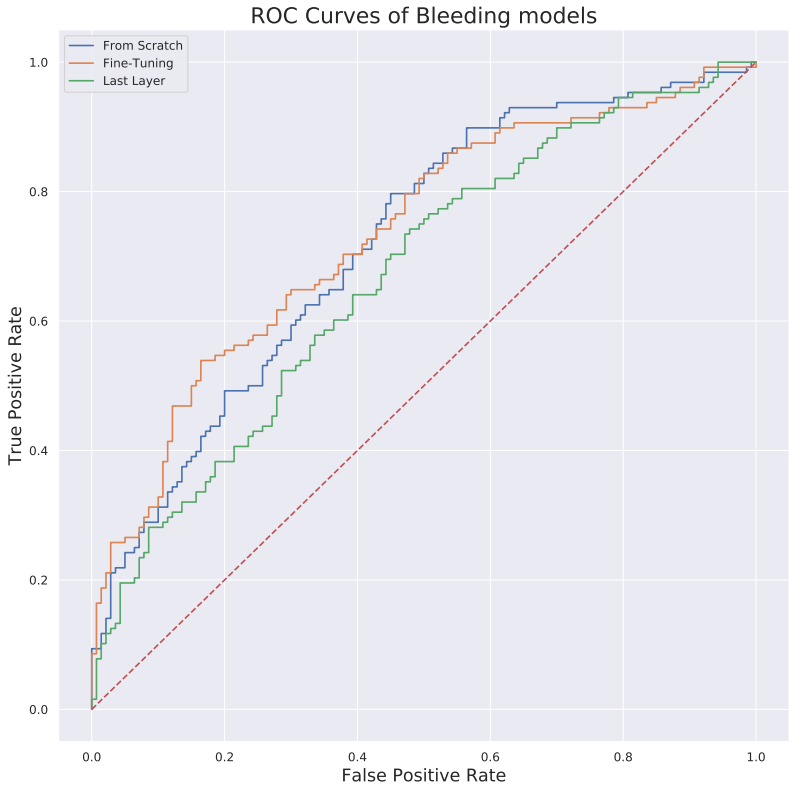


Figure 5.4: Comparison of ROC curves of the three scenarios considered

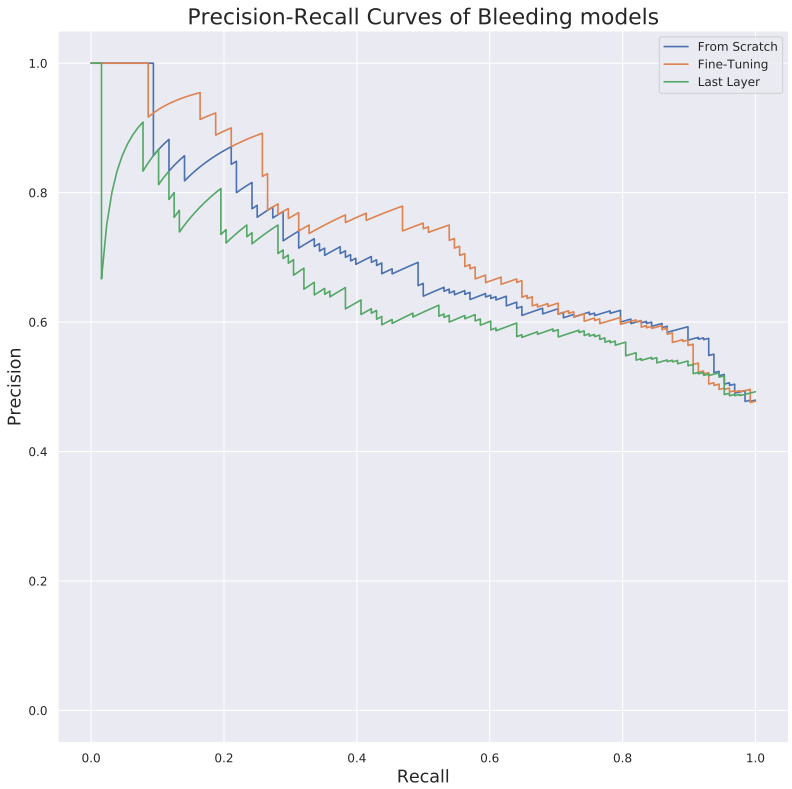


Figure 5.5: Comparison of precision-recall curves of the three scenarios considered

It is important to point out that the final goal of the research was not to create a model that predicts postoperative bleeding on solely chest X-rays; on which the available information is not enough to build a robust classifier, but to verify whether chest radiographs could give further information to the features processed by the RNN used in the previous work done. The results reported in Table 5.3 are therefore just a first step towards the final goal.

Model Interpretability

As bleeding can only be seen on chest radiographs when it is massive, it is interesting to investigate further what are the areas of the image that are considered to be important by the algorithm while making a prediction.

The grad-CAM method was exploited to analyze what areas of the image are considered to be relevant by the algorithm when making a prediction

Figure 5.6 shows results obtained by applying the grad-CAM method to four images in the test set, when the class considered is 1 (bleeding). Each image was chosen to represent a different category of the confusion matrix (TP, FP, TN, FN).

As a general observation, it can be noticed that the algorithm highlighted the lung area, or portion of it, to label an instance as positive (TP and FP). For samples labeled as negatives (TN and FN), the algorithm does not highlight an interesting

region, as a sign that the image did not show any characteristic to think postoperative bleeding would happen to the patient in question.

The images in Figure 5.6 should just be considered as a preliminary result. In order to evaluate whether the algorithm is finding something relevant in the radiographs and evaluate the wrongly labeled images, the help of a radiologist would be needed.

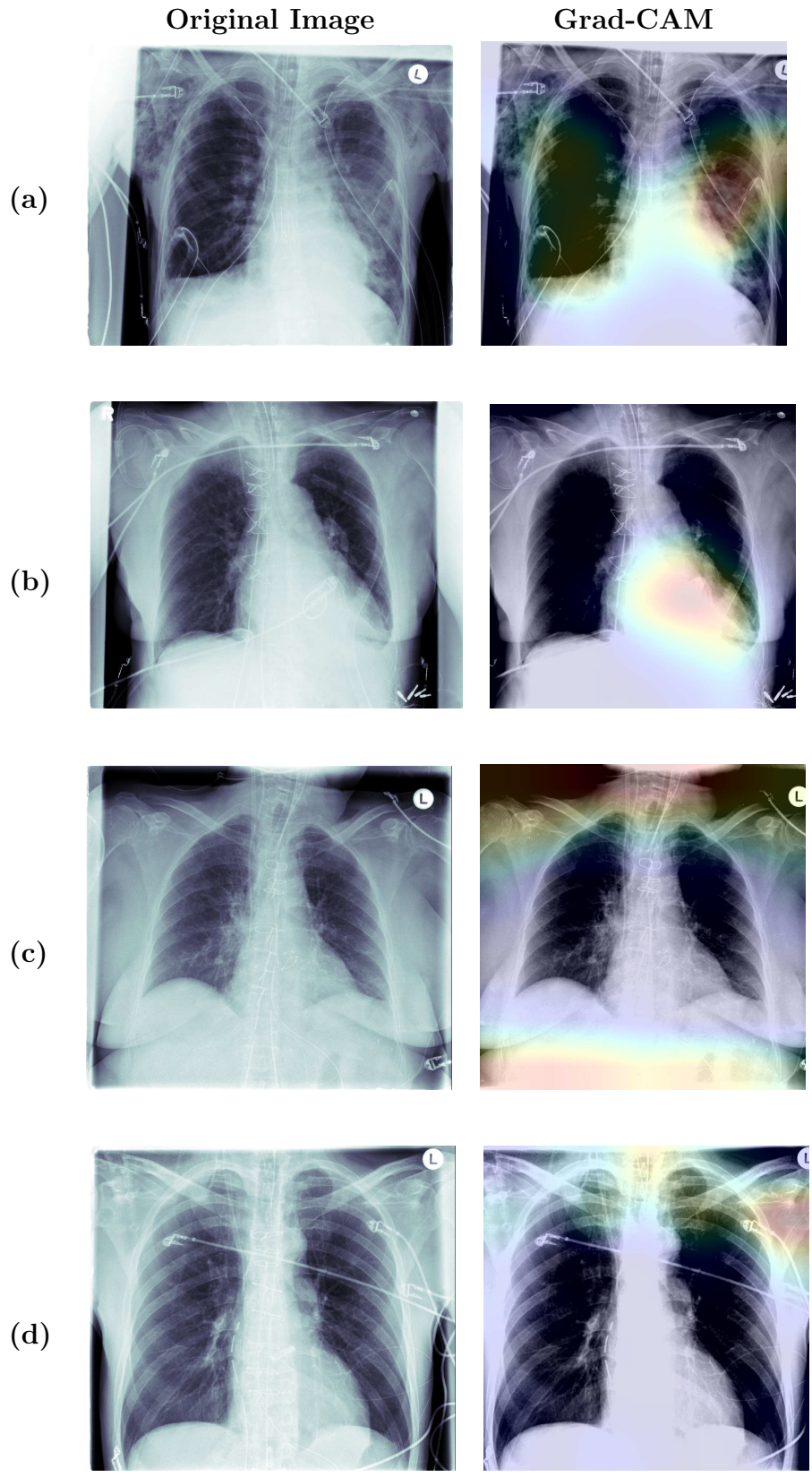


Figure 5.6: Model interpretation for four images in the test set using Grad-CAM. Case in (a) is a true positive. (b) shows a false positive. (c) and (d) represent respectively a true negative and a false negative examples

5.2.4 Challenges

A few challenges were encountered during this phase of the project and are described below.

Image Retrieval Time

Ideally, images should have been taken right after the surgery, within the first 3 hours after the monitor interval starts. 95.05% of the images were taken within the first 3 hours. The remaining radiographs were taken between 3 and 22 hours after surgery. The graph in Figure 5.7 shows the distribution of time difference between the end of the operation and the image capturing moment in the bleeding dataset.

A more important time delta to monitor is represented by the time between the image was taken and the bleeding event occurred, for patients that underwent a re-exploration surgery. It is important to check it because, in cases for which bleeding is massive, signs of it could be seen from the radiograph, therefore images taken closer to the second surgery may contain information that pictures taken earlier don't have.

The graph in Figure 5.8 shows for each bleeding sample in the test set, the time delta between the image was taken and the bleeding occurred, in relation to the error on the prediction of bleeding. It can be noticed than no clear pattern can be defined, as a sign that the prediction was not influenced by the time delta.

The monitoring time varied widely from patient to patient (from 30 minutes to 24 hours). For this reason, considering the absolute time delta may be misleading, as the same delta may mean the image was taken close to the beginning of the monitoring for one image and almost at the end for another. To cope with it, the relative time delta for each instance was computed by dividing the absolute delta with the monitoring length of the sample considered.

Figure 5.9 shows the error in relation with the relative time delta for each image. Two things can be noticed from this picture: first, that most of the images were taken in the first 20% of the monitoring time, a few after the 20% of the time passed and before 40% and only 3 in the last 60% of the monitoring time.

Second, for the 3 images taken in the last 60% of the monitoring time, there is not evidence of a particularly low error. To be sure that bleeding could not be seen on those images, a trained physician manually checked them and stated that there was no sign of hemorrhage.

A possible explanation of this is that the model does not probably consider as relevant to make the predictions the same image characteristics that a physician would, but finds instead patterns that are not easily visible by a human being.

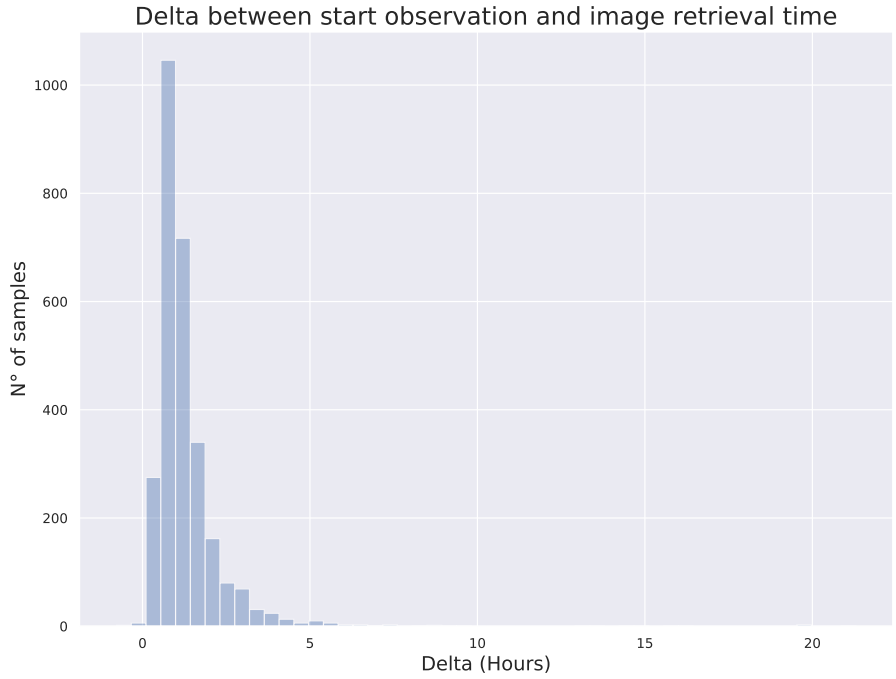


Figure 5.7: Distribution of time delta between the beginning of the monitoring time and the image capturing time for images in the bleeding dataset

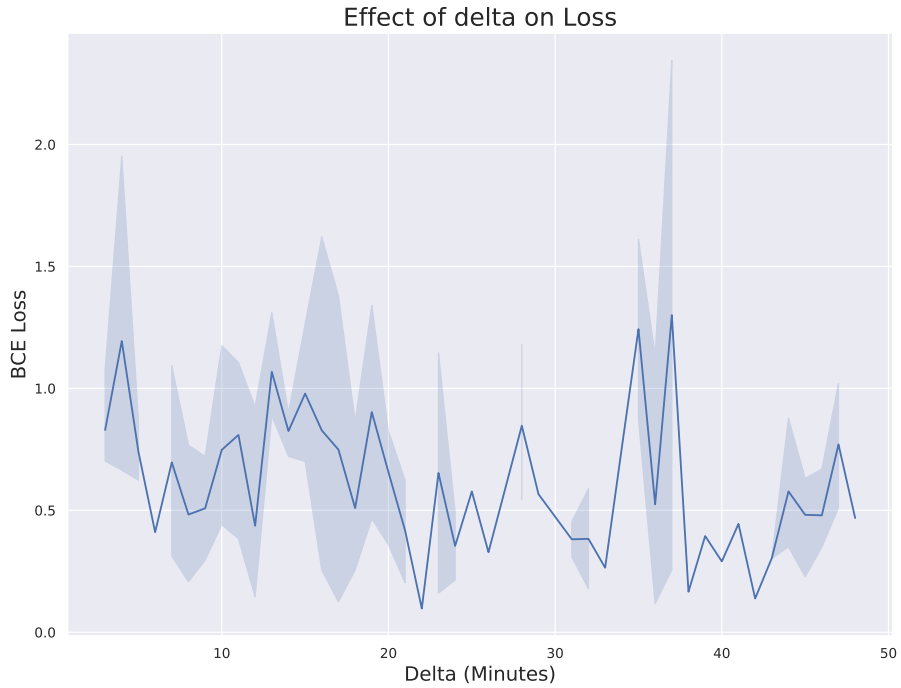


Figure 5.8: Effect of delta between the image capturing time and the end of monitoring window on the prediction error for bleeding patients in the test set

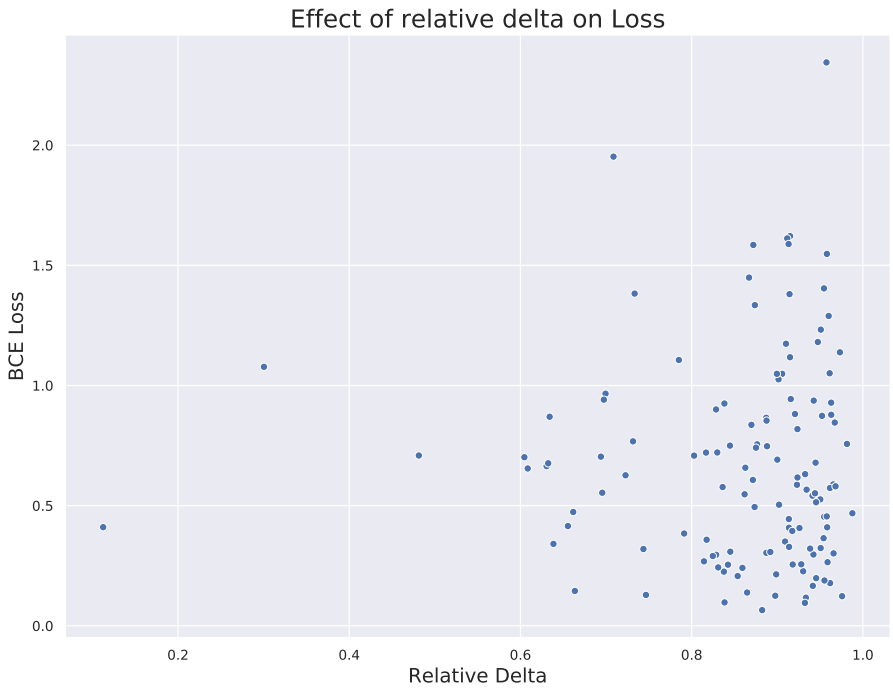


Figure 5.9: Effect of relative delta between the image capturing time and the end of monitoring window on the prediction error for bleeding patients in the test set

Model interpretability can help in understanding what patterns are delineated by the algorithm. A detailed analysis of the results obtained by applying grad-CAM to the test images may be useful to identify such patterns.

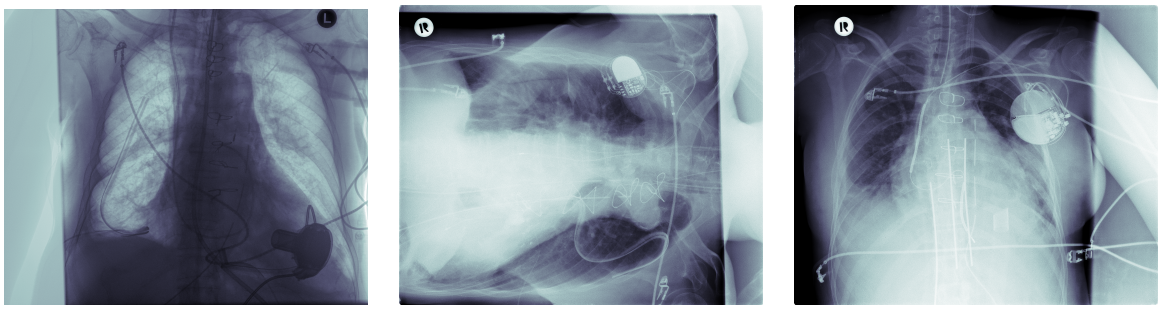
Anomalies

While examining the images retrieved from the system, a second potential problem was encountered. A few images in the bleeding dataset presented some anomalies that were not present in images from the public dataset.

These irregularities are probably common in radiographs and usually do not represent an obstacle for a physician that interprets them. However, these characteristics were not seen in images from the public datasets, on which the original network was trained. For this reason, they may constitute a problem, as the algorithm may face difficulties in classifying images with those anomalies.

The irregularities noticed can be seen in Figure 5.10 and were the following:

- some of the images had an inverted color scale
- the quality of a few images was really low. This is probably justifiable from the fact that all images in the bleeding dataset were taken on bedridden patients, in AP modality, leading to overall lower quality.



(a) (b) (c)

Figure 5.10: Examples of images in the bleeding dataset that presented anomalies. (a) has an inverted color scale, (b) is rotated and (c) presents a rectangular area surrounded by a brighter area.

- rotated images (up to 90 degrees) were present in the dataset. Although slight rotation was considered to augment the data in the public datasets, the algorithm was not trained to cope with images wrongly oriented.
- a rectangular area surrounded by a much brighter area could be noticed in many images. This difference in brightness may somehow influence the outcome.

Chapter 6

Model Integration

The following chapter describes the integration between the recurrent neural network, that processes the sequential information, and the convolutional neural network implemented to handle the X-ray images. Section 6.1 describes in detail the dataset considered. In section 6.2, the different experimental setups that were taken into account are described in detail. Results of the model integration are reported in Section 6.3.

6.1 Dataset

When integrating multiple sources using a late fusion or an intermediate fusion technique, it is easy to implement a model that is able to process missing data, and thus handles instances for which one or more sources is not available: For instance, the importance of each single source in the final prediction can be adjusted when one or more modalities are not available.

On the other hand, the final goal of the thesis was to investigate whether a model processing both chest radiographs and temporal data overcomes the performance of a model that does not consider the images. For this reason, from the original dataset, used to build the RNN in the previous work, only instances for which both the visual and the temporal information was available, were considered. The final dataset size was, therefore, the same as the one used to examine the effect of transfer learning in section 5.2, with a total of 2812 patients, of which 1388 experienced postoperative bleeding.

Another aspect to take into account was the date and time the image was taken. All images available were taken within the monitoring time of each patient. Some of them were taken right after the surgery, while others were obtained after a few hours. When dealing with RNNs, it is essential to avoid using information from a future point in time to make a prediction. For this reason, images should not be taken into

account for time points antecedent their actual capturing time. The X-ray image was, therefore, missing for some of the time points considered.

Moreover, each instance in the dataset had a different sequence length. This is because, if postoperative bleeding was observed at any time, the re-exploratory operation was performed right after, thus the sequence ended at that point. Additionally, to avoid the sequence length to influence the prediction in any way (as shorter sequence would imply that bleeding happened), the sequence lengths of patients from the control group were randomly modified to obtain the same final length distribution for both classes. Around 20% of the patients had a full monitoring time of 24 hours, while for others the time was shorter.

All instances were processed from the beginning of the sequence, even when the image was not available, since the prediction at each time point depends not only on the current values but also on the ones from previous timesteps. However, at each time step, only instances that had both the tabular and the visual data were taken into account to compute the loss and the metrics considered. This way, it was possible to estimate the impact of images on the model.

The plot in Figure 6.1 shows the percentage of data available at each time step. It can be noticed that for the first two time points, the available data was really low, as images were, in most cases, not available yet. For this reason, results on the test set were either impossible to compute (for the first time step, for which there were no samples with both inputs), or highly fluctuating (as only 19 instances were available for the second time point).

To avoid any wrong influence on the metrics, the results on the first two time points were not taken into account in the final average of the single metrics. To have a fair comparison, results for those time points were also not considered in the RNN evaluation either, even if enough data points in the test set were available in this case.

On the contrary, predictions at the end of the sequence were taken into account in the final evaluation, because they were considered robust enough (more than 20% of the test data are available at this point).

6.2 Experimental Setup

A total of three independent experiments were carried out, including the original RNN which was slightly adapted to the new settings and retrained. This section describes the individual setups in detail and the design choices made.

For the multimodality, both intermediate and late fusion strategies were considered. Early fusion strategies did not make sense in this setting, as data would have been destructured for concatenation (e.g. the image tensor should have been

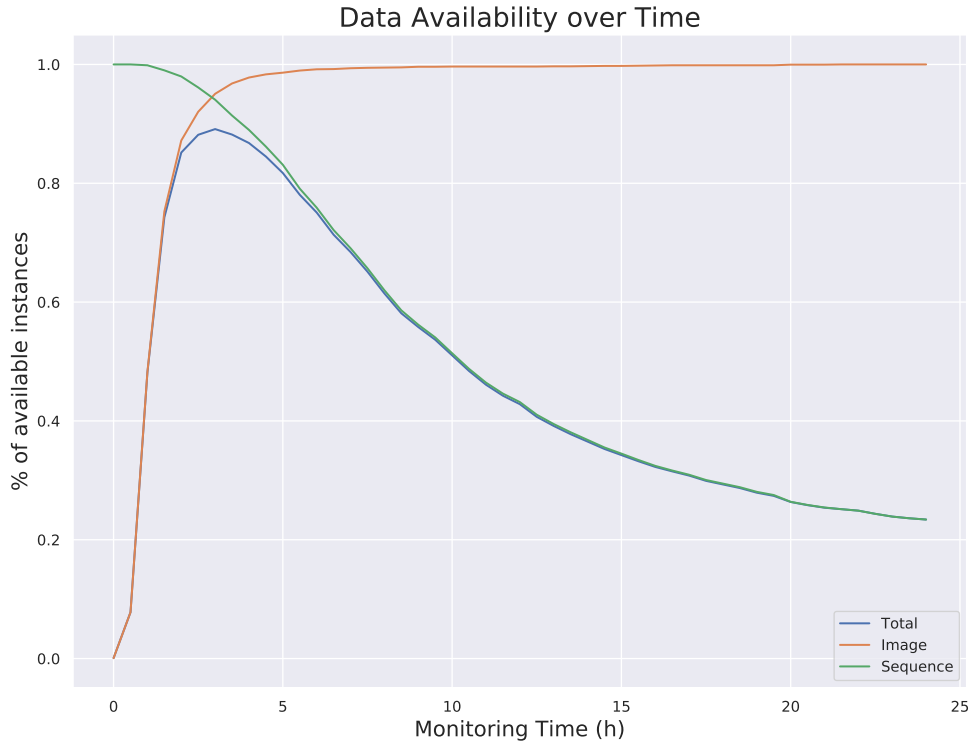


Figure 6.1: Availability of data in the dataset (train and test sets), in terms of image availability, sequence availability and total availability

flattened), thus losing important information and limiting the kind of network that could have been used.

For all the setups, Adam was again adopted as an optimization technique, with a learning rate of 0.001. BCE Loss was the chosen criterion, modified to ignore loss coming from data points with one missing modality.

The number of epochs and batch size were different for the various experiments.

All models were developed under a ten-fold cross-validation scheme, thus building a total of 10 models. For the final prediction on the test set, the 10 individual predictions were averaged. An early stopping technique was used, thus choosing for the final predictions the models from the epoch that reported the lowest loss on the validation fold.

A comparison between the different setups was done only by considering the final predictions on the test sets.

6.2.1 Setup 1: RNN

Before carrying out new experiments, the original recurrent neural network, which was initially implemented in Tensorflow, was re-implemented in Pytorch. Moreover, to have an unbiased comparison between the models, the RNN was retrained by

considering solely the patients for which both the image and the sequential information was available and tested on the final test set of 268 patients.

The RNN used in this phase has the same architecture as the one used previously, one GRU layer with 40 hidden nodes. The only difference between the architectures is that previously the tanh function inside the GRU cell was substituted with a sigmoid activation function, while in this case the original GRU cell, with tanh activation, was used.

The batch size was set to 1, as it was originally. The model was trained for a total of ten epochs. The performances of the new model, trained on a reduced dataset, were slightly lower than the ones of the original model.

6.2.2 Setup 2: Late Integration

In this experiment, the RNN processing the sequential data and the CNN processing the radiographs were trained independently to predict postoperative bleeding. Later on, late integration was done by averaging the results of the two models.

The RNN model was trained by considering the same settings used in experiment 1, while CNN was trained by considering a model pre-trained on the public X-rays datasets and fine-tuned on the bleeding dataset for 5 epochs, as described in section 5.2.

The single outputs were averaged before applying the sigmoid function. A weighted average was considered, and grid search was applied to find the best contribution for each modality.

Finally, the best results were obtained by giving the RNN model slightly more importance, thus making it contribute to the final prediction for 60%. The remaining 40% was obtained by the CNN prediction. A schematic overview of this integration strategy can be seen in Figure 6.2.

6.2.3 Setup 3: Intermediate Integration

In this setting, the potentiality of neural networks, that allow integrating multiple data sources at each point in the processing time, was exploited. The two individual networks were integrated at the end of the computational graph by combining their last layer, then further processed by two additional fully connected layers.

The architecture adopted in this experiment can be seen in Figure 6.3. For each network, the classifying layer was substituted with a layer with more nodes, or dropped (thus considering the outputs of the preceding layer), depending on the settings considered.

Different design choices were considered in this setting. First of all, the weights coming from the two modalities could be combined in different ways:

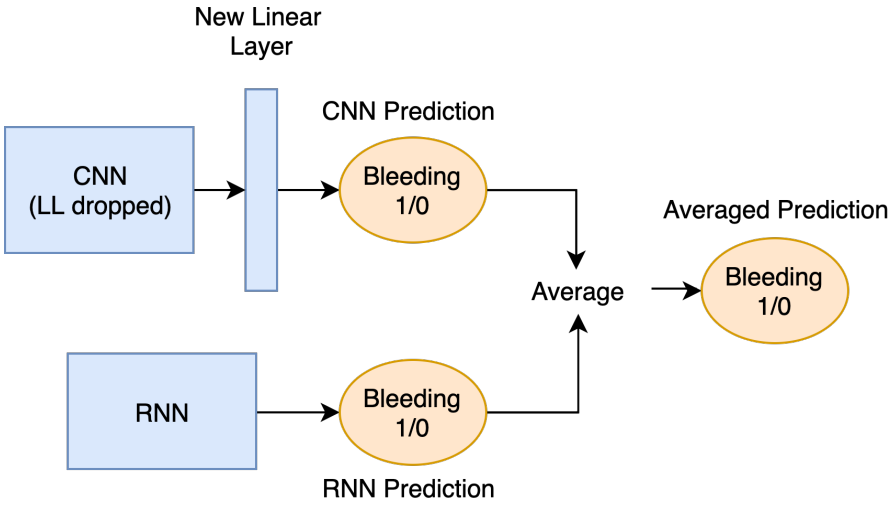


Figure 6.2: Schematic overview of late fusion integration strategy used in setup 2

- **Sum**: the outputs of the single networks were summed. To do that, the last layers of each network needed to have the same dimension. Each network was modified to have a final layer of size 512, whose parameters were then summed to obtain a unique layer.

- **Concatenation**: In this case, the individual linear layers were dropped, the weights coming from the last remaining layer of each modality were concatenated before feeding them to the new FC layers. It should be observed that, in this setting, the number of nodes coming from the CNN was much higher than the ones coming from the RNN (353 and 40 respectively).

- **Random Choice**: in the combination phase, either the weights coming from the RNN or the ones coming from the CNN were considered randomly for each node. Once again, to do that, the final layers of the networks were modified to a final dimension of 512 nodes each. This approach was inspired by an integrated model called Embrace Net [30].

The second aspect to consider was whether to import weights from pre-trained models for the RNN and the CNN or train the integrated architecture from scratch. The options considered were:

- Train the whole model from scratch.

- Import the weights both for the RNN and the CNN, and consider for the latter one the weights coming from the model trained on the public datasets (CheXpert and MIMIC-CXR).

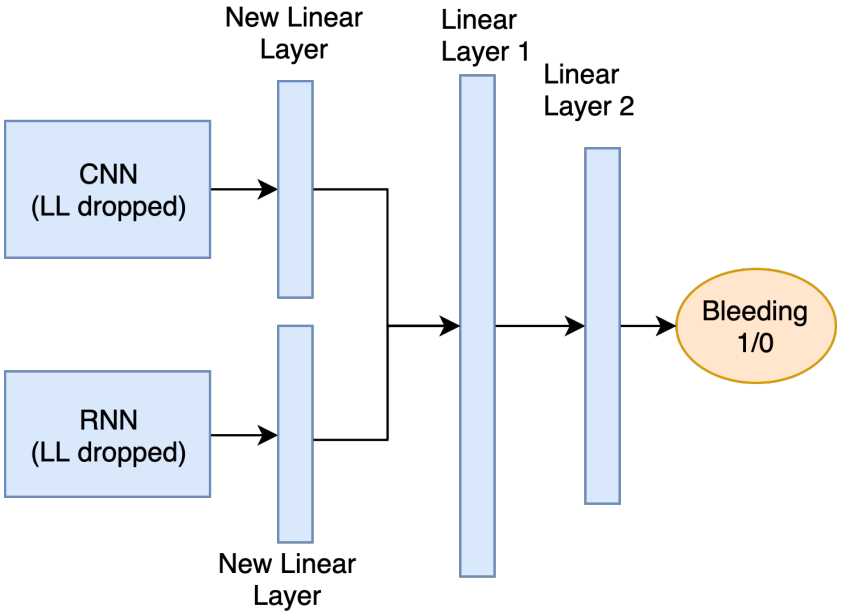


Figure 6.3: Schematic overview of intermediate fusion integration strategy used in setup 3

- Import the weights for both the RNN and the CNN, considering for the latter one the weights coming from the model pre-trained on the public datasets (CheXpert and MIMIC-CXR) and fine-tuned using images from the bleeding dataset.

It is important to point out that, while weights of models trained on different datasets could be imported at any time without issues, those of models trained on the same data used in the integration phase could only be imported when sure that the train and validation set used were the same, otherwise it would not be guaranteed that validation samples were never seen during the training phase. For this reason, when the pre-trained scenario was considered, the monomodal networks were retrained at each cross-validation phase, before the integrated model.

Choosing whether to import parameters from another model, also influenced the last design choice, which was whether to train the entire integrated model or only the FC layers of the network. In this case, the considered options were:

- Train the full model.
- Train only the last linear layers.
- Train first only the last linear layers and fine-tune the whole network for a few final epochs.

The final model architecture was composed of the original recurrent neural network, with the linear layer modified to a final size of 512 nodes, and the DenseNet architecture for the image processing part of the model, again with the last layer modified to match the RNN layer. At this point, weights were summed and further processed by two final fully connected layers.

Different architectural choices could have been made for the final part of the NN, for instance by considering a major number of layers. The final decision of keeping only two was made to avoid adding too much complexity to the model.

During the training phase of the integrated model, weights were imported from pre-trained individual networks, that were trained at each cross-validation step before the integrated classifier. For the image processing model, the model's weights were imported from a classifier trained on the public datasets and then fine-tuned with the images from the DHZB.

When training the multimodal model, the singular network's weights were frozen, thus training only the common FC layers.

Alternative Intermediate Integration Setup

An alternative intermediate integration setup was initially considered. The main idea behind this architecture was to process first the sole image and feed the output of the CNN (before the linear classifier) as an input of the RNN, concatenated to the sequential data.

This way, the images would have potentially had different weights at different timesteps, instead of common parameters for the whole sequence. A schematic overview of this architecture can be seen in Figure 6.4.

A few issues were encountered when considering this model. First of all, this way images would have contributed from the beginning and information from the future would have been used for some points. To cope with this, the CNN weights were set to 0 for all the points in time antecedent the capturing time. Moreover, since the RNN, in this case, takes as input a different vector of data, import weights from a pre-trained model was not possible.

Overall, the performances of this experiment did not show the expected results. Probably, different design choices could have given better performances.

However, there is no particular reason why images could be weighted differently at different timesteps. For this reason, the initial intermediate integration strategy was the one considered in the final evaluation.

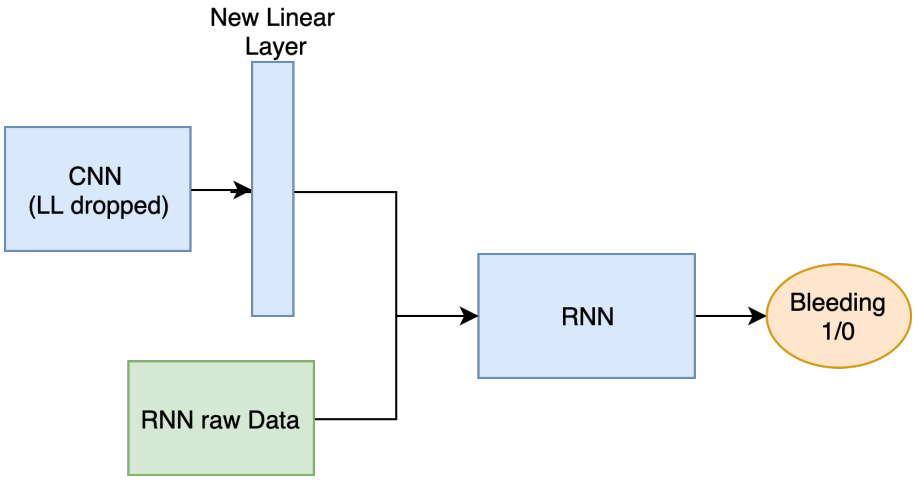


Figure 6.4: Schematic overview of the alternative intermediate fusion integration strategy proposed

6.3 Results

The final results for each experiment are summarized in Table 6.1. Moreover, ROC Curves and precision-recall curves for each classifier considered can be observed in Figures 6.5 and 6.6 respectively.

The performances presented were obtained by micro-averaging, thus averaging the predictions of each time point for each patient in the test set. The alternative to micro-averaging is macro-averaging, thus averaging the mean performances of each of the 49 time points (1 every 30 minutes for 24 hours). This approach was not adopted, due to the unbalanced distribution of available samples per each time step in the test set.

As expected, the recurrent neural network, trained on the reduced dataset of 2812 patients obtained results slightly worse than the ones of the initial model, trained on a bigger dataset. On the reduced dataset, both accuracy and AUC experienced a decrease of around 2%. This diminishment was considered negligible.

The results in Table 6.1 show that both the integration strategies proposed outperformed the monomodal model, with a slightly higher performance obtained with the late integration strategy.

To ensure that the results obtained were significant, a Wilcoxon signed-rank test was applied to the single accuracies for each time point in the test set. Wilcoxon test was considered instead of a more traditional t-test, to avoid making wrong assumptions on the distribution of the data.

The test was executed by taking each time-point considered for each patient in the test set (for a total of 6040 timepoints belonging to 268 patients) and assign 1 to it if the classifier's prediction for that point was correct, 0 otherwise (thus considering

	Accuracy	AUC	Precision	Recall	F1-Score
RNN	0.7795	0.8472	0.8509	0.6932	0.7672
Late Integration	**0.8386**	**0.8888**	**0.8737**	0.8012	**0.8359**
Intermediate Int.	0.8270	0.8853	0.8383	**0.8212**	0. 8297

Table 6.1: RNN and clinical baseline overall performance on the test set

the accuracies of the classifier on every single time point). Each classifier considered was then compared pairwise to the other two classifiers, based on those accuracies per timepoint.

Each statistical test verified the null hypothesis, which stated that there was no significant difference between the two models tested, versus the alternative hypothesis, according to which one model performed better than the other.

When testing the recurrent neural network versus the integrated model, the statistical test confirmed that the null hypothesis could be rejected, at a significance level of 0.01 (with a p-value much smaller than 0.01). Therefore, it can be said that the models including the chest radiographs performed significantly better than the model considering only the temporal data.

On the other hand, when the two integration experiments were tested together, the p-value obtained was 0.086, thus it could not be said, at the chosen alpha level of 0.01, that one integration strategy was better than the other.

Overall, adding images to the RNN model lead to an increase of AUC of 3.19 % and an improvement of the model's accuracy of 5.91 %.

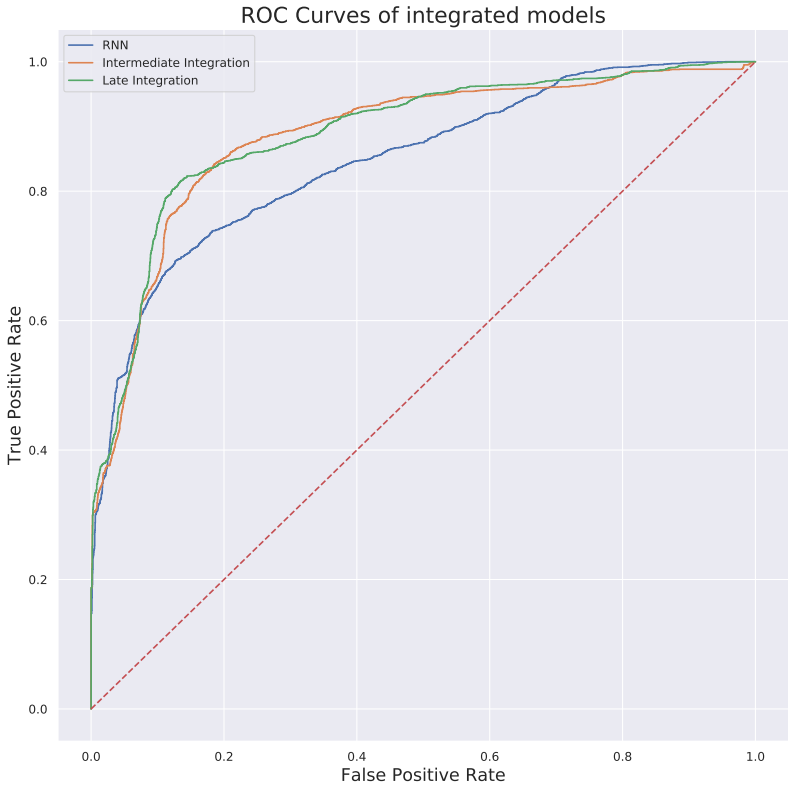


Figure 6.5: Comparison of ROC curves of the three experiments considered

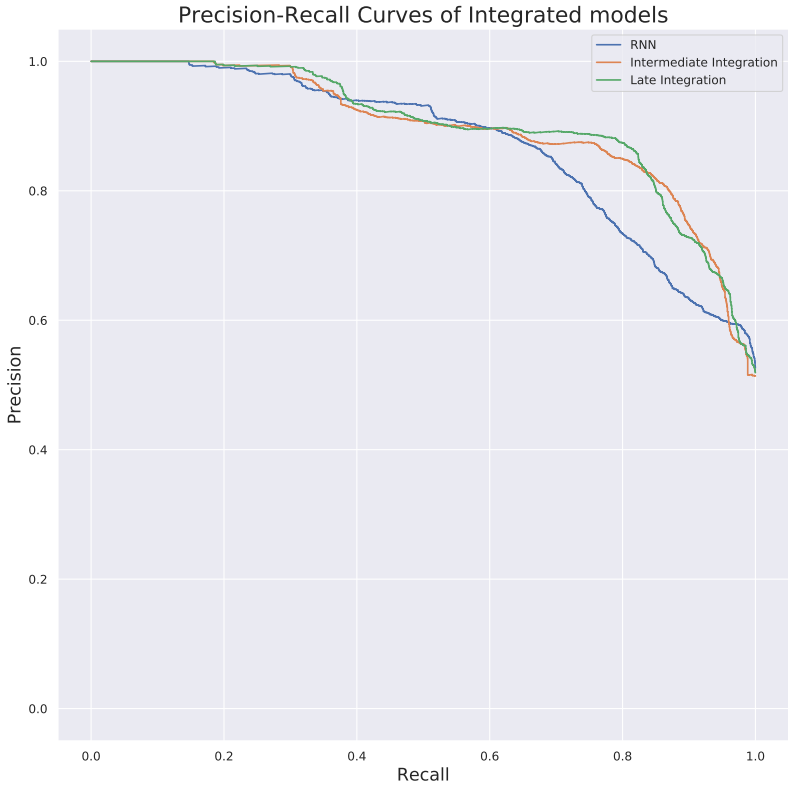


Figure 6.6: Comparison of precision-recall curves of the three experiments considered

6.3.1 Performance Comparison over Time

The results obtained can be further examined by observing another perspective: the performances obtained over time. Figures 6.7 and 6.8 shows respectively the accuracy and the AUC of the three setups considered in the 24 hours after the surgery.

A few observations can be made on those two plots. First of all, it can be noticed that the positive impact of images on the prediction increases over time. Initially, the performances of the three experiments seem to be comparable, while toward the end of the monitoring period, the gap between the classifiers becomes remarkable. While the accuracy over time for the integrated models keeps increasing, the accuracy of the RNN model flattens after an initial increase. After 24 hours, the gap between the integrated models and the RNN reaches 10% in accuracy.

The graph reporting the availability of the data at each timestep, in Figure 6.1 is useful to further investigate the trend. In it, it can be noticed that the amount of data used to build and test both the RNN and the integrated models towards the end is the same since at this point the chest radiograph was available for almost all the patients, thus the only data influencing the availability is the temporal information. On the other end, at the beginning of the sequence, indicatively for the first 7 hours of the monitoring period, the data available for the integrated models were less than the ones available for the single modality classifier.

The missing data at the beginning of the monitoring period justifies why the integrated model, which is initially trained and test on fewer samples, is not performing undoubtedly better from the beginning.

Secondly, it can be seen that the AUC of the three models follows a similar trend over time, while the accuracy fluctuates in a slightly different way. This is most likely because the threshold to decide whether to consider the case as bleeding or not is always set to 0.5, no matter the time step considered, while AUC shows the results independently from the threshold value. It may be possible that different thresholds should be considered for different points in time.

The value of the threshold highly depends on the task under evaluation. Setting the threshold to 0.5 is just a convention. However, further investigation should be made in this specific case to decide whether a different threshold would make more sense.

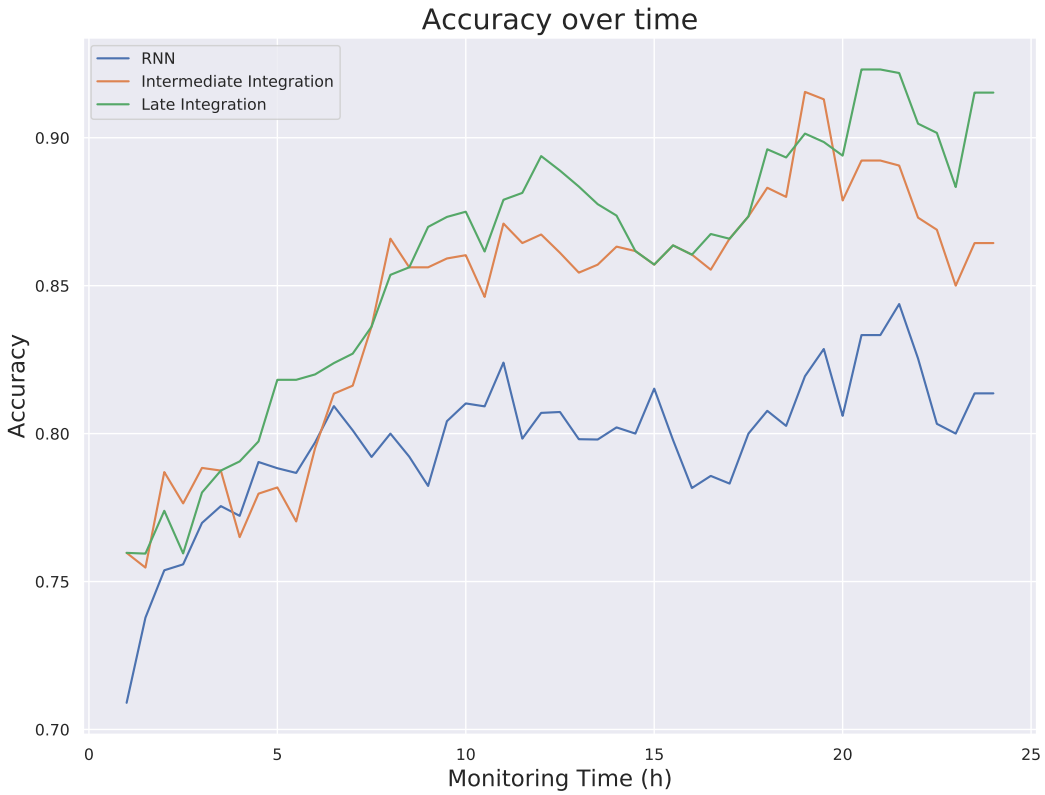


Figure 6.7: Mean accuracy over time-slices for the three setups considered

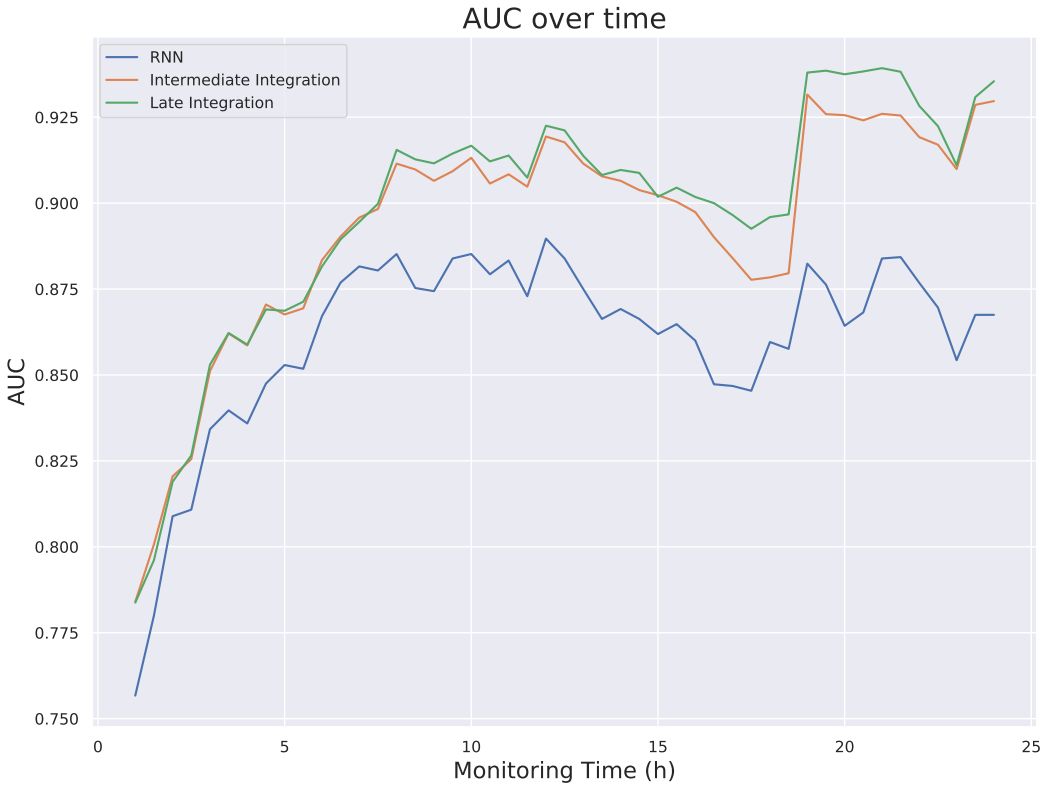


Figure 6.8: Mean area under curve over time-slices for the three setups considered

Increasing the threshold would mean labeling a case as bleeding with higher confidence. This way, the number of false positives, which represents cases in which bleeding is detected by the algorithm but not happening, decreases, together with the number of re-exploratory surgeries that would not be necessary.

On the other hand, it also means that the number of bleeding patients for which hemorrhage is not discovered by the algorithm increases, risking to detect the bleeding too late.

Decide whether to consider a different threshold should be discussed with a domain expert, that is able to state the percentage of false positives or false negatives that can be tolerated in the hospital's settings.

It may make sense to consider higher thresholds at the beginning of the monitoring time so that at this point only patients for which there is high confidence that the bleeding is ongoing are re-operated. The threshold could be then lowered so that at late points in time the risk of missing bleeding cases lowers.

Chapter 7

Conclusions

This thesis aimed at building a deep learning multimodal model for the prediction of complications in critical care and subsequently investigating the contribution of chest radiographs taken after cardiac surgery, for the prediction of postoperative bleeding on patients that have undergone major open-heart surgery in the monitoring time of 24 hours following the operation.

Firstly, a model processing solely images was developed and trained using two public datasets with more than 400,000 chest radiographs. It should be once again underlined that the final goal of this project was to bring additional information to the original RNN model, not to use images by themselves, since a model considering solely chest radiographs would not be robust enough to base the decision only on its outcome. Nonetheless, the first major finding of this project showed how transfer learning can be exploited in cases where not enough images are available, which is pretty common in healthcare projects, to improve the performances of a model. It should be once again pointed out that using a model pre-trained on the same kind of images (chest radiographs) was essential, as models pre-trained on more general datasets, such as ImageNet, usually do not perform better than non-pre-trained models on medical imaging classification tasks [20].

Secondly, different integration strategies were analyzed and implemented. Finally, two integrated models were chosen for the comparison with the RNN model, one adopting a late integration strategy, the other considering intermediate integration.

The experiments executed showed that including X-rays in classifiers that process the patient's vital parameters to predict unexpected complications allows building a model able to reach an accuracy of 0.8386 and a ROC AUC of 0.8888, outperforming the model that only considers temporal parameters by 5.91% in accuracy and 3.19% in AUC.

A few criticalities were encountered during the thesis development. First of all, the original dataset of patients had to be reduced, since chest radiographs

were not available for some patients, or were taken outside the monitoring period. Moreover, images should have been available for all patients from the beginning of the observational time. Unfortunately, in real settings, it usually takes some time between the end of surgery and the time the chest X-ray is taken, which may be influenced by various factors. For this reason, particular care was taken when integrating the data, to avoid building a model that considers information from a future point in time to make predictions.

The thesis was composed of various intermediate steps, each of which influenced the results of the following one. As described, different design choices were made at each phase. Due to the high number of choices and hyperparameters to tune, an exhaustive search was not possible. Potentially, a different choice taken at the beginning may have changed the outcome of the project. However, the most common settings were tested at each phase, leading to believe that the choices made were the most suitable for the task considered.

Machine learning methods are nowadays applied to various domains, thanks to their great potential in discovering non-conventional patterns in the data analyzed. However, it should be pointed out that the results obtained do not always carry valuable information or are not easily interpretable. To verify and enhance results obtained by an ML method, it is advisable to discuss potential choices and results with a domain expert.

In this thesis, the help of a physician was exploited in different situations, both to check the correctness of the input data, which is essential to building a robust classifier, and to make some model choices during the development phase. For instance, a domain expert was asked to define the targets that may relate to the bleeding task, among the ones available in [10] and [11].

It may have been of great help, but too time-consuming due to the size of the dataset, to have a clinician manually check the images retrieved from the hospital's PACS, to estimate the quality of the images and potentially retrieve new ones in case of particular issues.

The help of a physician could also be further employed in the future. In particular, a detailed discussion could be helpful to evaluate the possibility of adopting a different threshold or even different thresholds for different points in time.

Moreover, it may be useful to examine instances in which the chest X-ray image processing model performed particularly well or particularly poorly with a radiologist, to have an idea of why it is happening.

To have a complete view of what the model considers as important in an image to make the prediction, the interpretability of the image processing model should be further investigated by showing a radiologist the results obtained using the grad-CAM algorithm on the test set images to try to define common patterns and important

areas.

The interpretability method could be also applied to the multimodal model, to identify not only the most relevant pixels in the image but also the most important features within the sequential parameters.

As future work, the multimodal models developed could be extended for the prediction of the other two targets considered in [1]: renal failure and mortality.

To sum up, the thesis verified that a classifier built for the early prediction of postoperative bleeding, that can be exploited in real-time settings to support the doctor's diagnosis, can benefit from the information contained in chest radiographs to enhance its performance.

Similarly to other practical projects, the problems encountered during the development, regarding the data quality and availability, may have partially affected the final outcome. Building such a model with higher quality data may bring further improvement to the performance.

The best classifier developed adopted a late integration strategy, thus averaging the prediction of two single modality models and achieved a final AUC of 0.8888.

Acronyms

ANN	Artificial Neural Network
AP	Antero-Posterior
AUC	Area Under Curve
AUPRC	Area Under Precision-Recall Curve
BN	Batch Normalization
BCE	Binary Cross Entropy
CAM	Class Activation Mapping
CNN	Convolutional Neural Network
CSV	Comma Separated Value
CV	Cross Validation
DHZB	Deutsches Herzzentrum Berlin - German Heart Center Berlin
DICOM	Digital Imaging and COmmunications in Medicine
DL	Deep Learning
FC	Fully Connected
FN	False Negative
FP	False Positive
FT	Fine Tuning
FPR	False Positive Rate
GD	Gradient Descent
GPU	Graphics Processing Unit

GRU	Gated Recurrent Unit
ICU	Intensive Care Unit
LR	Learning Rate
LSTM	Long-Short Term Memory
ML	Machine Learning
MLML	Multi-Label Missing Label
NN	Neural Network
OTS	Off-The-Shelf
PA	Posterior-Anterior
PACS	Picture Archiving and Communication System
RGB	Red-Green-Blue
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
SGD	Stochastic Gradient Descent
SOTA	State Of the Art
TN	True Negative
TP	True Positive
TPR	True Positive Rate

Bibliography

[1] Alexander Meyer et al. "Machine learning for real-time prediction of complications in critical care: a retrospective study". en. In: *Lancet Respir Med* 6.12 (Dec. 2018), pp. 905–914 (cit. on pp. vii, ix, 1, 2, 38, 81).

[2] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015 (cit. on p. 17).

[3] B. Van Ginneken, B. M. Ter Haar Romeny, and M. A. Viergever. "Computer-aided diagnosis in chest radiography: a survey". In: *IEEE Transactions on Medical Imaging* 20.12 (Dec. 2001), pp. 1228–1241. ISSN: 1558-254X (cit. on p. 27).

[4] IC Mehta, ZJ Khan, and RR Khotpal. "Analysis and review of chest radiograph enhancement techniques". In: *Information Technology Journal* 5.3 (2006), pp. 577–582 (cit. on p. 27).

[5] Heber MacMahon et al. "Dual energy subtraction and temporal subtraction chest radiography". In: *Journal of thoracic imaging* 23.2 (2008), pp. 77–85 (cit. on pp. 28, 29).

[6] Linda G Shapiro and George C Stockman. *Computer vision*. Prentice Hall, 2001 (cit. on p. 28).

[7] Jun-Ichiro Toriwaki et al. "Pattern recognition of chest X-ray images". In: *Computer Graphics and Image Processing* 2.3-4 (1973), pp. 252–271 (cit. on p. 28).

[8] Xiaosong Wang et al. "Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2097–2106 (cit. on p. 30).

[9] Aurelia Bustos et al. "Padchest: A large chest x-ray image dataset with multi-label annotated reports". In: *arXiv preprint arXiv:1901.07441* (2019) (cit. on p. 30).

[10] Jeremy Irvin et al. *CheXpert: A Large Chest Radiograph Dataset with Uncertainty Labels and Expert Comparison*. 2019 (cit. on pp. 30, 32, 34, 35, 43, 45, 47, 52, 80).

[11] Alistair E. W. Johnson et al. *MIMIC-CXR: A large publicly available database of labeled chest radiographs*. 2019 (cit. on pp. 30, 45, 80).

[12] *Open-i: An open access biomedical search engine*. URL: https://openi.nlm.nih.gov. (cit. on p. 31).

[13] Wei Yang et al. "Cascade of multi-scale convolutional neural networks for bone suppression of chest radiographs in gradient domain". In: *Medical image analysis* 35 (2017), pp. 421–433 (cit. on p. 31).

[14] Alexey A Novikov et al. "Fully convolutional architectures for multiclass segmentation in chest radiographs". In: *IEEE transactions on medical imaging* 37.8 (2018), pp. 1865–1876 (cit. on p. 32).

[15] Mohammad R Arbabshirani et al. "Accurate segmentation of lung fields on chest radiographs using deep convolutional networks". In: *Medical Imaging 2017: Image Processing*. Vol. 10133. International Society for Optics and Photonics. 2017, p. 1013305 (cit. on p. 32).

[16] Junji Shiraishi et al. "Development of a digital image database for chest radiographs with and without a lung nodule: receiver operating characteristic analysis of radiologists' detection of pulmonary nodules". In: *American Journal of Roentgenology* 174.1 (2000), pp. 71–74 (cit. on p. 32).

[17] Pranav Rajpurkar et al. "Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning". In: *arXiv preprint arXiv:1711.05225* (2017) (cit. on p. 32).

[18] Ivo M Baltruschat et al. "Comparison of deep learning approaches for multi-label chest X-ray classification". In: *Scientific reports* 9.1 (2019), pp. 1–10 (cit. on p. 32).

[19] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255 (cit. on p. 32).

[20] Maithra Raghu et al. *Transfusion: Understanding Transfer Learning for Medical Imaging*. 2019 (cit. on pp. 32, 55, 79).

[21] Bolei Zhou et al. "Learning deep features for discriminative localization". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2921–2929 (cit. on p. 33).

[22] Ramprasaath R Selvaraju et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 618–626 (cit. on p. 34).

[23] Katrine Lawaetz Kristensen et al. "Reoperation for bleeding in cardiac surgery". In: *Interactive cardiovascular and thoracic surgery* 14.6 (2012), pp. 709–713 (cit. on p. 37).

[24] Paul Eisenberg and Nicholas Pesa. "Perioperative complications of cardiac surgery and postoperative care". In: *Cardiac Emergencies in the ICU, An Issue of Critical Care Clinics, E-Book* 30.3 (2014), pp. 527–555 (cit. on p. 37).

[25] Guillermo Gutierrez, HDavid Reines, and Marian E Wulf-Gutierrez. "Clinical review: hemorrhagic shock". In: *Critical care* 8.5 (2004), p. 373 (cit. on p. 37).

[26] M. Chen, T. Pope, and D. Ott. *Basic Radiology*. A Lange Medical Book. McGraw-Hill Companies,Incorporated, 2004. ISBN: 9780071410267 (cit. on p. 42).

[27] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf (cit. on p. 50).

[28] Hua Ma et al. "On use of partial area under the ROC curve for evaluation of diagnostic performance". In: *Statistics in medicine* 32.20 (2013), pp. 3449–3458 (cit. on p. 56).

[29] Donna Katzman McClish. "Analyzing a portion of the ROC curve". In: *Medical Decision Making* 9.3 (1989), pp. 190–195 (cit. on p. 56).

[30] Jun-Ho Choi and Jong-Seok Lee. "EmbraceNet: A robust deep learning architecture for multimodal classification". In: *Information Fusion* 51 (Nov. 2019), pp. 259–270. ISSN: 1566-2535 (cit. on p. 69).

[31] *Stanford Course CS231n: Convolutional Neural Networks for Visual Recognition*. URL: http://cs231n.stanford.edu/index.html.

[32] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[33] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 0387310738.

[34] Jürgen Schmidhuber. "Deep learning in neural networks: An overview". In: *Neural networks* 61 (2015), pp. 85–117.

[35] Sebastian Ruder. *An overview of gradient descent optimization algorithms*. 2016.

[36] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014.

[37] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015.

[38] *Understanding LSTM Networks*. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[39] Andrej Karpathy. *The unreasonable effectiveness of recurrent neural networks*. URL: http://karpathy.github.io/2015/05/21/rnn-effectiveness/.

[40] Gao Huang et al. *Densely Connected Convolutional Networks*. 2016.

[41] Lorien Y. Pratt, Jack Mostow, and Candace A. Kamm. "Direct Transfer of Learned Information Among Neural Networks". In: *Proceedings of the Ninth National Conference on Artificial Intelligence - Volume 2*. AAAI'91. Anaheim, California: AAAI Press, 1991, pp. 584–589. ISBN: 0-262-51059-6.

[42] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. *Multimodal Machine Learning: A Survey and Taxonomy*. 2017.

[43] Jiquan Ngiam et al. "Multimodal deep learning". In: *Proceedings of the 28th international conference on machine learning (ICML-11)*. 2011, pp. 689–696.

[44] Jennifer Williams et al. "DNN Multimodal Fusion Techniques for Predicting Video Sentiment". In: *Proceedings of Grand Challenge and Workshop on Human Multimodal Language (Challenge-HML)*. Melbourne, Australia: Association for Computational Linguistics, July 2018.

[45] Christian Szegedy et al. "Inception-v4, inception-resnet and the impact of residual connections on learning". In: *Thirty-first AAAI conference on artificial intelligence*. 2017.

[46] Jie Hu, Li Shen, and Gang Sun. "Squeeze-and-Excitation Networks". In: *CoRR* abs/1709.01507 (2017). URL: http://arxiv.org/abs/1709.01507.

[47] Marco Ranucci et al. "Surgical reexploration after cardiac operations: why a worse outcome?" In: *The Annals of thoracic surgery* 86.5 (2008), pp. 1557–1562.

[48] Robert M Bojar. *Manual of perioperative care in adult cardiac surgery*. John Wiley & Sons, 2009.

[49] H. Singh and J. Neutze. *Radiology Fundamentals: Introduction to Imaging & Technology*. Springer New York, 2011. ISBN: 9781461409441.

[50] Dina Zverinski. "Prediction of Postoperative Complications in Cardiac Surgery". MA thesis. Eidgenössische Technische Hochschule Zürich (ETH Zürich), 2017.

[51] *DICOM: Key Concepts*. URL: https://www.dicomstandard.org/concepts/.

[52] Terrance DeVries and Graham W Taylor. "Improved regularization of convolutional neural networks with cutout". In: *arXiv preprint arXiv:1708.04552* (2017).