UNIVERSIDAD
DE MÁLAGA

E.T.S.
INGENIERÍA
INFORMÁTICA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

GRADO EN INGENIERÍA DE LA SALUD

APRENDIZAJE PROFUNDO APLICADO A PROBLEMAS DE PREDICCIÓN DE SUPERVIVENCIA EN CÁNCER

DEEP LEARNING FOR CANCER SURVIVAL PREDICTION

Realizado por

MARÍA DEL ROCÍO CABELLO TOSCANO

Tutorizado por

FRANCISCO JAVIER VEREDAS NAVARRO

Departamento

LENGUAJES Y CIENCIAS DE LA COMPUTACIÓN

UNIVERSIDAD DE MÁLAGA

MÁLAGA, Junio 2019

Fecha defensa:

El Secretario del Tribunal

Resumen

El cáncer se cobró 18,1 millones de muertes a nivel mundial en 2018 y $87,8 billones para cuidados de salud durante el año 2014 en EEUU. El tremendo impacto que esta enfermedad supone a nivel mundial, junto con la disponibilidad cada vez mayor de datos genómicos y transcriptómicos, han potenciado el interés en incorporar tecnologías de vanguardia, como es el Aprendizaje Profundo (AI), a la lucha contra el cáncer. AI ha destacado en los últimos años, particularmente por el rendimiento de los modelos de Redes Neuronales Convolucionales (RNC) en reconocimiento de imágenes. El problema para el cual todos los modelos de este proyecto han sido entrenados es la predicción de supervivencia en cáncer en un conjunto discreto de intervalos de tiempo a partir de datos de RNA-Seq, debido a la importancia que el análisis de la supervivencia tiene en cuanto al estudio de los tratamientos contra el cáncer y su mejora. La propia naturaleza de los datos biológicos trae consigo algunos inconvenientes cuando se usan para entrenar modelos de RNC. Estos datos normalmente están formados por un número mucho mayor de variables (M) que de observaciones (N). Esto se conoce como *la maldición de la dimensionalidad* (en inglés, *the Curse of Dimensionality*) (M$>>$N). Otro inconveniente es la falta, a priori, de información espacial entre las variables biológicas. RNC son un tipo de modelo concreto de Aprendizaje Profundo que está especialmente pensado para el procesado de imágenes, en las cuales los píxeles que las componen se relacionan con sus píxeles vecinos. Esta relación se usa en las RNC para extraer más conocimientos de las observaciones y tener, en consecuencia, un mejor rendimiento. En este proyecto se proponen algunas estrategias para tratar de resolver estos dos inconvenientes. Con el objetivo de equipar a los perfiles de expresión génica con estructura, cinco estrategias han sido propuestas, aplicadas y comparadas. De manera similar, la estrategia de *aprendizaje por transferencia* conocida en inglés como *fine-tuning*, ha sido aplicada para tratar de resolver el inconveniente al que nos referimos como *la maldición de la dimensionaidad*. La comparación de estos modelos, todos entrenados con el mismo conjunto de variables y observaciones, ha sido realizada mediante el cálculo del Índice de Concordancia.

Palabras clave: Bioinformática · Redes Neuronales Convolucionales · RNA-Seq · Análisis de la Supervivencia · Cáncer

Abstract

Cancer claimed 18.1 millions deaths worldwide in 2018 and $87.8 billion for health-care in 2014 in USA. The tremendous impact this disease supposes worldwide, combined with the increasingly availability of genomic and transcriptomic data, have aroused the interest on incorporating cutting edge technologies, such as Deep Learning (DL), in the fight against cancer. DL has stand out in the last years, particularly because of the performance of the Convolutional Neural Networks (ConvNets) models in image recognition. The problem for which all models in this project have been trained is the prediction of cancer survival in a discrete set of time intervals, from RNA-Seq data, because of the importance survival analysis have in the study of cancer treatment and its improvement. The very nature of biological data brings some inconvenients when using it for training a ConvNet model. These data are usually composed by a much bigger number of features (M) than observations (N). This is known as *the Curse of Dimensionality* (M$>>$N). Other inconvenient is the lack, a priori, of spatial information among biological features. ConvNet is a DL model which is specially designed for image processing, in which the pixels composing them are related to its neighbour. This relation is used by ConvNets to extract more knowledge from observations and have, in consequence, a better performance. This project proposes some strategies to try to solve these two inconvenients. In order to equip gene-expression-profiles with structure, five strategies have been proposed, applied and compared. Similarly, the *transfer learning* technique known as *fine-tuning* have been applied to try to solve the inconvenient which we refer to as *the Curse of Dimensionality*. The comparison of these models, all trained with the same set of features and observations, has been made by calculating the Concordance Index (C-index) metric for each of them.

Keywords: Bioinformatics · Convolutional Neural Networks · RNA-Seq · Survival Analysis · Cancer

Agradecimientos

Me gustaría dar las gracias a mi tutor por haberme proporcionado los medios técnicos adecuados para la realización de este trabajo. Su atención y conocimientos me han sido muy valiosos.

También al grupo de investigación *Inteligencia Computacional y Biomedicina* (ICB) por darme la oportunidad de aportar mi granito de arena en un proyecto tan interesante.

En general, a todos aquellos (compañeras/os, docentes, amigas/os y familiares) que han aportado todo lo que estaba en su mano para ayudarme durante el transcurso de estos cuatro años.

Muchas gracias.

Contents

1 Introduction

1.1 Cancer and its socio-economical impact today

It is known as cancer the set of related diseases in which it is observed an uncontrolled process in cells division that gives place to tumors. This disease, if not well treated, ends up with the death of patients. Cancer deaths accounted 18.1 millions of deaths worldwide in 2018 (IARC, 2019). In particular, in Spain died around 113 thousand people in 2017 because of tumors, that supposes the 26.68 percent of all deaths over the year (figure 4) (INE, 2019). According to the Global Cancer Observatory (GLOBOCAN), this number is expected to increase worldwide over the years. In fact, they also expect that 1 out of 8 men and 1 out of 10 women will develop cancer in their lifetime (Bray et al., 2018). As shown in figure 3, the most diagnosed cancer is the one that affects breasts, even when males essentially do not suffer it. However, breast cancer survival expectations are quite high in view of having a really bigger number of incidences than deaths. Lung cancer expectations are not as good as breast's, being one of the most diagnosed cancer and whose number of caused death the highest (figure 3)

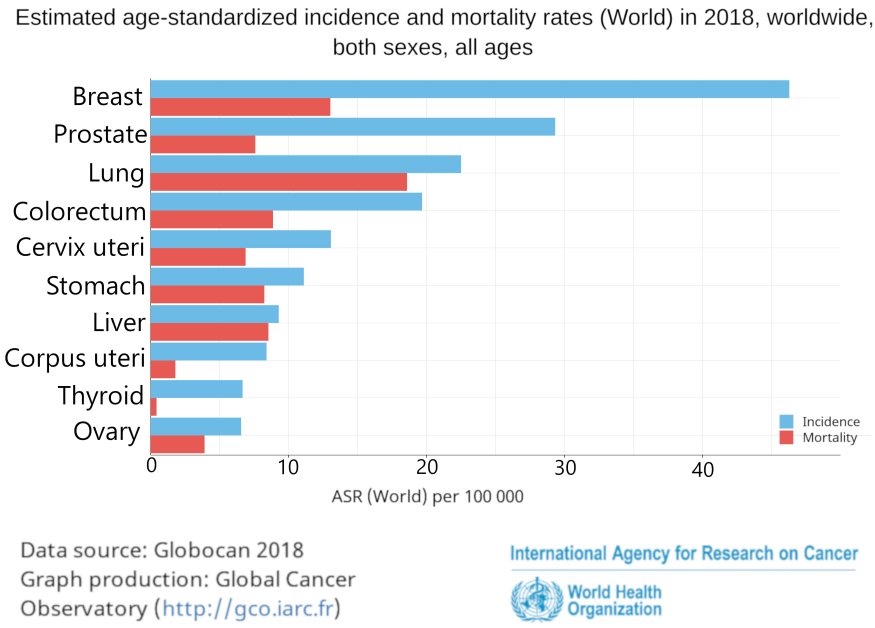


Figure 3: How cancer mortality and incidence vary depending on cancer sites, in terms of Age-Standardized Rate (ASR). ASR measures the number of new cases (incidence) or deaths (mortality) per 100.000 persons per year if population had a standard age structure. This is useful because age influences on cancer risk. Source: *Global Cancer Observatory* (IARC, 2019) (Adapted: Re-sized fonts)

All of this suppose a huge impact in today's society. Socially, patients do not only suffer physically but psychologically too, as well as their relatives. However, it is also important to mention the tremendous financial impact cancer supposes. For

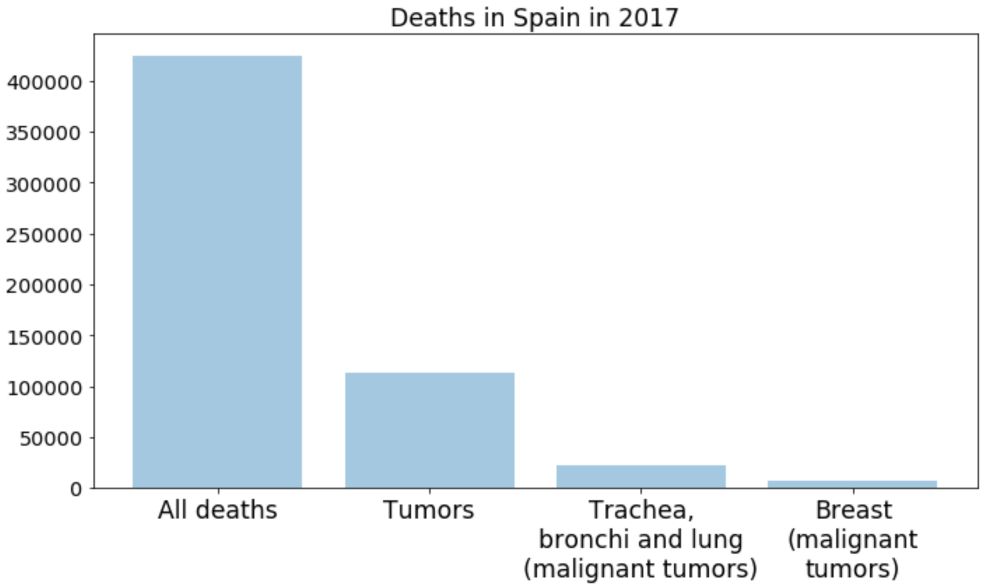


Figure 4: Number of deaths in Spain during 2017. First bar on the left represent all deaths, the second those caused in general by tumors, the third those caused by windpipe and lung tumors, and the fourth those caused by breast tumors. Source: *Instituto Nacional de Estadística - Spain* (INE, 2019).

both countries with public and private health insurances, important amounts of money are invested for fighting cancer either by the government or the patient himself. One example could be in the United States of America, where in 2014 approximately $87.8 billion were spent on cancer health care (Singleterry, 2017).

In this point the question is: how to improve cancer treatment in order to reduce both economical and living costs?

1.2 Survival analysis

As expected, there is a huge interest globally in finding new ways and improvements for treating cancer. One of them is based on studying patients survival, which aims to analyze the relation between surviving through time and some measurable features. Traditionally it has been done by using Proportional Hazard Models, commonly known as Cox Regressions (Cox, 1972), or Kaplan-Meier Estimators (Kaplan & Meier, 1958). Both are statistical methods belonging to the statistics branch known as survival analysis. In this field, it is studied how many time passes until an event happens. In this case, the event is deceasing because of cancer.

Cox Regressions estimates the probability of deceasing (λ) a patient has depending on some features (X_i) and in a precise moment of time (t) (equation 1).

$$\lambda(t, X_1, \cdots, X_n) = \lambda_0(t)\, exp(\Sigma_{i=1}^{n} \beta_i X_i) \tag{1}$$

On the other hand, Kaplan-Meier estimator evaluates the probability of deceasing that one or more patient populations

have along an interval of time (eg. figure 5). Both methods are really useful, for example, to determine which treatment is most suitable for each patient survival according to their own clinical or genetic conditions.

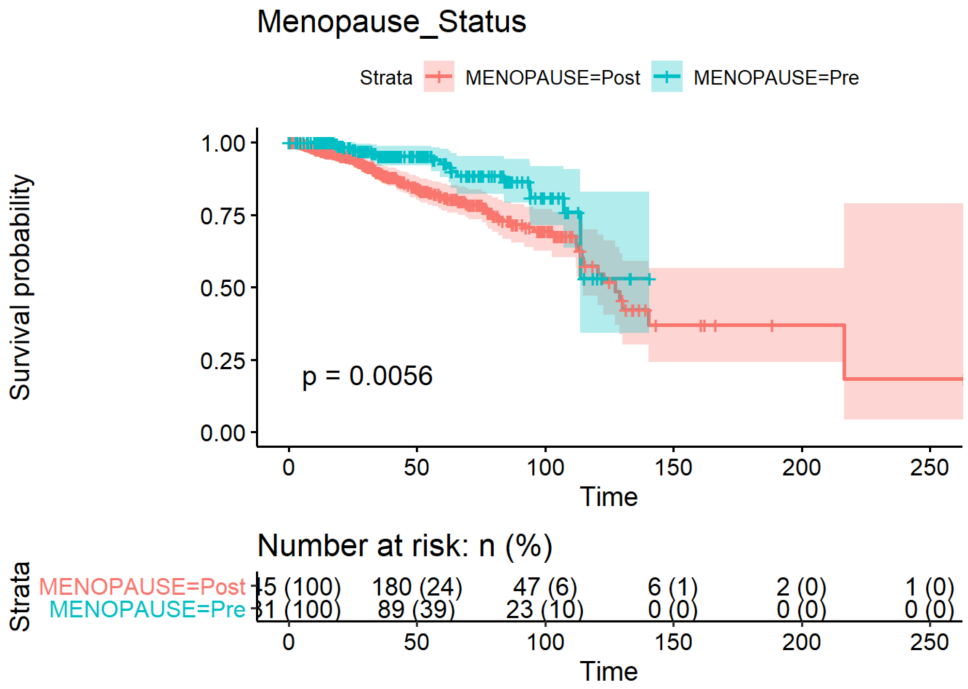


Figure 5: These Kaplan-Meier Curves represent the survival probability estimated for two groups of breast cancer patients, post-menopausal and pre-menopausal women, along 250 months. As observed, pre-menopausal patients are more likely to survive before 100 months than the post-menopausal ones. After this time it is not statistically reliable to extract any conclusion. Moreover, 'p=0.0056' represents that the fact of being pre or post-menopausal is significant when estimating breast cancer survival, because of a p-value lower than 0.05.
Data source: Pancancer database - TCGA

1.3 Genomics, Transciptomics and Cancer

Genes are sequences of DNA coding for molecules, which perform biological functions in the organism they belong to. The whole set of genes characterizing an organism or a specie is its genome, and its study is known as Genomics. Thanks to Next-Generation Sequencing (NGS) (Behjati & Tarpey, 2013) sequencing is increasingly cheaper and faster, and its output more precise and abundant. When genes are expressed, they are transcribed to RNA. The field studying transcription and in consequence gene expression is known as Transcriptomics. In this omic science, as in Genomics, has had a revolution too. In figure 6 it is shown how three of the main transcriptomics techniques have been gaining or losing popularity in terms of the number of references they had in Pubmed papers. As can be observed, in the last years RNA-Seq have had a great hype. This is probably the reason for MicroArrays to stabilize or even descend in popularity. RNA-Seq technology uses NGS to identify which genes are being expressed and this expression intensity in an specific moment and tissue/cell.

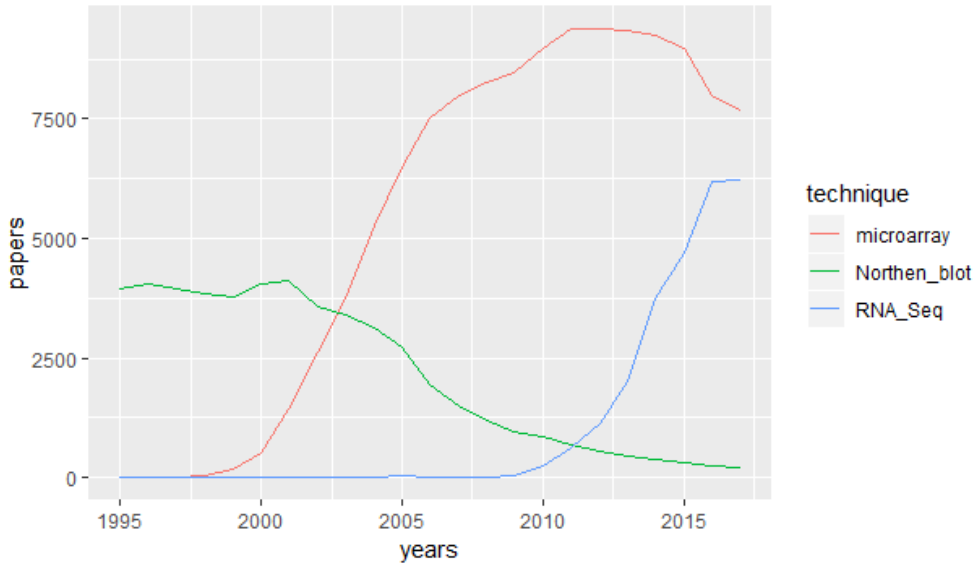


Figure 6: References to transcriptomics techniques through time in NCBI Pubmed (NCBI, 2019). Data retrieved from NCBI with Entrez.

The data obtained by these methods really helps in the investigation of diseases, such as cancer, because it makes easier to find out genomic implications in diseases or most suitable treatments for them, even brings us closer to precision medicine. There are plenty of existing projects investigating cancer from this point of view (van Lanschot, Bosch, de Wit, Carvalho, & Meijer, 2017). One of them is The Cancer Genome Atlas (TCGA), whose freely available data about gene expression profiles in cancer patients has been used in this final-degree project.

1.4 Artificial Intelligence, Deep Learning and Convolutional Neural Networks

Within Computational Sciences world, there is a field known as Artificial Intelligence (AI) which aims to equip machines with basic principles of human intelligence. One of this principles and with which which AI tries to equip machines is *learning*, and there is an specific branch in AI for it: *Machine Learning* (ML). ML algorithms and models try to learn from observations. Thus, they tend to abstract conclusions from an observation set.

This is really useful when the observation set is complex or too big. That is the case of genomic and transcriptomic data mentioned in the previous section. AI is a good way to manage this data in order to extract useful knowledge about health care improvements.

In Machine Learning, there is a subfield that has become very relevant because of its success and popularity. This is the subfield known as Deep Learning (DL). DL is based on Artificial Neural Networks (ANN), a ML model inspired by how neurons work in a biological context. As in brain, the basic unit in ANN and DL is the neuron. One neuron is represented as one lineal equation activated by a non-lineal function, whose inputs (independent variables) and output (dependent variable) symbolize synaptic connections which communicate neurons between each other. That is to say one neuron

output corresponds to an input for other neuron. In this way, neurons are arranged by layers, each layer containing an specific number of neurons (units) whose inputs are the output of the previous layer, and whose outputs are the inputs of the following layer. That is known as the network architecture (figure 7). But, how are this networks supposed to learn? People tend to learn from its own mistakes, and ANN are not different. There are many learning strategies but a well-known one is the Backpropagation Algorithm. The idea of this algorithm is to compute the committed error in the output layer, and based on it and by the use of derivatives, the synaptic weights are updated taking into account the weights in the way from the outputs to the inputs (backward). These weights are nothing more than the coefficients of the lineal equations that represent each neuron.

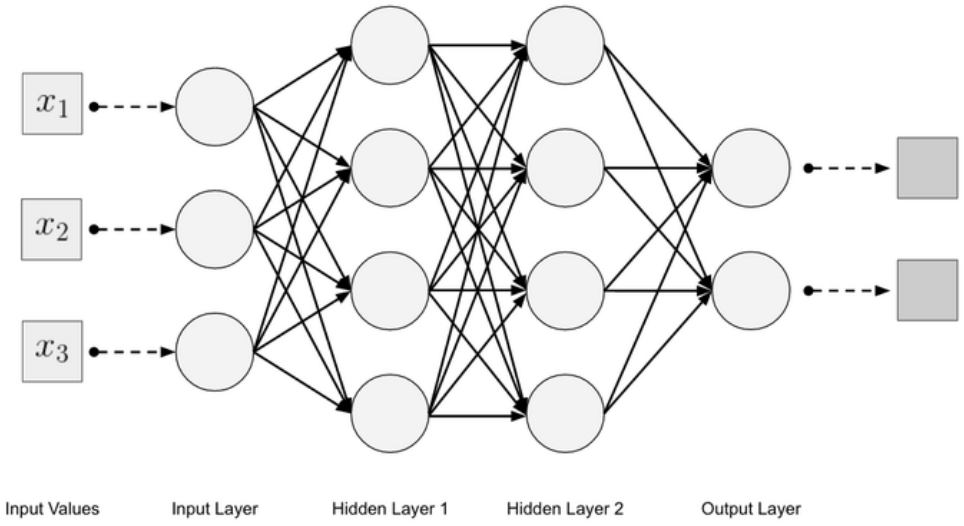


Figure 7: Network architecture of a two-hidden-layers Artificial Neural Network. Circles symbolize neurons/units and arrows the synaptic connections.

Deep learning (DL) is a subfield from Machine Learning that has become very relevant because of its success and popularity. But, which is the difference among DL and ANN? DL models are characterized by having a considerable number of layers, but having two hidden layers is enough for being considered DL, if the model is accompanied by typical algorithms of DL. For example, the use of *drop-out* and the *ReLU* function is characteristic of this types of models. It has stand out because of its good results in many application areas such as image recognition or natural language processing. More precisely, the model in DL that carries out the best results in image recognition is the known as Convolutional Neural Networks (ConvNets). Inputs in this networks are images, in other words, 2D matrices in which each number corresponds to the intensity of a pixel. Each 2D image could be composed by various channels, one channel for black and white images and three channels for RGB (colored) images. Even 3D matrices can be used when time is also a variable, that is the case of video inputs. The main idea that differentiate ConvNets is the use of convolutions. These are filters that recalculate each pixel value regarding its neighbour pixels (Chollet, 2018). The typical network architecture that is followed in ConvNets models is specified in the figure 8. Dense layers at the end of the model are based on ANN layers. Convolutional layers are those in which the mentioned filters are located. A convolutional layer is composed by four steps: Convolution, normalization, maxpolling and dropout. A convolution is

analogous to a filter. These layer recalculate each element in its input according to their neighbour elements. After doing it, the obtained new representation of the data is normalized and maxpolled, this is to say that only those higher values are taken into account. At the end, drop-out is applied. Drop-out is to randomly cut off some of the connections between two correlative layers. This is useful to reduce over-fitting, that is to say that the model is so specialized on the training data set that loss its ability of predicting from unknown data.

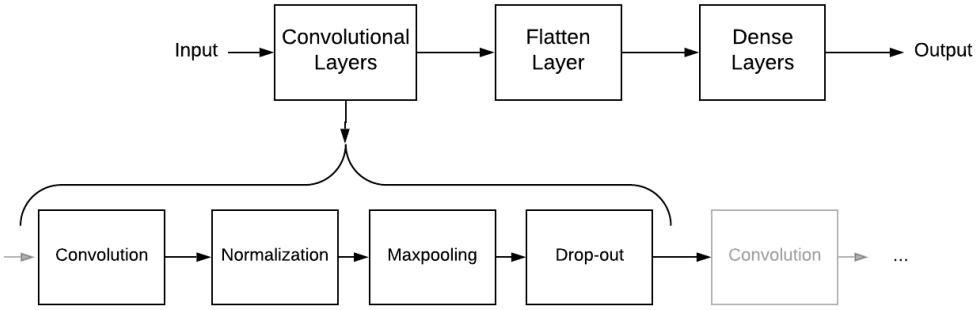


Figure 8: ConvNets schema of network architecture.

1.5 Deep Learning for Cancer Survival Prediction

Applying Deep Learning algorithms in biological and medical areas, with the same good results and acceptation as it has in other fields, is one of the challenges bioinformaticians are working on at the present. The very nature of the data prevailing in these areas make it difficult. On the one hand, DL and Big-Data go together, that is because DL needs big amounts of data for optimally learning and obtaining good results. Biological and clinical data sets usually have the particularity of having a really big number of features (M) than observations (N). This is known as ”the Curse of Dimensionality” and usually develop into bad models. On the other hand, data such as those of genomic nature suffers from not having spatial or temporal information which provide an intrinsic structure that could be exploded by the typical network architectures of Deep Learning, such as Convolutional or Recurrent Neural Networks. Those applications of DL in which a high performance is demonstrated are characterized by possess data with spatial structure, as it is the case of images. However, gene expression data do not have, a priori, this structure.

As expressed in the previous sections (1.1, 1.2 and 1.3), there is a big interest on improving cancer treatments and its results. One way on trying it is to study patients survival, which seem interesting to be done by using transcriptomic data, such as gene expression profiles, and Convolutional Neural Networks (ConvNets). Solving the problems presented in doing it (*The Curse of Dimensionality* and the lack of spatial information) is the main objective of this project

What it is proposed in this final-degree project is the comparison of ConvNets models, fed with gene expression data, in which ”the Curse of Dimensionality” and the lack of spatial information have been sought to solve, by using respectively

Transfer Learning and biological information of the genes. The problem in which to apply these models is the prediction of cancer survival.

It is important to note that this project is framed within the research project TIN2017-88278-C2-1-R of the I+D National Plan, granted to the group 'Inteligencia Computacional y Biomedicina' (ICB) on the Languages and Computational Sciences Department of the University of Málaga.

2 State of the Art

Predicting survival by the use of DL models has been frequently the objective of multiple projects in bioinformatics for the last years. Matsuo K, Purushotham S, Jiang B, et al., proved that DL models have a better performance than Cox ones, when predicting survival outcome in cervical cancer from clinical data (Matsuo et al., 2019). Due to they count with 768 observations and 40 features, they did not have to deal with *the Curse of Dimensionality*. Ilkyu Han et al. also demonstrate a better performance of DL against Cox Regressions in predicting cancer survival, using clinical features and C-index as performance measure (Han, Kim, Park, Kim, & Seo, 2018). However, these two projects move away from our purposes because they use clinical data and we will use RNA-Seq data instead. Chaudhary et al. do use RNA-Seq data too, and complement it with biological data for predicting liver cancer survival with DL models (Chaudhary, Poirion, Lu, & Garmire, 2018).

In all these mentioned projects it is implemented basic Deep Neural Network, but our aim is to implement ConvNets. As their input data is image-based, there are many projects in which ConvNets have been used for predicting from biological or clinical images. Mostly all of these ConvNets models are focused on classification problems, such as cancer classification (Lyu & Ling, 2018), in spite of survival prediction.

It exists some projects in the same line of ours in respect of equipping genes with structure, with the aim of use omic data as ConvNets input. For example, Shiyong Ma and Zhen Zhang used tree-map images for grouping gene expressions by functional annotation of genes in the same are of the image (Ma & Zhang, 2018), but they do not use any technique or information to sort these genes inside each group. This modification on their idea could arise better results. Other example is the paper of Giuseppe Jurman et al., in which the distance between genes is computed based on Gene Ontology (GO) data (Jurman et al., 2017).

3 Methods

In the development of this final-degree project, it has been followed the methodology proposed as general framework in data mining (see figure 9). This framework follows four steps: *Data obtaining*, in which to download the input and output data needed for feeding the models, *data preprocessing*, in which to clean and transform the obtained data as applicable, and *model adjustment and training* and *results analysis*, two steps intended for training data-mining models and evaluating them.

In figure 10, it is represented a more detailed schema of the work-flow that have been followed throughout the project. Note that it is based on the original data-mining framework and the actions and steps in it will be explained in the following sections.

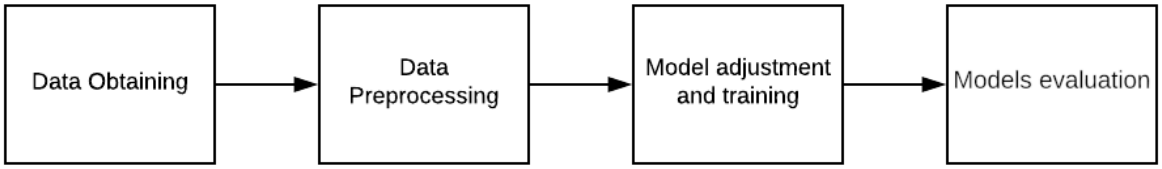


Figure 9: Steps followed in this final-degree project. (a) Typical data-mining framework. (b) More detailed schema, based on figure 7a, in which is represented the work-flow that has been followed in this final-degree project.

Regarding more technical aspects, the programming language that prevails in the implementation corresponding to the following sections is Python, although some R scripts have been implemented too. All the code have been implemented in Jupyter Notebook App by using Jupyter Notebooks, and it is available in the repository of github: https://github.com/emecete/DL_CancerSurvivalPrediction

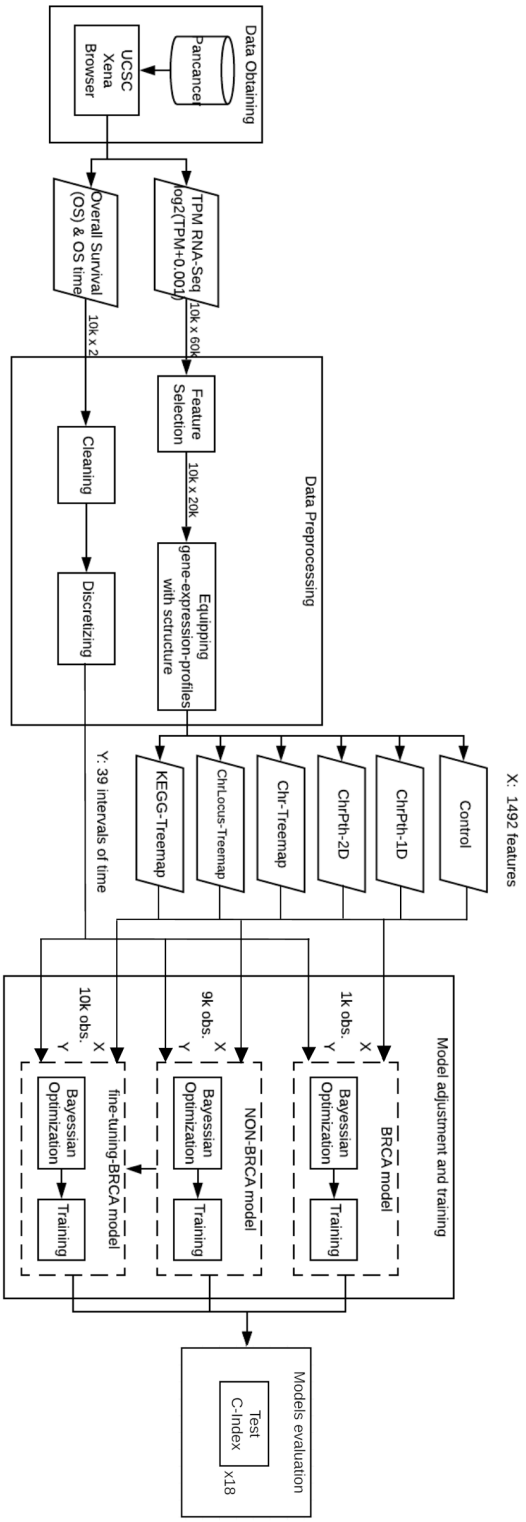


Figure 10: More detailed schema, based on figure 9, in which is represented the work-flow that has been followed in this final-degree project.

3.1 Data obtaining (and exploration)

The implemented models aim to predict survival on cancer patients, from their gene expression profiles. In consequence, model input is a gene expression profile and model output are the survival probabilities this patient has through a discrete interval of time. Both data sets have been obtained from a TCGA database known as Pancancer. This database is freely available on the public repository UCSC-Xena from University of California - Santa Cruz (UCSC).

Data have been obtained both by using its Python API (TCGADataobtaining.ipynb) and its web browser (https://xenabrowser.net/datapages/). In the first way, multiprocessing have been needed because of time-out error when retrieving big amounts of data.

It have been obtained gene expression data from a total of 10,535 patients, whose gene expression profiles correspond to 60,499 genes. All this patients do not suffer the same cancer but one of the 33 cancer types Pancancer database holds. Among these cancer types, the one more patients suffer on the obtained data is the *breast invasive carcinoma*, which we will refer to as *BRCA* (BReast CAncer). BRCA accounts 1,212 patients, this supposes the 11.5% of the total number of patients. At the view of figure 11, it can be assured that the number of observations in this data set is much lower than the number of features (around 60k), more specially if we focus on an specific type of cancer. As mentioned, the type of cancer that counts with the largest number of observations is *BRCA*, which seems like very numerous if compared with the second most numerous (*kidney clear cell carcinoma*), a type of cancer we have around 600 observations of. In view of these facts, it is clear that the proportion of observations and features in the obtained data supposes that the inconvenient of *the Curse of the Dimensionality* is present in our data.

This gene expression data comes from RNA-Seq. RNA-Seq, as expressed on section 1.3, is a transciptomics technique that uses Next-Generation Sequencing (NGS) to quantify the expression intensity a gene have in a cell or tissue. This is done by measuring the amount of RNA from this gene is present in the cell or tissue in a precise moment. In this case, RNA-Seq data obtained is measured in TPM (Transcripts Per Kilobase Million).

In the case of survival data, it have been obtained the overall survival (OS) and the OS-time (in days) of 10,532 patients (1,247 are BRCA patients). For each patient we have an OS value, which is 1 if patient failure or 0 if censored, after the specified days in the variable OS-time. A patient is censored when the follow-up study has finished and the event of deceasing has not been observed for the patient, or when the follow-up study has been interrupted, so that it is not possible to know if the patient dies because of the disease nor when.

Note that in our data set we count with more censored patients than deceased, and that both events occurs mainly during first 5 years (see figure 12). Typically, censored data is not taken into account when predicting survival, although models such as Cox Regression uses these observations. One of our objectives when training the corresponding ConvNet models is to use this censored data too.

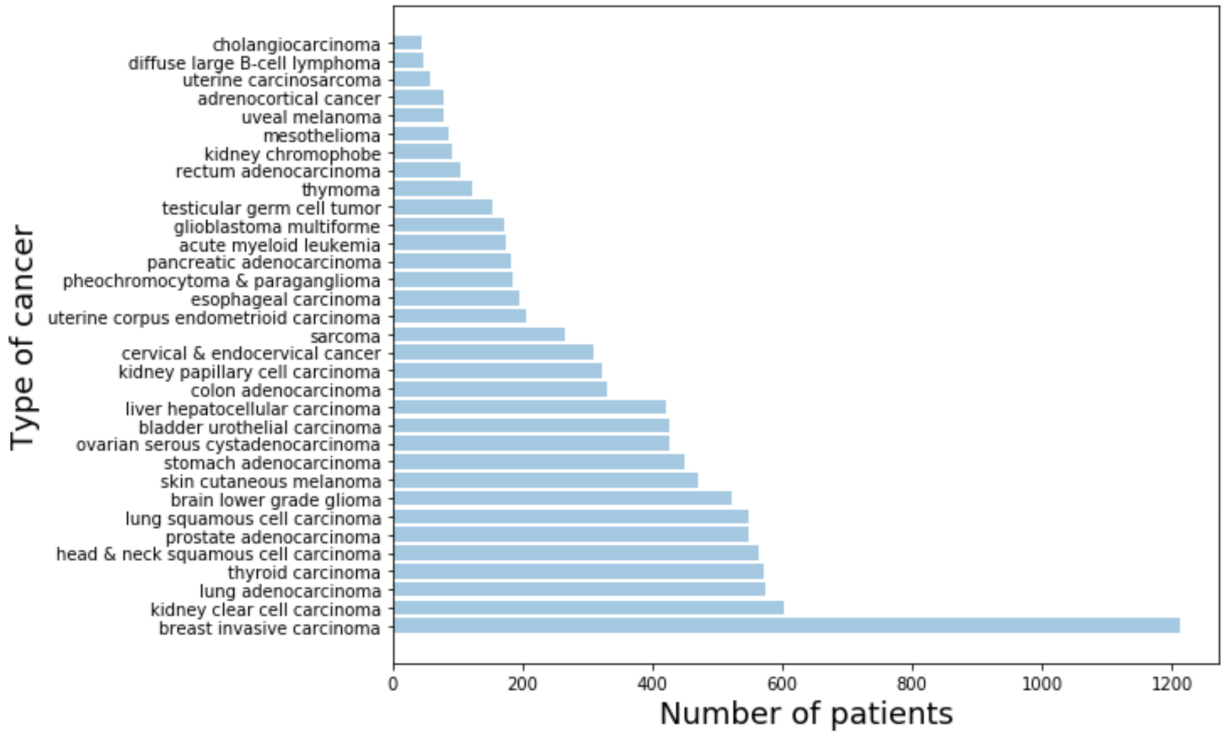


Figure 11: Available cancer types in Pancancer data set and the amount of available patients according to the type of cancer they are diagnosed with.

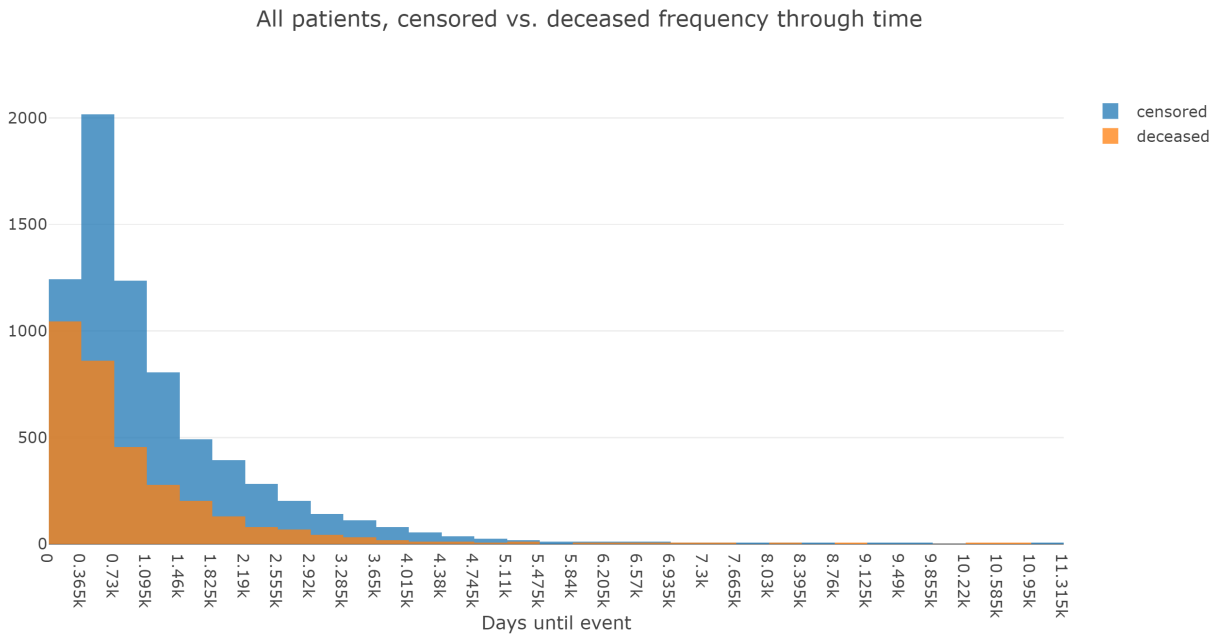


Figure 12: Histogram representing the distribution of Overall Survival data throughout time. This data belongs to Pancancer database.

3.2 Data Preprocessing

Apart from cleaning lost values, feature selection and the normalization and standardization of data we have used, in this step it is also included the different transformations that have been carried out in order to equip gene expression profiles with structure and to allow survival to be predicted in a discrete range of time.

With the aim of solving the lack of spatial information in gene expression data, five strategies, in which biological information of the genes were used to supply this data with structure, were developed. These were named as: *ChrPth-1D*, *ChrPth-2D*, *Chr-Treemap*, *ChrLocus-Treemap* and *Kegg-Treemap*. These strategies are illustrated on section 3.2.2.

3.2.1 Gene expression profiles - Feature selection

There was no need of cleaning lost values, because there was not any of them in the retrieved data. In terms of normalization and standardization, neither operation was needed due to data being already normalized and standardized by log2(TPM+0.001) transformation.

In the problem of predicting breast cancer patients survival, we had around 60 k features (genes) and 1 k observations (patients), which is the pure example of *the Curse of Dimensionality*, one of the problems that want to be solved in this project. The first thing that was done in the intention of solving it was to apply an unsupervised filter strategy for feature selection:

Let be X an $m \times n$ matrix representing our gene-expression-profiles data set, where m is the number of observations and n the number of features. Being i and j integer numbers ($1 \leq i \leq n$ and $1 \leq j \leq m$), $\bar{X}_i$ is the arithmetic average of the gene expression levels for the ith feature, $\tilde{X}_i$ is the median of the gene expression levels for the ith feature and X_{ij} the gene expression level of the jth patient for the ith feature. In the first place, those genes whose expression values through all patients kept constant were dropped, these were those whose standard deviation (equation 2) were equal to zero. In consequence, 5,055 features were dropped. After this first screening, the Median-Absolute Deviation (MAD) (equation 3) of the remaining genes was computed, and those 20 k genes with the highest values of MAD were considered as the starting gene set with which to provide structural information to gene expression profiles.

$$\sigma_i = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(X_{ij} - \bar{X}_i)^2} \tag{2}$$

$$MAD_i = median(|X_{ij} - \tilde{X}_i|) \tag{3}$$

In figure 13 are represented both the standard deviation and MAD from those 20,000 genes that were selected by this strategy. As expected, these measures are correlated.

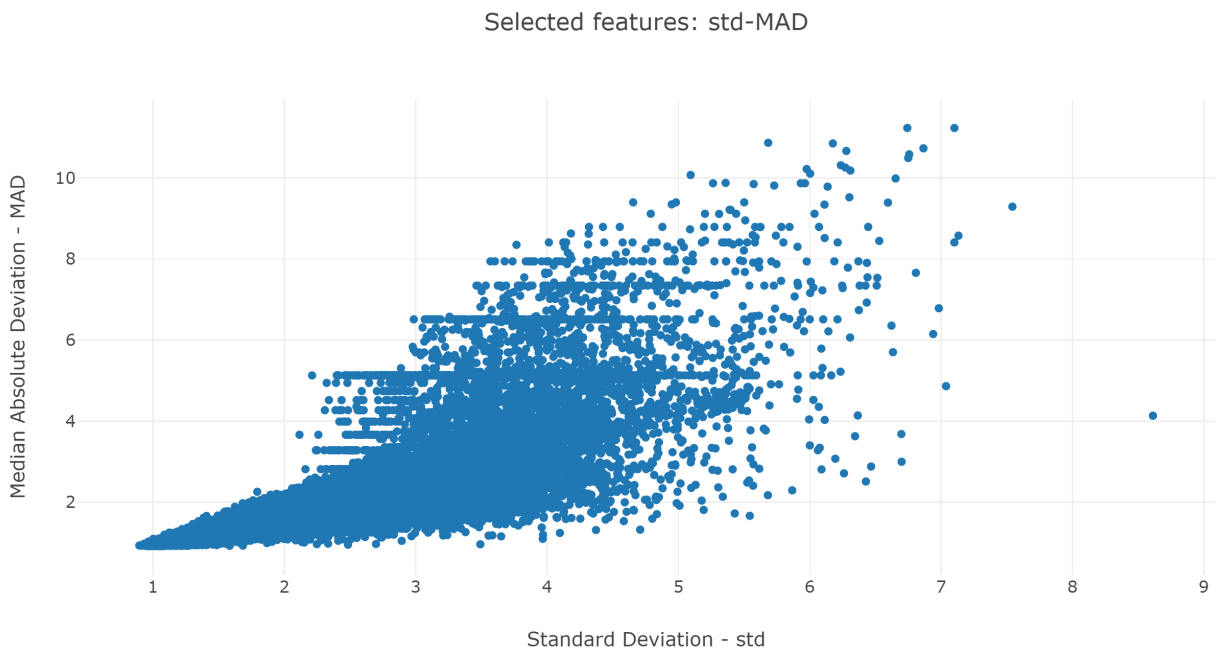


Figure 13: Relation between standard deviation (x axis) and MAD (y axis) of the 20k selected genes.

3.2.2 Equipping gene-expression-profiles with structure

Channels based

A black and white image is nothing else than a two-dimensional matrix, whose numeric values represent the intensity of each pixel in a grey scale (from 0 to 255, 0 is black and 255 is white). On the other hand, RGB images are composed by three channels, one per each primary color: Red, Green and Blue. That is to say that RGB images are three dimensional matrices whose shape is (width, height, 3). The colors that are observed in a RGB image are based on the combination of these three channels and, in consequence, how much each channel influences each pixel. The idea of channels is exploited in this set of strategies.

A biological pathway is a sequence of molecular interactions in a cell. Genes are directly involved with biological pathways and it exists databases, such as KEGG Pathway, from which we can know the pathways each gene participates in. If two genes participate in the same pathway, their expression could be related. Thus, it is interesting to use the idea of pathway participation as a strategy for equipping gene-expression-profiles with structure.

The two strategies grouped in *channels based* join the idea of channels and biological pathways in the same strategy. Additionally, the chromosomal locus of a gene is also taken into account in these strategies.

For each gene-expression-profile has been created a set of channels: one channel per pathway. In particular, 33 channels were created because only those pathways (33 pathways) in which more than 100 genes (from the selected 20k genes) are involved were taken into account. In this way, in each channel is represented the same gene-expression-profile, but the expression level of those genes that are not involved in the corresponding pathway are set as 0 (as

unexpressed in the pathway). Simultaneously, all genes are sorted according to their chromosomal locus. This is the position in which a gene is located in terms of chromosome. In figure 17 there is a graphical representation of the amount of genes we had per chromosome, and the idea of *chromosomal locus* is more detailed in the following sections. Afterwards, only those genes that are usable in all strategies at the same time were maintained, with the aim of honestly compare them. In this way, the number of genes was reduced to 1,492.

- ChrPth-1D

In this strategy, the gene-expression-profiles of 10,535 patients have been transformed to a set of 2D **vectors**. Each gene is represented as the combination of 33 channels and all genes are sorted according to their chromosomal locus. The shape of each gene-expression-profile is: (1492, 33) in terms of (genes, pathways), that is to say that each gene-expression profile is transformed to a set of 33 vectors. In figure 14 it is shown an example: the gene-expression-profile of the patient *TCGA-02-0047-01*. In this figure are represented 33 vectors of length 1492, and 100 of those 1492 genes are zoom for detail. It can be observed how the same expression value is repeated in different pathways and how some of the genes that are close in their chromosome intervene in one or several pathways at the same time.

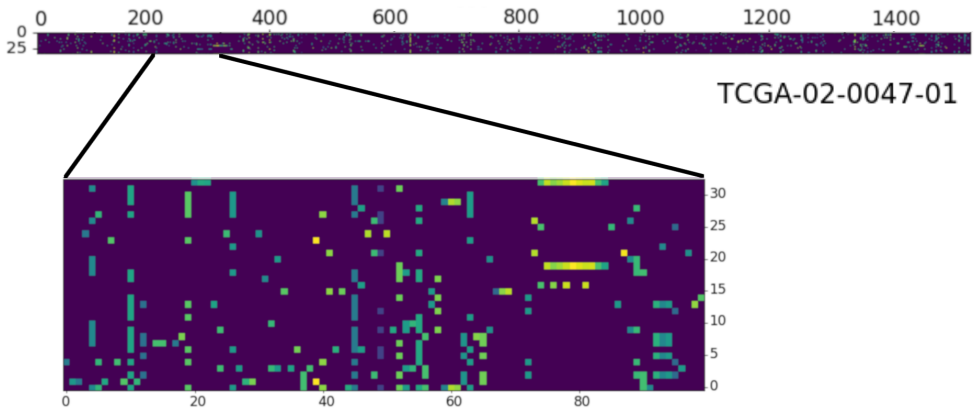


Figure 14: Example of gene-expression-profile after being transformed by strategy *ChrPth-1D*. X axis represents genes and Y axis represents pathways (33 channels). Expression levels are mapped into colors. Those expression levels in dark purple are equal to zero; those genes that are not involved in the specific pathway.

In this strategy, ConvNets go trough each vector and by the use of its convolutional filters extracts information from each level of expression in relation to those genes that precede and proceed, but not in relation to those genes that are in the same chromosome but in a more far position.

- ChrPth-2D

In this strategy, the gene-expression-profiles of 10,535 patients have been transformed to a set of 2D **matrices**. As in

strategy *ChrPth-1D*, each gene is represented as the combination of 33 channels and all genes are sorted according to their chromosomal locus. However, in this strategy each vector conforming a gene-expression-profile is reshaped to a matrix. The dimensions of these matrices are $n \times n$, where $n = \lfloor \sqrt{number\ of\ genes} \rfloor$, in this case $n = 39$. As the number of elements in each matrix is higher than the number of genes, there are extra elements that are padded with zero values. In figure 15a is represented where each gene is placed in the matrices when reshaping. Note that the final black row symbolize those padded values. The shape of each gene-expression-profile is: (39, 39, 33), that is to say that each gene-expression profile is transformed to a set of 33 matrices (1 pathway, 1 channel). In figure 15b it is shown an example: the gene-expression-profile of the patient *TCGA-3C-AAAU-01*. In this figure are represented 3 matrices of dimensions 39×39. Each pixel symbolizes the expression level of each gene, and those pixels in dark-purple stand for genes that are not involved with the corresponding pathway.

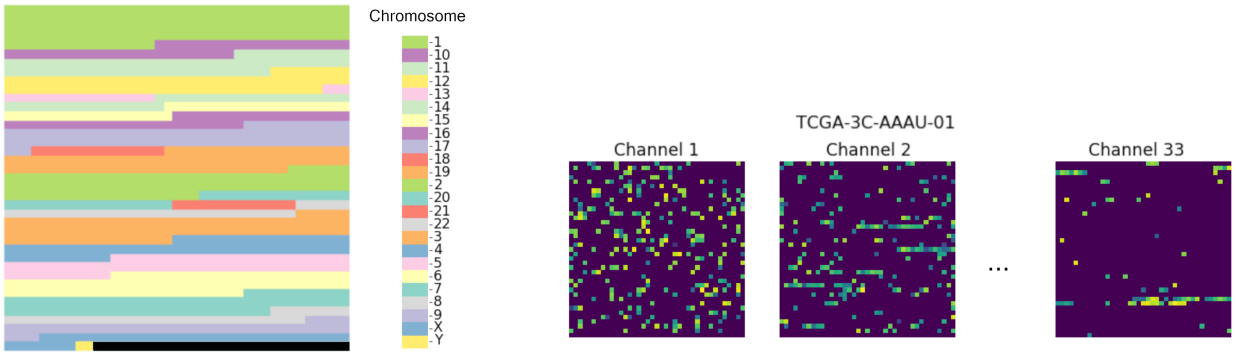
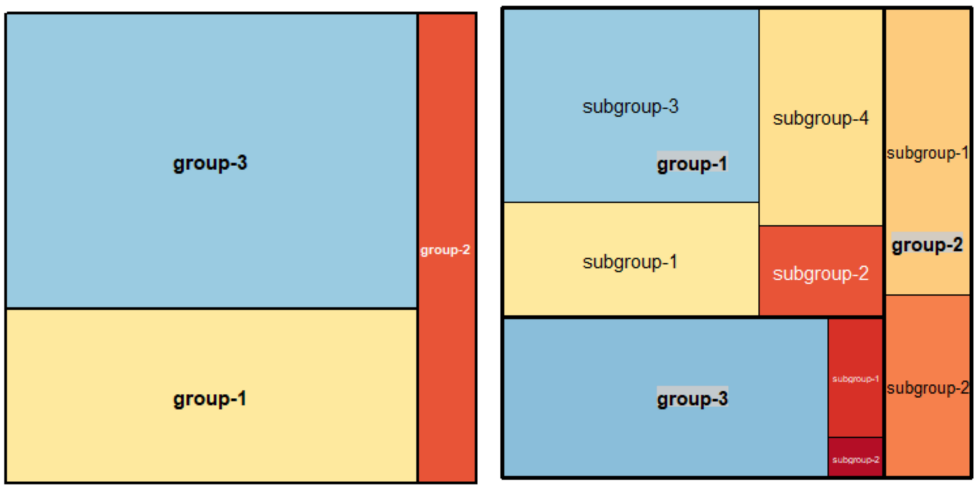


Figure 15: Transformation of gene-expression-profiles in strategy *ChrPth-2D*. (a) A map in which it is shown where each gene is placed in the matrix according to the chromosome it belongs to. (b) Example of gene-expression-profile after being transformed by the strategy. Each gene-expression-profile is composed by 33 matrices, but in this figure only three have been shown.

In this strategy it is tried that ConvNets not only extract information from the expression of genes in relation to those that precede and proceed, but also from other more distant genes that belongs to the same chromosome. However, due to the way in which the vectors are reshaped, as shown in the figure 15a, there are some genes that end up moving away from those that are in their same chromosome (e.g. those genes in chromosome 13).

Treemap based

A treemap is an hierarchical representation of hierarchical data. More precisely, rectangular treemaps are rectangular figures composed by rectangular subfigures of different sizes. Each rectangle in a treemap represents an hierarchy, and its size is related with a measurable characteristic of the hierarchy, for example the amount of data composing it. In figure 16, two examples of treemaps are shown. In them it can be visualized an hierarchical representation that results in three groups (figure 16a), and also in multiple subgroups (figure 16b).



(a) (b)

Figure 16: Example of treemaps images. (a) Only one level hierarchy. (b) Two levels hierarchy.

In this project, three of the strategies dedicated to equip gene-expression-profiles with structure are based on rectangle treemap concept. Their main idea is to generate one rectangle treemap image per patient. In this way, each treemap contains the gene-expression-profile of its patient. Features are grouped and sorted in the image according to biological aspects with the objective of representing closer the expression level of those genes that are biologically related.

All treemap images generated in the strategies that are detailed down below represent each feature as rectangles which are colored according to its gene expression level. Concurrently, these rectangles are grouped according to a biological hierarchy and sorted inside each group by some measure or biological aspect.

R CRAN provides a package called *treemap*. This package has been used for generating all treemap images in this project. Although the main language that have been used is Python, for this task it has been preferred to use R, since the R-package *treemap* provides a better documented than the provided by the Python library *squarify*. These two options were evaluated and finally it was chosen the use of R Scripts because of the advantages the *treemap* package presents. Concurrently, while (Ma & Zhang, 2018) re-implemented this R package for similar purposes, in this project it has been used the original functions *treemap* supplies. The function that has been mostly employed from the package is the named as itself: *treemap*.

The 10,535 images per strategy corresponding the patients in our data set were built by employing the function *treemap*, which belongs to the R package named likewise, as mentioned before. The corresponding R Scripts were executed during around 9 hours per strategy in 10 of the 12 cores of a GPU NVIDIA 1080. For this parallelization of the execution was employed the functions of the R package named *snow*.

- Chr-Treemap

Chromosomes are molecules of DNA in which genes are located. In the case of homo sapiens, as a diploid cell organism, all their cells contain all their chromosomes duplicated in terms of gene composition. The number of pairs of chromosomes contained in the nucleus of an homo sapiens cell is 22, along with the sexual pair, whose chromosomes are XX (female) or XY (male). Thus, each gene belongs to one of the 24 different chromosomes a cell have (from 1 to 22, along with X and Y), and those that are located in the same chromosome are more likely to interact with each other because of their physical proximity.

This idea is employed to generate the treemap images of this strategy, that is to say that the hierarchy adopted is the belonging of the genes to one chromosome (hence its name: *Chr-Treemap*). In this way, each gene-expression-profile conforms a treemap image in which the expression of those genes belonging to the same chromosome are placed in the same area of the image. In consequence, each treemap is composed by 24 groups of genes. The fact of reuniting all genes belonging to the same chromosome in the same area is expected to equip gene-expression-profiles with structure due to those genes belonging to the same chromosome are physically closer and are more likely to interact with each other. This interaction could lead to a correlated expression. The internal order of each hierarchy, this is, the order in which genes belonging to one of the 24 chromosomes are placed, is the level of gene expression they have. Due to for each patient this order is different and the position of genes in the image would vary depending on the patient, the average of each gene expression throughout all patients is computed and used for sorting genes inside each group in the treemap images.

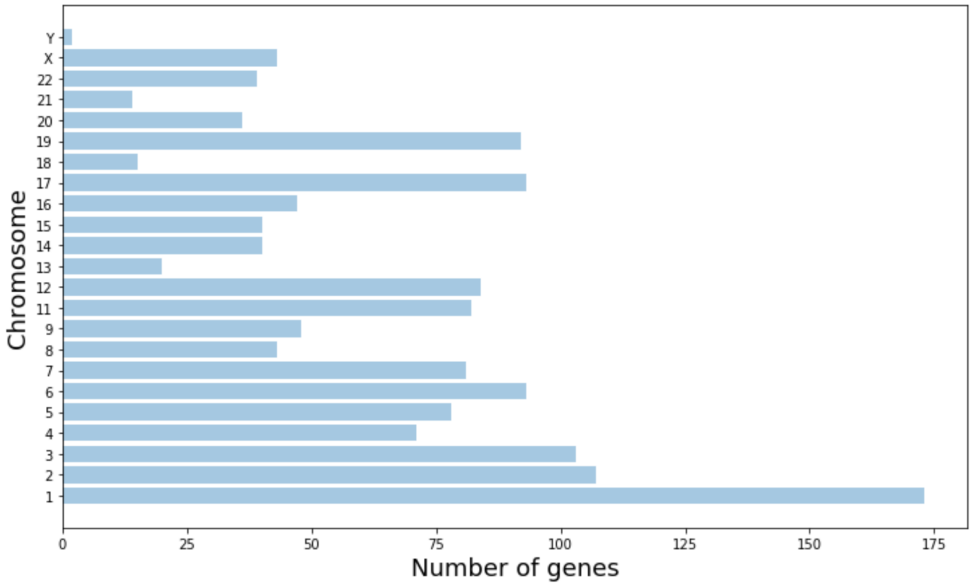


Figure 17: Amount of genes per chromosome in the set of genes to be represented in strategies *Chr-Treemap* and *ChrLocus-Treemap*

In order to create this hierarchy, it is needed to know which chromosome each gene in our data set belongs to. For that purpose, additional information about the genes have been downloaded. Once this information is added to the

gene-expression-profiles data set, the number of features is reduced from 20,000 to 17,111 because not all genes are available in the additional data set. Afterwards, only those genes that are usable in all strategies at the same time were maintained, with the aim of honestly compare them. In this way, the number of genes with which to built the treemap images was reduced to 1,492. The chromosome for which we count with the greatest number of genes is chromosome 1, for which we count with around 750 genes, that is nearly a half of the full amount of genes. The other half of genes are distributed along other chromosomes. Chromosome Y is the less numerous in genes. It was expected to be like that because this gene is only present in males, and its size as a molecule is small, then it also includes less genetic information than the rest of chromosomes. See figure 17 for a graphical representation of the frequency of genes per chromosomes to be represented in each treemap image.

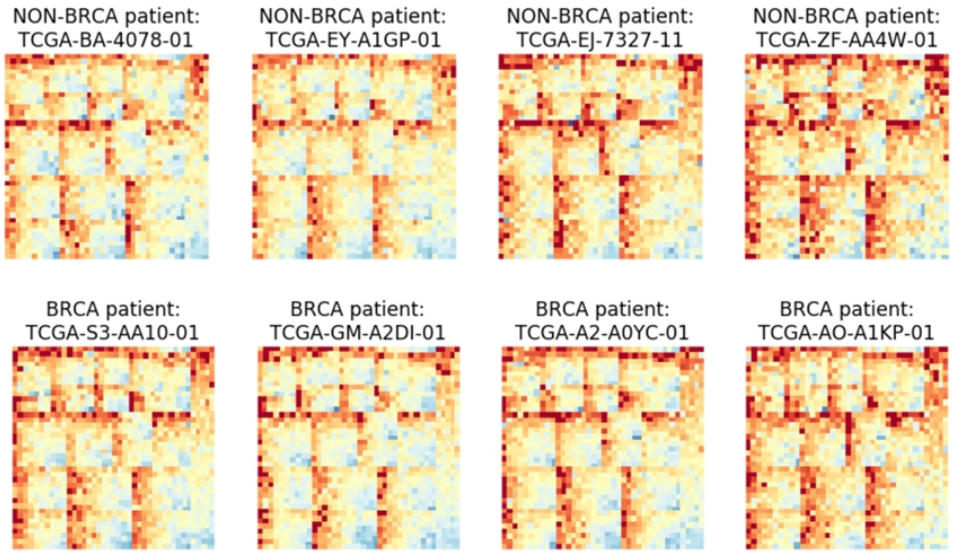


Figure 18: These treemap images have been obtained by applying the strategy named as *Chr-Treemap* with the aim of equipping gene-expression-profiles with structure. Eight treemap images representing gene-expression-profiles are shown: in the upper row there are four treemap images from random non-BRCA patients, and in the lower four treemap images from random BRCA patients.

Some of the images generated by this strategy are shown in figure 18. At first glance, and because of sorting genes according to their average expression, groups in the treemap images are clearly delimited. This seems successful to the human eye, mainly for clearly showing that there are groups conformed in these images, but it could involve disadvantages for ConvNets. For example, the great color contrast between groups or the delimitation this contrast supposes in the images. These facts are implicitly due to the sorting criteria and they may cause the models learn from these patterns to a larger extent than other important facts for the prediction of survival, such as changes in the expression of a gene related to changes on the survival of its patient. For these reasons, it is assumed that this sorting criteria is not the best, and in strategy *ChrLocus-Treemap* other criteria is proposed to solve these inconvenients.

- ChrLocus-Treemap

This strategy goes one step further than the previous: *Chr-Treemap*. As in it, genes are grouped according to the

chromosome they belong to (from 1 to 22, along with X and Y), but in this strategy it is also considered the position each of them have inside their chromosome.

Chromosomal locus are fixed positions in a chromosome in which it is located, for example, a gene. Knowing the exact position in which a gene is located inside its chromosome allows sorting them in a way that those genes physically closer, and in consequence more likely to interact with each other, were closer in the treemap image. Thus, instead of using the average of each gene expression throughout all patients as sorting criteria inside each group, as it is done in the *Chr-Treemap* strategy, the exact location of a gene inside the chromosome (chromosomal locus) is here the sorting criteria to sort all genes belonging to the same chromosome.

In the same data set in which we obtained the chromosome each gene belongs to, it is also specified the start and end locus of a gene and its transcript in the chromosome. Thus, once again it is used this data set, both for creating groups based on chromosomes and for sorting genes inside each group based on the locus coding the start of each gene. As happened in *Chr-Treemap*, once this information is added to the gene-expression-profiles data set, the number of features is reduced from 20,000 to 17,111 because not all genes are available in the additional data set. Afterwards, only those genes that are usable in all strategies at the same time were maintained, with the aim of honestly compare them. In this way, the number of genes with which to built the treemap images was reduced to 1,492.

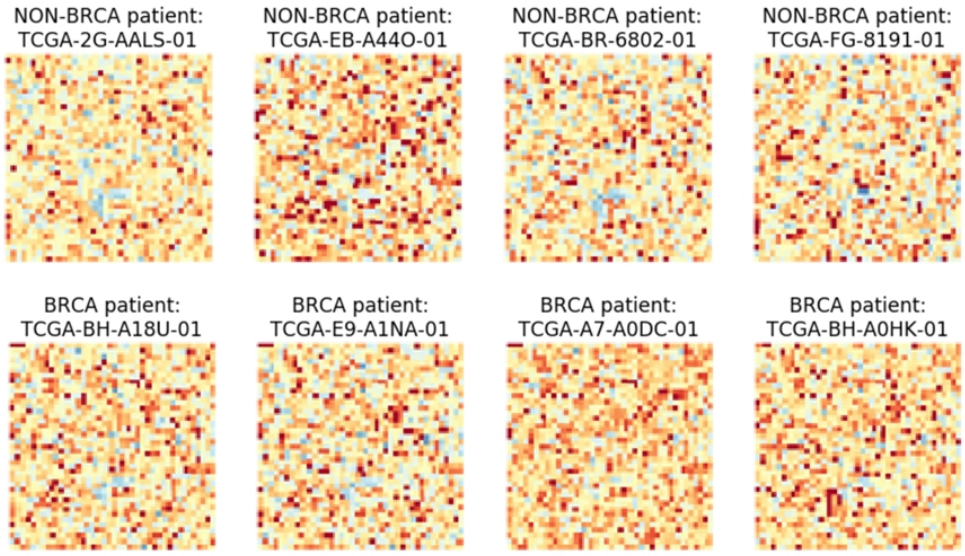


Figure 19: These treemap images have been obtained by applying the strategy named as *Kegg-Treemap* with the aim of equipping gene-expression-profiles with structure. Eight treemap images representing gene-expression-profiles are shown: in the upper row there are four treemap images from random non-BRCA patients, and in the lower four treemap images from random BRCA patients.

Some of the images generated by this strategy are shown in figure 19. Unlike *Chr-Treemap* images (figure 18), in these images cannot be appreciated the areas in the image that correspond with each group (chromosome). Instead, a better and more logical sorting have been applied inside each group, enabling those chromosomes that are physically closer, and whose expression is correlated, to result in a big amount of neighbour pixels with a similar color.

- Kegg-Treemap

Kyoto Encyclopedia of Genes and Genomes (KEGG) is a biological database resource which aims to electronically represent biological systems. Due to this purpose, KEGG assembles many different databases in which data about different biological aspects are stored. One of the organisms KEGG supports data of is *homo sapiens*, and among all its databases it is KEGG BRITE. KEGG BRITE captures functional hierarchies of KEGG objects. These objects are, for example, genes, pathways and organisms inter alia.

In this strategy, and as its name indicates, KEGG has been used for equipping gene-expression-profiles throughout treemap images generation. More precisely, data from KEGG BRITE has been used. It allows us to know the functional annotation of homo sapiens genes, that is to say in which biological functions these genes are involved. In this way, functional annotations acts as hierarchies/groups in the treemap images that were generated.

By the use of the KEGG web API, multiple queries are requested to KEGG BRITE database to obtain the information needed in this strategy. These queries are done by using the function *get* from the Python library *requests*, and the obtained data is saved as *.csv* files. In table 1 are specified the queries that were used when retrieving KEGG BRITE data. However, these data were not enough because we count with genes from Pancancer database and Pancancer database uses Ensembl ids for identifying genes. Then, it was also downloaded a table that relate each Ensembl gene id with its Hugo Gene Name (from https://raw.githubusercontent.com/jvivian/docker_tools/master/gencode_hugo_mapping/attrs.tsv). Ensembl is a biological database which is focused on genomes, and Hugo Gene Names are the human gene names approved by HGNC (Hugo Gene Nomenclature Committee).

Query	**Obtained data**	**Obtained data size**
https://rest.kegg.jp/list/hsa	Gene ids of homo sapiens (hsa) genes in KEGG, related to their Hugo Gene Names	38680 pairs
https://rest.kegg.jp/get/br:br0892	Availables functional annotations in KEGG BRITE database	112 functional annotations
https://rest.kegg.jp/link/hsa/brite	Gene ids of homo sapiens (hsa) genes in KEGG, related to their functional annotation ids in KEGG BRITE	32712 pairs

Table 1

This time, join together gene-expression-profiles data and the corresponding hierarchical information was more complex than it was in *Chr-Treemap* and *ChrLocus-Treemap* strategies. With this purpose, first of all it was necessary to map all those Ensemble gene ids we had to the corresponding Hugo Gene Names. After doing it, it was possible to link the genes we had from Pancancer with their functional annotations. The complete mapping that were carried out in doing this is schematized in figure 20.

Once this information is added to the gene-expression-profiles data set, the number of features is reduced from 20,000 to 7,299 because not all genes are available in the additional data set. Afterwards, only those genes that are usable

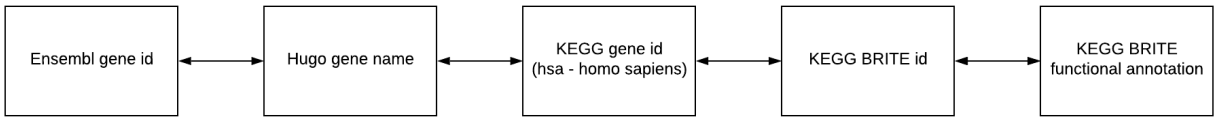


Figure 20

in all strategies at the same time were maintained, with the aim of honestly compare them. In this way, the number of genes with which to built the treemap images was reduced to 1,492.

It is important to mention that one gene can have more than one functional annotation. In consequence, those 1,492 genes appeared in the images duplicated as many times as functional annotations they have, because they belong at the same time to different groups. More precisely, there are a total of 3,619 genes with overlap, of whom only 1,492 are unique genes.

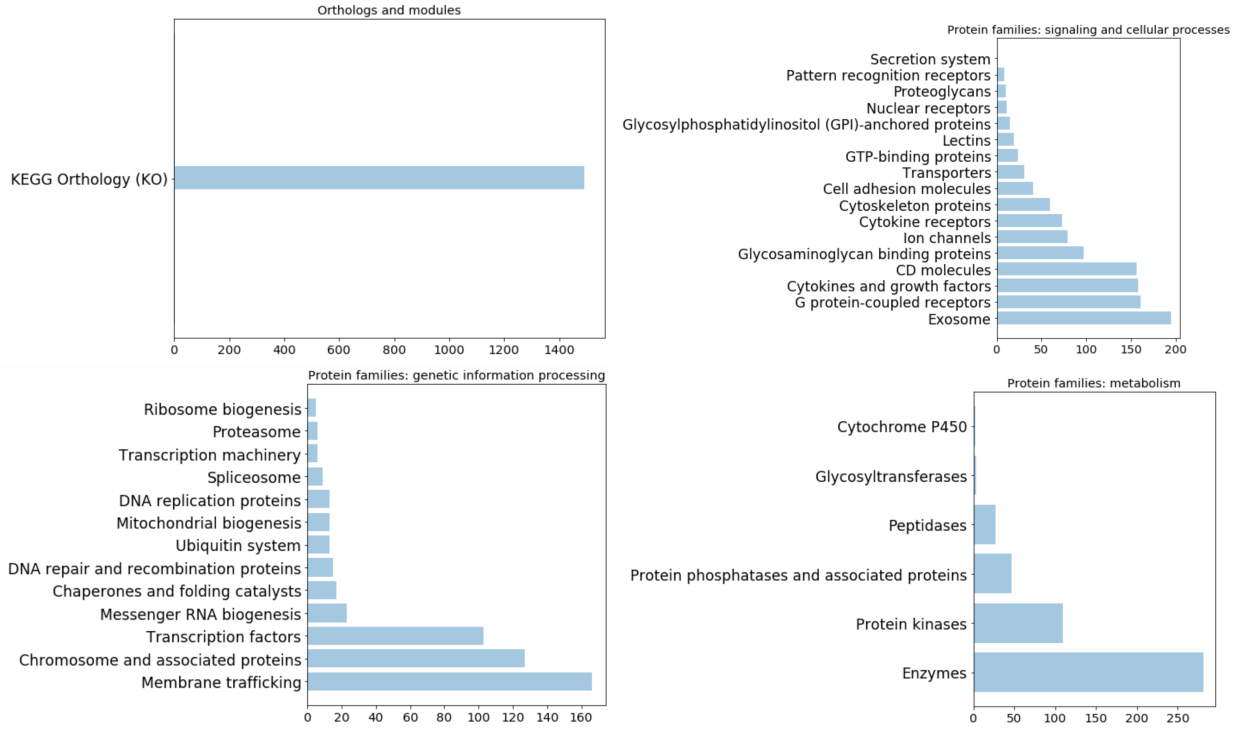


Figure 21: Functional annotation groups from KEGG BRITE (4 groups), and its corresponding functional annotations (total of 37 subgroups). Each chart represent a group of functional annotations and its bars the corresponding subgroups and the amount of genes we count with per each one.

According to the functional annotation hierarchy, genes represented by this strategy are grouped in four main groups, from which arise a total of 37 subgroups. These are those corresponding to the hierarchies of KEGG BRITE and, in consequence, the shown in figure 21. In contrast to the previous detailed strategies (*Chr-Treemap* and *ChrLocus-Treemap*), in this strategy a two level hierarchy is used, as in the example of figure 16b.

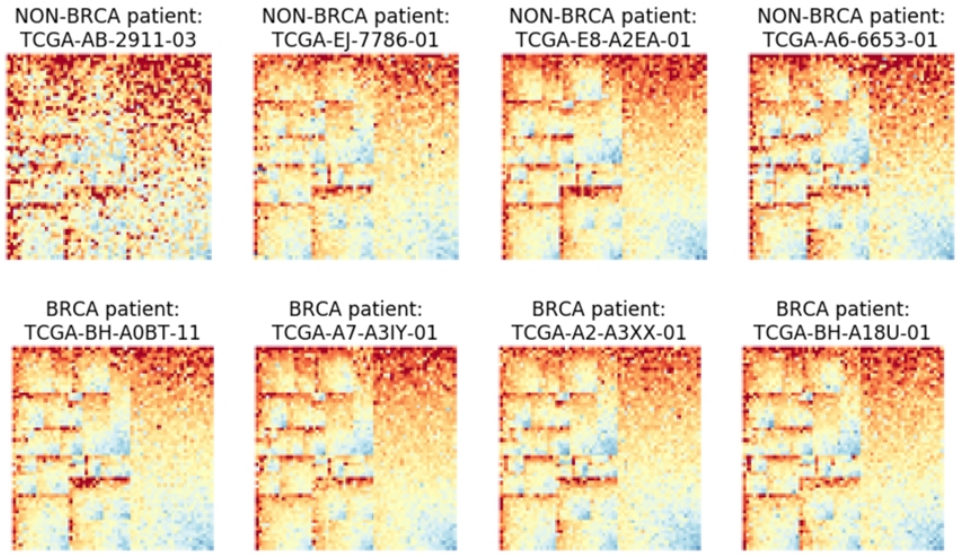


Figure 22: These treemap images have been obtained by applying the strategy named as *Kegg-Treemap* with the aim of equipping gene-expression-profiles with structure. Eight treemap images representing gene-expression-profiles are shown: in the upper row there are four treemap images from random non-BRCA patients, and in the lower four treemap images from random BRCA patients.

Some of the images generated by this strategy are shown in figure 22. As well as in strategy *Chr-Treemap*, at first glance, and because of sorting genes according to their average expression, groups in the treemap images are clearly delimited. This seems successful to the human eye, mainly for clearly showing that there are groups conformed in these images, but it could involve disadvantages for ConvNets. For example, the great color contrast between groups or the delimitation this contrast supposes in the images. These facts are implicitly due to the sorting criteria and they may cause the models learn from these patterns to a larger extent than other important facts for the prediction of survival, such as changes in the expression of a gene related to changes on the survival of its patient.

Control

This section is not about an strategy for equipping gene-expression-profiles with structure, quite the contrary. To test whether these strategies improve the performance of the ConvNets models, the gene expression profiles have been randomly rearranged and subsequently used to train a ConvNet model in the same way as the other models have been trained. It is important to mention that each control model have been trained with five different rearrangements in order to reduce randomization and make it more reliable.

3.2.3 Overall Survival (OS) data cleaning and discretization

As indicated in section 3.1, Overall Survival data (OS and OS-time) is available for a total of 10,532 patients. Among all this patients, 1,247 are patients of BRCA and 9,285 are patients of the other 32 cancer types held by Pancancer. However, a total of 81 patients, 20 from BRCA and 61 from non-BRCA sets, had lost values and had to be dropped

from the data set. In consequence, 10,451 of the 10,532 patients are included in the data set for training and testing the models.

After removing lost values, an important step was carried out in order to enable predicting survival along an interval of time, instead of doing it for a precise moment of time. For that purpose, the follow-up study is delimited to an interval of five years, from 0 to 1,825 days. It was decided like that because, as shown in figure 12, the majority of the OS events takes place during this interval of time (from 0 to 5 years). In this way, OS data is transformed to a set of vectors, one vector per patient and representing their survival along 39 equally spaced intervals for follow-up time. This means that follow-up data (OS) have been discretized through time in 39 equally spaced intervals, each interval gap was around 48 days ($\frac{5\times365}{39}$).

Let be V a $m \times n$ matrix, where m is the number of patients (in this case 10,451) and n is the number of intervals (in this case 39). Being i and j integer numbers ($1 \leq i \leq m$ and $1 \leq j \leq n$), v_i is the survival vector of the ith patient and v_{ij} is a logical value (1 or 0) that indicates if the ith patient has survived to the jth interval of time or not (1 if survived, 0 if not). In order to create this matrix, it is used the function *make_surv_array* from *nnet_survival.py* (https://github.com/MGensheimer/nnet-survival/blob/master/nnet_survival.py) (Gensheimer & Narasimhan, 2019). This function takes three vectors as input: t of size m, f of size m and b of size $n+1$. Vectors f and t are the OS (1 if failure and 0 if censore of the patient) and OS-time (follow-up time) vectors respectively. Vector b indicates the set of days in which each follow-up interval begins. It is assumed that intervals for follow-up time are equally spaced ($b_j - b_{j+1} = K$, where K is a constant representing the gap of time among intervals). Each vector v_i is computed according to four possible situations:

- If the ith patient dies or is censured ($f_i = 1$) in a moment of time $t_i > b_n + K$, then v_i is a vector in which all its elements are 1. That is to say that the event occurs out of the follow-up space of time (in this case, after 5 years).
- If the ith patient dies ($f_i = 1$) in a moment of time $t_i <= b_n + K$ and $b_j <= t_i < b_j + K$, then the elements of the vector v_i are 1 from 0 to j and 0 from $j+1$ to n. That is to say that v_i is 1 until the interval in which the patient dies.
- If the ith patient is censored ($f_i = 0$) in a moment of time $t_i <= b_n + K$ and $b_j + \frac{K}{2} <= t_i < b_j + K$, then the elements of the vector v_i are 1 from 0 to j and 0 from $j+1$ to n. That is to say that the patient has demonstrated to survive the interval of time in which the event happened.
- If the ith patient is censored ($f_i = 0$) in a moment of time $t_i <= b_n + K$ and $t_i < b_j + \frac{K}{2}$, then the elements of the vector v_i are 1 from 0 to $j-1$ and 0 from j to n. That is to say that the patient has **not** demonstrated to survive the interval of time in which the event happened.

These four conditions and its consequences are schematized in figure 23 in order to make it easier to understand the idea implemented on the function *make_surv_array*.

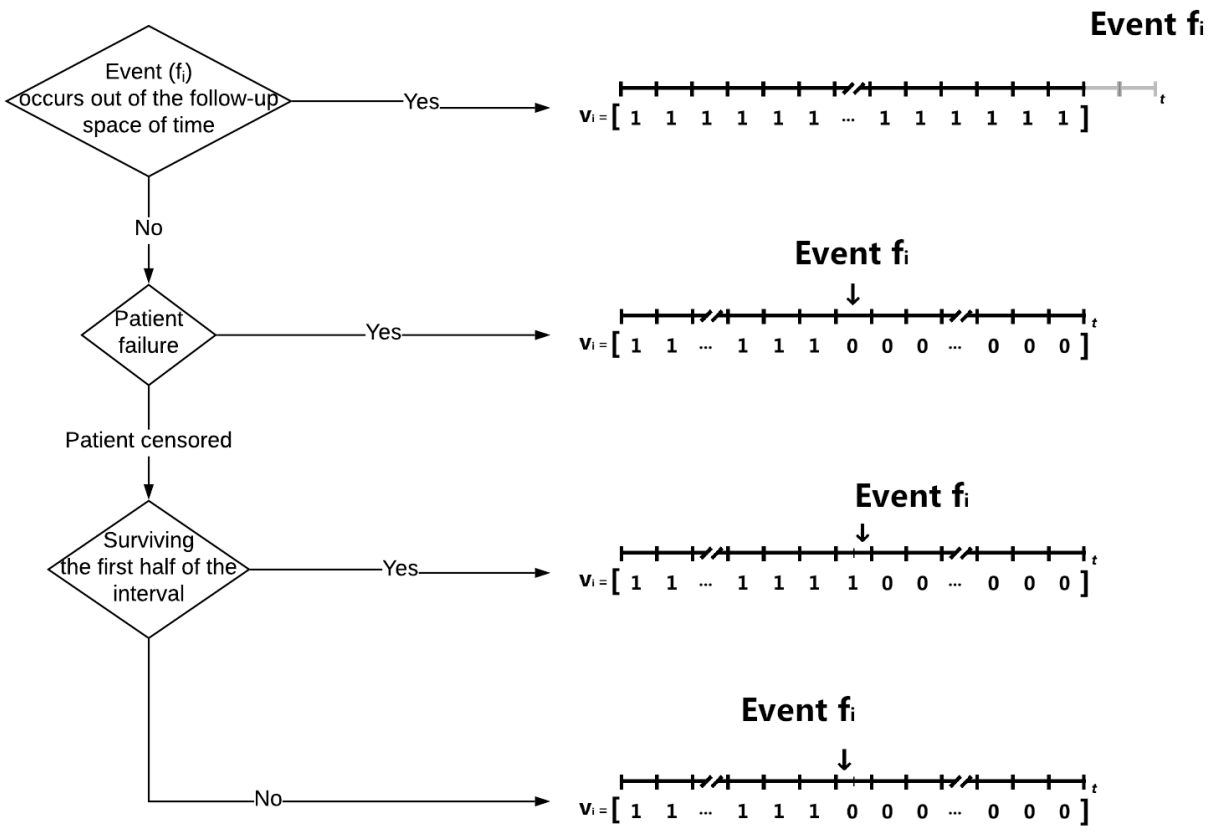


Figure 23: Function *make_surv_array* performance for discretizing follow-up data.

3.3 Model adjustment and training

The main Deep Learning model type implemented in this project is ConvNets, and have been implemented in the Keras Deep Learning framework, by the use of the Python library *Keras*.

As expressed in section 1.4, the main building block of all neural network models is the layer, but they are more complex than that. A Neural Network model is composed by: layers, input and output data, a loss function and an optimizer. At the moment of designing one of this models, it is necessary to decide not only the number of layers, their composition in units and many other hyperparameters, but also what is going to act as the *loss* function, the *optimization* strategy and the *monitoring* of the model while training.

In this section it is going to be specified both the design of the implemented ConvNets architectures and the strategies for the optimization of hyperparameters, model training and model testing.

3.3.1 Loss function: *log-likelihood*

A loss function computes the committed error while training a model. This error is used by the optimizer mechanism in order to decide how correctly update the weights of the model and make it learn by minimizing the error.

Due to the nature of the output data in our models (vector of survival probabilities), it is suitable to use an appropriate loss function.

Once again, it has been used a function from *nnet_survival.py* (https://github.com/MGensheimer/nnet-survival/blob/master/nnet_survival.py) (Gensheimer & Narasimhan, 2019). The function *surv_likelihood* uses the functions from *Keras.backend* to implement a custom loss function. Those survival vectors (v_i) described in section 3.2.3, can be analogously understood from other point of view that is more suitable for the description of the *log-likelihood* function. This time, the survival of a patient is going to be composed by two vectors: sup_f and sup_s. Both are of length n (number of follow-up intervals). The vector sup_s represents those intervals that have been survived by the ith patient ($v_i = sup_s$), while sup_f represents the interval in which the patient fails if he/she did fail. In this way, censoring is controlled.

When a patient fails ($OS = 1, OS.time = t$), the jth component of sup_s and sup_f are defined as:

$$sup_s(j) = \begin{cases} 1 & \text{if } t \geq t_j \\ 0 & \text{if } t < t_j \end{cases}$$

$$sup_f(j) = \begin{cases} 1 & \text{if } t_{j-1} \leq t < t_j \\ 0 & \text{if not} \end{cases}$$

On the other hand, when a patient is censored ($OS = 0, OS.time = t$), the jth component of sup_s and sup_f are defined as:

$$sup_s(j) = \begin{cases} 1 & \text{if } t \geq \frac{t_{j-1}+t_j}{2} \\ 0 & \text{if not} \end{cases}$$

$$sup_j(j) = 0$$

Finally, for each patient, log-likelihood function is defined as:

$$loglikelihood = \sum_{j=1}^{n} ln(1 + sup_s(j)(sup_p(j) - 1) + ln(1 - sup_f(j)sup_p(j)) \tag{4}$$

where sup_p is the vector of survival probabilities predicted by the model. With the aim of maximize the loglikelihood, the loss function is the negative of 4.

3.3.2 Bayesian Optimization of hyperparameters

All ConvNets implemented in this project have many things in common. All of them use the same loss function (section 3.3.1) and the same optimizer: Adam (Tomori et al., 2018). Their output layer is composed by 39 units, which are activated by a sigmoid function (equation 5) unlike all the other layers that are activated by ReLU function (*rectified linear unit*) (equation 6). In consequence, the output of all models is a vector of probabilities, one per follow-up interval.

$$S(x) = \frac{e^x}{e^x + 1} \tag{5}$$

$$ReLU(x) = x^+ = \max(0, x) \tag{6}$$

They have also in common the typical structure of a ConvNet: convolutional layers, flatten layer and dense layers. However they count with many hyperparameters that vary depending on the model. These hyperparameters are chosen with the aim of obtaining the more suitable model to optimize an objective function.

The Python library *hyperopt* was used to optimize the following hyperparameters:

- Number of convolutional layers (two or three), and for each one of these layers, its number of units and its kernel and filter size.
- Number of dense layers (two or three), and for each one of these layers its number of units.
- How much drop-out to be applied on each layer.
- The learning rate, this is the magnitude of change when updating the weights.
- The size of each batch, this is the number of observations to be evaluated at the same time during training.

This library (*hyperopt*) provides many functions to perform what is known as Bayesian Optimization. The Bayesian Optimization (also known as Sequential model-based optimization, SMBO) is a strategy for optimizing the result of an objective function that depends on a Machine Learning model without the use of derivation. It is needed the prior definition of an space of hyperparameters to be tested. Unlike many others strategies for hyperparameters tuning, Bayesian Optimization do not search among all possible combinations in the space of hyperparameters, but uses an algorithm in order to carry out a guided search. The package *hyperopt* provides two search algorithms: random search and Tree-of-Parzen-Estimators (TPE) algorithm (J. Bergstra & Kégl, 2011).

In our project, the objective function is the Concordance Index (C-Index), whose meaning is detailed in section 3.4.

This objective function measures the goodness of a model which implicitly depends on their hyperparameters. By using Bayesian Optimization, it is chosen a set of hyperparameters, belonging to a previously defined space of hyperparameters, that in this case maximize the C-Index function. The search algorithm that have been used is TPE.

3.3.3 Fine-tuning

As mentioned in section 3.1, the number of observations to train the models is: 1,212 for BRCA patients and 9,323 for non-BRCA patients. In order to predict the survival of those patients suffering from breast cancer, there is therefore no data set with adequate dimensions to train a ConvNet model and obtain good results. ConvNet models, as DL ones, need of a higher proportion of observations than features, otherwise it is about *the Curse of Dimensionality*.

In this project it has been implemented and trained three models per each one of the strategies for equipping genes-expression-profiles with structure. This is a total of 18 models. These models differ in the type of cancer that has been diagnosed to the patients whose data are used to train the model. These are: *non-BRCA models*, *BRCA models* and *fine-tuning-BRCA models*. In the last one, *fine-tuning-BRCA models*, a *transfer learning* strategy have been applied in order to solve the problem about the dimensionality of BRCA data set. The others, BRCA and non-BRCA, just are models trained with BRCA and non-BRCA data, respectively.

Fine-tuning is a transfer learning strategy in which some layers of an already trained model are re-trained with new and related data in order to solve a different problem. In this project, *fine-tuning-BRCA models* consist in re-training the set of dense layers of the optimum non-BRCA model with BRCA data. These layers are 6 or 9, depending on the pre-trained model architecture. In this way, the obtained model have been trained with around 10k observations for predicting BRCA patients survival, instead of the 1k observations used in *BRCA models*.

3.3.4 Cox Regression

Additionally to the ConvNets models, three Cox Regression models have been implemented in order to compare its performance with that of the ConvNets models. These Cox models are one per data set: BRCA data, non-BRCA data and both BRCA and non-BRCA data.

To train these Cox Regression models, the functions *CoxPHFitter()* and *CoxPHFitter().fit()* from the Python library *lifelines* were employed. With the aim of comparing these models with those of ConvNets, it was tried to train them with the same set of genes (1,492), but it was not possible. Cox models do not deal with collinear data. In figure 24 a heat map shows how correlated is each gene with the others. As it can be seen at first sight, these are highly correlated.

The number of genes was reduced to 149 by removing collinear pairs. There were only maintained those genes with a correlation lower than 0.25. This feature selection was performed in R and by the use of the R package *Caret* and its function *findCorrelation*.

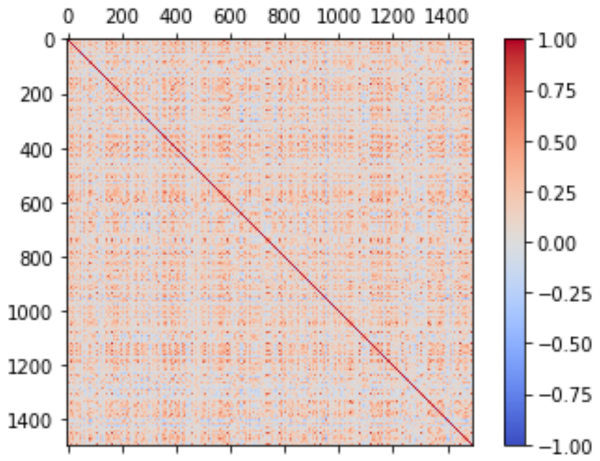


Figure 24: Correlation between the 1,492 genes used for training all models in this project.

3.4 Models Evaluation

In this project it has been implemented and trained three models per each one of the strategies for equipping genes-expression-profiles with structure: *non-BRCA models*, *BRCA models* and *fine-tuning-BRCA models*. This is a total of 18 models. Comparing the performance of these models is equivalent to compare the performance of the strategies for equipping gene-expression-profiles with structure and the *transfer learning* strategy.

There are many measures for analyzing predictive models performance, but due to the format of the output the generated ConvNets models had, the measure that seems more appropriate to be used in this project is the Concordance Index (C-index). C-index is the frequency of concordant pairs among all pairs of patients in the test data set. A concordant pair of patients is that one in which the risk of deceasing of the patient that deceases in a later moment is higher than the one that deceases in an earlier. C-index values goes from 0 to 1. Those models whose C-index is 0.5 or less are no good models.

In this project, it has been computed the C-Index value of the predicted survival vectors of probabilities in a range of follow-up time from 0 to 2 years, instead of the predicted complete range (from 0 to 5 years). The three last years have not been taken into account because of the small amount of observations whose OS event has occurred after 2 years of follow-up (as shown in figure 12). The smaller the number of observations, the lower the reliability of the predicted data. Furthermore, the two first years after the diagnosis of cancer is more relevant for clinicians.

4 Results and discussion

4.1 ConvNets Models

As a result of this final-degree project, 18 ConvNet models have been produced for cancer survival prediction. One per strategy for equipping gene-expression-profiles with structure and the type of cancer for which to predict.

Data have been splitted into the same subsets for training (64%), validation (16%) and testing (20%), in order to honestly compare the models. In table 2 it is specified the number of observations that compose these subsets. It is important to bear in mind that for the BRCA data set there is a much smaller number of observations and yet the number of variables is equally large.

	non-BRCA	**BRCA**	**BRCA fine-tuning**
train	5903	764	6667
validation	1845	240	2085
test	1476	192	1668
total	9224	1196	10420

Table 2: Number of observations for training, validating and testing each model.

Regarding their hyperparameters, these have been chosen in order to obtain an optimum model by the use of the strategy known as Bayesian Optimization. The selected hyperparameters per each one of the 15 models (irrespective of *control* models) are on table 4. The controls models have not been taken into account in this table because these are models in which the ordering of the variables has been randomized and have been optimized five times each (in order to reduce randomization of their results), so that five models have been obtained whose hyperparameters are different and are not of interest.

After optimizing each model hyperparameters, the resulting models have been tested with the data in their test subset. In table 3 and in figure 25, the C-index value obtained for each model is shown.

If something is clear at the view of the results is that the fact of having a much smaller number of observations to train *BRCA models* is affecting the performance of these models. This was expected and it is the result of what have been mentioned many times in this project: *the Curse of Dimensionality*. With the aim of solving it, a *fine-tuning* strategy was carried out and two of the six models in which *fine-tuning* is applied have better performance than those in which only BRCA data is used. These are *ChrPth-1d* and *ChrLocus-TreeMap*. Additionally, two of the remaining four *fine-tuning models* have mostly the same performance than the models in which the low-dimensional data set is used. It is not clear why this is happening, but maybe it depends on the randomness in which observations have been splitted into testing and training, or even on the goodness of the pre-trained models.

When comparing, in general, the performance of the models corresponding to each strategies for equipping gene-expression-profiles with structure, it does not seem that these strategies are giving much better results than a random

ordering of the genes (control).

The C-index value obtained for each control model is, as specified in section 3.2.2, the average of five C-index values. This was done in order to reduce the randomization of the resulting models, due to these were trained with gene-expression-profiles that were randomly rearranged. In consequence, it is worth to mention that the results in table 3 and figure 25 about the control strategy have an standard deviation of around 0.03.

	non-BRCA	**BRCA**	**BRCA fine-tuning**
Control	0.74229	0.69347	0.64573
ChrPth-1D	0.76154	0.63397	0.72129
ChrPth-2D	0.76403	0.64862	0.64444
Chr-TreeMap	0.73836	0.64885	0.65201
ChrLocus-TreeMap	0.75574	0.64578	0.71125
Kegg-TreeMap	0.74996	0.68527	0.63648

Table 3: C-Index values obtained when testing each ConvNets model.

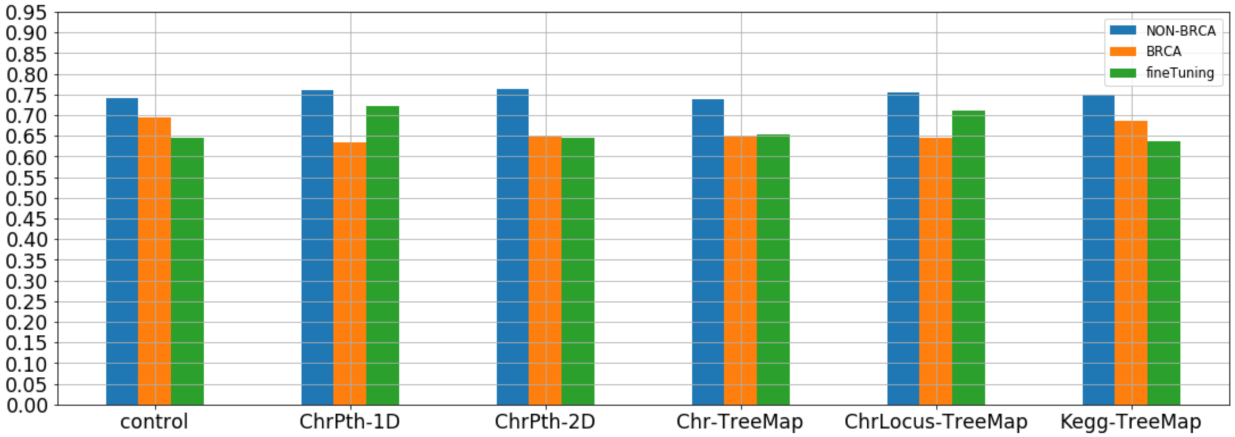


Figure 25: C-Index values obtained when testing each ConvNets model.

It is very important to highlight that these results have not been obtained by any validation strategy, as *Cross-Validation* or *Nested-Cross-Validation*, but have been tested in the same and unique partitions of data whose length have been specified in table 2.

		ChrPth-1D			ChrPth-2D			Chr-TreeMap			ChrLocus-TreeMap			Kegg-TreeMap		
		BRCA	non-BRCA	fine-tuning	BRCA	non-BRCA	fine-tuning	BRCA	non-BRCA	fine-tuning	BRCA	non-BRCA	fine-tuning	BRCA	non-BRCA	fine-tuning
Learning Rate		0.002	0.002	0.005	0.0001	0.002	0.001	0.0001	0.003	0.0001	0.003	0.01	0.008	0.0005	0.001	0.0001
Batch Size		64	16	16	64	16	64	32	16	128	64	32	32	16	16	64
Conv. layers		3	2		2	2		2	3		3	3		2	2	
Conv. layer 1	units	16	32		32	32		16	128		32	128		32	128	
	kernel	3	10		5	5		3	5		3	3		3	3	
	pool	5	2		2	2		2	2		2	2		2	2	
	drop-out	0.75	0.75		0.5	0.25		0.75	0.75		0.25	0.25		0.5	0.25	
Conv. layer 2	units	32	64		16	64		128	32		16	128		128	128	
	kernel	5	10		3	5		3	3		5	5		5	3	
	pool	5	2		2	2		2	2		3	2		2	2	
	drop-out	0.75	0.5		0.25	0.25		0.75	0.25		0.25	0.5		0.25	0.75	
Conv. layer 3	units	32	-		-	-		-	64		64	32		-	-	
	kernel	5	-		-	-		-	5		3	3		-	-	
	pool	3	-		-	-		-	2		2	2		-	-	
	drop-out	0.25	-		-	-		-	0.5		0.5	0.5		-	-	
Dense layers		2	3	3	2	2	2	2	3	2	3	2	2	2	2	2
Dense layer1	units	128	16	16	32	64	32	128	64	64	16	64	64	16	64	16
	drop-out	0.25	0.25	0.25	0.25	0.25	0.25	0.5	0.5	0.5	0.25	0.5	0.5	0.25	0.5	0.25
Dense layer2	units	32	128	128	32	32	128	16	128	64	128	16	128	128	64	16
	drop-out	0.75	0.25	0.5	0.25	0.25	0.75	0.25	0.5	0.75	0.75	0.25	0.5	0.25	0.25	0.75
Dense layer3	units	-	128	-	-	-	-	-	64	-	16	-	-	-	-	-
	drop-out	-	0.25	-	-	-	-	-	0.25	-	0.5	-	-	-	-	-

Table 4: Optimum hyperparameters obtained by applying Bayesian Optimization

4.2 Cox Regression

As mentioned in section 3.3.4, Cox Regression models have been implemented and training from the same gene-expression-profiles than in ConvNets models, but the number of genes composing these observations have been reduced to 149 in order to avoid collinearity.

The obtained Cox Regression models have been tested and their corresponding C-Index values are in the table 5.

These models, in spite of having good results, are not entirely reliable since they do not meet the proportional hazard assumptions and therefore can not be fully affirmed that they are better than the ConvNets models obtained in these final-degree-project.

	non-BRCA	**BRCA**	**all**
C-Index	0.73	0.82	0.72

Table 5: C-Index values obtained when testing each Cox Regression model.

5 Conclusions

In this project five strategies have been developed in order to provide gene expression profiles with structure, and therefore, the models of Convolutional Neural Networks that are trained with this type of data have a better performance. After evaluating them, it can be concluded that, a priori, it does not seem that any of the strategies are giving rise to better results than those obtained without any additional structural information. In spite of everything, this conclusion is not completely definitive if one takes into account that no validation strategy has been used, such as *Cross Validation*, which reduces the chances of obtaining results that are too optimistic or pessimistic due to random reasons when separating data sets.

On the other hand, it have been also tried to solve the problem known as the *curse of dimensionality* by using the strategy of *transfer learning* known as *fine-tuning*. In this case, some positive results have been obtained, but not all. Therefore, it could be concluded that this is an appropriate strategy to try to solve the problem and it would be a good idea to continue studying this way.

Regarding the comparison of traditional models, models of Cox Regression have been estimated for the resolution of the same problem. Despite having higher values of C-index, it can not be concluded that they are better since the models obtained do not comply with the the proportional hazard assumptions.

5.1 Future work

In order to obtain truly reliable results, it will be tried to evaluate the ConvNets models obtained through the use of validation strategies, as already mentioned. The *nested cross validation* would be a good way to do it. This strategy consists of dividing the data set into *folds* and dedicating some of them to train and others to test, in an alternating manner, resulting in all the data having acted as training data and test data at some time. In this way, the final result will be the average of all the values obtained according to a performance metric (C-Index in this case), and will represent the performance of the strategy used, instead of the performance of a particular model.

Regarding the comparison of traditional models, models of Cox Regression have been estimated for the resolution of the same problem. Despite having higher values of C-index, it can not be concluded that they are better since the models obtained do not comply with the proportional hazard assumptions.

Because none of the strategies for equipping gene-expression-profiles with structure has been successful, new strategies based on other types of biological information will be proposed and developed, such as information about the regulation of genetic expression carried out by micro-RNAs.

6 Conclusiones

En este proyecto se han desarrollado cinco estrategias con el fin de dotar a los perfiles de expresión génica con estructura, y por tanto, que los modelos de Redes Neuronales Convolucionales (RNC) que se entrenen con este tipo de datos tengan un mejor rendimiento. Tras la evaluación de los mismos se puede concluir que, a priori, no parece que ninguna de las estrategias esté dando lugar a mejores resultados que los obtenidos sin ninguna información estructural adicional. Pese a todo, esta conclusión no es del todo definitiva si se tiene en cuenta que no se ha utilizado ninguna estrategia de validación, como puede ser la *Validación Cruzada*, que reduzca las posibilidades de estar obteniendo resultados demasiado optimistas o pesimistas por razones de azar a la hora de separar los conjuntos de datos.

Por otro lado, también se ha tratado de resolver el problema conocido como *la maldición de la dimensionalidad* mediante el uso de la estrategia de *aprendizaje por transferencia* conocida en inglés como *fine-tuning*. En este caso, sí se han obtenido algunos resultados positivos, pero no todos. Por lo tanto, se podría concluir que se trata de una estrategia adecuada para tratar de resolver el problema y sería una buena idea seguir estudiando esta idea.

En cuanto a la comparación de los modelos tradicionales, se han estimado modelos de Regresión de Cox para la resolución del mismo problema. Pese a contar con unos valores mayores de C-index, no se puede concluir que sean mejores puesto que los modelos obtenidos no cumplen con el criterio de la proporcionalidad de los riesgos.

6.1 Trabajos futuros

Con el fin de poder obtener unos resultados realmente fiables, se tratará de evaluar los modelos RNC obtenidos mediante el uso de estrategias de validación, tal y como ya se ha comentado. La *validación cruzada anidada* sería una buena manera de hacerlo. Esta estrategia consiste en dividir el conjunto de datos en cajas y dedicar algunas de ellas a entrenamiento y otras a test, de manera alternada, dando lugar a que todas los los datos hayan actuado con datos de entrenamiento y datos de test en alguna ocasión. De esta forma el resultado final será la media de todos los valores obtenidos según una metrica de rendimiento (C-Index en este caso), y representará el rendimiento de la estrategia utilizada, en lugar del rendimiento de un modelo en concreto.

Por otro lado, también se debe resolver el problema de que los modelos de Regresión de Cox obtenidos no cumplan con el criterio de proporcionalidad de los riesgos, con el objetivo de obtener una comparación más fiable. Además, la implementación de otros modelos de Machine Learning tradicionales, como son las Redes Neuronales Simples o las Máquinas de Soporte Vectorial, será realizada para comparar los modelos de Deep Learning con los tradicionales.

Debido a que ninguna de las estrategias para equipar a los perfiles de expresión génica con estructura ha tenido éxito, se propondrán y desarrollarán estrategias nuevas basadas en otros tipos de información biológica, como puede ser información sobre la regulación de la expresión génica llevada a cabo por micro-RNAs.

References

Behjati, S., & Tarpey, P. S. (2013). What is next generation sequencing. *Arch Dis Child Educ Pract Ed*, *98*(6), 236–238. doi: 10.1136/archdischild-2013-304340

Bray, F., Ferlay, J., Soerjomataram, I., Siegel, R. L., Torre, L. A., & Jemal, A. (2018). Global cancer statistics 2018: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA: a cancer journal for clinicians*, *68*(6), 394–424. doi: 10.3322/caac.21492

Chaudhary, K., Poirion, O. B., Lu, L., & Garmire, L. X. (2018, Mar). Deep Learning-Based Multi-Omics Integration Robustly Predicts Survival in Liver Cancer. *Clin. Cancer Res.*, *24*(6), 1248–1259.

Chollet, F. (2018). *Deep Learning with Python* (Manning Publications Company, Ed.).

Cox, D. R. (1972). Regression Models and Life-Tables. , *34*(2), 187–202. doi: 10.1111/j.2517-6161.1972.tb00899.x

Gensheimer, M. F., & Narasimhan, B. (2019). A scalable discrete-time survival model for neural networks. *PeerJ*, *7*, e6257.

Han, I., Kim, J. H., Park, H., Kim, H. S., & Seo, S. W. (2018, Sep). Deep learning approach for survival prediction for patients with synovial sarcoma. *Tumour Biol.*, *40*(9), 1010428318799264.

IARC. (2019). *International agency for research on cancer. Global Cancer Observatory: Cancer today.* Retrieved 2019-05-28, from http://gco.iarc.fr/today/home

INE. (2019). *Instituto nacional de estadística (spain). INE: Deaths by cause 2017.* Retrieved 2019-05-29, from http://www.ine.es/jaxi/Datos.htm?path=/t15/p417/a2017/l0/&file=01001.px

J. Bergstra, Y. B., R. Bardenet, & Kégl, B. (2011, Jul). Algorithms for Hyper-parameter Optimization. *Proc. Neural Information Processing Systems 24*, 2546–2554.

Jurman, G., Maggio, V., Fioravanti, D., Giarratano, Y., Landi, I., Francescatto, M., ... Furlanello, C. (2017, Oct). Convolutional neural networks for structured omics: OmicsCNN and the OmicsConv layer. *arXiv e-prints*, arXiv:1710.05918.

Kaplan, E. L., & Meier, P. (1958). Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association*, *53*(282), 457-481. doi: 10.1080/01621459.1958.10501452

Lyu, J., & Ling, S. H. (2018, 07). Using Multi-level Convolutional Neural Network for Classification of Lung Nodules on CT images. *Conf Proc IEEE Eng Med Biol Soc*, *2018*, 686–689.

Ma, S., & Zhang, Z. (2018). OmicsMapNet: Transforming Omics Data to Take Advantage of Deep Convolutional Neural Network for Discovery. *arXiv*, arXiv:1804.05283.

Matsuo, K., Purushotham, S., Jiang, B., Mandelbaum, R. S., Takiuchi, T., Liu, Y., & Roman, L. D. (2019, Apr). Survival outcome prediction in cervical cancer: Cox models vs deep-learning model. *Am. J. Obstet. Gynecol.*, *220*(4), 1–381.

NCBI. (2019). *Pubmed - us national library of medicine national institutes of health.* Retrieved 2019-06-25, from https://www.ncbi.nlm.nih.gov/pubmed

Singleterry, J. (2017, April). *The costs of cancer; adressing patient costs* (Tech. Rep.). American Cancer Society Cancer Action Network.

Tomori, S., Kadoya, N., Takayama, Y., Kajikawa, T., Shima, K., Narazaki, K., & Jingu, K. (2018, Jul). A deep learning-based prediction model for gamma evaluation in patient-specific quality assurance. *Med Phys*.

van Lanschot, M., Bosch, L., de Wit, M., Carvalho, B., & Meijer, G. (2017). Early detection: the impact of genomics. *Virchows Arch*, *471*(2), 165-173. doi: 10.1007/s00428-017-2159-2