Tumor Classification Using Interpretable Neural Networks

André Ferreira
andre.c.n.ferreira@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

February 2020

Abstract

Cancer constitutes a complex set of diseases, which originate from genetic mutations. A human genome has around 3 billion base pairs, which can be seen as units of genetic information. Knowing this, it seems to be nearly impossible for a person to single-handedly understand all the information residing in our cells, which consequently makes cancer appear as a problem out of reach for human knowledge. Fortunately, advances in machine learning have brought us other superhuman-like skills, such as autonomous vehicles, automatic translation, art generation, and professional level gaming AI. These advances have not yet fully reached the health domain, in big part due to the associated models being hard to interpret. In this paper, the task of classifying tumor types is addressed, with multiple machine learning models of varying complexity, and with an applied interpretability technique that allows for a better explanation of the underlying logic. Surprisingly, in this experiment, all tested models performed at a similarly high level, reaching over 0.98 AUC, with only a small subset of RNA features determining the primary tumor type. We hope that these results help bring a simpler view on the main characteristics of cancers and that it opens up further research into more detailed analysis, aided by machine learning models that are both high performing and explainable.

Keywords: Tumor, Genomics, Classification, Machine Learning, Deep Learning, XAI, Interpretability

1. Introduction

Over the years of scientific research and technological development, humankind has been stretching life expectancy and improving quality of life, even until later stages. However, there have been some roadblocks that go on setting a cap on this path of improvement, usually in the form of fatal diseases whose cure has yet to be found. As the second leading cause of death worldwide [3], cancer is without a doubt one of the major issues that are preventing us from reaching longer and better lives. What is making its cure so difficult to achieve is the vast complexity of the problem. There are multiple types of tumors, with even some debate around how many there are in total and how to define them. It can originate from basically any part of the body and present different behaviors, which also bring different resistance to treatments. And to complicate it even further, we don't yet know for sure what causes tumors to appear, despite some evidence of influence from smoking and eating habits, making prevention and treatment the confusing, ineffective mess that it currently is. What we do know for sure is that tumors represent genetic mutations, an uncontrolled multiplication of cells with altered genetic information. Furthermore, with the increasing collection and availability of multiple sources of clinical and behavioral data, we could look for patterns so as to get a better understanding of these diseases. But, even if we ignore these other data sources and focus on genomics, the human body has millions of cells, each one with 23 chromosomes, each one with thousands of genes, constituting a human genome of over 3 billion base pairs, which are units of genetic information in our DNA. Considering all of this, it is unrealistic to think that a human doctor can analyze every detail on his or her own. This is where machine learning can be helpful. A computer can perform computations much faster than a human and is not restricted by biological factors such as tiredness, hunger or stress. When paired with advanced machine learning models, particularly artificial neural networks, we can get significant advancements in fields so varied as computer vision (including autonomous vehicles [20]), natural language processing (including conversational bots [37]), reinforcement learning (including beating world champions of strategic videogames [26] and demonstrating collaborative intelligence [8]), art generation (including melody

creation [12], style transfer [15] and painting generation [27]) and even in some health research (including diagnosis [28] and prognosis [28, 6]). Nevertheless, the health sector has had a relatively slow adoption of these techniques, out of lack of trust [29] [1]. In a medicine scenario, where actions can have consequences as serious as detecting or not a disease, using or not the right treatment, and ultimately as letting the patient live or die, mistakes are very costly. So, health professionals must be very careful with their decisions, using deeply studied theory and empirical evidence to validate the option, not just blindly believe in what an algorithm outputs in a computer. As such, in these sensible topics, either machine learning is not applied at all or simpler, lower performance models are implemented, as they are easier to understand. To add to this problem, while deep learning has had success in sequential and unstructured data, it has not been as good in tabular data, such as genomics. In these datasets, decision tree based models have had an advantage, as demonstrated in Kaggle challenges [32], due to their current state-of-the-art performance and better interpretability with their simpler branching-like logic.

Realizing the importance of the topic, the lack of trust in machine learning and the unrealized potential of neural networks in tabular data, we experiment in tumor primary type classification, comparing the application of state-of-the-art decision tree based models [9] and artificial neural networks, all while interpreting the models without performance losses.

2. Related Work

Despite the still relatively unexplored application of deep learning to genomics, recent research review papers [13, 39] show the rise of this combination, arguing the potential for deep neural networks to achieve state-of-the-art results and their flexibility, being suited for tasks such as tumor genomes, RNA analysis, DNA accessibility, protein–protein interaction network analysis, clustering and even synthetic data generation, among others. Both papers also mention the need for model interpretation, no only to be able to understand and trust the reasoning behind it but also to extract new insights through feature importance. This way, researchers can see how different inputs influence the outputs and, knowing the salient features in each case, investigate further. As for how to interprate the models, both papers suggest perturbation-based methods, where each feature's influence is determined by how the output changes on variations of the feature's value, and gradient-based methods, where each feature's influence is determined by the derivative of the output on the respective feature. It's important to notice that, while perturbation-based methods are model-agnostic, working similarly for any model type, the gradient-based methods are most relevant for artificial neural networks, considering the speed increase that we get by doing gradient backpropagation instead of running multiple samples of perturbations.

In genomics, several papers use clustering, which can help discover for instance groups of interesting genes and what is common among them. When using deep learning, the most sought neural network architecture is the autoencoder [10], and variations of it, as it tends to be the most robust at finding patterns even through the noise, thanks to the model's learning by encoding and decoding data repeatedly. It's also worth noting Katherine A. Hoadley et al paper [18], which works on similar data, compared to this paper, from the TCGA dataset and clusters around 10,000 tumor samples on multiple platforms of data, namely aneuploidy, mRNA, miRNA, DNA methylation, and RPPA. Although they use clustering techniques such as iCluster [31] and COCA [19] instead of deep learning, it is relevant to notice the derived conclusions, particularly that the obtained clusters were determined mainly by cell-of-origin, the tissue and body part from which the tumor started developing, with other information such as copy-number aberrations also having relevant, yet less significant, contribution to the clustering. This makes a case for the current paper's tumor classification, as the model should learn more about what are the other features relevant for each anatomically defined tumor type and how important they actually are, by running a feature importance analysis on the trained models.

As mentioned, through deep learning's flexibility, there are papers addressing many prediction tasks [22, 4, 11, 36], even implementing models such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). However, contrary to a MultiLayer Perceptron (MLP), a simpler neural network configuration that we use, CNN's and RNN's each depend on specific assumptions about the data, in terms of spacial and/or sequential dependencies, which aren't true for tabular data, as is the case of this paper's genomics inputs. For a better comparison, it is more interesting to compare with other cancer type classification studies such as that of Yuchen Yuan et al [38], which gets near 0.7 test accuracy on 12 tumor types classification, with the help of interesting gene data processing ideas. However, the data used is not as heterogeneous as the TCGA data used in this paper, having only somatic data and less than half the number of tumor types to classify. Furthermore, the authors do not benchmark their DeepGene model against other deep learning alternatives nor gradient boosting models such as XGBoost, which could possibly

outperform the authors' custom model.

In terms of interpretability techniques, alongside some generic approaches that require changes to the neural networks, such as attention weights [7, 35], there are biomedical papers that even mold their models so that the whole architecture resembles real, predetermined concepts [24, 14]. While these ideas can make it more straightforward to interpret the model, the forced strict restriction of the model's shape can amper the performance and is not easily scalable to other tasks, as a well-known theoretical graph must be defined beforehand. As such, we focus on model-agnostic approaches, meaning that they do not interfere with model design nor its performance. Having in mind that there are simpler alternatives such as occlusion windows and mask filters [34], we implement the more established and theoretically sound SHAP values [23], which are described in more detail in the next background section.

3. Background

3.1. XGBoost

Decision trees have long been a part of data science toolboxes, considering their flexibility in addressing classification and regression problems, as well as their simple and intuitive logic. Similarly as to how people plan an action in detail, the decision tree works by analysing one feature at a time, branching into different possible scenarios, with more features to enquiry, based on the received input value.

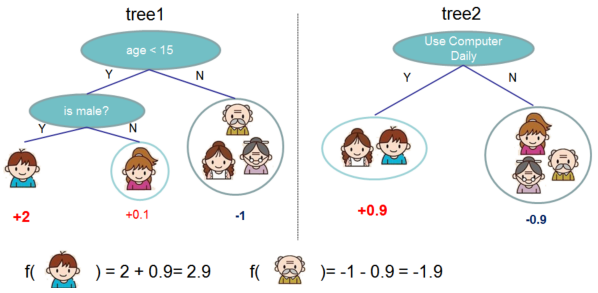


Figure 1: Example of two decision trees in a task of predicting if someone plays videogames, with bagging scores for two samples bellow.

Over the years, through research that aimed to improve these intuitive machine learning models, new variants of decision trees have been developed. Initially, ensembles of decision trees were used, in what was referred to as bagging. Then, with random forests, trees are also added to an ensemble, but using just a random subset of features. The idea of boosting gave a different approach to ensembling, as it trains small decision trees, called weak learners, building them iteratively based on the previous trees' performance, focusing on their misclassified samples. Moving into gradient boosting, the idea of building a strong model out of several weak learners remain, although now with the use of an overall loss function, in an iterative functional gradient descent optimization. In other words, now each model being added (i.e. each function) is chosen based in a way that most minimizes the loss, advancing in its negative gradient.

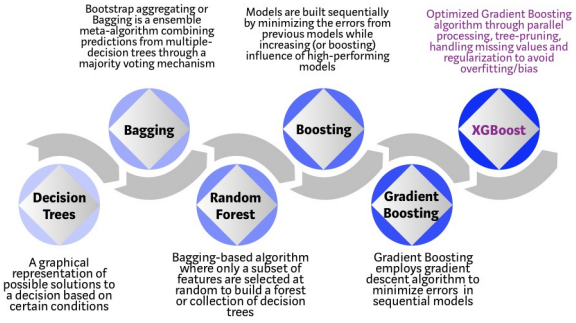


Figure 2: Evolution of decision tree based models.

XGBoost [9], which stands for eXtreme Gradient Boosting, essentially constitutes an highly optimized version of gradient boosting, with parallel computing and overall efficient management of hardware resources. It is actually so efficient, that it has been used in winning solutions of multiple online competitions [32], all while decreasing the time it takes to train the model [25].

3.2. Logistic Regression

As a simple but popular machine learning algorithm, logistic regression is based on a weighted sum (with weights w, with the same n dimension as the input x; optional bias parameter b) similarly to a traditional linear regression, followed by an activation function (h) that returns class probabilities ($y(x)$):

$$y = h\left(\sum_{i=1}^{n} w_n x_n + b\right)$$

The activation function used depends if the model is a binary (2 classes) or multinomial (3 or more classes) logistic regression. While both achieve the same purpose, of normalizing output scores to get class probabilities which sum to 1, they must be changed according to the number of classes. In a binary task, we use the sigmoid function:

$$h(x) = \frac{1}{1 + e^{-x}}$$

Notice how, if the output score x is a very large positive number, the output h is 1, i.e. 100% class probability. On the other hand, if the output score x is a very large negative number, the output h is 0, i.e. 0% class probability. Meanwhile, in a multinomial task with k classes, we use the softmax

function:

$$h(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}}$$

The same reasoning of the sigmoid function applies to softmax, only now we have multiple logits x. In fact, the sigmoid is equivalent to the softmax formula when using two inputs x.

With this simple logic, in go the inputs out goes an activation, a binary logistic regression is very similar to a perceptron, the core building block of artificial neural networks, which was modeled after a simplified version of a brain neuron. The difference is that logistic regression can only have probabilistic activation functions (sigmoid or softmax), while a perceptron can have any kind of non-linear function in its output. Through this biologically inspired structure, as depicted on Figure 3, a perceptron on its own can assume a range of many different functions. And through the activation function, it can either output a score in a certain range, predict a variable's value or classify an input with a given probability (logistic regression), thanks to the non-linearity of the activation function.

Figure 3: Simplified images representing a neuron and a perceptron, showing structural and behavioural similarities.

3.3. Multilayer Perceptron

When stacking multiple perceptrons, the artifical neural networks constitutes what is known as a Multi-Layer Perceptron (MLP). This way, each perceptron still follows the same formulation as explained in section 3.2, with the difference that, after the first layer, they receive as inputs the outputs of the previous layer's perceptrons. As seen in the two dimensional example of Figure 4, the addition of a layer allows for more complex model behaviour, which can for instance separate classes of data that were not separable with less layers. In this example, it is possible to see that a single perceptron represents a linear function, which cannot fully separate the data. With an additional layer, the output is based on a weighted sum of three individual perceptrons, which can be imagined as three different functions or decision boundaries, which results in a nonlinear output that can effectively separate the example's data.

In this MLP configuration, the first layer is called the input layer, the last one is the output layer and every possible one between them is an hidden layer,

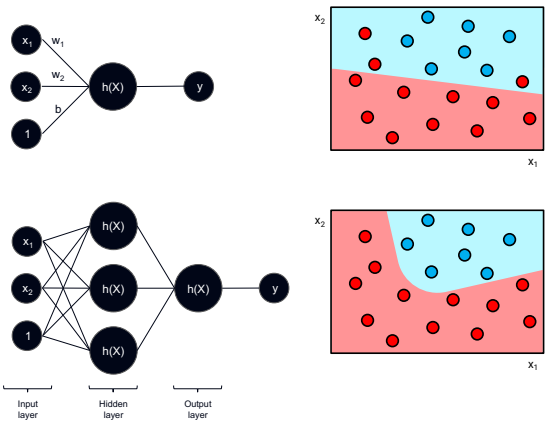


Figure 4: Comparison of the complexity brough by the addition of layers to an artificial neural network, shown through a simple two dimensional example. Each color represents a different class.

as their outputs are internal to the model, not values to be seen. Through this combination, each layer's output is the input to the next layer, until the output layer is reached and the final result is calculated. The very fact that these models tend to have multiple layers is what originated the name deep learning. Overall, each layer brings increasingly higher abstraction, allowing for the extraction of more complex features from the data. Not only does this tend to improve the performance, it is also a feature inspired by nature. For instance, in CNN's, a artificial neural network architecture that is frequently used in computer vision, research has shown that, for instance, in a face classification problem, the first layers only detect edges and lines in the image, the intermediate layers detect facial parts such as eyes and nose, while the final layers activate only to specific faces. Other studies show that the visual cortex present in the human brain is also divided in different parts, like in layers, with the neurons activating to increasing abstraction, from edges in the primary visual cortex to faces in the inferior temporal cortex [16]. However, it is not always a good idea to keep adding layers. With each layer added, and even for each unit (perceptron) added, the total number of parameters that the model needs to learn increases. This is because each unit has its own weights which are learned in the training process. As such, with a finite dataset, after a certain amount of layers, or complexity of the model, the performance starts to decrease due to overfitting. In other words, when there are too many parameters to learn, the model is not able to learn features that are generalizable to unseen data, and just forces the weights to match the training data, even if in irregular and very context specific patterns.

Besides paying attention to the number of layers, there are other techniques in deep learning designed to avoid as much as possible the overfitting. Similarly to other machine learning methods, it is possible to apply regularization to the weights. In regularization, weights have their absolute values minimized, either through L1 regularization, with $Costfunction = Loss + \lambda \sum |w|$, or L2 regularization, with $Costfunction = Loss + \lambda \sum w^2$. This helps avoid the excessive dominance of a set of weights over the others, which makes the model use more evenly other features and, as so, become more robust to overfitting. In the case of L1 regularization, it also promotes feature selection by allowing some weight values to get closer to zero.

Dropout is another method used to prevent overfitting. Similarly to regularization, it tries to prevent the model from prioritizing too much certain features and use more evenly all the available information. But instead of minimizing the weights, it focuses on training every part of the network. It does so by randomly deactivating nodes from the network in each training iteration, according to a certain probability, which represents a hyperparameter. It is a very common practice in deep learning and was introduced by Geoffrey Hinton in 2012 [17], being that his student Alex Krizhevsky's famous winning model Alexnet [21] was one of the first practical uses of this method.

In order to be able to train the model, there must be a way to find out how to change each of its weights in the direction that minimizes the cost function. To do this, deep learning relies on backpropagation. In the same way that the data flows in feedforward mode from the inputs to the outputs, the error can be backpropagated from the output to each node's weight, based on the chain rule, e.g. $\frac{\partial E}{\partial w} = \frac{\partial E}{\partial y} \frac{\partial y}{\partial w}$. As such, it is just a matter of knowing the derivatives of each segment of the neural network, mainly the nodes' activation function and the weighted sum, and apply the chain rule to get the gradient for each weight. Since the gradients represent the variation in the model's error, it is desirable to find the maximum gradient value and make the weights vary in the opposite direction. Then, to apply the gradients and update the weights, an optimization function is used. The simplest and one of the first optimization algorithms to be used was the Stochastic Gradient Descent, which just subtracts the gradients multiplied by a learning rate to the weights. Other, more complex optimizers are now more commonly used, such as Stochastic Gradient Descent with momentum, which essentially adds speed to the optimization in order to try to avoid getting stuck in local minima, RMSprop is also used a lot, adding elements such as automatically adjusted learning rate and tries to avoid oscillations, and Adam is perhaps the most popular option, combining RMSprop and momentum.

3.4. SHAP

In 2017, Scott Lundberg and Su-In Lee published the paper "A Unified Approach to Interpreting Model Predictions" [23]. As the title suggests, the authors proposed a new method to interpret machine learning models that unifies previous ones. They found out that 7 other popular interpretability methods (LIME, Shapley sampling values, DeepLIFT, QII, Layer-Wise Relevance Propagation, Shapley regression values, and tree interpreter) all follow the same core logic: learn a simpler explanation model from the original one, through a local linear model. Because of this, the authors call them additive feature attribution methods. Essentially, for each sample x that we want to interpret, using the output of model f, we train a linear model g, which locally approximates f on sample x. However, the linear model g does not directly use x as input data. Rather, it converts it to x', which represents which features are activated (for instance, $x'_i = 1$ means that we're using feature i, while $x'_i = 0$ means that we're "removing" feature i). As such, and considering that we have M features and $M + 1$ model coefficients (named ϕ), we get the following equation for the interpreter model:

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i$$

And, having the mapping function h_x that transforms x' into x, the interpreter model should locally approximate model f by obeying to the following rule, whenever we get close to x' (i.e. $z' \approx x'$):

$$g(z') \approx f(h_x(z')), \text{ if } z' \approx x'$$

Knowing that the sample x that we want to interpret naturally has all features available (in other words, x' is a vector of all ones), this local approximation dictates that the sum of all ϕ should equal the model's output for sample x:

$$\sum_{i=0}^{M} \phi_i = f(x)$$

Each coefficient ϕ, being this a linear model, relates to each feature's importance on the model. For instance, the bigger the absolute value of ϕ_i is, the bigger the importance of feature i is on the model. Naturally, the sign of ϕ is also relevant, as a positive ϕ corresponds to a positive impact on the model's output (the output value increases) and the opposite occurs for a negative ϕ. An exception here is ϕ_0. There is no feature 0, so it is not associated with any feature in particular. In fact, if we have

an all zeroes vector z' as an input, the output of the interpreter model will be $g(0) = \phi_0$. In theory, it should correspond to the output of the model when no feature is present. Practically, what is done in SHAP is that ϕ_0 assumes the average model output on all the data, so that it represents a form of starting point for the model before adding the impact of each feature. Because of this, we can see each of the remaining coefficients as each feature's push on the base output value (ϕ_0) onto a bigger or smaller output (depending on the coefficient's sign), with the sum of all of these feature related coefficients resulting in the difference between the output on the current sample and the average output value.

There's an additional interesting aspect of the ϕ coefficients. For everyone except ϕ_0, which again does not correspond to any single feature importance, the authors defend that there is only one formula that simultaneously accomplishes three desirable properties:

1: Local accuracy

When approximating the original model f for a specific input x, local accuracy requires the explanation model to at least match the output of f for the original input x.

$$f(x) = g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x'_i$$

2: Missingness

If the simplified inputs (x') represent feature presence, then missingness requires features missing in the original input to have no impact.

$$x'_i = 0 \Rightarrow \phi_i = 0$$

3: Consistency

Let $f_x(z') = f(h_x(z'))$ and $z' \setminus i$ denote setting $z'_i = 0$. For any two models f and f', if

$$f'_x(z') - f'_x(z' \setminus i) \geq f_x(z') - f_x(z' \setminus i)$$

for all inputs $z' \in 0, 1^M$, then $\phi(f', x) \geq \phi(f, x)$.

And the formula that fulfills these properties, considering M features, is the following:

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(|M| - |z'| - 1)!}{|M|!}[f_x(z') - f_x(z' \setminus i)]$$

Having this theoretically sound basis, SHAP implements a model-agnostic interpretability process, allowing any model type to be interpretable. Now the trade-off becomes not that of performance versus interpretability, but rather detailed explanation against computation speed. In order to get the most accurate feature importance, we need to set a large background set, filled with samples that are representative of the dataset, to use in the inactive features ($z' = 0$) in each iteration of SHAP values calculation. Another possibility is to use one of SHAP's versions that are fine tuned to specific models, such as Deep SHAP which uses neural networks' backpropagation property in order to find out faster the influence of each feature through their gradients.

4. Experiments

4.1. Data

Having an interest in exploring genomics data, we chose to use the TCGA dataset, one of the largest open datasets for cancer related genetic data. But instead of extracting it raw from the source, considering the heterogeneous data and the desire to compare with the paper from Katherine A. Hoadley et al [18], the pre-processed data files associated to the paper were used instead. The processing that was already done was not complex enough to restrict our own needs and it included important processes such as batch effect corrections. In terms of feature selection and data processing, the following decisions were made:

- Gathered data from the RNA, copy number ratio and clinical sources.
- Normalized all continuous data into z-scores, with an average value of 0 and standard deviation of 1.
- Removed all features that had at least 40% missing values.
- Performed missing value imputation by interpolation on chromosome data, as it was presented in spatial sequence, and by filling the remaining with the respective feature's average value (which is 0 after normalization).
- Converted all categorical features into a numeric format, with special attention given to the non-binary features "race" and "ajcc_pathologic_tumor_stage". For these two features, embedding layers were created to work alongside the MLP model, while the other models received them as one hot encoded.
- Joined each sample's chrommosome data in a single column per chromosome, averaging the values along their regions.
- Used the primary tumor type as the label, obtaining a total of 33 classes, as seen in table 2.
- The final dataframe comprised of nearly 10k samples, with over 20k features.

Model	AUC	Weighted AUC
LR	**0.99978**	**0.99974**
SVM	0.99933	0.99916
XGBoost	0.99777	0.99786
MLP	0.99165	0.99439

Table 1: Benchmark of models trained in the main tumor type classification task, measured on the test set. The logistic regression has slightly better results, both in the standard and the weighted AUC metrics.

4.2. Model Performance

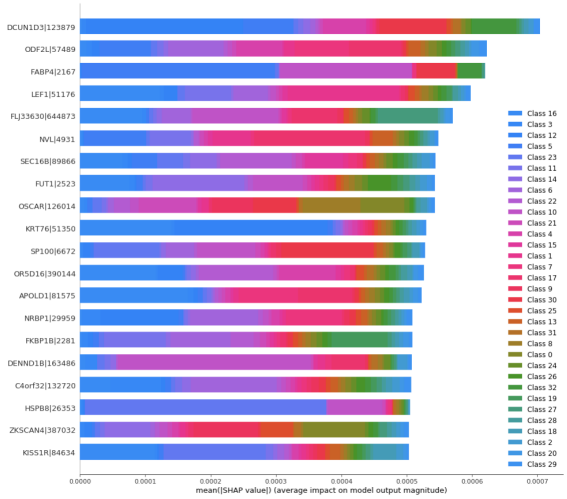
Considering the high dimensionality of the problem, with more than 20k features and almost 10k rows, it would be expected that a complex model such as a multilayer perceptron or boosted trees would clearly outperform simpler models (i.e. with less parameters and nonlinearities). Furthermore, with the apparent complexity of the classification task, predicting the tumor type based on genetic and clinical data, this idea was reinforced. However, as we can see in table 1, not only did all of the tested models excel in the task, with over 0.9 AUC, the model that reached the best performance was of logistic regression (LR) type, one of the simplest possibilities. This is particularly impressive considering the profound imbalance of the dataset, with the most represented class having more than 1k samples and the least represented only having 29 samples. The unexpected high performance of all models might be caused mainly by one of two scenarios. Either the task is actually quite easy to handle, with the different tumor types being clearly characterized by a subset of genes or clinical features, or there was a data leakage problem. This later concept means that there could have been an unexpected additional information present in the data, specifically while training the models, possibly revealing or at least being highly correlated with the ground truth. In either cases, with the classification problem being easy in the sense of heavily depending on a small group of features, it is understandable that small models built to find rather simple decision boundaries can better approximate simplistic patterns in the data than more complex, highly nonlinear models. In order to investigate further on this topic, with the purpose of visualizing the models' logic and identifying the possible cause of the exceptional performance, we can interpret the models through SHAP values.

4.3. Interpretability

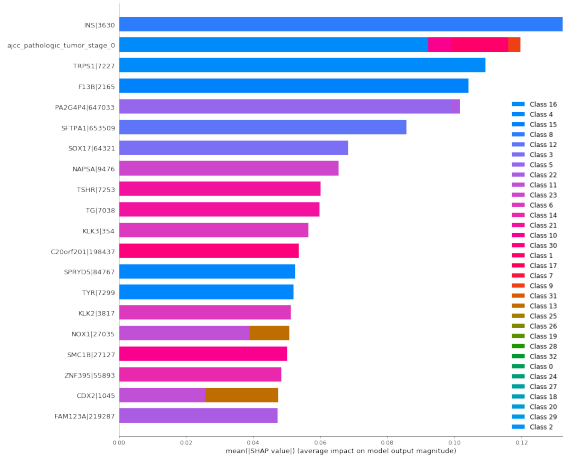
In order to interpret the models, SHAP values were calculated on 1k test samples for each one, although with different algorithms. For the XGBoost, the TreeExplainer module was used, considering its optimizations for decision tree based models, such that the calculations only took around 19 seconds. For the logistic regression (LR), a similar approach was taken by using LinearExplainer, with its own optimizations for linear models, with the calculations taking just 6 seconds. Meanwhile, the MLP was interpreted with KernelExplainer, a module that is model-agnostic, which made it take a much longer time at around 4 hours and 55 minutes. The SVM model did not go through the interpretability segment, as it was not the best performing model and it would require the use of the KernelExplainer again, demanding a lot of computation time. All of this processes were run in similar cloud computing environments, so we can really see how the complexity of the models and the existence of SHAP optimizations can affect the time required to obtain feature importance scores.

Having the SHAP values calculated, we can use SHAP's integrated visualization tools to analyze the results. To start off, it is good practise to look at an overview of the most important features, by viewing the average SHAP value of each feature along the selected 1k test samples. We can do that using the summary plot, where it is even possible to simultaneously view the data for all output classes, as seen on figure 5. In this plot, we can see the features ranked by their average absolute importance, with the colors indicating the contribution of the impact on each class. Therefore, if for instance there is a big segment of a feature's corresponding bar with a single color, it means that that feature is particularly important for the class associated to the color. Knowing this, we can immediately see an interesting difference between the models. While the neural network (MLP) and the logistic regression (LR) models show a diffused behaviour, where features are relevant across multiple tumor types, given their multicolored rows, the XGBoost model appears to make a clear distinction between the classes, having the features being relevant mostly just to a single class, according to its almost monochromatic bars. Another difference between these overview plots is that there is a more significant drop in importance for XGBoost's features, while for the MLP and the LR the features show a closer level of importance between themselves. What the three models have in common is that they tend to value RNA genetic data the most, with all the top important features corresponding to genes except the indicator of the tumor stage 0 for the XGBoost model. No direct comparison should be made regarding the actual SHAP values indicated in the X axis, as the different models have different output types and therefore different scales.

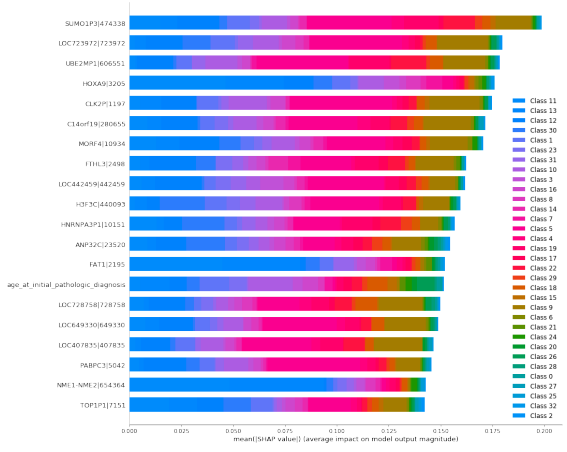
Going in a more detailed view, we can continue on



(a) MLP



(b) XGBoost

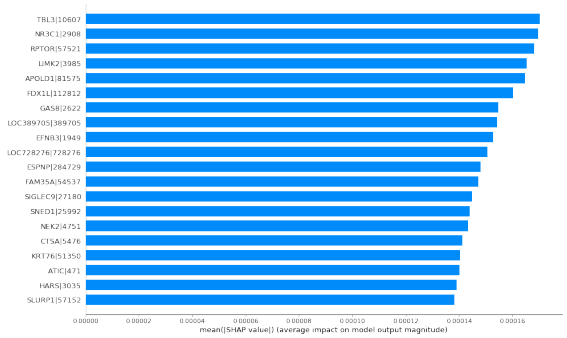


(c) LR

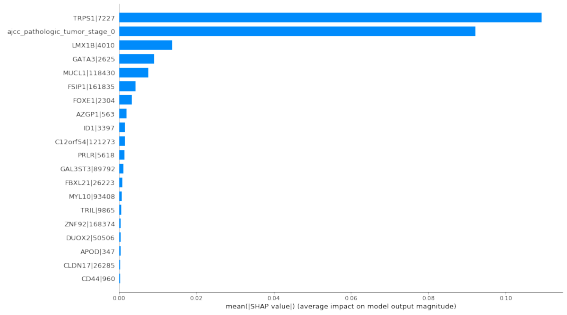
Figure 5: Feature importance summary. In these plots, features are ranked in descending order of importance, based on their average SHAP value in 1k test samples, for all the 33 classes. Each color represents the contribution from each class.

the summary plots, but focus on one output class, i.e. one tumor type, at a time. This is particularly relevant to confirm how distinctive the XGBoost model's behaviour might be for different classes and how diffuse the remaining ones are. These figures 6, 7 and 8 clearly confirm that XGBoost focuses on a very small subset of features, usually only requiring one or two features to identify the tumor type, while MLP and LR have a more disperse contribution of bigger and different features sets. The XGBoost plots appear to give strength to the possibility of data leakage, considering the sole reliance on a handful of features to identify all output classes. In fact, according to academia databases [2] and studies [33], the most important features identified by the XGBoost model are in fact referenced as being in high expression specifically in the corresponding tumor types. However, if the remaining SHAP interpretation was true to the models, we can see that the MLP and LR models were able to obtain similar performance using a larger combination of other features. As such, perhaps there are certain genes that are very correlated to the tumor types, but still are not an obvious solution, considering that for these two models the optimization found the patterns in combinations of features to be better than the other features in isolation.

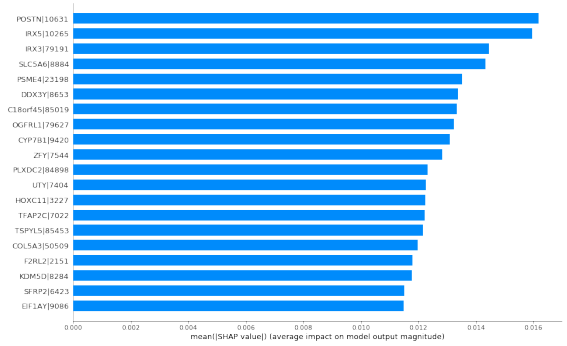
We can keep on descending to a deeper level of detail, by visualizing samples one by one through waterfall plots. In this kind of graph, the feature importance is demonstrated by how each feature contributes to the final output, comparing to the reference of that class's average output score. Each row corresponds to a single or a subset of features, in the case of the last row, being again ordered in descending importance, and the size of each bar corresponds to the magnitude of the shift in the output caused by the corresponding feature. The colors now distinguish if the feature increases (red) or decreases (blue) the output value. In figure 9, we use a breast cancer sample, as this is the most represented tumor type on the dataset and therefore the one that the models should be more knowledgeable about. Again, we see XGBoost's reliance on just a few features, in this case the four times higher than average expression of the TRPS1 gene and the fact that the cancer is not on stage 0 (although this value decreases the probability that this sample corresponds to breast cancer), and much more disperse contribution of features, particularly in the case of logistic regression. Curiously, the best performing model, even if just by a small margin, is actually the logistic regression one, which uses features in more similar levels of importance and even genes that are not as commonly used as cancer markers. This confirms that there are multiple solutions to the cancer classification problem and might suggest



(a) MLP

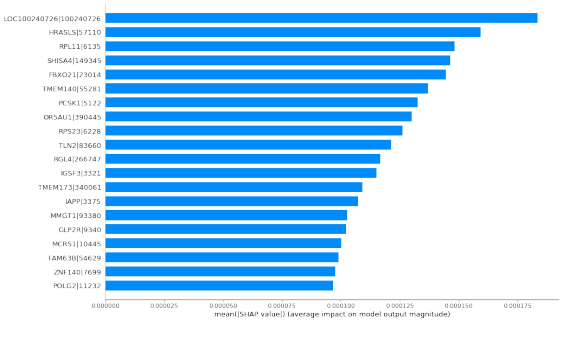


(b) XGBoost

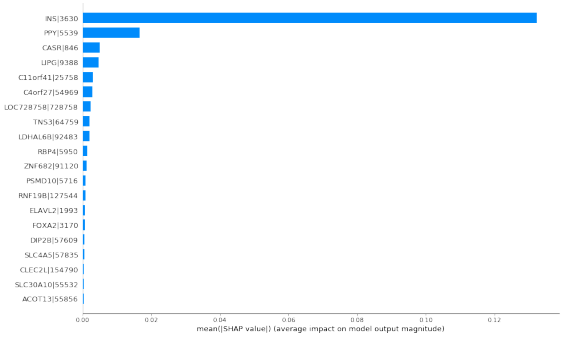


(c) LR

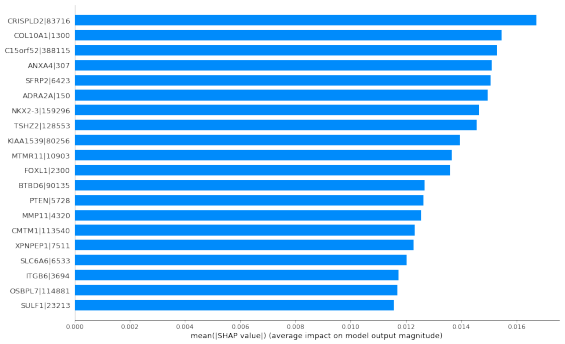
Figure 6: Feature importance summary just for the breast cancer class. TRPS1 and POSTN are genes that are commonly found in higher expression in breast cancer than in other tumors or normal tissues. The XGBoost model takes into account the indication of whether the sample is in tumor stage 0 or not, which suggests that the model might adapt its logic according to how developed the tumor is.



(a) MLP



(b) XGBoost



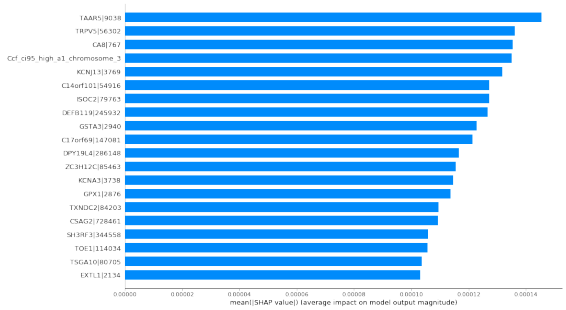
(c) LR

Figure 7: Feature importance summary just for the pancreas cancer class. INS is a gene that is commonly found in higher expression in pancreas cancer than in other tumors or normal tissues.

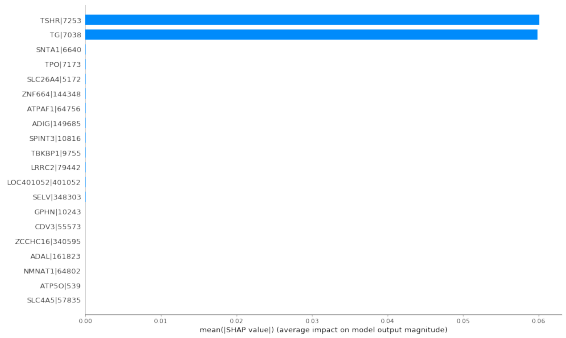
further analysis of the less regarded genes identified by the MLP and LR models.

5. Conclusion

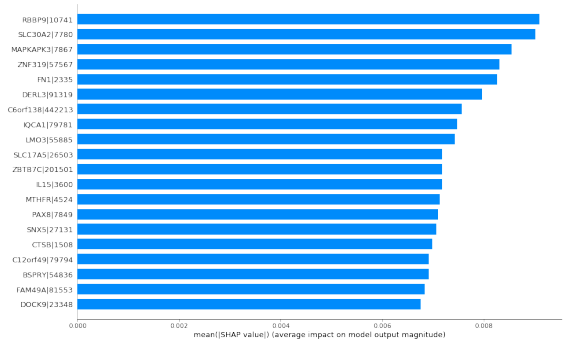
In this paper, we sought to train a neural network model on TCGA data to identify tumor types, ideally with state-of-the-art results, and then be able to interpret the model. A neural network was in fact trained and interpreted as intended, showing that, provided that there is enough time and computing resources, deep learning no longer needs to be seen as an untrustable black box. However, the other baseline machine learning models obtained performances just as good or even better than the proposed neural network. Even simpler architectures such as logistic regression and support vector machine achieved performance of over 0.9 AUC, even on a weighted version which is impressive for such an imbalanced dataset. This should in part be due to a data leakage problem, in the sense that some genes are correlated so much with tumor types that they essentially give away the ground truth. Still, despite the XGBoost model emphasizing just a handful of already well known cancer markers, there are less studied combinations of genes that



(a) MLP



(b) XGBoost

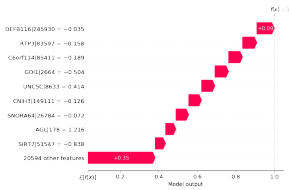


(c) LR

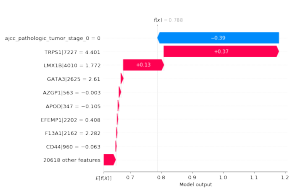
Figure 8: Feature importance summary just for the thyroid cancer class. TSHR and TG are genes that are commonly found in higher expression in thyroid cancer than in other tumors or normal tissues.

served a major rule in some models, which might suggest new research possibilities in cancer genetics. It is important however to take this interpretation results with a pinch of salt, as the diversity of SHAP explainer algorithms and the possibly small amount of background data used could have resulted in feature importance that is not entirely true to the models.

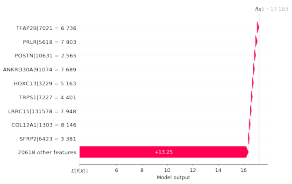
As future work, we should go over the results with other researchers and medical staff, identifying then a new project where data leakage is not an issue. After doing that, more innovative models can be experimented, such as Regularization Learning Networks [30] and TabNet [5], as well as a more rigorous and standard approach to interpretability.



(a) MLP



(b) XGBoost



(c) LR

Figure 9: Feature importance on a specific sample from the most common class (breast cancer). Colors represent the direction of the feature's contribution, where red means that it increases the output score, while blue means that it decreases. The size of the bars indicate the magnitude of the shift in the output value that the corresponding feature causes.

Acknowledgements

I hereby thank FCT for financing with a scholarship, as well as my supervisors, Alexandra Carvalho and Susana Vinga, for motivating me to go forward with the research project. The results published here are in whole or part based upon data generated by the TCGA Research Network: https://www.cancer.gov/tcga.

References

[1] Communication: Building Trust in Human Centric Artificial Intelligence — Digital Single Market.

[2] The Human Protein Atlas.

[3] Cancer fact sheet, 2018.

[4] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nature Biotechnology*, 33(8):831–838, 8 2015.

[5] S. O. Arik and T. Pfister. TabNet: Attentive Interpretable Tabular Learning. 8 2019.

[6] A. Avati, K. Jung, S. Harman, L. Downing, A. Ng, and N. H. Shah. Improving Palliative Care with Deep Learning. 11 2017.

[7] D. Bahdanau, K. Cho, and Y. Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. 9 2014.

[8] B. Baker, I. Kanitscheider, T. Markov, Y. Wu, G. Powell, B. McGrew, and I. Mordatch. Emergent Tool Use From Multi-Agent Autocurricula. 9 2019.

[9] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 13-17-August-2016, pages 785–794. Association for Computing Machinery, 8 2016.

[10] H. Cui, C. Zhou, X. Dai, Y. Liang, R. Paffenroth, and D. Korkin. Boosting Gene Expression Clustering with System-Wide Biological Information: A Robust Autoencoder Approach. *bioRxiv*, page 214122, 2017.

[11] F. Dutil, J. P. Cohen, M. Weiss, G. Derevyanko, and Y. Bengio. Towards Gene Expression Convolutions using Gene Interaction Graphs. 6 2018.

[12] J. Engel, C. Resnick, A. Roberts, S. Dieleman, D. Eck, K. Simonyan, and M. Norouzi. Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders, 2017.

[13] G. Eraslan, Avsec, J. Gagneur, and F. J. Theis. Deep learning: new computational modelling techniques for genomics, 7 2019.

[14] N. Fortelny and C. Bock. Knowledge-primed neural networks enable biologically interpretable deep learning on single-cell sequencing data. *bioRxiv*, page 794503, 2019.

[15] L. A. Gatys, A. S. Ecker, and M. Bethge. A Neural Algorithm of Artistic Style. 8 2015.

[16] Grace Lindsay. Deep Convolutional Neural Networks as Models of the Visual System: Q&A, 2018.

[17] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. 7 2012.

[18] K. A. Hoadley, C. Yau, T. Hinoue, D. M. Wolf, A. J. Lazar, E. Drill, R. Shen, A. M. Taylor, A. D. Cherniack, V. Thorsson, R. Akbani, R. Bowlby, C. K. Wong, M. Wiznerowicz, F. Sanchez-Vega, A. G. Robertson, B. G. Schneider, M. S. Lawrence, H. Noushmehr, T. M. Malta, S. J. Caesar-Johnson, J. A. Demchok, I. Felau, M. Kasapi, M. L. Ferguson, C. M. Hutter, H. J. Sofia, R. Tarnuzzer, Z. Wang, L. Yang, J. C. Zenklusen, J. J. Zhang, S. Chudamani, J. Liu, L. Lolla, R. Naresh, T. Pihl, Q. Sun, Y. Wan, Y. Wu, J. Cho, T. DeFreitas, S. Frazer, N. Gehlenborg, G. Getz, D. I. Heiman, J. Kim, M. S. Lawrence, P. Lin, S. Meier, M. S. Noble, G. Saksena, D. Voet, H. Zhang, B. Bernard, N. Chambwe, V. Dhankani, T. Knijnenburg, R. Kramer, K. Leinonen, Y. Liu, M. Miller, S. Reynolds, I. Shmulevich, V. Thorsson, W. Zhang, R. Akbani, B. M. Broom, A. M. Hegde, Z. Ju, R. S. Kanchi, A. Korkut, J. Li, H. Liang, S. Ling, W. Liu, Y. Lu, G. B. Mills, K. S. Ng, A. Rao, M. Ryan, J. Wang, J. N. Weinstein, J. Zhang, A. Abeshouse, J. Armenia, D. Chakravarty, W. K. Chatila, I. de Bruijn, J. Gao, B. E. Gross, Z. J. Heins, R. Kundra, K. La, M. Ladanyi, A. Luna, M. G. Nissan, A. Ochoa, S. M. Phillips, E. Reznik, F. Sanchez-Vega, C. Sander, N. Schultz, R. Sheridan, S. O. Sumer, Y. Sun, B. S. Taylor, J. Wang, H. Zhang, P. Anur, M. Peto, P. Spellman, C. Benz, J. M. Stuart, C. K. Wong, C. Yau, D. N. Hayes, J. S. Parker, M. D. Wilkerson, A. Ally, M. Balasundaram, R. Bowlby, D. Brooks, R. Carlsen, E. Chuah, N. Dhalla, R. Holt, S. J. Jones, K. Kasaian, D. Lee, Y. Ma, M. A. Marra, M. Mayo, R. A. Moore, A. J. Mungall, K. Mungall, A. G. Robertson, S. Sadeghi, J. E. Schein, P. Sipahimalani, A. Tam, N. Thiessen, K. Tse, T. Wong, A. C. Berger, R. Beroukhim, A. D. Cherniack, C. Cibulskis, S. B. Gabriel, G. F. Gao, G. Ha, M. Meyerson, S. E. Schumacher, J. Shih, M. H. Kucherlapati, R. S. Kucherlapati, S. Baylin, L. Cope, L. Danilova, M. S. Bootwalla, P. H. Lai, D. T. Maglinte, D. J. Van Den Berg, D. J. Weisenberger, J. T. Auman, S. Balu, T. Bodenheimer, C. Fan, K. A. Hoadley, A. P. Hoyle, S. R. Jefferys, C. D. Jones, S. Meng, P. A. Mieczkowski, L. E. Mose, A. H. Perou, C. M. Perou, J. Roach, Y. Shi, J. V. Simons, T. Skelly, M. G. Soloway, D. Tan, U. Veluvolu, H. Fan, T. Hinoue, P. W. Laird, H. Shen, W. Zhou, M. Bellair, K. Chang, K. Covington, C. J. Creighton, H. Dinh, H. V. Doddapaneni, L. A. Donehower, J. Drummond, R. A. Gibbs, R. Glenn, W. Hale, Y. Han, J. Hu, V. Korchina, S. Lee, L. Lewis, W. Li, X. Liu, M. Morgan, D. Morton, D. Muzny, J. Santibanez, M. Sheth, E. Shinbrot, L. Wang, M. Wang, D. A. Wheeler, L. Xi, F. Zhao, J. Hess, E. L. Appelbaum, M. Bailey, M. G. Cordes, L. Ding, C. C. Fronick, L. A. Fulton, R. S. Fulton, C. Kandoth, E. R. Mardis, M. D. McLellan, C. A. Miller, H. K. Schmidt, R. K. Wilson, D. Crain, E. Curley, J. Gardner, K. Lau, D. Mallery, S. Morris, J. Paulauskis, R. Penny, C. Shelton, T. Shelton, M. Sherman, E. Thompson, P. Yena,

J. Bowen, J. M. Gastier-Foster, M. Gerken, K. M. Leraas, T. M. Lichtenberg, N. C. Ramirez, L. Wise, E. Zmuda, N. Corcoran, T. Costello, C. Hovens, A. L. Carvalho, A. C. de Carvalho, J. H. Fregnani, A. Longatto-Filho, R. M. Reis, C. Scapulatempo-Neto, H. C. Silveira, D. O. Vidal, A. Burnette, J. Eschbacher, B. Hermes, A. Noss, R. Singh, M. L. Anderson, P. D. Castro, M. Ittmann, D. Huntsman, B. Kohl, X. Le, R. Thorp, C. Andry, E. R. Duffy, V. Lyadov, O. Paklina, G. Setdikova, A. Shabunin, M. Tavobilov, C. McPherson, R. Warnick, R. Berkowitz, D. Cramer, C. Feltmate, N. Horowitz, A. Kibel, M. Muto, C. P. Raut, A. Malykh, J. S. Barnholtz-Sloan, W. Barrett, K. Devine, J. Fulop, Q. T. Ostrom, K. Shimmel, Y. Wolinsky, A. E. Sloan, A. De Rose, F. Giuliante, M. Goodman, B. Y. Karlan, C. H. Hagedorn, J. Eckman, J. Harr, J. Myers, K. Tucker, L. A. Zach, B. Deyarmin, H. Hu, L. Kvecher, C. Larson, R. J. Mural, S. Somiari, A. Vicha, T. Zelinka, J. Bennett, M. Iacocca, B. Rabeno, P. Swanson, M. Latour, L. Lacombe, B. Têtu, A. Bergeron, M. McGraw, S. M. Staugaitis, J. Chabot, H. Hibshoosh, A. Sepulveda, T. Su, T. Wang, O. Potapova, O. Voronina, L. Desjardins, O. Mariani, S. Roman-Roman, X. Sastre, M. H. Stern, F. Cheng, S. Signoretti, A. Berchuck, D. Bigner, E. Lipp, J. Marks, S. McCall, R. McLendon, A. Secord, A. Sharp, M. Behera, D. J. Brat, A. Chen, K. Delman, S. Force, F. Khuri, K. Magliocca, S. Maithel, J. J. Olson, T. Owonikoko, A. Pickens, S. Ramalingam, D. M. Shin, G. Sica, E. G. Van Meir, H. Zhang, W. Eijckenboom, A. Gillis, E. Korpershoek, L. Looijenga, W. Oosterhuis, H. Stoop, K. E. van Kessel, E. C. Zwarthoff, C. Calatozzolo, L. Cuppini, S. Cuzzubbo, F. DiMeco, G. Finocchiaro, L. Mattei, A. Perin, B. Pollo, C. Chen, J. Houck, P. Lohavanichbutr, A. Hartmann, C. Stoehr, R. Stoehr, H. Taubert, S. Wach, B. Wullich, W. Kycler, D. Murawa, M. Wiznerowicz, K. Chung, W. J. Edenfield, J. Martin, E. Baudin, G. Bubley, R. Bueno, A. De Rienzo, W. G. Richards, S. Kalkanis, T. Mikkelsen, H. Noushmehr, L. Scarpace, N. Girard, M. Aymerich, E. Campo, E. Giné, A. L. Guillermo, N. Van Bang, P. T. Hanh, B. D. Phu, Y. Tang, H. Colman, K. Evason, P. R. Dottino, J. A. Martignetti, H. Gabra, H. Juhl, T. Akeredolu, S. Stepa, D. Hoon, K. Ahn, K. J. Kang, F. Beuschlein, A. Breggia, M. Birrer, D. Bell, M. Borad, A. H. Bryce, E. Castle, V. Chandan, J. Cheville, J. A. Copland, M. Farnell, T. Flotte, N. Giama, T. Ho, M. Kendrick, J. P. Kocher, K. Kopp, C. Moser, D. Nagorney, D. O'Brien, B. P. O'Neill, T. Patel, G. Petersen, F. Que, M. Rivera, L. Roberts, R. Smallridge, T. Smyrk, M. Stanton, R. H. Thompson, M. Torbenson, J. D. Yang, L. Zhang, F. Brimo, J. A. Ajani, A. M. A. Gonzalez, C. Behrens, o. Bondaruk, R. Broaddus, B. Czerniak, B. Esmaeli, J. Fujimoto, J. Gershenwald, C. Guo, C. Logothetis, F. Meric-Bernstam, C. Moran, L. Ramondetta, D. Rice, A. Sood, P. Tamboli, T. Thompson, P. Troncoso, A. Tsao, I. Wistuba, C. Carter, L. Haydu, P. Hersey, V. Jakrot, H. Kakavand, R. Kefford, K. Lee, G. Long, G. Mann, M. Quinn, R. Saw, R. Scolyer, K. Shannon, A. Spillane, J. Stretch, M. Synott, J. Thompson, J. Wilmott, H. Al-Ahmadie, T. A. Chan, R. Ghossein, A. Gopalan, D. A. Levine, V. Reuter, S. Singer, B. Singh, N. V. Tien, T. Broudy, C. Mirsaidi, P. Nair, P. Drwiega, J. Miller, J. Smith, H. Zaren, J. W. Park, N. P. Hung, E. Kebebew, W. M. Linehan, A. R. Metwalli, K. Pacak, P. A. Pinto, M. Schiffman, L. S. Schmidt, C. D. Vocke, N. Wentzensen, R. Worrell, H. Yang, M. Moncrieff, C. Goparaju, J. Melamed, H. Pass, N. Botnariuc, I. Caraman, M. Cernat, I. Chemencedji, A. Clipca, S. Doruc, G. Gorincioi, S. Mura, M. Pirtac, I. Stancul, D. Tcaciuc, M. Albert, I. Alexopoulou, A. Arnaout, J. Bartlett, J. Engel, S. Gilbert, J. Parfitt, H. Sekhon, G. Thomas, D. M. Rassl, R. C. Rintoul, C. Bifulco, R. Tamakawa, W. Urba, N. Hayward, H. Timmers, A. Antenucci, F. Facciolo, G. Grazi, M. Marino, R. Merola, R. de Krijger, A. P. Gimenez-Roqueplo, A. Piché, S. Chevalier, G. McKercher, K. Birsoy, G. Barnett, C. Brewer, C. Farver, T. Naska, N. A. Pennell, D. Raymond, C. Schilero, K. Smolenski, F. Williams, C. Morrison, J. A. Borgia, M. J. Liptay, M. Pool, C. W. Seder, K. Junker, L. Omberg, M. Dinkin, G. Manikhas, D. Alvaro, M. C. Bragazzi, V. Cardinale, G. Carpino, E. Gaudio, D. Chesla, S. Cottingham, M. Dubina, F. Moiseenko, R. Dhanasekaran, K. F. Becker, K. P. Janssen, J. Slotta-Huspenina, M. H. Abdel-Rahman, D. Aziz, S. Bell, C. M. Cebulla, A. Davis, R. Duell, J. B. Elder, J. Hilty, B. Kumar, J. Lang, N. L. Lehman, R. Mandt, P. Nguyen, R. Pilarski, K. Rai, L. Schoenfield, K. Senecal, P. Wakely, P. Hansen, R. Lechan, J. Powers, A. Tischler, W. E. Grizzle, K. C. Sexton, A. Kastl, J. Henderson, S. Porten, J. Waldmann, M. Fassnacht, S. L. Asa, D. Schadendorf, M. Couce, M. Graefen, H. Huland, G. Sauter, T. Schlomm, R. Si-

mon, P. Tennstedt, O. Olabode, M. Nelson, O. Bathe, P. R. Carroll, J. M. Chan, P. Disaia, P. Glenn, R. K. Kelley, C. N. Landen, J. Phillips, M. Prados, J. Simko, K. Smith-McCune, S. VandenBerg, K. Roggin, A. Fehrenbach, A. Kendler, S. Sifri, R. Steele, A. Jimeno, F. Carey, I. Forgie, M. Mannelli, M. Carney, B. Hernandez, B. Campos, C. Herold-Mende, C. Jungk, A. Unterberg, A. von Deimling, A. Bossler, J. Galbraith, L. Jacobus, M. Knudson, T. Knutson, D. Ma, M. Milhem, R. Sigmund, A. K. Godwin, R. Madan, H. G. Rosenthal, C. Adebamowo, S. N. Adebamowo, A. Boussioutas, D. Beer, T. Giordano, A. M. Mes-Masson, F. Saad, T. Bocklage, L. Landrum, R. Mannel, K. Moore, K. Moxley, R. Postier, J. Walker, R. Zuna, M. Feldman, F. Valdivieso, R. Dhir, J. Luketich, E. M. Pinero, M. Quintero-Aguilo, C. G. Carlotti, J. S. Dos Santos, R. Kemp, A. Sankarankuty, D. Tirapelli, J. Catto, K. Agnew, E. Swisher, J. Creaney, B. Robinson, C. S. Shelley, E. M. Godwin, S. Kendall, C. Shipman, C. Bradford, T. Carey, A. Haddad, J. Moyer, L. Peterson, M. Prince, L. Rozek, G. Wolf, R. Bowman, K. M. Fong, I. Yang, R. Korst, W. K. Rathmell, J. L. Fantacone-Campbell, J. A. Hooke, A. J. Kovatich, C. D. Shriver, J. DiPersio, B. Drake, R. Govindan, S. Heath, T. Ley, B. Van Tine, P. Westervelt, M. A. Rubin, J. I. Lee, N. D. Aredes, A. Mariamidze, J. M. Stuart, C. C. Benz, and P. W. Laird. Cell-of-Origin Patterns Dominate the Molecular Classification of 10,000 Tumors from 33 Types of Cancer. *Cell*, 173(2):291–304, 4 2018.

[19] K. A. Hoadley, C. Yau, D. M. Wolf, A. D. Cherniack, D. Tamborero, S. Ng, M. D. Leiserson, B. Niu, M. D. McLellan, V. Uzunangelov, J. Zhang, C. Kandoth, R. Akbani, H. Shen, L. Omberg, A. Chu, A. A. Margolin, L. J. Van't Veer, N. Lopez-Bigas, P. W. Laird, B. J. Raphael, L. Ding, A. G. Robertson, L. A. Byers, G. B. Mills, J. N. Weinstein, C. Van Waes, Z. Chen, E. A. Collisson, C. C. Benz, C. M. Perou, J. M. Stuart, R. Abbott, S. Abbott, B. A. Aksoy, K. Aldape, A. Ally, S. Amin, D. Anastassiou, J. T. Auman, K. A. Baggerly, M. Balasundaram, S. Balu, S. B. Baylin, S. C. Benz, B. P. Berman, B. Bernard, A. S. Bhatt, I. Birol, A. D. Black, T. Bodenheimer, M. S. Bootwalla, J. Bowen, R. Bressler, C. A. Bristow, A. N. Brooks, B. Broom, E. Buda, R. Burton, Y. S. Butterfield, D. Carlin, S. L. Carter, T. D. Casasent, K. Chang, S. Chanock, L. Chin, D. Y. Cho, J. Cho, E. Chuah, H. J. E. Chun, K. Cibulskis, G. Ciriello, J. Cleland, M. Cline, B. Craft, C. J. Creighton, L. Danilova, T. Davidsen, C. Davis, N. D. Dees, K. Delehaunty, J. A. Demchok, N. Dhalla, D. DiCara, H. Dinh, J. R. Dobson, D. Dodda, H. V. Doddapaneni, L. Donehower, D. J. Dooling, G. Dresdner, J. Drummond, A. Eakin, M. Edgerton, J. M. Eldred, G. Eley, K. Ellrott, C. Fan, S. Fei, I. Felau, S. Frazer, S. S. Freeman, J. Frick, C. C. Fronick, L. L. Fulton, R. Fulton, S. B. Gabriel, J. Gao, J. M. Gastier-Foster, N. Gehlenborg, M. George, G. Getz, R. Gibbs, M. Goldman, A. Gonzalez-Perez, B. Gross, R. Guin, P. Gunaratne, A. Hadjipanayis, M. P. Hamilton, S. R. Hamilton, L. Han, Y. Han, H. A. Harper, P. Haseley, D. Haussler, D. N. Hayes, D. I. Heiman, E. Helman, C. Helsel, S. M. Herbrich, J. G. Herman, T. Hinoue, C. Hirst, M. Hirst, R. A. Holt, A. P. Hoyle, L. Iype, A. Jacobsen, S. R. Jeffreys, M. A. Jensen, C. D. Jones, S. J. Jones, Z. Ju, J. Jung, A. Kahles, A. Kahn, J. Kalicki-Veizer, D. Kalra, K. L. Kanchi, D. W. Kane, H. Kim, J. Kim, T. Knijnenburg, D. C. Koboldt, C. Kovar, R. Kramer, R. Kreisberg, R. Kucherlapati, M. Ladanyi, E. S. Lander, D. E. Larson, M. S. Lawrence, D. Lee, E. Lee, S. Lee, W. Lee, K. V. Lehmann, K. Leinonen, K. M. Leraas, S. Lerner, D. A. Levine, L. Lewis, T. J. Ley, H. I. Li, J. Li, W. Li, H. Liang, T. M. Lichtenberg, J. Lin, L. Lin, P. Lin, W. Liu, Y. Liu, Y. Liu, P. L. Lorenzi, C. Lu, Y. Lu, L. J. Luquette, S. Ma, V. J. Magrini, H. S. Mahadeshwar, E. R. Mardis, M. A. Marra, M. Mayo, C. McAllister, S. E. McGuire, J. F. McMichael, J. Melott, S. Meng, M. Meyerson, P. A. Mieczkowski, C. A. Miller, M. L. Miller, M. Miller, R. A. Moore, M. Morgan, D. Morton, L. E. Mose, A. J. Mungall, D. Muzny, L. Nguyen, M. S. Noble, H. Noushmehr, M. O'Laughlin, A. I. Ojesina, T. H. O. Yang, B. Ozenberger, A. Pantazi, M. Parfenov, P. J. Park, J. S. Parker, E. Paull, C. S. Pedamallu, T. Pihl, C. Pohl, D. Pot, A. Protopopov, T. Przytycka, A. Radenbaugh, N. C. Ramirez, R. Ramirez, G. Rätsch, J. Reid, X. Ren, B. Reva, S. M. Reynolds, S. K. Rhie, J. Roach, H. Rovira, M. Ryan, G. Saksena, S. Salama, C. Sander, N. Santoso, J. E. Schein, H. Schmidt, N. Schultz, S. E. Schumacher, J. Seidman, Y. Senbabaoglu, S. Seth, S. Sharpe, R. Shen, M. Sheth, Y. Shi, I. Shmulevich, G. O. Silva, J. V. Simons, R. Sinha, P. Sipahimalani, S. M. Smith, H. J. Sofia, A. Sokolov, M. G. Soloway, X. Song, C. Sougnez, P. Spellman, L. Staudt, C. Stewart, P. Stojanov, X. Su, S. O. Sumer,

Y. Sun, T. Swatloski, B. Tabak, A. Tam, D. Tan, J. Tang, R. Tarnuzzer, B. S. Taylor, N. Thiessen, V. Thorsson, T. Triche, D. J. Van Den Berg, F. Vandin, R. J. Varhol, C. J. Vaske, U. Veluvolu, R. Verhaak, D. Voet, J. Walker, J. W. Wallis, P. Waltman, Y. Wan, M. Wang, W. Wang, Z. Wang, S. Waring, N. Weinhold, D. J. Weisenberger, M. C. Wendl, D. Wheeler, M. D. Wilkerson, R. K. Wilson, L. Wise, A. Wong, C. J. Wu, C. C. Wu, H. T. Wu, J. Wu, T. Wylie, L. Xi, R. Xi, Z. Xia, A. W. Xu, D. Yang, L. Yang, L. Yang, Y. Yang, J. Yao, R. Yao, K. Ye, K. Yoshihara, Y. Yuan, A. K. Yung, T. Zack, D. Zeng, J. C. Zenklusen, H. Zhang, J. Zhang, N. Zhang, Q. Zhang, W. Zhang, W. Zhao, S. Zheng, J. Zhu, E. Zmuda, and L. Zou. Multiplatform analysis of 12 cancer types reveals molecular classification within and across tissues of origin. *Cell*, 158(4):929–944, 8 2014.

[20] A. Karpathy. PyTorch at Tesla, 2019.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. Technical report, 2012.

[22] J. Lanchantin, R. Singh, Z. Lin, and Y. Qi. Deep Motif: Visualizing Genomic Sequence Classifications. 5 2016.

[23] S. M. Lundberg and S.-I. Lee. A Unified Approach to Interpreting Model Predictions, 2017.

[24] J. Ma, M. K. Yu, S. Fong, K. Ono, E. Sage, B. Demchak, R. Sharan, and T. Ideker. Using deep learning to model the hierarchical structure and function of a cell. *Nature Methods*, 15(4):290–298, 4 2018.

[25] V. Morde. XGBoost Algorithm: Long May She Reign! - Towards Data Science, 2019.

[26] OpenAI. OpenAI Five, 2018.

[27] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. Semantic Image Synthesis with Spatially-Adaptive Normalization. 3 2019.

[28] A. Rajkomar, E. Oren, K. Chen, A. M. Dai, N. Hajaj, P. J. Liu, X. Liu, M. Sun, P. Sundberg, H. Yee, K. Zhang, G. E. Duggan, G. Flores, M. Hardt, J. Irvine, Q. Le, K. Litsch, J. Marcus, A. Mossin, J. Tansuwan, D. Wang, J. Wexler, J. Wilson, D. Ludwig, S. L. Volchenboum, K. Chou, M. Pearson, S. Madabushi, N. H. Shah, A. J. Butte, M. Howell, C. Cui, G. Corrado, and J. Dean. Scalable and accurate deep learning for electronic health records. 1 2018.

[29] M. T. Ribeiro, S. Singh, and C. Guestrin. "Why Should I Trust You?" Explaining the Predictions of Any Classifier.

[30] I. Shavitt and E. Segal. Regularization Learning Networks: Deep Learning for Tabular Datasets, 2018.

[31] R. Shen, Q. Mo, N. Schultz, V. E. Seshan, A. B. Olshen, J. Huse, M. Ladanyi, and C. Sander. Integrative Subtype Discovery in Glioblastoma Using iCluster. *PLoS ONE*, 7(4):e35236, 4 2012.

[32] Shivam Bansal. Historical Data Science Trends on Kaggle, 2018.

[33] M. Uhlén, L. Fagerberg, B. M. Hallström, C. Lindskog, P. Oksvold, A. Mardinoglu, Sivertsson, C. Kampf, E. Sjöstedt, A. Asplund, I. M. Olsson, K. Edlund, E. Lundberg, S. Navani, C. A. K. Szigyarto, J. Odeberg, D. Djureinovic, J. O. Takanen, S. Hober, T. Alm, P. H. Edqvist, H. Berling, H. Tegel, J. Mulder, J. Rockberg, P. Nilsson, J. M. Schwenk, M. Hamsten, K. Von Feilitzen, M. Forsberg, L. Persson, F. Johansson, M. Zwahlen, G. Von Heijne, J. Nielsen, and F. Pontén. Tissue-based map of the human proteome. *Science*, 347(6220), 1 2015.

[34] J. van der Westhuizen and J. Lasenby. Techniques for visualizing LSTMs applied to electrocardiograms. 5 2017.

[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 2017-December, pages 5999–6009. Neural information processing systems foundation, 2017.

[36] K. Wnuk, J. Sudol, K. B. Givechian, P. Soon-Shiong, S. Rabizadeh, C. Szeto, and C. Vaske. Deep learning with implicit handling of tissue-specific phenomena predicts tumor DNA accessibility and immune activity. *bioRxiv*, page 229385, 4 2019.

[37] Y. M. Yaniv Leviathan. Google AI Blog: Google Duplex: An AI System for Accomplishing Real-World Tasks Over the Phone, 2018.

[38] Y. Yuan, Y. Shi, C. Li, J. Kim, W. Cai, Z. Han, and D. D. Feng. DeepGene: an advanced cancer type classifier based on deep learning and

somatic point mutations. *BMC Bioinformatics*, 17(S17):476, 12 2016.

[39] J. Zou, M. Huss, A. Abid, P. Mohammadi, A. Torkamani, and A. Telenti. A primer on deep learning in genomics. *Nature Genetics*, 51(1):12–18, 1 2019.

6. Appendix

6.1. Code

Code used for this paper is publicly available on GitHub: https://github.com/AndreCNF/tcga-cancer-classification

Data can be retrieved from the following link: https://gdc.cancer.gov/about-data/publications/pancanatlas

6.2. Labels

Tumor type	Abbreviation	Code	Samples
Acute Myeloid Leukemia	LAML	32	109
Adrenocortical carcinoma	ACC	0	77
Bladder Urothelial Carcinoma	BLCA	1	398
Brain Lower Grade Glioma	LGG	22	509
Breast invasive carcinoma	BRCA	16	1047
Cervical squamous cell carcinoma and endocervical adenocarcinoma	CESC	10	291
Cholangiocarcinoma	CHOL	29	36
Colon adenocarcinoma	COAD	11	436
Esophageal carcinoma	ESCA	31	161
Glioblastoma multiforme	GBM	19	148
Head and Neck squamous cell carcinoma	HNSC	5	507
Kidney Chromophobe	KICH	20	66
Kidney renal clear cell carcinoma	KIRC	14	514
Kidney renal papillary cell carcinoma	KIRP	7	285
Liver hepatocellular carcinoma	LIHC	15	350
Lung adenocarcinoma	LUAD	23	503
Lung squamous cell carcinoma	LUSC	12	495
Lymphoid Neoplasm Diffuse Large B-cell Lymphoma	DLBC	2	47
Mesothelioma	MESO	24	81
Ovarian serous cystadenocarcinoma	OV	17	295
Pancreatic adenocarcinoma	PAAD	8	158
Pheochromocytoma and Paraganglioma	PCPG	25	161
Prostate adenocarcinoma	PRAD	6	473
Rectum adenocarcinoma	READ	13	156
Sarcoma	SARC	9	242
Skin Cutaneous Melanoma	SKCM	4	448
Stomach adenocarcinoma	STAD	30	403
Testicular Germ Cell Tumors	TGCT	26	133
Thymoma	THYM	28	103
Thyroid carcinoma	THCA	21	475
Uterine Carcinosarcoma	UCS	18	56
Uterine Corpus Endometrial Carcinoma	UCEC	3	521
Uveal Melanoma	UVM	27	80

Table 2: Tumor types used as labels in the classification task.