Data Challenge 2020 - Predicting lung cancer survival time, Owkin

Antoine Yang, Léon Zheng
ENS Paris-Saclay
{antoine.yang, leon.zheng}@polytechnique.edu

Abstract

As part of the "Multi-Scale model and Convolutional Neural Networks" MVA course teached by Stéphane Mallat, we choose to take part to the Owkin challenge. The task is to design a model to best predict lung cancer survival times from CT scan images and clinical data. For that, we start by analyzing the data and radiomics features provided. Then we present our two complementary heuristics to build a model: one built on traditional features, and another built on deep features extracted without any a priori information. In the first one, we start by showing that the problem, because of right censored data, cannot be reduced to a simple regression approach. We then reproduce the Cox PH baseline model. To improve it, we extract radiomics features also from Laplacian of Gaussian and wavelet transforms of scan images. However, we observe that a large number of radiomics features are highly correlated and non informative. Therefore we present multiple feature selection (and associated cross-validation model training) approaches: recursive feature elimination, LASSO regression and Random Survival Forest. In a second approach, we aim at extracting powerful features from CT scans without any prior information. For that, we first simplify the problem by only considering the 2D slice with most tumor for each patient, and try to extract features with a pretrained ResNet-18 model. To improve it, we finetune the ResNet by integrating it into a network that takes as input the best 2D slice to predict risk, and that can be trained with negative log partial likelihood loss and used to predict survival times in an end-to-end fashion. We also evaluate the possibility of finetuning ResNet with a simple classification problem. To improve the end-to-end approach, we further add information, by computing in parallel an attention map from the associated mask. We notably analyze the previous results using Grad-Cam. To exploit all image information provided, we finally propose a semantic segmentation 3D U-Net model used both to recover missing masks and for feature extraction. Our best public test submission yields an intermediate academic score of 77.26% C-Index, ranked 1st out of 9.

1. Data Analysis

The goal of this challenge is to predict survival time of patients diagnosed with lung cancer, based on personal clinical data and 3-dimensional radiology images scanned by computed tomography (CT). The evaluation metric of the challenge is the C-Index [9], which measures how well the model is able to rank the orders of survival time.

The training dataset contains 300 patients: 162 patients who died during the experience, and 138 patients who escaped the study. The small size of the training set is a major difficulty of the challenge. For each patient, we are given clinical data, CT scan images with their masks, and 53 radiomics features extracted by the challenge providers.

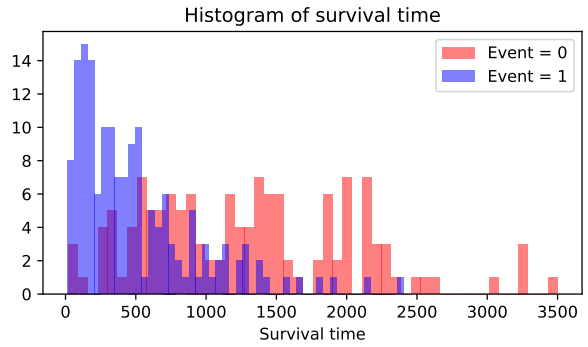


Figure 1: Histogram of patients' survival time in the training set.

Supervision The provided labels are right censored and can be thought as a weak supervision: in the training set, each patient is described by an observation time and an event indicator. The former is the time at which we observe the patient. The latter is a binary class and represents whether the patient is alive (Event $= 0$) or not (Event $= 1$) at this time.

Clinical data The clinical data describes each patient by its age, the source data it originates from, and the TNM

staging of the cancer. It also contains information about the type of cancer cells : adenocarcinoma, large cell, squamous cell carcinoma. In order to use these features in a regression model for survival analysis, we preprocess the data and use dummies variables instead of categorical variables for the type of cancer cells and for the source dataset. We choose to fill the missing ages by the mean age value.

Images and radiomics features Description in Section 1.1 and 1.2.

Missing masks Some images are provided with an empty mask, although provided radiomics features actually describe a tumor for these patients (Patient ID 256, 3, 391, 327, 263 in the training set, and Patient ID 198, 199, 234, 50, 311, 404 in the test set). For the training set, we will not consider patients without mask when we use features derived from an extraction using the mask. For the test set, when we use features that have to be extracted with a mask, we will use the predicted mask of our semantic segmentation model (see Section 3.5).

1.1. Images analysis and visualization

We are given 300 train images and 125 test images of size $92 \times 92 \times 92$ (1-grey-channel 3D images) representing CT scans and binary masks of the same size indicating the presence or absence of a tumor at each pixel. There are many challenges that arise from the task of extracting useful information from the images: we have a small training set, the images themselves are rather small (which might actually be an advantage in terms of computational requirement), they are 3-dimensional with 1 grey channel (while most recent Computer Vision methods study 2D RGB images), many slices barely contain any tumor pixel (although we have the corresponding masks).

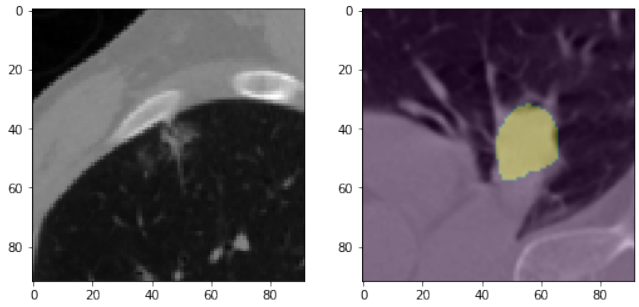


Figure 2: Image slices examples without/with mask

We used matplotlib, skvideo and io to visualize the scans and the masks. We noticed that images are generally nicely centered around the tumor, that the scale is rather similar from one scan to another, that some 2D slices contain no or little information about the tumor and that the patients with missing mask actually have a tumor.

1.2. Radiomics features analysis

Radiomics features are quantitative features computed from radiology imaging data, for which the tumoral zones have been annotated by medical experts. These annotated zones are the masks of the images. We will see that these features can encode prior information that we know about the problem.

The 53 radiomics features provided by the challenge have been extracted from the original CT images (and we can guess this has been done with the ground truth masks for the 11 missing masks), using `PyRadiomics` package [22]. In order to have a deeper understanding of this library, we reproduced the radiomics feature extraction, converting to Simple ITK format all scans and masks and using the `PyRadiomics` feature extractor from `Python`. In fact, one can extract in total 130 features from the original CT images. However, we will directly use the provided 53 features in our pipeline and ignore the other original features, because they already cover most important aspects according to the challenge providers. For each 3D CT image provided with its mask, `PyRadiomics` extractor can compute three kind of features: shape features (shape module), intensity features (first order module), textural features (GLCM, GLSZM, GLRLM modules).

Shape features These features describe the three-dimensional size and shape of the tumor region, which is delimited in the mask by an expert. The shape features include the volume of the tumor, surface of the tumor, compactness of the tumor, spherical disproportion, diameter of the tumor in 3D or in a 2D slice. These features represent the prior information that the shape of the tumor has an impact on the prediction of survival time. Indeed, we can assume that a patient with a bigger tumor has a low chance of survival.

Intensity features These features are basic first order statistics describing the distribution of voxel intensities within the image region defined by the mask, including energy, entropy, mean, standard deviation, minimum, maximum, quantile, skewness, Kurtosis (peakness), uniformity... They can summarize quantitatively the gray intensity information contained in an image.

Textural features These are quantitative features analyzing image texture, computed from a descriptor of gray level intensities. Discretization of gray intensities reduces noises but also normalizes intensities across all the patients, which allows for direct comparison of the computed features between the patients. In the GLCM module [8], the descriptor

is the Gray Level Co-occurrence Matrix, which represents the probability that a combination of two pixels of intensity i and j appears in the image, given that the two pixels are separated by a given distance and a given orientation. In the GLSZM module [20], the descriptor is the Gray Level Size Zone matrix, which counts the number of connected voxels that share the same gray level intensity. In the GLRLM module [6], the descriptor is the Gray Level Run Length Matrix, which counts the number of times a gray level i appears consecutively j times in a specified direction.

These textural features computed from gray level intensities descriptors capture information about relative positions between each gray levels in the image, which cannot be described by first order statistics. They measure quantitatively the coarseness or fineness of the texture, groupings of voxels with similar gray-level values, local intensity variation, homogeneity, contrast. They are built to have some invariance properties, like rotational invariance.

Preprocessing In regression model like Cox-PH which estimates linear coefficients β_i for each explanatory variable x_i, we need to normalize all the features x_i to avoid numerical issues during the estimation. However it is not necessary to normalize the features when using random forest models, since tree classifiers only rely on splitting value at each node.

2. Prediction with radiomics features as prior information

In this approach, we train simple prediction models by only using clinical data and radiomics features. We will see that prior information of the CT images, in particular texture, multi-scale information, can be encoded effectively with radiomics features. We then evaluate the performance of this approach so that we can compare it to the other approach without prior information in Section 3.

2.1. Classical regression approach

A possible simplification to the survival prediction problem is to approximate the survival regression as a classical regression. In this very rude approximation, we consider that the observation time is the time of death, even if the patient escaped the study. This means that we ignore the fact that data are right censored. We can then use an AutoML library `autosklearn` to perform regression on the 53 provided radiomics features with clinical data to predict survival time of the patients. The obtained C-Index score with this classical regression approach is 0.6783 on the public test set, which is slightly worse than the baseline provided by the challenge (see Section 2.2). This experience shows that this survival analysis problem cannot be reduced to a classical regression approach, certainly because they are too many right censored data. We define a non admissible pair as a pair of patients (i, j) where both are censored or patient i died at $t = k$ and patient j is censored at $t < k$. There are 11404 such pairs, which represents 25% of the 44850 possible pairs of patients in the training set (see Figure 1).

2.2. Baseline model: Cox PH model

The baseline model proposed by the challenge providers is the Cox proportional-hazards model [2], and is adapted to take into account right censored data.

Proportionality of hazards In this model, the hazard function $h(t|x)$ of each individual described by a vector of explanatory variables $x \in \mathbb{R}^p$ is defined as:

$$h(t|x) = h_0(t) \times \exp(\beta^T x) \tag{1}$$

where $t \geq 0$ represents a time variable, $\beta \in \mathbb{R}^p$ is the parameter of the model, and $h_0(t)$ is the baseline hazard function not depending on the feature values. The hazard function $h(t|x)$ can be interpreted as the risk that a patient described by the features x dies at instant t, knowing that he was alive just the instant before. One can define the cumulative hazard function as:

$$H(t|x) = \int_0^t h(u|x)du \tag{2}$$

which can be interpreted as the accumulation of the hazard over time.

During the Cox regression, we only estimate the parameter β from the right censored data and the features describing the patients, and not the baseline hazard function, since we are only interesting in the hazard risk ratio $\frac{h(t|x)}{h(t|x')} = e^{\beta^T(x-x')}$ between two patients.

The main assumption is the proportionality of the hazards: impacts of features $(x_i)_{i\in[\![1,p]\!]}$ on the hazard risk $h(t|x)$ are the same over time. Therefore, it is a simple model where we don't consider time dependant explanatory variables x. We can then interpret the impact of any explanatory variable $x_i \in \mathbb{R}$ from its parameter $\beta_i \in \mathbb{R}$ estimated during the regression: reduction in hazard if $\beta < 0$, increase in hazard if $\beta > 0$, and no effect on hazard if $\beta = 0$.

Reproducing the baseline The model is quite well suited to the problem, since C-Index score achieved by the challenge providers is 0.6909 on the public ranking, when using a subset of 8 appropriate variables. We reproduced the baseline and achieve a mean C-Index score of 0.6809 on a 5-fold cross validation, and 0.6701 on the public test set. One main issue of this baseline model is the high variability of the C-Index from one cross validation to another one, going from 0.6051 to 0.7491.

2.3. Extracting additional radiomics features

To improve the baseline model, we decide to extract more radiomics features to capture a maximum of prior information, in particular textural and multi-scale features, as it is done in many works for survival prediction with CT images, like in [19] [13].

The extraction of radiomics features with `Pyradiomics` can be either performed directly on the original image, like for the 53 provided radiomics features, or from a filtered image obtained by a wavelet transform, or a Laplacian of Gaussian (LoG) filtering. We will denote the extracted features by "original features", "wavelet features" and "LoG features". In order to capture good prior information for our prediction model, wavelet and LoG features should be extracted with adapted parameters for the filtering. We use the same mask for the original and the filtered image. Because of the missing mask issue described in Section 1, some wavelet and LoG features may not be accurate for patients with missing mask.

Wavelet transform Consider a low pass 1D filter L (scaling) and a high pass 1D filter H (wavelet). We apply 3D wavelet transform on the original CT image denoted U to obtain 8 wavelet decomposition denoted $(F_x, F_y, F_z) * U$, for $F_x, F_y, F_z \in \{L, H\}$. Each decomposition is obtained by convolving the image U along the x, y, z axis using respectively the filters F_x, F_y, F_z:

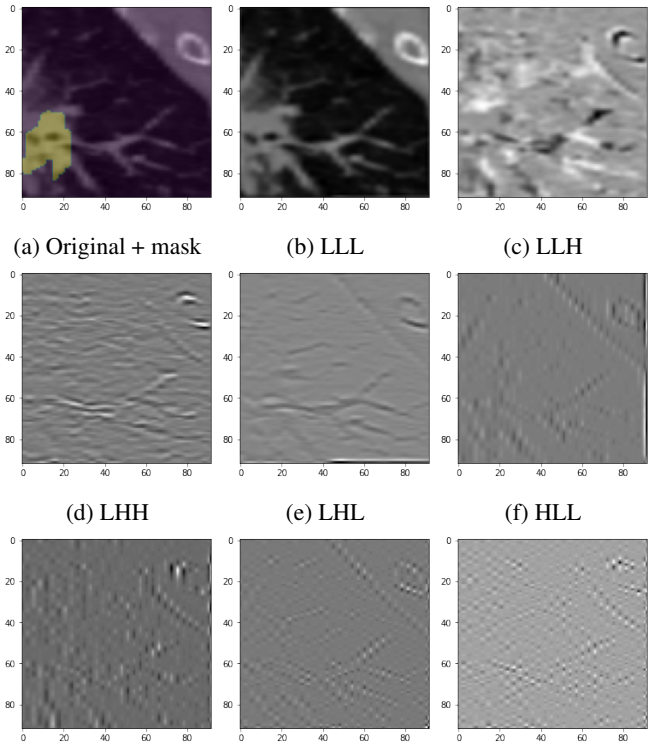
$$\sum_{k',l',m'} F_x(k')F_y(l')F_z(m')U(k-k', l-l', m-m').$$

Image examples of wavelet transform are shown in Figure 3. We then extract radiomics features (intensity features and textural features, see Section 1.2) from these filtered images.

Using these wavelet features allows us to capture multi-scale information of the original image. With different combinations of filters, we enhance low or high frequencies of the image. When using a low pass filter, the filtered image is blurred and captures large scale intensity variation. When using a high pass filter, the filtered image captures only high frequency information, and the obtained image is focused on edges and patterns in texture.

Laplacian of Gaussian filtering We also apply Laplacian of Gaussian filtering to the original CT image to extract LoG features. This filtering is obtained by convolving the image with the Laplacian of the Gaussian kernel, defined as:

$$G(x, y, z, \sigma) = \frac{1}{(\sigma\sqrt{2\pi})^3} \exp\left(-\frac{x^2+y^2+z^2}{2\sigma^2}\right). \quad (3)$$



(a) Original + mask (b) LLL (c) LLH

(d) LHH (e) LHL (f) HLL

(g) HLH (h) HHL (i) HHH

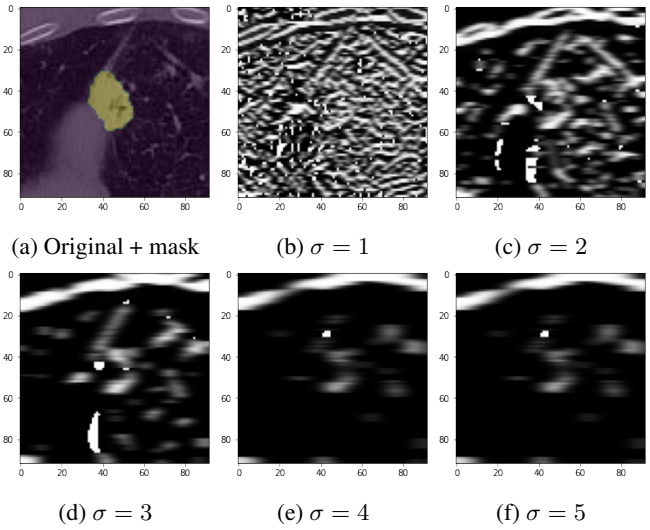
Figure 3: Slices of the 8 wavelet decompositions of a CT image, for a patient. Original mask is shown in yellow.

This Laplacian of Gaussian filtering can be decomposed into two steps: first, the image is convolved by a Gaussian kernel, which is used to smoothen the image, because it can be seen as a pooling operator; then, the obtained image is convolved by a Laplacian kernel, which is sensitive to areas with rapidly changing intensities, and therefore enhances edges in the filtered image. The parameter σ determines the size of the Gaussian kernel: low σ values enhances fine textures, while high σ values enhances coarse textures. Examples of CT images filtered by a Laplacian of Gaussian is shown in Figure 4. We then extract radiomics features (intensity features and textural features, see Section 1.2) from filtered CT images with a LoG of parameters $\sigma = 1$ and $\sigma = 3$, to capture both fine and coarse textures.

Prior information that we use when extracting LoG features is the fact that CT images have relevant information to capture in their texture when predicting survival time.

2.4. Exploring radiomics features

Once we extracted the wavelet and LoG features, if we add the 53 provided original features and the clinical data, we have in total 802 features for our survival regression model. Using directly all these features into a Cox PH model described in Section 2.2 leads to worse results than



(a) Original + mask (b) $\sigma = 1$ (c) $\sigma = 2$

(d) $\sigma = 3$ (e) $\sigma = 4$ (f) $\sigma = 5$

Figure 4: Slices of CT images filtered by Laplacian of Gaussian, with different kernel size σ, for a patient. Original mask is shown in yellow.

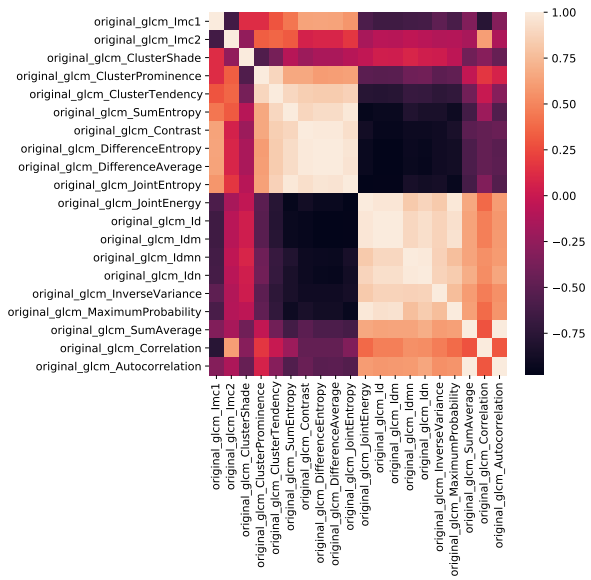


Figure 5: Spearman correlation matrix between GLCM original features.

the baseline: we obtain a mean C-Index of 0.6330 on a 5-fold cross validation. Cox PH model doesn't generalize well with large number of features: the model with 802 features overfits on the training set, which has only 300 patients. The C-Index achieved on the training set is 0.8467, compared to 0.6947 on the training set for the baseline model with a subset of 8 preselected features (see Section 2.2). Therefore, it is necessary to perform feature selection to overcome this generalization issue. We explore the 802 radiomics features to analyze if some of them are non informative or redundant.

2.4.1 Correlations

Some radiomics features are highly correlated since they are variations of the same measure. For example, several shape features describe the compactness of the tumor, and several textural features describes the same kind of gray level variation in the image, as it is shown in Figure 5. This Spearman correlation matrix illustrates some redundancy between GLCM original features. For illustration purpose, we only show correlations for one group of features, but one can reproduce this correlation matrix for all the features and also observe a lot of highly correlated features.

2.4.2 Classification approach

One can simplify the survival prediction problem as a binary classification problem, by splitting the population into low and high survival probability populations. The first category includes patients who died during the experience (Event $= 1$) before the time T_{low} days. The second category includes patients who are still alive at time T_{high}, whether they died during the experience (Event $= 1$) or escaped the study (Event $= 0$). According to the histogram of patient's survival time in Figure 1, we can choose for example $T_{\text{low}} = 300$ and $T_{\text{high}} = 1400$, so that we construct a reduced training set of 130 people (66 low survival, 64 high survival chances) split in 80% for training and 20% for validation. We can then study the characteristics of this reduced dataset. To explore the 802 radiomics features, we analyze to what extent it is possible to classify the patients of this reduced training set using these 802 features. We use a random forest model to perform this classification, to overcome generalization and high correlation issues. Random forest can also establish a ranking of feature permutation importance [1], so that we can identify discriminating variables for the classification.

The mean accuracy achieved in this classification is 0.85 on a 5-fold cross validation, which is not perfect. In Figure 6, we observe that most importance features (measured by permutation importance) are wavelet features, but median value and standard deviation of the importance weights are all similar, which makes the interpretation of discriminating variables difficult. This analysis shows that is it difficult to characterize perfectly low and high survival probability patients with the considered 802 radiomics features. In a certain way, this experience shows the difficulty of the dataset: one cannot expect to obtain perfect survival prediction with these radiomics features.

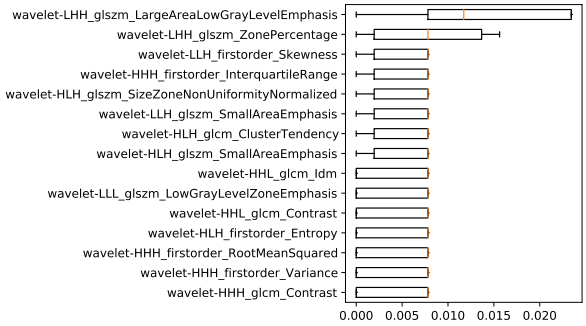


Figure 6: Permutation importance box plot for radiomics features (top 15 among 802, ranked by mean value) established by a random forest model classifying patients as low or high survival probability individuals. Orange line shows median value.

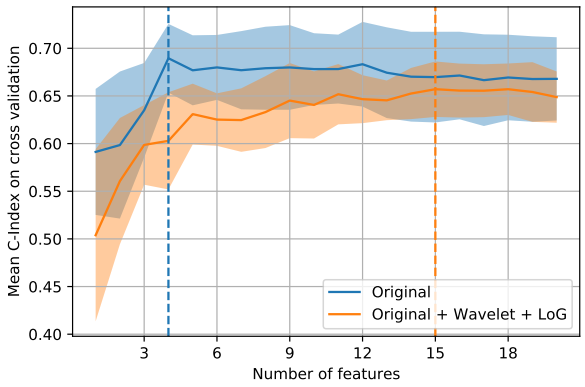


Figure 7: Recursive feature elimination for original features (blue curve) and all radiomics features (orange curve), using Cox-PH baseline model. At each number of features N, the mean C-Index from the cross validation of the model learned on the N selected features is represented on the graph, with its standard deviation. Dashed line represents optimal number of features according to cross validation.

2.5. Selecting radiomics features

As a result of the previous exploration, feature selection is crucial to eliminate redundant, non informative radiomics features. We tried several feature selection strategies: recursive feature elimination, LASSO regression and random survival forest.

2.5.1 Recursive feature elimination

We perform recursive feature elimination which eliminates p weak features at each iteration until a given number N of features is reached. Weak features are those who have a parameter coefficient β_i close to zero, since zero coefficients correspond to features with no impact on survival prediction. In our experience, p represents 10% of the total feature number. Once we selected the N features with this elimination method, we evaluate a Cox-PH model learned on these N with a 5-fold cross validation. In Figure 7, we present the mean C-Index score obtained during the cross validation of the model learned on the N features selected from the recursive feature elimination method.

This simple heuristic can reduce the number of features with good performances, but has no guarantee of selecting the best subset of features in terms of prediction performances, because at each iteration, discarded features considered as weak might in fact achieve good performances when they are combined with other features. This might be the explanation why in Figure 7, recursive feature elimination gives bad C-Index score (around 0.65 for the best cross validation score, 15 features selected) when used with all the 802 radiomics features. In this case, it is better to restrict the selection only among original features (mean C-Index of 0.69, 4 features selected), but this leads to the loss of prior information we considered with wavelet and LoG features.

2.5.2 LASSO regression

The Cox PH model presented in Section 2.2 solves a regression problem during the estimation, and one can use L^1 feature selection in this regression. To properly define the penalized regression problem, consider we have data of the form $(y_1, x_1, \delta_1), \cdots, (y_n, x_n, \delta_n)$ where $x_i \in \mathbb{R}^p$ is the vector of features values, y_i is the time of failure if $\delta_i = 1$, or the time of right-censoring if $\delta_i = 0$. Let $t_1 < t_2 < \cdots < t_m$ the list of the increasing failure times, assuming that at each time there is a unique event, and denote $j(i)$ the index of the observation failing at time t_i. In the setting where n the number of patients is larger than p the number of features, the partial likelihood [21] of the Cox model is given by:

$$L(\beta) = \prod_{i=1}^{m} \frac{\exp\left({x_{j(i)}}^T \beta\right)}{\sum_{j \in R_i} \exp\left({x_{j(i)}}^T \beta\right)} \tag{4}$$

where R_i is the set of indices j with $y_i \geq t_i$. During the regression of the Cox PH model, the negative log of the partial likelihood is minimized to get the optimal regression parameter β. To reduce the number of non-zero coefficients β_i, one can penalize the previous objective function with an elastic-net penalty:

$$\hat{\beta} = \arg\min \log L(\beta), \text{ subject to } \|\beta\|_1 \leq \lambda \tag{5}$$

where λ is the regularization parameter. This gives the LASSO regression problem of regularized Cox model. By controlling the value of λ, we can control the sparsity of the parameter β, that means the number of non-zero coefficients β_i. When $\beta_i = 0$, the feature x_i doesn't contribute

to survival prediction. Therefore, selected variables by the LASSO regression are the features x_i for which β_i is non zero.

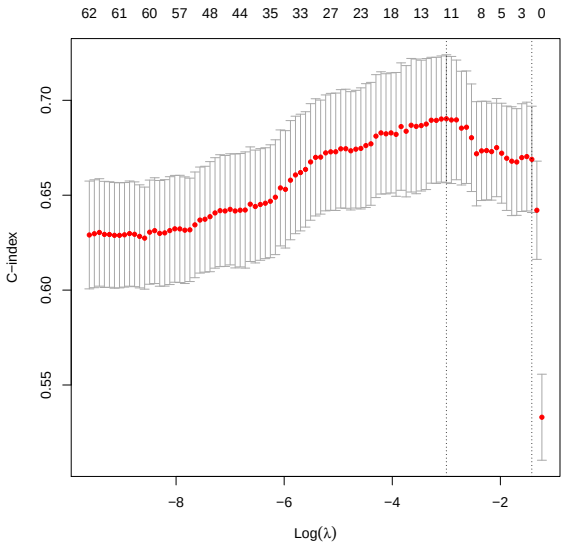
Experience We use `glmnet` package [5] in `R` to solve this LASSO regression problem on our dataset, and obtained an estimated parameter $\hat{\beta}(\lambda)$ of the regularized Cox model, for each regularization parameter λ. We then evaluate these regularized models on a 5-fold cross validation. Each value of λ corresponds to a number of non zero coefficients. Two values of λ are considered at the end of the experience: $\lambda_{\min}$ which gives minimum mean cross-validated error; and λ_{1se} which gives the most regularized model such that error is within one standard error of the minimum. We then select the features for which the corresponding coefficients of $\hat{\beta}(\lambda_{\min})$ or $\hat{\beta}(\lambda_{\min})$ are non zero. Choosing $\lambda_{\min}$ leads to the optimal selection for the training set considered, but λ_{1se} will lead to more sparse solution, which gives a worse score but a better chance of generalization.

Figure 8 shows that best C-Index score obtained with parameter $\hat{\beta}(\lambda_{\min})$ is 0.6903 when considering only original features (11 non zero coefficients), and 0.6724 when considering all features (18 non zero coefficients). We will not consider parameters obtained with λ_{1se}, since they are too sparse, with only 1 or 2 non zero coefficients. We conclude that LASSO regression feature selection works better with less variables, and it is not a good strategy to use if we want to select features among wavelet and LoG features, because of their large number ($p > n$ in expression (4) context).

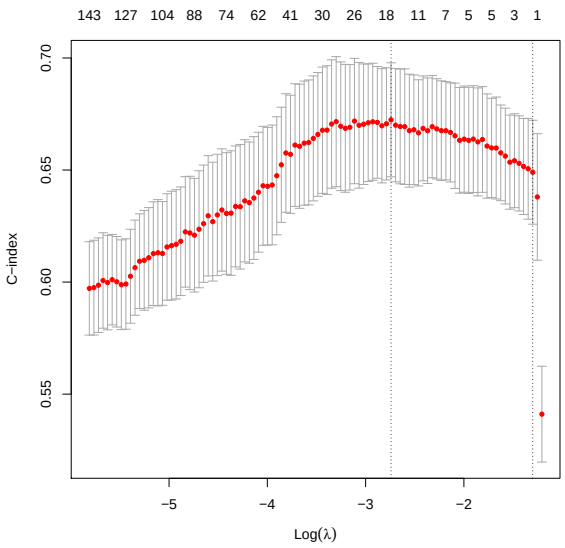
2.5.3 Random survival forest

Random survival forest [11] (RSF) is an ensemble tree method for right censored data analysis. It relies on survival trees growing on bootstrap samples from the original data: the split at each node uses the variable that maximizes survival differences between children. Each trees computes an estimate of the cumulative hazard function (CHF) from expression (2). Then, the average CHF over all trees is used to compute the prediction error on the out-of-bag data.

Variable importance For each feature x_i, one can compute its variable importance in the following way: during the training, when a split of x_i is encountered, a random child is assigned to the node; then, the variable importance is the difference between the original forest's prediction error and the one computed with the perturbed forest. High variable importance indicates that the feature have large predictive ability. However, it is not necessary that a forest looses performance in prediction when we remove a feature with high variable importance: when a variable correlated to another one with high variable importance is removed, the prediction error will likely remain the same.



(a) Original features only



(b) All radiomics features (original, wavelet, LoG)

Figure 8: LASSO regression for several values of regularization parameter λ. We represent the C-Index score on 10-fold cross validation (mean and std) of the regularized Cox model, trained on original features (top) and all features (bottom), for different values of λ. The corresponding number of non-zero coefficients is shown at the top of the graph. Full line represents $\lambda_{\min}$, while the dashed line represents λ_{1se} (see Section 2.5.2).

Ranking and optimal number of features Given a set of features, we can perform feature selection based on vari-

able importance. Firstly, we tune random survival forest hyperparameters on the training dataset using all the features: each model is evaluated with a 5-fold cross validation. We then use the optimal random survival forest model to establish a ranking on the variables based on their variable importance. For more precision in the ranking, one can prune the last $p\%$ features in this ranking ($p = 50$ will be used in the following experience), and repeat the previous step with the reduced subset of features. Pruning features might increase the accuracy in the ranking, since variable importance computed with less variables has smaller standard deviation; but pruning too much variables might discards important features for the prediction. Secondly, we determine the optimal number of features we select from the previous ranking to get the higher C-Index on cross validation. For each number of features N, we select the N highest variable importance features in the ranking, and evaluate with a 5-fold cross validation a random survival forest model trained on this subset of features. The final selected variables are the top $N_{\max}$ variables in the variable importance ranking achieving the highest mean cross validation score.

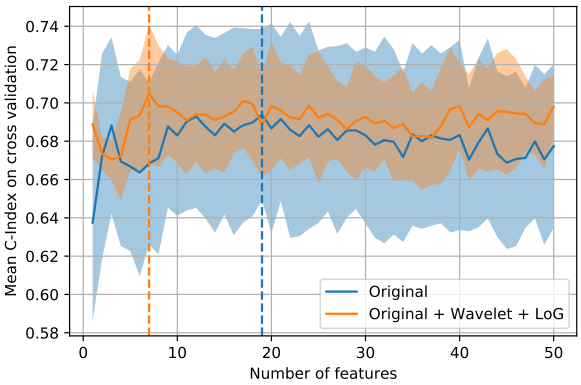


Figure 9: Random survival forest feature selection with variable importance. A RSF trained on the training set ranks features by their computed variable importance. Blue curve considers only a ranking on original features, while orange curve considers a ranking over all radiomics features. For each number of selected features N, cross validation is performed on a RSF model trained on the top N features of the ranking, and mean value of the cross validation with its standard deviation is represented in this graph. Dash lines represents the optimal number of features to select at the top of the ranking.

Experience We implement the previous algorithm, and use it to select radiomics features. In order to compare the impact of each group of radiomics features (original, wavelet, LoG) on prediction, we first use this algorithm to select only among original features (62 features), then to select among all features (original, wavelet, LoG, 802 features). In Figure 9, we observe that wavelet features and LoG features can contribute to improve C-Index score on cross validation. The optimal number of features when selecting among all radiomics features is 7, reaching a mean C-Index score of 0.7047, and variability in cross-validation score is smaller. In comparison, when selecting features only among original features, the best mean C-Index score achieved is 0.6939 (18 features selected). In Figure 10, we observe that selected features that are highly predictive are all wavelet and LoG features.

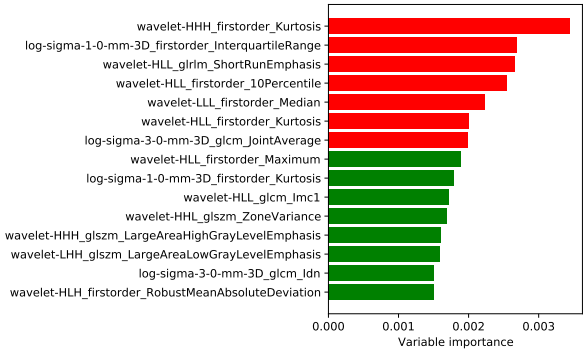


Figure 10: Variable importance ranking (top 15 over all 802 radiomics features) computed by RSF trained on the training set. Red variables are the selected features which corresponds to the optimal number of features during the cross validation evaluation described in Section 2.5.3.

We see that the advantage of using RSF and features ranking with variable importance is that it can scale with large numbers of features, because of generalisation properties of random forests.

		Selected features	Total
RFE	Ori.	2 original, 2 clinical	4
	All	2 original, 11 wavelet, 2 clinical	15
RSF	Ori.	16 original, 2 clinical	18
	All	5 wavelet, 2 LoG	7
LASSO	Ori.	6 original, 5 clinical	11
	All	1 original, 14 wavelet, 3 clinical	18

Table 1: Summary of selected features, using recursive feature elimination (RFE), random survival forest (RSF), and LASSO regression. We first select among original features only (Ori.), then among all features.

2.6. Training model and evaluating prediction

Features selected from different strategies are summarized in Table 1. According to the chosen feature selection strategy, we train and tune a specific model on the subset of selected features to predict survival time:

- for recursive feature elimination strategy, we use a Cox PH model preseted in Section 2.2;
- for LASSO regression strategy, we use a Coxnet model [17] which is the regularized version of Cox PH model;
- for random survival forest strategy using variable importance ranking, we simply use a random survival forest presented in Section 2.5.3.

C-Index score on 5-fold cross validation and on public test set is shown in Table 2. The highest scores achieved on the public and private test set are respectively 0.7209 and 0.7726, and both are achieved with the LASSO regression strategy for feature selection, applied only on original features. This is unexpected, since the best score on cross validation 0.7230 is achieved by RSF feature selection strategy with wavelet and LoG features. Overall we observe that selecting among all features (original, wavelet LoG) gives high C-Index on cross validation than selecting only among original features, but lower C-Index on public score. The score achieved on private test set is also surprising, since its deviation from the mean cross validation C-Index is much larger than the standard deviation (0.0729 compared to 0.0198).

		Cross validation score	Public score
RFE	Ori.	0.6974 ± 0.0319	0.7001
	All	0.6940 ± 0.0325	0.6497
LASSO	Ori.	0.6997 ± 0.0198	$\mathbf{0.7209}$
	All	0.7101 ± 0.0068	0.6805
RSF	Ori.	0.6938 ± 0.0451	0.6880
	All	$\mathbf{0.7230} \pm 0.0273$	0.6716

Table 2: C-Index score on 5-fold cross validation and on the public test set. For each features selection strategy, we train the corresponding predictive model, see Section 2.6. Feature selection is applied on original features only (Ori.), and then on all radiomics features.

Overfitting issues In other words, it seems that, when considering wavelet and LoG features, our algorithm for predicting survival time is overfitting on cross validation. This is not impossible, since our feature selection methods evaluate the optimal number of selected features with cross validation. Choosing in this way the optimal number of features might lead to overfitting on cross validation. In this case, it is possible that selected features, which are mainly features extracted from high-pass wavelet transforms or LoG filtering with small parameter σ (as we can see in Figure 10), are more focused on noises than on the content of CT image, since these features mainly describe high frequencies. The model learned on these noises cannot generalize well on new data.

3. Computer Vision pipeline with no a priori information

In the continuity of the course, we thought it would be interesting to compare the previous approach and an image-based approach that aims at extracting powerful features only from images and masks without any a priori information (in opposite to the tailored radiomics features), and eventually learn how to predict survival times from these features. The advantage of such an approach is that it can be used in an end-to-end fashion.

For that, we might start from a pretrained Deep Learning model as our training set is rather small. However, the Deep Learning models that have established themselves as state-of-the-art on many Computer Vision tasks in the last decade are most often provided with pretrained weights on 2D images with RGB channels. We will present how we can bring the problem down to such a CNN, then explore how to further finetune it and analyze its performances.

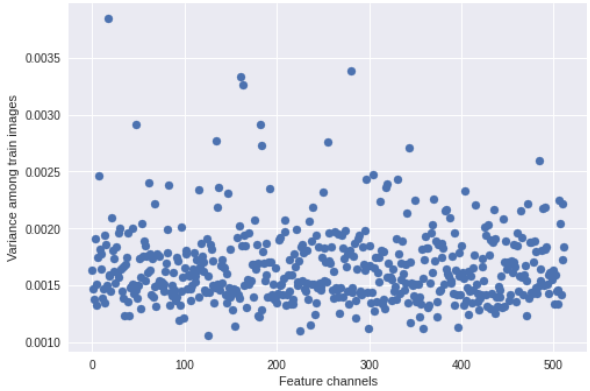
With such small training set, a key tradeoff to achieve here is to use enough information from the images data without overfitting: we will present different approaches we tried to achieve that trade-off.

3.1. CNN Feature Extraction

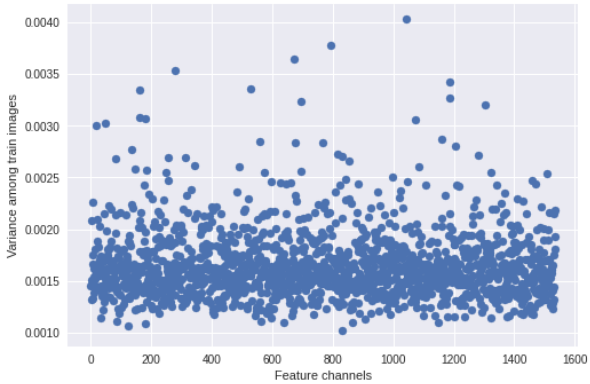
To restrict ourself to a simpler 2D problem and only keep the most discriminative information about the tumor image, we start by extracting only the 2D slice (among all dimensions) that has maximum number of tumor pixels for each patient, to be able to use pretrained 2D CNNs on it. For that, we then repeat it 3 times to have a 3-channel RGB-like input (of dimension $92 \times 92 \times 3$, bilinearly resized to $224 \times 224 \times 3$, that we pass through a feature extractor without any other finetuning.

We choose a simple but efficient ResNet-18 [10] (with its PyTorch implementation and weights after training on ImageNet [3]), just removing the last classification layer. Its architecture is basically made up with an input convolution, batch normalization, ReLU, max pooling, 4 convolutional blocks with residual connections and an average pooling. Therefore, we obtain 512 dimensional feature vectors for each image. This can also be done separately for each dimension, thus outputing 3 512 dimensional feature vectors for each image.

However, we found that the features obtained have almost no discriminative power: they all have less than 0.005 variance. We tried to reduce the dimension of the feature space via PCA and to use RSF on it but as expected, this did not perform well. Similar observations were done on the features extracted from separate dimensions.



(a) Best Slice across all dimensions



(b) 3 Best Slices

Figure 11: Variance of the different deep feature extracted.

3.2. Image based End-to-End Pipeline

Doing transfer learning without any finetuning as done previously is not very efficient as ResNet-18, like many pretrained available models, is trained on ImageNet which is made up with 224×224 RGB images (which is very different of our dataset even with our preprocessing). Therefore we implement a neural network composed with a pretrained ResNet feature extractor, a 3-layer perceptron (hidden size 128 and 32, output size 1, with ReLU activations, dropout of probability 0.5 between layers to avoid overfitting) that predicts the risk of patients. The weights of all but the last two convolutional blocks of ResNet are kept frozen.

Inspired by [12], we train with the negative log partial likelihood. This loss can be thought as an extension of Cox PH regression to non linear functions (here the neural network function). Instead of considering linear relationship $\beta^T x$ in expression (1) for expressing the impact of explanatory variables on hazard risk, DeepSurv [12] models this impact with a neural network function $f_\theta(x)$ parameterized by θ. The negative log partial likelihood of the Cox model is then given by:

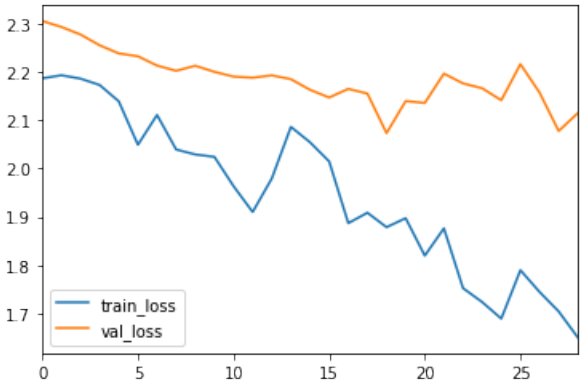
$$L(\theta) = \prod_{i=1}^{m} \frac{\exp\left(f_\theta(x_{j(i)})\right)}{\sum_{j \in R_i} \exp\left(f_\theta(x_{j(i)})\right)} \tag{6}$$

where we have considered the same setting and notations as the expression (4) in Section 2.5.2.

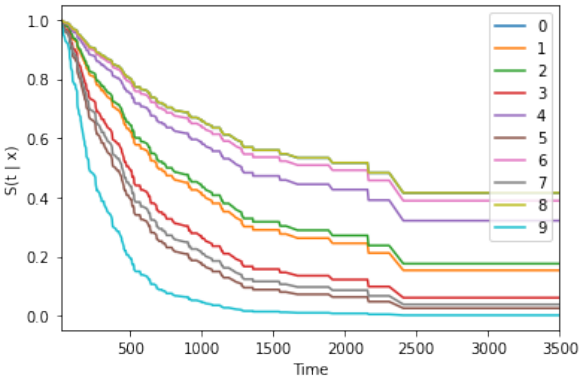
We built on the PyCox library [7] to inherit CoxPH-like functionalities, and notably to use this loss to have an interesting supervision of our model exploiting both censored and uncensored data.

We split train images 80% for the training itself, 20% for the validation set, and use SGD optimizer with learning rate 0.0005 (chosen using [18] heuristic), momentum 0.9 [14], and weight decay 3×10^{-4}, for 20 epochs (plus eventually early stopping), with batch size 20.

After training, we compute baseline hazards as CoxPH is semi-parametric, and we predict survival. The best validation C-Index obtained was 0.72, and corresponded to 0.64 public test C-Index.



(a) Training and validation loss evolution



(b) Risk Prediction on 10 validation individuals

Figure 12: End to End training

As seen on Figure 12, a substantial compromise (which was handily optimized with the previous parameters) is to

achieve a good training (the training loss decreases) without overfitting (the validation loss also decreases). One idea not used here is data augmentation (we describe it in the next part actually): we were able to implement proper torch dataloaders (so to apply easily torchvision transformations) and actually train, but it is actually more tricky to use PyCox functionalities (notably computing baseline hazards) with dataloaders for evaluation afterwards.

To further analyze what our model is actually doing, we implemented a Grad-Cam extractor [16] that enables the visualization of most important parts for network's risk prediction. As seen on Figure 13, these parts are not always focused on the tumor, and typically always contain an external part, which means that the model does not really make its predictions mostly on the tumor itself, so it is likely not extracting features characterizing it.

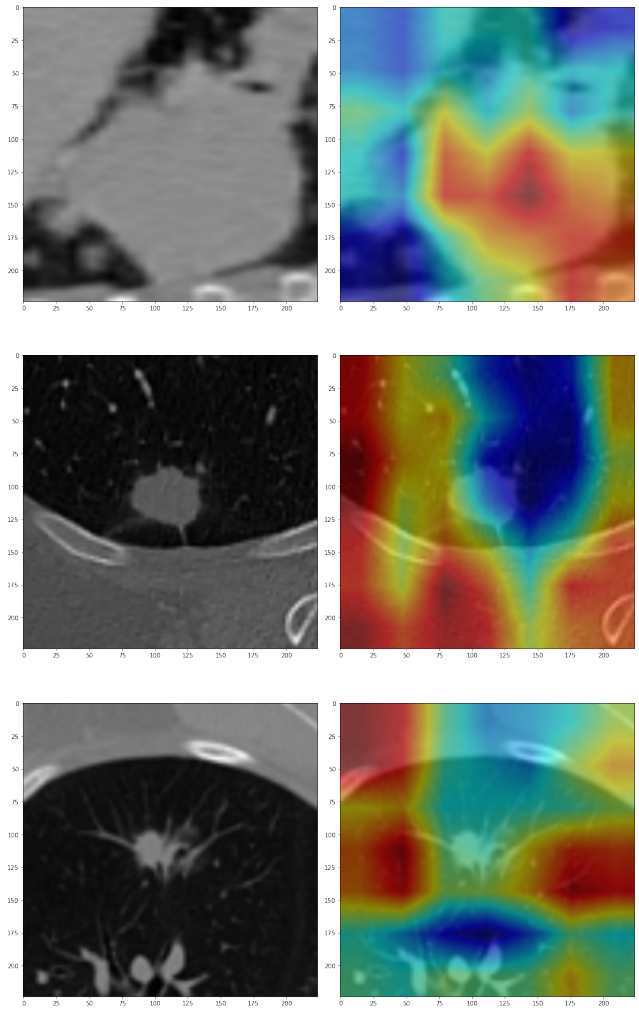


Figure 13: Grad-Cam example on ResNet + MLP model. Red zones are considered as important for prediction.

3.3. Simplifying the problem: classification

We thought the risk prediction may be a too complicated problem to solve with only 300 slices of data. Following previous observation from Section 2.4.2 that we could distinguish 2 classes (those who have low chance of survival and those who have a high chance of survival), we use the reduced training set previously described splitted in 80% for training and 20% for validation.

We once again only consider the best slice for the patient (preprocessed in a similar fashion as previously), and ResNet as classifier (just replacing the last layer by a 2-classes classifier), freezing all weights but those of the last convolutional block. We train this network with Cross Entropy Loss, SGD optimizer with learning rate 0.001, momentum 0.9 [14], and weight decay 3×10^{-4}, for 10 epochs, with batch size 16.

We use as additional data augmentation a slight color jittering (brightness 0.1, contrast 0.1, saturation 0.1, hue 0.1), random horizontal flip (probability 0.5), random vertical flip (probability 0.5) and random rotation (15 degrees) to avoid overfitting.

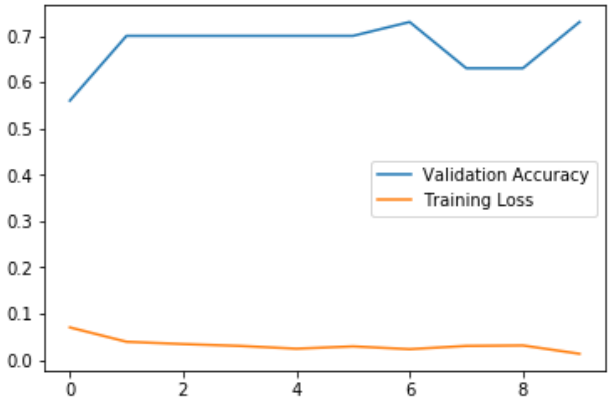


Figure 14: Training Loss and Validation Accuracy evolution

We obtain at the end a validation accuracy of 73% (as seen in figure 14, there is no much improvement after the first 2 epochs actually), which is not very satisfying given the problem is so much simplified, and it is significantly outperformed by the classical approach.

Once again, we use Grad-Cam [16] to analyze what parts of the image are actually discriminative for the network so that he predicts a class for a given image. As previously, only rarely does it actually bases its prediction mainly on the actual tumor as seen in Figure 15.

At this point, we thought the bottleneck of all previous approaches might be due to the limitation to a single 2D slice, which is a simpler data to exploit code-wise and in terms of computational cost. So we thought about trying to use more information provided by 3D images and masks.

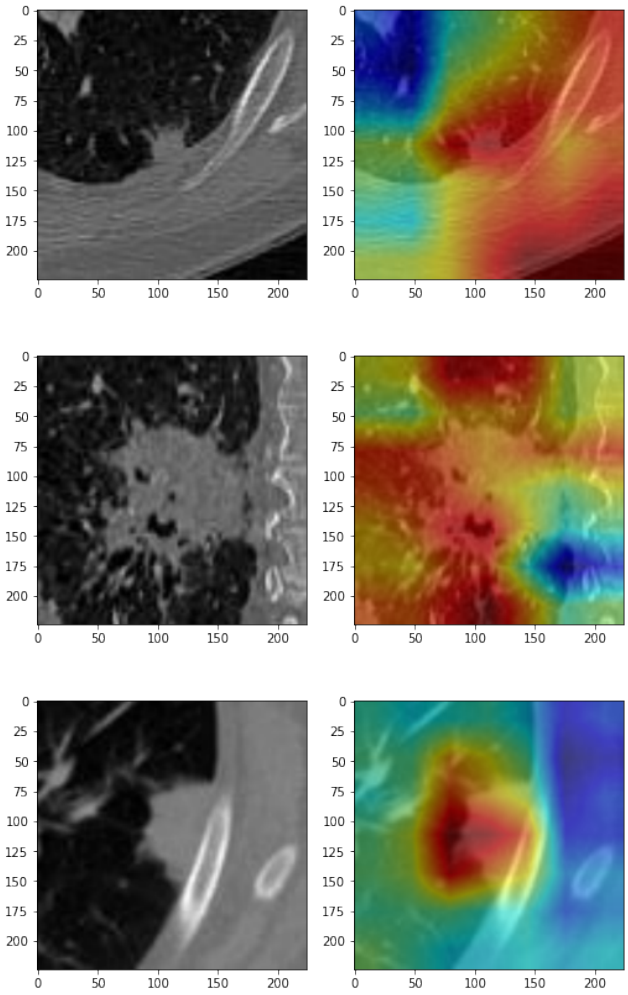


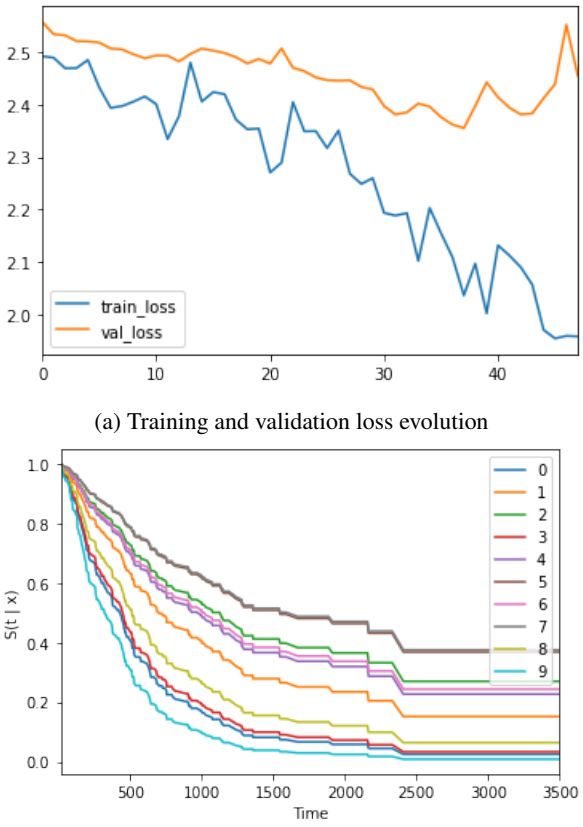
Figure 15: Grad-Cam example on ResNet Binary Classifier

3.4. Attention-based CNN

We can modify the end-to-end network described in Section 3.2 to further use the mask. For this, inspired by [4], we add another branch in parallel computing attention weights from the 2D mask corresponding to the best slice (with a 2D convolution similar to the first one in ResNet-18, with 1 input channel, 64 output channels, stride 2, kernel size 7, padding 3). The output of this attention map is then multiplied with the output of the first convolution of ResNet-18, and this result is then passed to all other ResNet modules.

For the training parameters, we use SGD optimizer with learning rate 0.00025 (chosen using [18] heuristic), momentum 0.9 [14], and weight decay 3×10^{-4}, for 20 epochs (plus eventually early stopping), with batch size 20.

The best validation C-Index obtained was 0.72, and corresponded to 0.62 public test C-Index, which is unfortunately not better than the network described in Section 3.2. The reason for this might be that it is hard to learn all the weights of the attention layer with such a small dataset. We tried to use different learning rates for different layers but could not get better validation results.



(a) Training and validation loss evolution

(b) Risk Prediction on 10 validation individuals

Figure 16: End to End training of the CNN + Attention

3.5. Binary Image Semantic Segmentation

Masks are indeed given for all 425 (train and test) patients but 11. However, we thought training an image binary semantic segmentation model could be interesting not only to recover the masks of the 11 others, but also to hopefully have a decent feature extractor, learnt exploiting all the image data that we have, at the cost of more complex implementation and a significantly longer training.

Therefore we implemented a 3D-UNet model [15], inspired by a modified implementation (`https://github.com/pykao/BraTS2018-tumor-segmentation`, from which we actually corrected what we believe is a mistake at `https://github.com/pykao/BraTS2018-tumor-segmentation/issues/2`). We chose this model because it is well known in the biomedical sector.

Its architecture is fully convolutional, and consists first in a succession of 5 convolutional blocks (context aggregation pathway) that encode increasingly abstract representation of the image. This is followed by upsampling blocks (localization pathway) that recombine these representations with shallower features to precisely localize the structures of interest.

Images and Masks are resized to $128 \times 128 \times 128$ (via nearest neighbours interpolation) to meet the architecture requirements. We use all but 30 images from training images and test images as training set, 30 other training images as validation set (as we do have supervision for it) and the patients with missing mask as test set (Patients ID 3, 256, 263, 327 and 391 from training images, and 50, 198, 199, 234, 311, 404 from test images). The reason for such a split is that we want to have good segmentation (and feature extraction) for test images.

We use Adam optimizer with learning rate 0.0001, $\beta = (0.9, 0.999)$, $\epsilon = 10^{-8}$, and weight decay 10^{-5}, for 2 epochs with batch size 2 (due to Google Colab memory requirements). We use as loss a combination of negative dice index and binary cross entropy, defined as:

$$Dice(pred, target) = 2\frac{target \cap pred}{target \cup pred}$$

$$\begin{aligned}\mathcal{L}(pred, target) &= \lambda \times CE(pred, target) \\ &+ (1 - Dice(pred, target)) \end{aligned} \quad (7)$$

where $pred$ are the classes predicted probabilities, and $target$ is a one-hot encoding of the ground truth class. The weight of cross entropy is handtuned at $\lambda = 0.25$ Using Cross-Entropy Loss only leads to good accuracy indeed, but does not lead to good segmentation (nor precision / recall) as there is a significant imbalance between the two classes. Indeed, there is much more no tumor pixels than tumor pixels, which motivates the choice of such a loss. Additionally, Dice Index or IoU is a natural choice of metric to evaluate the quality of a semantic segmentation.

Because with such a small dataset, we are very likely overfitting, we use as data augmentation random flip (90, 180 and 270 degrees with equiprobability) along each axis. Note that 3D data augmentation is not supported in Torchvision as it relies on the PIL library. Possibilities for further data augmentation rely on using specific libraries such as medicaltorch (`https://github.com/perone/medicaltorch`) or torchio (`https://github.com/fepegar/torchio`). For the sake of simplicity, we chose to implement our data augmentation techniques with numpy directly.

In the second epoch, as seen in Figure 17, we have a decent 0.75 average training dice index (which is a relevant information because the test images are in our training set). However, we have 0.58 average validation dice index, which highlights some failure of generalization.

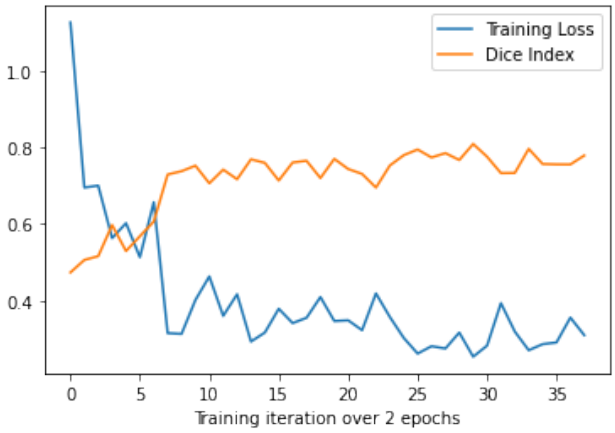
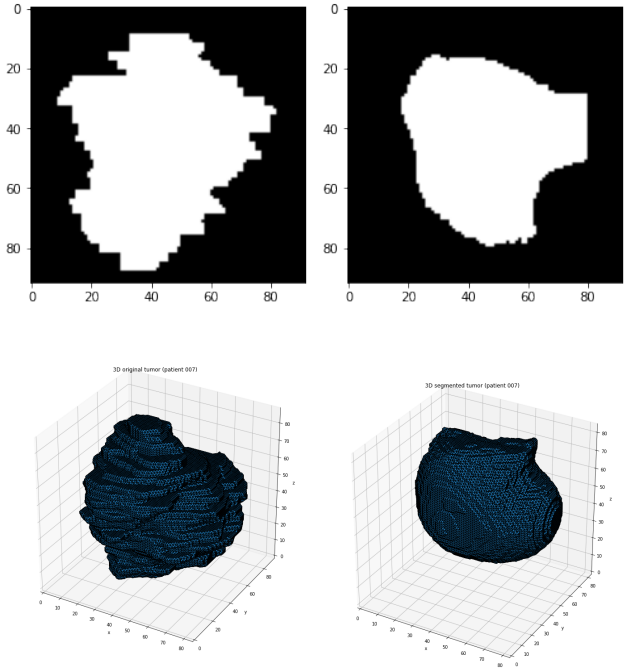


Figure 17: Training Loss and Training Dice Index evolution over 2 epochs (an iteration corresponds to 10 batches)

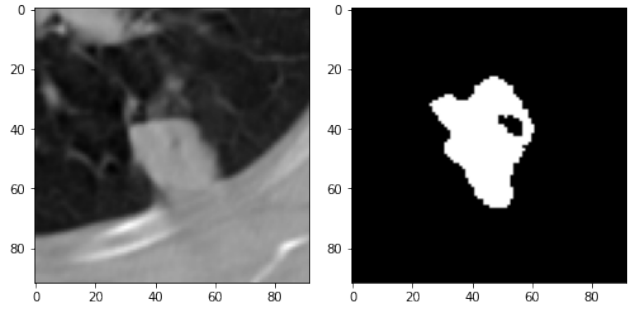
In Figure 18 and 19, segmentation examples, both in 2D and 3D, from training and test sets are provided. We use this segmentation model to recover the 11 missing masks (well, given the performances of the segmentation, it does not recover it perfectly, but this is still decent to perform the pyradiomics feature extraction as done in the previous section).



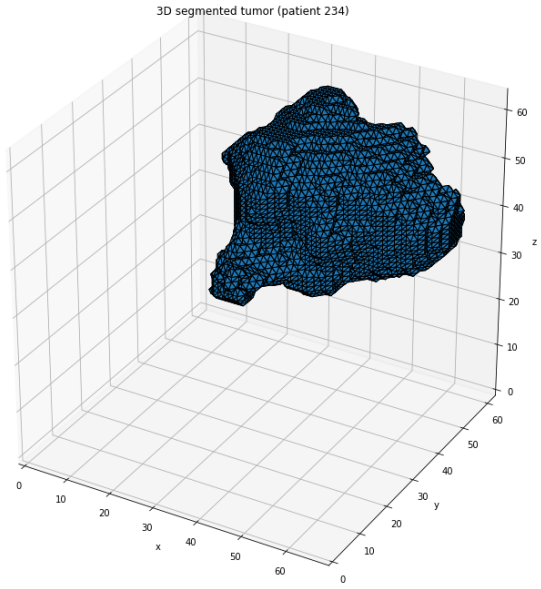
(a) Ground Truth

(b) Segmentation Result

Figure 18: Segmentation comparison of an scan from the training set



(a) Random Slice (b) Segmentation Result



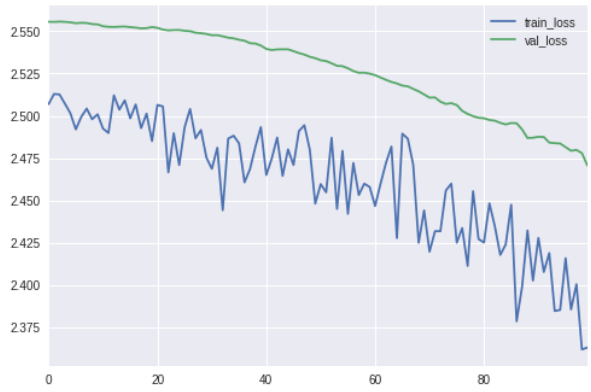
(c) 3D Segmentation result

Figure 19: Segmentation example on a patient from the test set (empty mask)

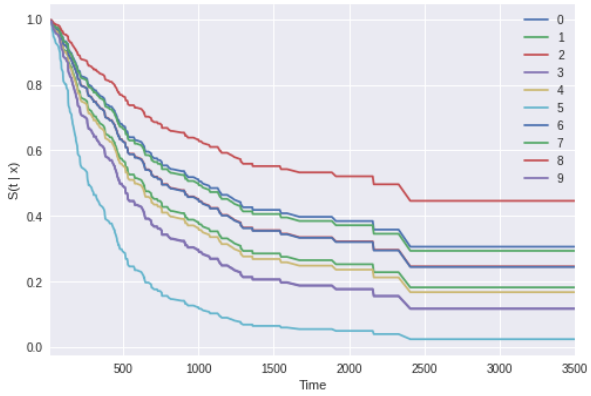
We can also use our segmentation model to extract $256 \times 8 \times 8 \times 8$ feature maps at the most abstract level of the representation for each patient. They do not present low variance like the ResNet ones. We tried putting them into the end-to-end computer vision pipeline to see what they are worth. They do not present the same variance problem, but are very high dimensional which makes it hard to use them on such small dataset.

The architecture used is a 3D average pooling of size (2,2,2) which results in an input of size 16384 to a multilayer perceptron of hidden size 2048, 128 and 32 with dropout 0.5 between layers and relu activations (the risk computation is done similarly as before after the mlp).

Unfortunately, the validation C-Index is 0.67, which is probably due to the difficulty of exploiting such high dimensional representation (using a bigger pooling results in too big of a loss of information hence worse results).



(a) Training and validation loss evolution



(b) Risk Prediction on 10 validation individuals

Figure 20: Training of a risk predictor from segmentation feature maps

4. Conclusion

On the one hand, exploiting images without any a priori information proved to be a very difficult challenge. To sum up, we implemented visualizations of 2D slices, 3D images (as videos) and 3D masks as geometric forms. We tried to simplify the problem by extracting the slice with most tumor, first for a ResNet-18 feature extraction resulting in unmeaningful features, then for an end-to-end pipeline inspired by DeepSurv but taking as input images only, which enabled to finetune a ResNet-18 with negative log partial likelihood loss. This resulted in the best (but unsatisfying) result of this pipeline (0.72 validation C-Index, 0.64 public test C-Index). We also tried to finetune ResNet-18 by classifying low / high survival patients. We implemented Grad-Cam to analyze the results and found both previous

approaches not to make their prediction as we would like them to. We tried using more information first by adding mask information to the previous end-to-end pipeline to compute an attention map, but this did not perform better than previously. We tried using all the images information by implementing a 3D U-Net model to segment the tumors, that had at best 0.75 average training dice index and 0.58 average validation dice index. We used it for feature extraction (which showed to be difficult because of the high dimensionality) and to recover 11 missing masks in the dataset. Other interesting ideas not tried include using a 3D-CNN trained in end-to-end either on images multiplied by mask, either with a parallel attention layer (just like the model used for 2D slice previously). Other things we would like to explore would be to use further data augmentation (we only used rotations), and mix deep learning features with traditional features.

On the other hand, radiomics features are traditional features used to represent CT images and are adapted to this survival prediction problem. They can encode prior information that we know about CT images, because they capture multiscale information with wavelet features, and texture information with Laplacian of Gaussian features. However, most of the radiomics features are redundant or non-informative, so feature selection is necessary in order to train simple predictive models with these features. We tried recursive feature elimination, LASSO regression and random survival forest as feature selection strategies. We observed that selecting among all features (original, wavelet and LoG) gives higher C-Index on cross validation than selecting only among original features (0.7209 for random survival forest feature selection strategy), but not on public and private test set, where the highest scores are achieved by LASSO regression only with original features (respectively 0.7209 and 0.7726 for public and private score). We conclude that our feature selection methods overfit on cross validation, when we consider all the 802 radiomics features. Future works should focus on how to make our method more robust to avoid this overfitting, in order to use efficiently the wavelet and LoG features and improve survival prediction.

References

[1] Leo Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[2] David Robert Graham Cox. Regression models and life-tables. 1972.

[3] Jia Deng and et al. Imagenet: A large-scale hierarchical image database. *IEEE conference on computer vision and pattern recognition*, 2009.

[4] S. Eppel. Classifying a specific image region using convolutional nets with an roi mask as input. 2018.

[5] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

[6] Mary M. Galloway. Texture analysis using gray level run lengths. 1975.

[7] M. Geck. Pycox: Computing with (finite) coxeter groups and iwahori–hecke algebras. 2012.

[8] Robert M. Haralick, K. Sam Shanmugam, and Its'hak Dinstein. Textural features for image classification. *IEEE Trans. Systems, Man, and Cybernetics*, 3:610–621, 1973.

[9] Frank E. Harrell, Robert M. Califf, David B. Pryor, Kerry Lamont Lee, and Robert A. Rosati. Evaluating the yield of medical tests. *JAMA*, 247 18:2543–6, 1982.

[10] K. He, S. Zhang, X. andRen, and J. Sun. Deep residual learning for image recognition. *In Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[11] Hemant Ishwaran, Udaya B. Kogalur, Eugene H. Blackstone, and Michael S. Lauer. Random survival forests. 2008.

[12] J. L. Katzman, U. Shaham, A. Cloninger, J. Bates, T. Jiang, and Y. Kluger. Deepsurv: personalized treatment recommender system using a cox proportional hazards deep neural network. 2018.

[13] Stefan Leger, Alex Zwanenburg, and Christian Richter et al. A comparative study of machine learning methods for time-to-event survival data for radiomics risk modelling. In *Scientific Reports*, 2017.

[14] Yurii Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). 1983.

[15] Olaf RONNEBERGER, Philipp FISCHER, and Thomas. BROX. U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*, 2015.

[16] Ramprasaath R. Selvaraju and et al. Grad-cam: Visual explanations from deep networks via gradient-based localization. *Proceedings of the IEEE international conference on computer vision*, 2017.

[17] Noah Simon, Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for cox's proportional hazards model via coordinate descent. *Journal of Statistical Software*, 39(5):1–13, 2011.

[18] L. N. Smith. Cyclical learning rates for training neural networks. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017.

[19] Li Sun, Songtao Zhang, Hon-Xun Chen, and Lin Luo. Brain tumor segmentation and survival prediction using multimodal mri scans with deep learning. In *Front. Neurosci.*, 2019.

[20] Guillaume Thibault, Jesús Angulo, and Fernand Meyer. Advanced statistical matrices for texture characterization: Application to cell classification. *IEEE Transactions on Biomedical Engineering*, 61:630–637, 2014.

[21] Robert Tibshirani. The lasso method for variable selection in the cox model. *Statistics in medicine*, 16 4:385–95, 1997.

[22] Joost J.M. van Griethuysen, Andriy Fedorov, Chintan Parmar, Ahmed Hosny, Nicole Aucoin, Vivek Narayan, Regina G.H. Beets-Tan, Jean-Christophe Fillion-Robin, Steve Pieper, and Hugo J.W.L. Aerts. Computational radiomics system to decode the radiographic phenotype. *Cancer research*, 77 21:e104–e107, 2017.