Predicting lung cancer survival time (Healthcare & Computer Vision)

Oualid El Hajouji
Department of Computational and Mathematical Engineering
Stanford University
`oualidel@stanford.edu`

Joachim Sasson
Department of Computational and Mathematical Engineering
Stanford University
`jsasson@stanford.edu`

1 Introduction

This project is part of a challenge [6] proposed by the company Owkin. The goal of this challenge is to develop methods to predict survival time of patients diagnosed with lung cancer. In order to complete this challenge, Owkin provides a dataset containing segmented radiology images (Computed Tomography scanner or CT scan) and numerical features. Predicting the survival time is essential in order to predict a patient's risk and adapt treatment strategies. We can list several challenges in this project. The most obvious ones are the following:

- The small number of patients
- Mix of image and structured tabular data
- Predicting survival time is very different from a classic regression problem

The last point is extremely important. This is a survival analysis problem, and not a regression task. Indeed, we saw in the description of the dataset that the survival time is in fact a time until an event is observed. This event is characterized by the censor variable. If the observed event is the patient's death, the survival time is simple to determine. However, if the patient is *censored* (he moved in another coutry, the clinical trial is interrupted before the patient's death...) we only have access to a lower bound for the survival time.

Thus, the task consists in properly ordering survival times between patients rather than accurately predicting the survival times. That is why the **Concordance Index** metric (C-index) is the metric of interest here. The C-index is a generalization of the AUC which can take into account censored data. The C-index evaluates whether predicted survival times are ranked in the same order as their true survival time. Indeed because of censored data, the actual ordering of patient survival is not always available. The pair (i, j) of patients is said **admissible** if patients i and j are not censored, or if patient i dies at $t = k$ and patient j is censored at $t > k$. On the contrary, if both patients are censored or if patient i died at $t = k$ and patient j is censored at $t < k$, the pair is **not admissible**. Indeed in that last case we are not sure if patient j has died before patient k'. The proper definition of the C-index is thus:

$$C = \frac{\#\{\text{Concordant pairs}\}}{\#\{\text{Admissible pairs}\}} \tag{1}$$

Where concordant pairs are admissible pairs of patients that are correctly classified. Therefore, we notice that a score of 0.5 will correspond to a random prediction while a score of 1 corresponds to a perfect prediction.

2 Dataset

To each patient corresponds one lungs 3-D CT scan (Figure 1), and one binary segmentation mask that locates the tumor volume. This kind of data is particularly appropriate in tumor detection since it detects the difference of density between lungs (filled of air) and tumors (composed of dense tissues). The mask was built based on annotations from radiologists, who delineated by hand the tumor(s).

- Input: Images (one scan and one mask per patient), radiomics features (an ensemble of 53 numerical features per patient, extracted from the scan), clinical data (age of the patient, TNM staging of the cancer ...)
- Output: the time to the event along with an indicator of the event (1 if the event is the patient's death, 0 if the patient escaped the study).

	Training	**Test**
# of patients	300	42
Median survival time (days)	644	532
Censor	45%	45%

Table 1: Description of the dataset

For each patient			
1 CT scan	53 numerical features	Clinical data (age, stage of the tumor)	Survival time (days) Censor variable 1 = censored patient 0 = death observed

Table 2: Composition of the Dataset

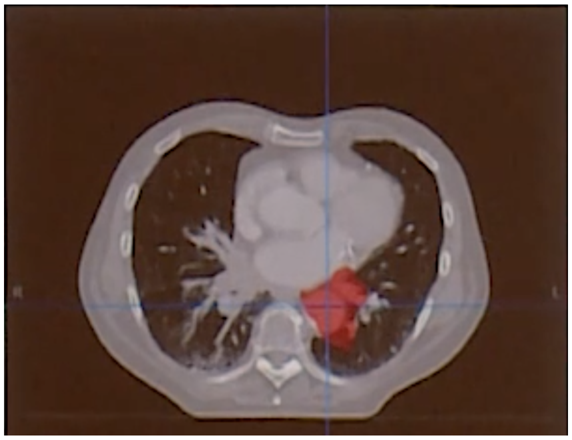

Figure 1: CT scan of a lung tumor

3 Related work

Several survival analysis methods incorporate deep learning techniques. We plan to explore and adapt to our problem three different kinds of algorithm: Cox-based, RNN-based and DeepHit.

- Cox-based deep learning algorithms, such as Cox-nnet [2] and DeepSurv [4], are based on the Cox proportional hazards model. It is a semi-parametric survival analysis method which consists in modeling the hazard function (probability of death at time t) as a product of a parametric hazard ratio depending on the features and a baseline hazard. Deep learning can then be applied to learn the optimal parameters, in terms of likelihood, of the hazard ratio.

- RNN-based survival analysis models [3, 8] are promising as well, due to the sequential and time-based nature of the problem. RNN-SURV [3], at each time step, computes an embedding of the features which goes through a recurrent layer, and a probability of death is computed after that layer. The model is trained with a combination of a cross-entropy loss and an upper-bound of the C-index, presented in Section **??**.

- DeepHit [5] uses a deep neural network to learn the distribution of survival times directly. It can handle competing risks (several events that can potentially occur) be we are mainly interested in the single event model.

None of these models have considered non-tabular inputs such as images. We plan to adapt one or several of these architectures so that they take CT scans as input as well, by taking advantage of CNN architectures. As a baseline, we will use a Cox-based model with numerical inputs.

4 Baseline: Cox Proportional Hazard Model

The very first step of the analysis is to generate predictions of the survival time with the numerical features (clinical and radiomics data) provided. For this purpose, we use the **Cox Proportional Hazard** model. In this model, the risk of a patient with characteristics x at time t is:

$$h(t|x) = h_0(t) \exp(\beta^T x) \tag{2}$$

where $h_0(t)$ is the baseline proportional risk factor that reflects the underlying hazard for subjects with all covariates being equal to zero. The parameter we estimate here is the vector β. This model has a good interpretation since a negative value of the parameter β_k indicates that the feature x_k is protective while a positive value of β_k indicates a worse prognosis.

4.1 Likelihood Estimation of β

Let us set the notations: we consider m patients and suppose for the sake of simplicity that we have m different survival times $t_1, ..., t_m$. For patient i we observe (x_i, y_i, δ_i) where x_i are the features (clinical and radiomics), y_i is the patient's survival time and δ_i is the indicator that the patient is non censored (e.g the patient died). Finally, let $\mathcal{R}(t) := \{i | y_i \geq t\}$ be the set of patients who are at risk of death at time t.

In this context, we define the partial likelihood as follows:

$$L(\beta) = \prod_{i, \delta_i = 1} \frac{h_0(y_i) \exp(\beta^T x_i)}{\sum_{k \in \mathcal{R}(y_i)} h_0(y_i) \exp(\beta^T x_k)} = \prod_{i, \delta_i = 1} \frac{\exp(\beta^T x_i)}{\sum_{k \in \mathcal{R}(y_i)} \exp(\beta^T x_k)} \tag{3}$$

This likelihood is thus the product over all non censored patients of the proportional risks of each patient with respect to the others. For the purpose of Cox regression, we want to maximize this likelihood. The Jacobian and Hessian matrices of the log-likelihood are given by:

$$\begin{cases} l'(\beta) = \sum_{i,\delta_i=1} \left(x_i\beta - \log\left(\sum_{j,y_j \geq y_i} \exp(\beta^T x_j) \right) \right) \\ l''(\beta) = -\sum_{i,\delta_i=1} \left[\frac{\sum_{j,y_j \geq y_i} \exp(\beta^T x_j) x_j x_j^T}{\sum_{j,y_j \geq y_i} \exp(\beta^T x_j)} - \frac{\left(\sum_{j,y_j \geq y_i} \exp(\beta^T x_j) x_j \right) \left(\sum_{j,y_j \geq y_i} \exp(\beta^T x_j) x_j^T \right)}{\left(\sum_{j,y_j \geq y_i} \exp(\beta^T x_j) \right)^2} \right] \end{cases} \tag{4}$$

Given these expressions, we maximize the log-likelihood using a Newton-Raphson algorithm and deduce an estimation $\hat{\beta}$ of β.

4.2 Implementation and Results

Based on expert recommendations [6] we selected the following features: the tumor sphericity, the tumor's surface to volume ratio, the tumor's maximum 3D diameter, the dataset of origin, the N tumoral stage, the tumor's joint entropy, the tumor's inverse difference and the tumor's inverse difference moment. This method results in a C-index of 0.687 on the dev set.

5 Deep Learning Cox-based method

5.1 Method

The DeepSurv method [4] is a Cox-based Deep Learning method. The idea of this method is to generalize the Cox proportional hazard model by modeling the log-risk function as a neural network instead of a linear function (which is originally $\beta^T x$). By introducing non-linearity, we are able to model more complex risk functions. There is, indeed, no reason for the ground truth log-risk to depend linearly on the features. Then, the loss function, defined in 5, corresponds to the negative log-likelihood (3), but with the original log-risk replaced by the network output $h_\theta(x)$. The loss function is optimized during training using the Adam optimization.

$$L(\theta) = -\frac{1}{N_{\delta_i=1}} \sum_{i=1,\delta_i=1}^{m} \left(h_\theta(x_i) - \log \sum_{k \in \mathcal{R}(y_i)} \exp(h_\theta(x_k)) \right) \tag{5}$$

5.2 Results

We implemented the DeepSurv method in Python, by making use of the Pycox package [7], which is a time-dependent prediction package based on Pytorch. The Neural Network architecture is described in Table 5.2. Since we have very few data points for training (250), the Neural Network is prone to overfitting. To correct that, we added Dropout layers to the network. With this method, we improved the baseline on the dev set with a C-index of 0.708.

Neural Network h_θ			
Layer	Number of outputs	Parameter	Activation function
Input x	56	-	-
FC	32	-	ReLU
Dropout	-	0.4	-
FC	16	-	ReLU
Dropout	-	0.3	-
FC	8	-	ReLU
Dropout	-	0.2	-
FC	1	-	None

Table 3: Architecture of the DeepSurv neural network

6 Incorporating image data

6.1 Model

The dataset that we have contains images that we can take advantage of. The most challenging part of the project is thus to select the best architecture in order to harness the full predictive power of this data. We incorporate those images to the model and as of now we make the choice to operate a regression on the survival time. The reason for this change of approach is that we want to fully exploit the predicting potential of the image data.

The final architecture we adopted for our model is a ResNet18 Neural Network. This network is composed of four building blocks which are all composed of successions of:

- 2 dimensional CNNs
- Batch Normalization units
- Fully Connected layers with ReLu activations
- 2 dimensional MaxPool units

These four building blocks are followed by an Average Pooling layer and a linear unit since we are working on a regression task. The total number of trainable parameters for the model is 11179512.

For the sake of efficiency, as suggested in the class, we used **Transfer Learning** to carry this regression task. Thus, we chose a pre-trained Neural Network and froze all the layers except the fully connected layers and the first two dimensional convolutional layer. The pre-trained network was loaded from [1].

We then trained our model to minimize the Mean Square Error Loss between the survival times provided in the data set and the outputs generated by the ResNet. At the end of our iterative process to find the best overall architecture, we chose the following set of parameters to carry out this regression task:

- Number of batches: 16, which amounts to 19 epochs in our training.
- Optimizer: Adam with $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$

More importantly, we use Early Stopping with a patience parameter of 10 in order to reduce overfitting of the training data, which is a consequence of the small size of the data set.

6.2 Analysis of the Results

By running the model on the test set, we come up with a Concordance Index of 0.72. This result confirms the predictive potential of the image data since the Cox model of part 5 on radiomics and clinical data led to a Concordance Index of 0.708. This fact is not surprising since radiomics data are in fact derived from the analysis of CT scans. Finally, using deep learning techniques on image leads to a better ordering of patients' death risk. Our final model enables us to give an accurate estimation of which patient has the most important risk for 72% of the pairs of admissible patients.

References

[1] `"https://download.pytorch.org/models/resnet18-5c106cde.pth"`. Online; accessed 10 November 2020.

[2] Travers Ching, Xun Zhu, and Lana X Garmire. "Cox-nnet: an artificial neural network method for prognosis prediction of high-throughput omics data". In: *PLoS computational biology* 14.4 (2018), e1006076.

[3] Eleonora Giunchiglia, Anton Nemchenko, and Mihaela van der Schaar. "RNN-SURV: A deep recurrent model for survival analysis". In: *International Conference on Artificial Neural Networks*. Springer. 2018, pp. 23–32.

[4] Jared L Katzman et al. "DeepSurv: personalized treatment recommender system using a Cox proportional hazards deep neural network". In: *BMC medical research methodology* 18.1 (2018), p. 24.

[5] Changhee Lee et al. "DeepHit: A Deep Learning Approach to Survival Analysis With Competing Risks." In: *AAAI*. 2018, pp. 2314–2321.

[6] *Owkin challenge: Predicting lung cancer survival time*. `https://challengedata.ens.fr/challenges/33`. Online; accessed 30 September 2020.

[7] *Pycox: Time-to-event prediction with PyTorch*. `https://github.com/havakv/pycox`. Online; accessed 30 September 2020.

[8] *WTTE-RNN: Hackless churn modeling*. `https://ragulpr.github.io/2016/12/22/WTTE-RNN-Hackless-churn-modeling/`. Online; accessed 30 September 2020.