Unpublished - draft

Digital Object Identifier Undefined

Binary Classification of Neuroblastoma using Convolutional Neural Networks

PIERRE MINIER[1] and BEGONYA GARCIA-ZAPIRAIN[2], (Member, IEEE)
[1]Telecommunication department, ENSEIRB-MATMECA, Talence 33400, France
[2]eVida Research Group, University of Deusto, Bilbao 48007, Spain

Corresponding author: Begonya Garcia-Zapirain (e-mail: mbgarciazapi@deusto.es).

This work was supported by the eVida Research Group of the University of Deusto.

ABSTRACT This study focuses on the classification of tumours using convolutional neural networks based on histological images. Specifically, the investigation targets neuroblastoma, a malignant tumour that predominantly affects infants and children and develops in nerve cells. The paper begins by reviewing existing approaches used for automatic classification of neuroblastoma differentiation grades. Subsequently, an algorithm for partitioning databases is proposed. It is designed for biomedical imaging applications with a limited number of images of varying dimensions to respect dataset independence, target proportions, and equal diversity. Then, pretrained convolutional neural networks on ImageNet are modified and fine-tuned. ResNet, VGG, and Inception families are considered. The results suggest that a proposed modified architecture of Inception v.3 provides the best performance for binary classification, with an accuracy of 88.43% and a loss of 0.320 with cross entropy. Finally, an analysis is conducted on metric distributions acquired from the completion of 12,500 training sessions. This analysis aims to facilitate a deeper understanding of the proposed partition and the inherent relationship between accuracy and loss metrics.

INDEX TERMS classification, CNN, dataset independence, fine-tuning, neuroblastoma, partitioning

I. INTRODUCTION

NEUROBLASTOMA is a type of cancer that develops from immature cells of the sympathetic nervous system, and is often diagnosed in children under 5 years old [1]. This cancer can be aggressive and spread rapidly to other parts of the body by metastasis. Diagnosis is usually made by histological imaging, which provides an overview of the composition of the tumour cells, including whether they are mature or immature. The so-called immature cells are more likely to spread to other parts of the body, and therefore make the disease more severe.

To assist medical doctors in grading the severity of neuroblastoma, a classification system called Shimada was developed in the late 90s [2]. Figure 1 presents a simple representation of it. This system is based on the aspect of the tumour cells observed, taking into account their morphology and arrangement. Nevertheless, histological analysis remains a complex and tedious task, requiring both expertise and time.

The investigated solution involves the use of machine learning algorithms, and especially Convolutional Neural Networks (CNN). CNNs are algorithms capable of processing large quantities of images and recognizing complex features in the data. These algorithms are therefore particularly well suited to histological analysis. Patterns extracted by CNNs are not necessarily the same as the ones used by pathologists. It is therefore possible to improve the accuracy and robustness of diagnoses, in contrast to more traditional approaches that rely on algorithms mimicking the doctor's approach.

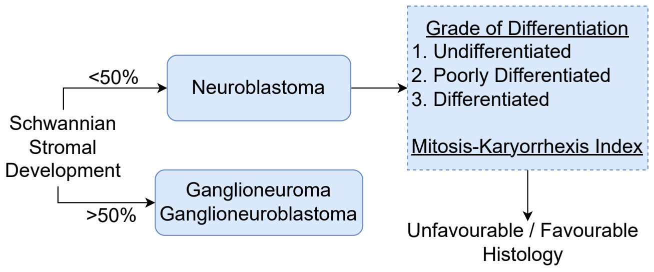


FIGURE 1. A simplified version of the Shimada Classification. FH: Favourable Histology, UH: Unfavourable Histology.

Histology is the study of tissues and cells at the microscopic level. It is a fundamental discipline in anatomy and biology that involves the examination of tissues to understand their structure and function. Whole Slide Images (WSIs) are digital versions of these tissues. They are created by scanning stained tissue sections at high resolution, and capturing an image of the entire slide that can be numerically analysed. In general, the classification of histopathological images is a challenging task due to variations in illumination and shapes.

Figures 2 and 3 present examples of histological WSIs. In this work, only Poorly Differentiated and Differentiated cells are classified to estimate the Grade of Differentiation.

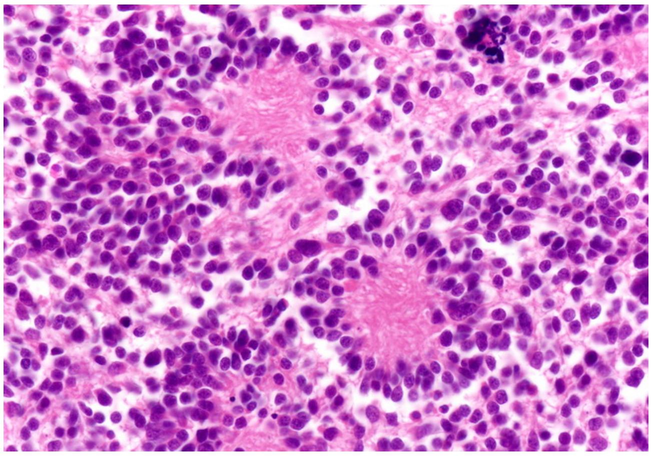


FIGURE 2. WSI labelled as «Poorly Differentiated».

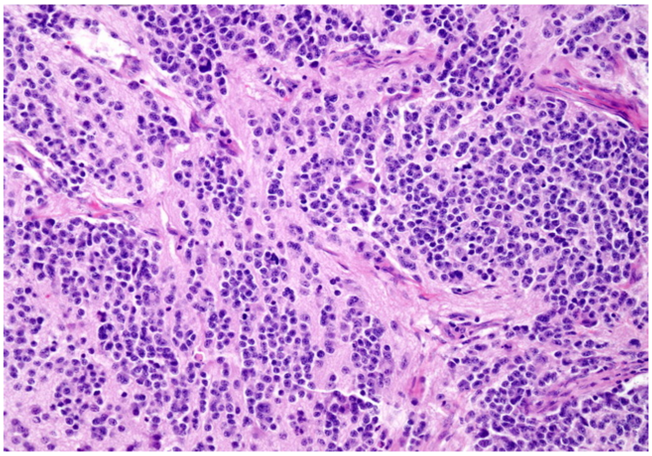


FIGURE 3. WSI labelled as «Differentiated».

II. EXISTING SOLUTIONS

This section delves into the examination of four specific methods for the classification of the Grade of Differentiation based on Neuroblastoma WSIs.

A. SEMANTIC SEGMENTATION FOR FEATURE EXTRACTION (OCTOBER 2008)

In October 2008, a first computer-aided evaluation of neuroblastoma on WSIs is presented [3]. The dataset consisted of 36 cases of neuroblastoma, covering all three subtypes of neuroblastic grading defined by the Shimada System. The images were processed at various scales, starting at a resolution of 64×64 and increasing as necessary, until 512×512.

Semantic segmentation in the Lab colour space was used to divide the image into five classes: nuclei, cytoplasm, neuropil, red blood cells, and background. The segmentation employed a clustering-based approach, maximizing the interclass distance over the intraclass distance. Only two regions were retained: cytoplasm and neuropil. Based on it, 24 features were constructed to describe the current scale-level studied, utilizing four kind of metrics (entropy, mean, variance, and homogeneity) across the three channels (L, a, b). To improve accuracy, the features from the current scale-level and lower scale-levels were combined into a pool, chosen through the Sequential Floating Forward Selection (SFFS) algorithm.

B. PATCHED COMPLETED LOCAL BINARY PATTERN (APRIL 2018)

The Patched Completed Local Binary Pattern (PCLBP) method [4] presents another approach for classifying histopathological images. The study uses a dataset of 1043 images from The Tumour Bank at Kids Research at The Children's Hospital at Westmead.

The proposed method comprises four distinct steps to achieve its objective. Firstly, the preprocessing stage involves dividing the entire image into square patches of equal size. This division facilitates the extraction of localized information from the image. Secondly, the feature extraction step employs the CLBP algorithm to compute two types of binary patterns, namely Sign Binary Patterns (SBP) and Magnitude Binary Patterns (MBP), from each patch. The CLBP algorithm defines a neighbourhood based on a specified radius and number of neighbours.

Next, the feature encoding phase focuses on creating a feature vector for each patch. This is accomplished by generating histograms of the SBP and MBP, which provide a simplified representation of the image. Lastly, the classification step involves comparing the feature vector of the target image with those of other images. Based on this comparison, the final classification of the image is determined.

C. CONVOLUTIONAL DEEP BELIEF NETWORK (MAY 2018)

In May 2018, a Convolutional Deep Belief Network (CDBN) was used [5] with the same dataset of the PCLB study.

The first step involved preprocessing the images using the whitening method. This technique aimed to equalize the variance of the images, removing any correlations in the intensity values. Next, the feature extraction step utilized a CDBN. By stacking multiple Convolutional Restricted Boltzmann Machines (CRBM) hierarchically, the CDBN effectively captured important features from the images. In this study, three CRBM layers were used, incorporating max-pooling to improve the results.

Following the feature extraction, the feature encoding step involved applying a K-Means algorithm. This algorithm generated centroids, forming a codebook that represented the features of the images. To create a concise representation for each image, a histogram was constructed using the formed code words. Finally, the classification step employed a Support Vector Machine (SVM) to classify the images. Leveraging the extracted features and the established codebook, the SVM algorithm made the final classification based on learned patterns and decision boundaries.

D. SIFT/SURF FOR A BOVW CLUSTERING (NOVEMBER 2020)

In November 2020, a study [6] pairs the classical algorithms Scale-Invariant Feature Transform (SIFT) and Speeded Up Robust Features (SURF) with a Bag of Visual Words (BOVW) representation. The same dataset used in two previous covered studies [4], [5].

Firstly, the images were preprocessed by cropping them to a standardized size. Then, feature extraction was performed using SIFT and SURF algorithms. Class balancing was implemented by downsampling the over-represented class features. The remaining features were encoded using BOVW with K-means clustering. Finally, classification was conducted using an SVM. The study found that combining SIFT and SURF did not significantly improve accuracy compared to using either algorithm alone. However, the accuracy was enhanced by employing resampling techniques such as SMOTE or NearMiss, making the models more robust.

E. PERFORMANCE OUTCOMES

The Table 1 provides the reached performances of the four previous algorithms.

Paper	Date	Framework	Accuracy
[3]	10/08	Semantic Segmentation	88%
[4]	04/18	PCLB, Histogram for encoding	76%
[5]	05/18	Boltzmann Machines	84%
[6]	11/20	SIFT, BOVW, K-Means, SVM	90%

TABLE 1. Algorithms classifying the Grade of Differentiation on WSIs.

F. OTHER RELATED WORK TO THE GRADE OF DIFFERENTIATION OF NEUROBLASTOMA

1) Classification based on Hover-Net (October 2022)

A pre-trained Hover-Net model on the PanNuke database is used to accurately segment cells in histological images [7]. Subsequently, various morphological features were extracted for each nucleus, including area, perimeter, convex area and eccentricity. These data were combined with patient-related information such as sex, age, and the nuclear degradation rate in tumour cells (called Mitosis-Karyorrhexis Index [8] (MKI)). Subsequently, classification algorithms like Random Forest and K-NN were trained to classify patients into two groups: high-risk and low-risk. The best model classifies with an accuracy of 86.5%.

2) New characteristics for the Shimada System (January 2023)

The authors [9] recommend to incorporate image entropy (S), fractal dimension (FD) and lacunarity (λ) features alongside the two primary ones, namely Grade of Differentiation and MKI [8], for a more robust and refined classification. S measures the complexity or randomness of the image; FD characterizes the roughness or irregularity of the structures within the image; and λ measures the heterogeneity of the image texture. These parameters were combined into a machine learning algorithm and reached an accuracy of 82% to detect tumour malignancy.

III. DATABASE

A. IMAGE DIMENSION

In the context of a CNN application, it is essential for the input images to have uniform dimensions. Typically, popular networks like VGG, ResNet, and DenseNet use an input size of 224×224 pixels. This particular size was chosen due to its divisibility into 32×32 pixel squares, which facilitates efficient parallel processing on GPU architectures. Moreover, this size is generally sufficient for capturing the requisite patterns for recognition. However, it should be noted that this is not an absolute rule, as the Inception architecture, for instance, employs a format of 299×299 pixels.

In this study, VGG, ResNet, and Inception architectures are examined. To ensure consistency in the datasets and enable comparison of training sessions across different architectures, the images are cropped to dimensions of 250×250 pixels. For models using the 224×224 pixel format, a random crop is performed at each iteration to form an image batch for the given optimisation process. For the 299×299 pixel format, resizing is achieved through interpolation. The choice of the 250×250 pixel format minimizes pixel loss since the patches do not overlap, and the dimensions of WSIs in our database are often multiples of 250.

The use of cropping is preferred over large resizing, as it allows for a significant increase in the number of patches without distorting the morphology or existence of the observed cells. Also, a pre-crop step is done to get rid of artefacts or vignetting on the border. However, patching WSIs has a drawback: not all patches contain cells. Therefore, patches must be reviewed and filtered.

B. DATASET DEFINITIONS

Three datasets must be created. These datasets are called «training», «validation» and «test» . One should partition the set of valid patches with the following ideal proportions: 70%, 15% and 15%. The training dataset is used to learn the weights, while the validation dataset to supervise this training. The test dataset is only used when the model is trained and allows its performance to be evaluated independently of the data used during the training stage.

C. INDEPENDENCE AND DIVERSITY OF THE DATASETS

The difficulty revolves around the notion of independence. Indeed, patches obtained from the same original image possess a certain correlation and must therefore be associated with the same dataset. Otherwise, there are two risks. Firstly, of having an overfitted model, i.e., a model that has memorized all the patterns without generalizing well. Secondly, of being unable to detect this overfitting.

Moreover, achieving a proper partitioning is not straightforward, as the number of valid patches varies depending on the original dimensions of the WSIs and the pre-cropping.

Furthermore, the datasets must be sufficiently diverse to enable good generalization (the 'train' dataset), effective training monitoring (the 'validation' dataset), and accurate measurement of real-world performance (the 'test' dataset). For example, it is important to avoid a situation where very large WSIs dominate the small validation and test datasets.

D. ITERATIVE FILING PARTITIONING

The Iterative Filing Partitioning (IFP) method is the name of the developed algorithm to partition a given set of correlated patches into independent and diverse datasets. One class label is considered at a time.

1) Problem Statements

The objective is to partition a set of patches into three groups P_1, P_2, and P_3. This means that the pixel proportions p_1, p_2, and p_3 associated with the partitions P_1, P_2, and P_3 must sum to 1. One sets $p_1 > p_2 \geq p_3$. A key assumption of the problem is $p_1 > 0.5$. Indeed, a partition must contain at least half of the available pixels, and will be associated with the training dataset. Additionally, target diversities d_1, d_2, and d_3 are assigned to partitions P_1, P_2, and P_3. In our case, $d_1 = d_2 = d_3 = 1/3$.

2) Initialisation

Patch families are sorted according to their number in descending order. Families that can not be assigned to P_2 and P_3 because they are too large are assigned to P_1. If N_1 families are assigned to P_1, then P_2 and P_3 are deficient in N_1 families respectively. So in turn, $2 \times N_1$ remaining families with the largest number of patches are assigned to P_2 and P_3. If a family cannot be assigned, it is ignored and will be assigned later. In this case, another family with fewer patches is chosen to reach N_1 for P_2 and P_3.

IFP is sensitive to the initialization where P_1 is filled with N_1 images. A hyperparameter μ is introduced such that N_1 is the number of families gathering a proportion of total pixels greater than or equal to $\mu \times min(p_2, p_3) = \mu \times p_3$. This hyperparameter μ belongs to $]0, 1]$ and allows P_1 absorbing the families saturating too quickly P_2 and P_3, preventing them from reaching their target diversity.

3) Iterations

Families are alternately assigned to a partition. The largest remaining ones are always examined first. Partitions P_2 and P_3 are considered filled when no remaining family can prevent them from exceeding the proportions $p_2 + \varepsilon$ and $p_3 + \varepsilon$, or when all families have been assigned. In the end, any remaining families are assigned to P_1.

The hyperparameter ε allows an excess, because a situation where $p_2 = 16\%$ for a target of 15% is equivalent to a situation where $p_2 = 14\%$ in terms of raw error.

4) Optimisation

An exhaustive search for the optimal (μ^*, ε^*) is performed. Let the proportions p_i and the diversities d_i effectively obtained be noted with an eff power. The metric (1) is equivalent to averaging the average differences reached on the proportions and diversities.

$$D(\mu,\varepsilon) = \frac{1}{6}\left[\sum_{i=1}^{3}|p_i - p_i^{eff}(\mu,\varepsilon)| + \sum_{i=1}^{3}|d_i - d_i^{eff}(\mu,\varepsilon)|\right] \tag{1}$$

$$(\mu^*, \varepsilon^*) = \underset{\substack{\mu\in[0,1] \\ \varepsilon\in[0,1]}}{\operatorname{argmin}}\, D(\mu,\varepsilon) \tag{2}$$

In practical terms, equation (2) means finding the coordinates or areas minimizing a heatmap containing all the values of D for the μ and ε hyperparameters, like the one presented in Figure 4, and retaining the corresponding partition.

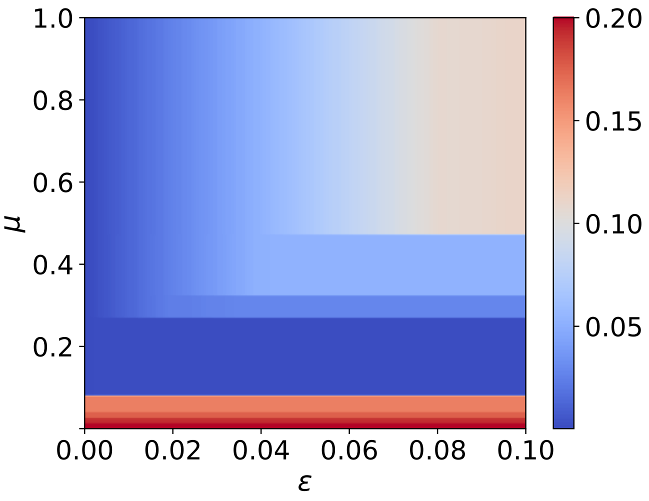


FIGURE 4. Heatmap to minimize, i.e. finding the bluest point or area.

5) Discussion

In the proposed metric (1), an implicit equivalence is formulated: a discrepancy of 1% in proportion is considered equally incorrect as a discrepancy of 1% in diversity. However, the current study does not establish that this equivalence is the optimal choice for a CNN training application. It is possible that alternative correspondences exist, wherein the two sums in equation (1) are differently weighted.

E. CLASS IMBALANCE

Class imbalance occurs when a dataset has one of its classes under-represented in terms of number of images. This leads to generalization problems during training and makes it difficult to interpret certain metrics during the evaluation phase. Two strategies exist: subsampling the over-represented classes or artificially creating new images for the minority classes.

1) Challenge

The challenge is twofold: solving the imbalance class problem with a number of transformations, and if possible increasing all classes with the remaining transformations. To achieve this, the two extreme classes must first be considered: the one with the fewest images and the one with the most images.

In our application framework, few images are available, so augmentation methods are used. Only global geometric transformations (rotation and flip) are considered, and not transformations involving local geometric deformations, in order to preserve the characteristics of the cells. For square format images, there are seven transformations that do not alter the internal geometries.

2) Assigning Transformations

Let r the ratio between the two extreme classes, then balancing requires $r-1$ transformations. The collective increase takes place at a rate of 1 for r. Therefore, by noting n the number of transformations applied to the majority class, it is necessary to provide $r \times n + (r-1)$ for the minority class.

For the intermediate classes, the same formula applies, only the coefficient r differs, which should necessarily be less than or equal to the minority class one.

3) Application

Let t the number of available transformations. The relation (3) optimizes the use of these transformations, with r the ratio of the minority class previously defined.

$$r \times n + (r-1) \leq t \tag{3}$$

By isolating n, it can be deduced that all classes are minimally augmented by $\left\lfloor \frac{t+1}{r} \right\rfloor - 1$ transformations, with $\lfloor . \rfloor$ as the floor operator. However, there is no available transformation for collectively increases all classes when $(t+1)/r < 1 \Leftrightarrow t < r-1$. Therefore, the Number of Global Class Augmentation (NGCA) is given by equation (4).

$$NGCA = \begin{cases} \left\lfloor \dfrac{t+1}{r} \right\rfloor - 1 & \text{if } t \geq r-1 \\ 0 & \text{else} \end{cases} \tag{4}$$

In our database, 61 WSIs provided 1,570 patches distributed into two classes: Differentiated (D) and Poorly Differentiated (PD). The D class was in the minority with a ratio $r = 2$ on the three datasets. Balancing and augmentation were achieved with all $t = 7$ transformations for D and three transformations for the majority class PD. Table 2 gives the raw numbers of images before and after balancing and augmentation. The Final Augmentation column describes the sample distribution in the datasets used for training.

Class (Dataset)	Starting	Balancing	Final Augmentation
PD (training)	734	734	**2,936**
D (training)	420	840	**3,360**
PD (validation)	127	127	**508**
D (validation)	82	164	**656**
PD (test)	127	127	**508**
D (test)	80	160	**640**

TABLE 2. Number of samples in the datasets through transformations

F. STANDARDISATION

The order in which class imbalance addressing and standardization should be performed lacks a universal rule. Nevertheless, it is typically advised to balance datasets before scaling them. This recommendation stems from the fact that achieving precise classification outcomes necessitates consistent application of scaling to all classes. This practice guarantees that features are scaled uniformly, which can prove crucial for various machine learning algorithms.

To standardize a given dataset of RGB images, the mean, and the variance are computed across the three-colour channels of each image. Each pixel value is then transformed by subtracting the mean and dividing by the standard deviation, ensuring that the dataset is centred around zero and has a standard deviation of one.

This operation is performed on training, validation, and test datasets independently to ensure that each one is transformed to have a similar distribution. It also prevents information leakage between datasets, as standardization is not performed on the entire database as a whole.

G. EXPERIMENTS

In order to test the approach developed in this section III, several experiments are performed. The objective is to study the behaviour of the training curves when the training and the validation datasets are correlated. This correlation means that patches belonging to the same family are found in these two datasets. The test dataset always remains independent of the two others, allowing to expose, when necessary, the training biases.

It is important to note that the same architecture, the same optimisation algorithms and the same hyperparameters (learning rate, momentum, batch sizes, number of epochs) are used. Only the database partition is modified.

1) Experiment 1: complete independence

In experiment 1 illustrated by the Figure 5, all datasets are independent. The objective is to understand the dynamics occurring throughout the training process for the selected model. This one is designed to not generalize on the data and to be able to memorize the samples of the training dataset.

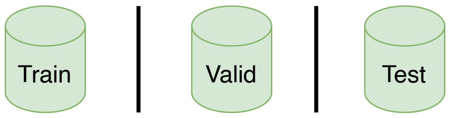


FIGURE 5. Experiment 1: training and validation datasets are not correlated.

2) Experiment 2: low correlation

In experiments 2 illustrated by the Figure 6, the datasets are more or less correlated, but remain relatively independent because the exchange involves a single patch per family and some families contain more than 150. The number of patches in each dataset remains unchanged after this exchange, so the diversity criteria is respected.

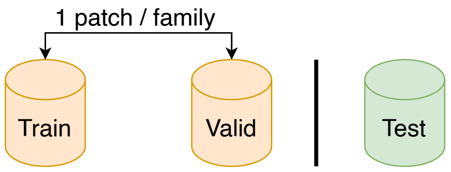


FIGURE 6. Experiment 2: training and validation datasets are lowly correlated.

3) Experiment 3: high correlation

In experiments 3 illustrated by the Figure 7, the samples for the training and the validation datasets are randomly assigning. The correlation is significant and stronger than in Experiments 2.

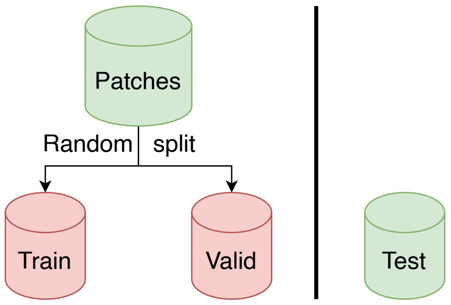


FIGURE 7. Experiment 3: training and validation datasets are highly correlated.

4) Results

Each experiment was conducted ten times. Figure 8 depicts the mean loss across epochs for both training and validation datasets. The magnitude of loss increases with greater independence between the datasets. Experiment 2 exhibited a similar pattern to Experiment 1 with a minor magnitude, indicating that some correlation is present.

However, the dynamics of Experiment 3 differed significantly, with the plain and dot red curves closely aligned,

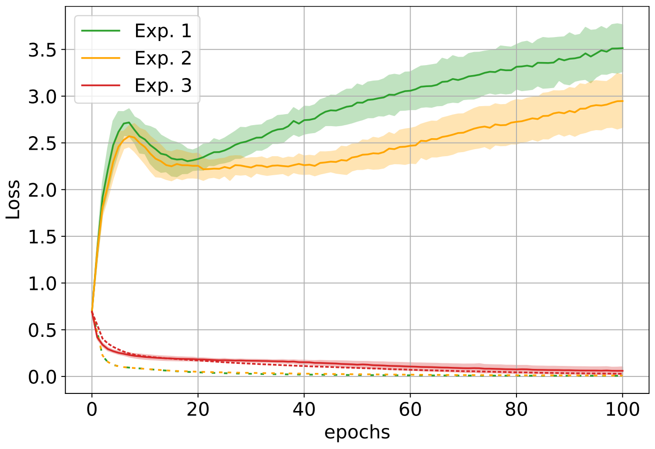


FIGURE 8. Loss for the three experiments. Validation in solid lines, opaque areas represent the means $\pm$ their standard deviations. Training in dotted lines.

suggesting effective generalization to the validation dataset. Notwithstanding, the independent test dataset resulted in a loss of 0.7, resembling the initial epoch, implying an important correlation issue between the training and validation datasets during the training phase.

An additional indication of correlation is in the delay between the green and orange dotted curves and the red one. In Experiment 3, the model requires more time to reach a specific loss threshold due to the increased pattern complexity resulting from the correlation.

IV. PROPOSED METHOD

A. OVERVIEW

To address the issue of a limited number of images, pre-trained backbones are often employed with transfer learning. However, the optimisation process remains time-consuming, making it impractical to test all possible combinations of hyperparameters. Consequently, only the learning rate and the batch size are investigated in this study, while keeping the optimizer and loss function unchanged (Stochastic Gradient Descent (SGD) and Binary Cross Entropy (BCE)).

To begin, a preliminary step is taken to identify promising backbones that require minimal modifications to the original architecture. These selected backbones are then subjected to more time-intensive strategies for further evaluation. The first strategy involves introducing non-linearities at the end of the backbone, while the second one entails unfreezing additional layers from their original weights. Figure 9 provides an overview of all aforementioned procedures.

B. METRICS

1) Search Grid

During the training process, a level of randomness is present. Hence, when searching for optimal hyperparameters, such as the network version, learning rate, or batch size, several training sessions are conducted using the same values.

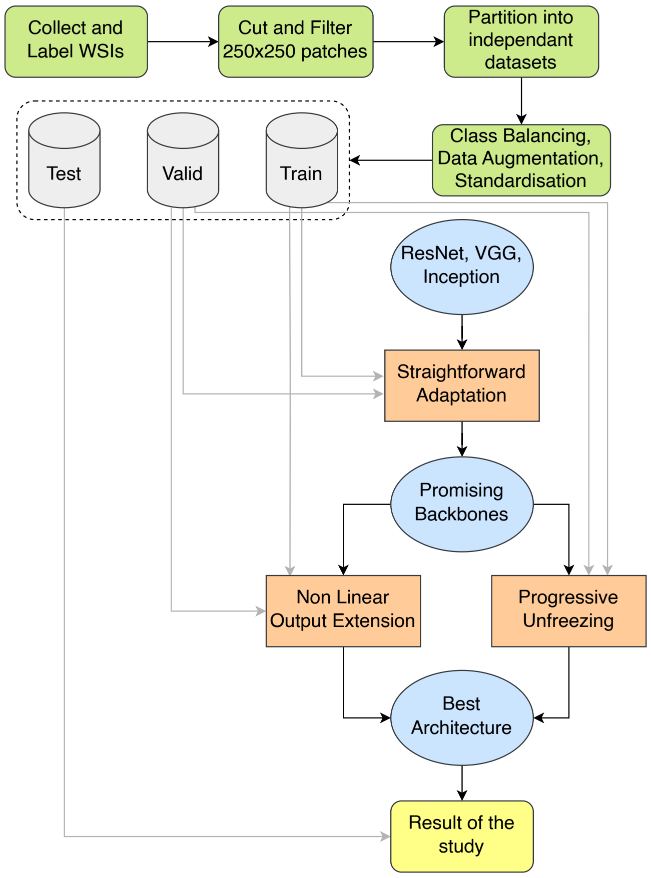

Collect and Label WSIs
Cut and Filter 250x250 patches
Partition into independant datasets
Class Balancing, Data Augmentation, Standardisation
Test
Valid
Train
ResNet, VGG, Inception
Straightforward Adaptation
Promising Backbones
Non Linear Output Extension
Progressive Unfreezing
Best Architecture
Result of the study

FIGURE 9. Overview of the method.

2) Limitations of Validation Curves

In general, the curves obtained on the training dataset are better than those on the validation dataset. The reason is that the optimisation is performed on the training one. Nevertheless, in few and rare cases, a particular initialization can lead to the curves where the validation is better than the training on the first epochs. In such scenarios, solely relying on the validation curve to determine the performance of the model is inadequate.

For the accuracy, the maximum of the minimum of the two curves is considered. For the loss, the minimum of the maximum of both curves is considered since the loss must be minimized, and the accuracy maximized.

C. CNN BACKBONES

1) VGG

Visual Geometry Group (VGG) [10] repeats small (3x3) convolution blocks several times and stack them on top of each other. The dimension reduction is performed by max pooling layers: it keeps a single maximum value for an entire window.

The last layer of the VGG architecture is a fully connected with 4096 inputs and 1000 outputs for the 1000 ImageNet classes. The classification part is independent of the versions and consists of 3 fully connected layers. Only the number of convolutional layers becomes more important as the versions are higher. VGG is available in four versions: 11, 13, 16, and 19. The relationship between the version v and the number of convolutional layers c can be expressed as $v = c + 3$.

2) ResNet

Residual Network (ResNet) [11] uses basic modules that contain two parallel branches. The main one performs a linear transformation, and the residual one adds the original input to the output of the main branch. This avoids the problem of the vanishing gradient that occurs when the depth of the network is increased.

The last layer of the ResNet architecture is a fully connected layer that varies in the number of inputs. For versions 18 and 34, it has 512 inputs, while for versions 50, 101, and 152, it has 2048 inputs. This layer has 1000 outputs and a bias term. The 1000 outputs correspond to the probability of each of the 1000 classes in the ImageNet dataset.

3) Inception

Inception [12] combines several different types of convolution blocks into a single module. These blocks include convolutions of different filter sizes (1x1, 3x3, 5x5) and pooling operations. The objective is to allow the network to capture patterns at different spatial scales. There are several versions of Inception, ranging from Inception v.1 to Inception v.4. Its final layer is a fully connected with 2048 inputs and 1000 outputs.

4) Performance on ImageNet

Table 3 presents the performance on ImageNet of the backbones previously discussed. Weights leading to these results are available online at https://pytorch.org/vision/stable/models.html#classification.

Acc @5 denotes the accuracy of accurately identifying the class within the first five predicted categories.

FLOPS stands for "Floating Point Operations Per Second. It is a metric used to measure the computational cost of training architecture models.

Backbone	Acc @1	Acc @5	No. Params	FLOPS
ResNet-18	69.76%	89.08%	11.7M	2B
ResNet-34	73.30%	91.42%	22M	4B
ResNet-50	76.15%	92.87%	26M	4B
ResNet-101	77.37%	93.56%	45M	8B
ResNet-152	78.31%	94.06%	60M	12B
VGG-11	69.02%	88.63%	133M	8B
VGG-13	69.93%	89.25%	133M	11B
VGG-16	71.59%	90.38%	138M	15B
VGG-19	72.38%	90.88%	144M	20B
Inception v.3	77.29%	93.45%	27.2M	5.71B

TABLE 3. Pre-trained backbone performances on ImageNet.

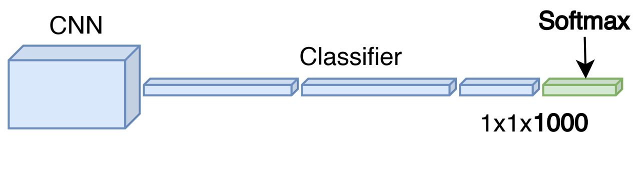
Other pre-trained backbones on ImageNet can perform a classification task. These include the EfficientNet [13] and

DenseNet [14] families. But our study is limited in time, and so arbitrary choices lead to these technologies being discarded.

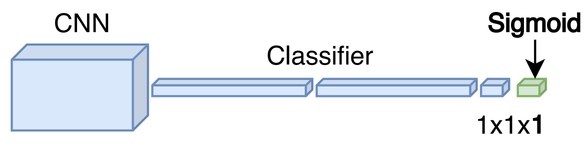
V. STRAIGHTFORWARD ADAPTATION

A. IDEA

The straightforward adaptation involves making minimal modifications to the architecture. Specifically, the final layer must be modified to have a number of outputs that corresponds to the number of classes being predicted. Exception in the case of binary classification where a single output can be adequate, as illustrated by Figure 10. The adapted layer is subsequently trained using the specific dataset. The other layers keep their original weights.



(a) ImageNet Classifier (pretrained).



(b) Binary Classifier (used).

FIGURE 10. Straightforward adaptation of the classifier.

B. RESULTS

Figure 11 shows the performance of the different backbones considered with a straightforward adaptation. Three families of backbones are delimited by dotted grey lines: Inception, ResNet and VGG. The optimisation is performed on the loss only, and the corresponding accuracy at the same epoch

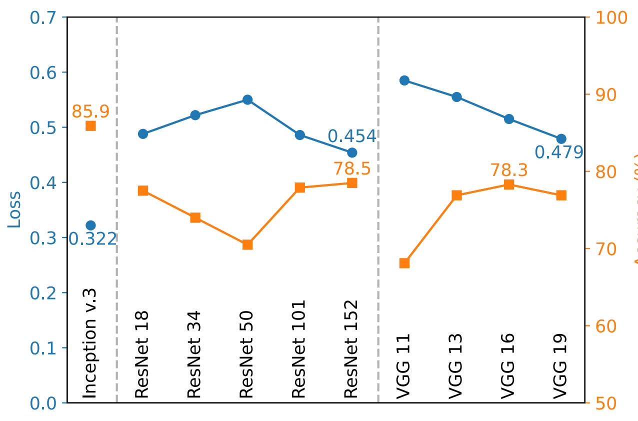


FIGURE 11. Best loss and corresponding accuracy for the studied backbones with a straightforward adaptation.

is given. The best backbone according to both metrics is Inception v.3 with an accuracy of 85.9%.

An interesting point from this figure is the noticeable behaviour when considering more and more sophisticated VGG architectures. As the number of convolutional layers increases, the loss drops. However, there is no proportional or simple link between the accuracy and the loss, whereas there is a kind of mirror effect for ResNet curves.

The best ResNet architectures are versions 18 and 152. Unlike VGG, loss does not decrease with increasing versions. This can be explained by the variation in the classifier part. In fact, the last layer is a fully connected layer that varies in the number of inputs. For versions 18 and 34, it has 512 inputs, while for versions 50, 101, and 152, it has 2048 inputs.

Finally, promising backbones, as named in Figure 9, are Inception v.3, ResNet 18, ResNet 152 and VGG 19. In the following, more computational strategies are tested to improve their respective loss.

VI. NON-LINEAR OUTPUT EXTENSION

A. IDEA

A more computational approach consists in keeping the backbone with its trained weights and replacing the last layer with more than one to be trained. Between these new layers, a non-linearity function is added. This gives the network more degrees of freedom during the training phase.

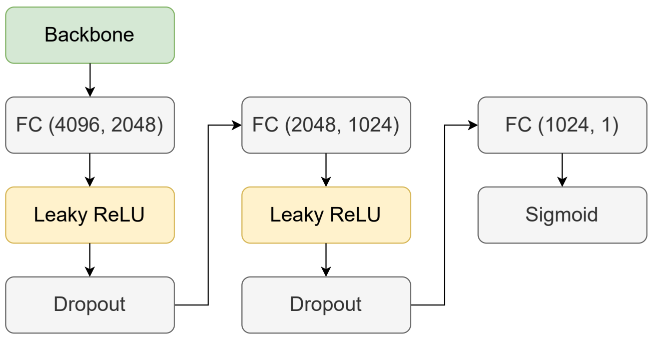


FIGURE 12. Example of a non-linear output extension. Two non-linearities are added. FC(a, b): Fully Connected with a inputs and b outputs.

The added layers are fully-connected, and the number of inputs is halved from the previous one. A dropout of 0.5 is set to randomly select half of neurons and ignore them during one epoch to prevent overfitting. Figure 12 illustrates this kind of classifier with 2 non-linearities in yellow.

B. RESULTS

Figure 13 provides the results for the non-linear output extension strategy. The loss is given as a function of the number of non-linearities introduced. The abscissa 0 corresponds to the result obtained with the straightforward adaptation with the addition of 0 non-linearity (figure 11). Each point was obtained via a search grid.

For all tested backbones, a right-chosen extension provides an equal or a better loss than the straightforward adaptation.

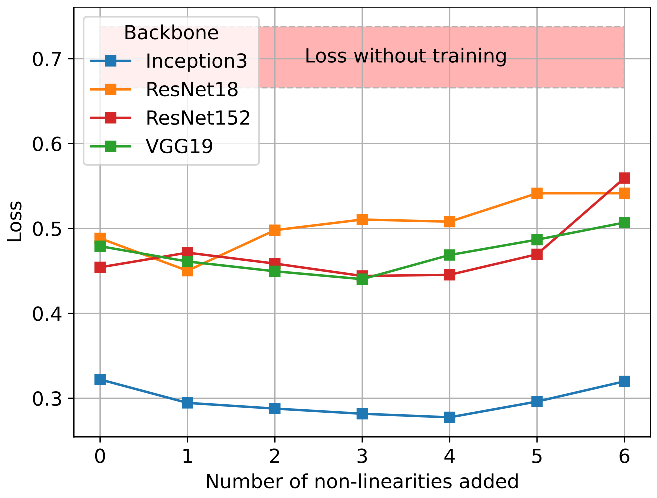


FIGURE 13. Loss obtained with a non-linear output extension.

In particular, the Inception v.3 backbone with four non-linearities added reaches a loss of 0.278. Once a certain number of layers have been added, the loss function becomes higher.

The red opaque region represents the loss obtained from the training and validation datasets at epoch 0. It corresponds to the average ± standard deviation obtained from all the training sessions carried out during this work. Within this red domain, models are akin to random decision-making. This area is justified later, in section IX.

VII. PROGRESSIVE UNFREEZING

A. IDEA

In the context of CNN, first layers extract general features, while last layers become more specific to the problem being studied. The straightforward adaptation approach consists in modifying only the last layer to fit the number of expected outputs. This layer is represented in orange in Figure 14.

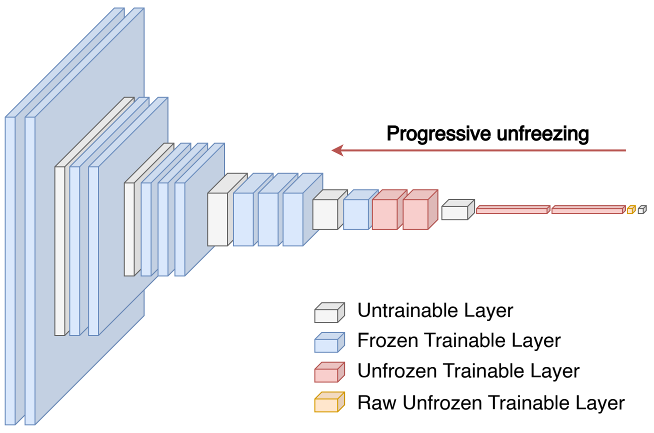


FIGURE 14. Example of a network with four unfrozen layers which are fine-tuned, while the last one is still trained from a random initialisation.

This section develops this approach further. The weights of the last layer are still randomly initialized, and a certain number of layers preceding the last one are unfrozen and trained from the ImageNet initialization. This idea is illustrated by the red arrow in Figure 14: more and more layers are unfrozen in order to find the right tuning approach.

B. RESULTS

Results are given in Figure 15. The approach seems appropriate for VGG 19. Indeed, a gain of almost 0.2 on loss is observed: this corresponds to a reduction of 40% comparing to the straightforward adaptation. Progressive unfreezing is optimal here for six layers initialized using ImageNet weights. Beyond that, the amount of data does not seem to allow optimisation deeper into the network.

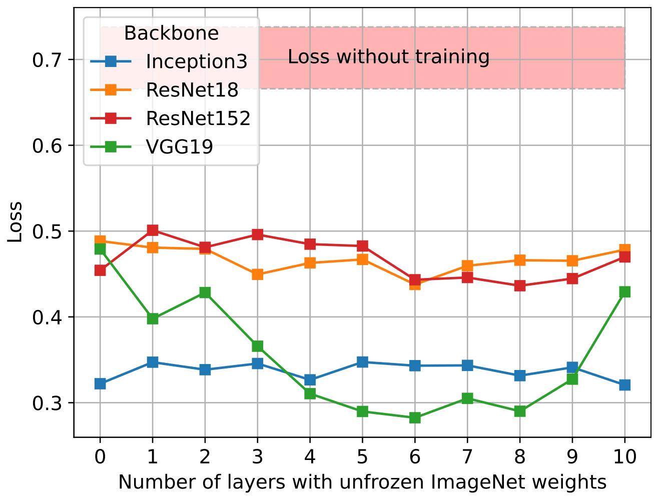


FIGURE 15. Loss obtained with the progressive unfreezing strategy.

For other networks, the approach seems less convincing: in particular for Inception v.3. This can be explained by its parallel architecture, which requires many more layers to be unfrozen. However, there is a risk that fine-tuning will no longer be possible, and the loss will increase, as is the case with VGG 19 with more than 7-8 unfrozen layers.

VIII. RESULTS SUMMARY

The table 4 summarizes the best metrics obtained for the three adaptation strategies studied. The model retained is Inception v.3 with an extension of four non-linearities. Figures

Strategy	CNN	Modification	Loss	Acc.
Straightforward	Inception v.3	/	0.322	85.90%
Extension	Inception v.3	4 non-linearities	**0.278**	**88.36%**
Unfreezing	VGG 19	6 unfrozen layers	0.282	88.09%

TABLE 4. Best results on the validation dataset

16 and 17 show training curves of the best training session performed, which led to a loss of 0.278 on the validation dataset. The learning rate is 3×10^{-5}, the batch size 8, and the optimizer SGD. For those same parameters, on the

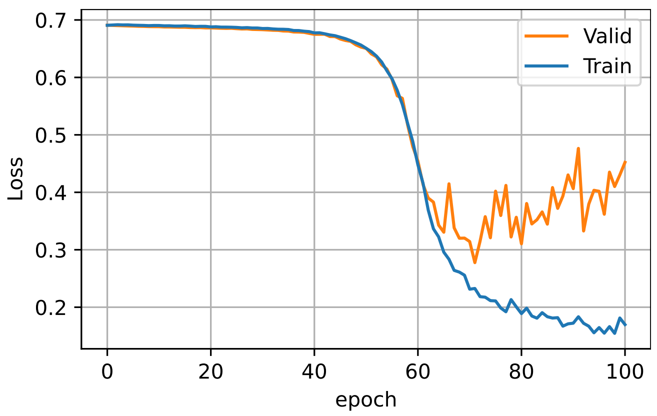


FIGURE 16. Best loss curve. Obtained with Inception v.3 as backbone and an extension of four non linearities as illustrated on Figure 12.

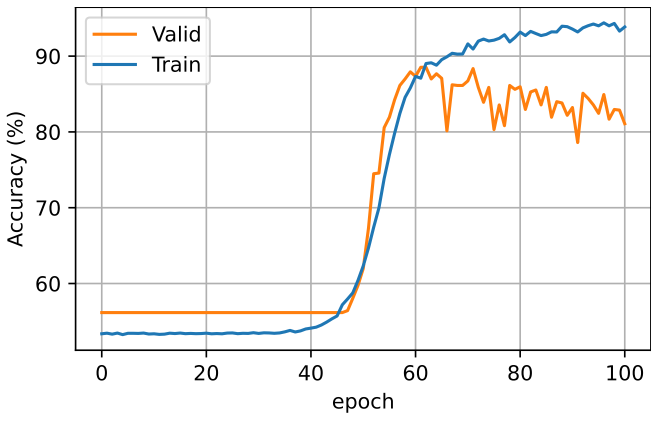


FIGURE 17. Accuracy curve corresponding to Figure 16.

10 training sessions performed, the mean of the best loss is 0.310 and the mean of the best accuracy 86.95%. On the test dataset, independent of those used for training, the loss is **0.320** and the accuracy is $\mathbf{88.43\%}$.

IX. DISTRIBUTION

Throughout this study, a total of 12,500 training sessions were conducted, generating a substantial volume of data. Those results offer ample opportunity for analysis. Leading to a better comprehension of the metrics under investigation, namely the loss and accuracy, as well as the performed database partitioning.

A. LOSS DISTRIBUTION WITHOUT TRAINING

Figure 18 depicts the distribution of the loss across individual training sessions conducted at epoch 0. It represents the initial state of the model prior to any optimisation. It is noteworthy that the distributions of used datasets in the training process show a reassuring level of overlap.

Without few outliers after 0.85, this loss distribution seems to be Gaussian. This is why the loss without training is represented by an area of the mean $\pm$ the standard deviation in Figures 13 and 15.

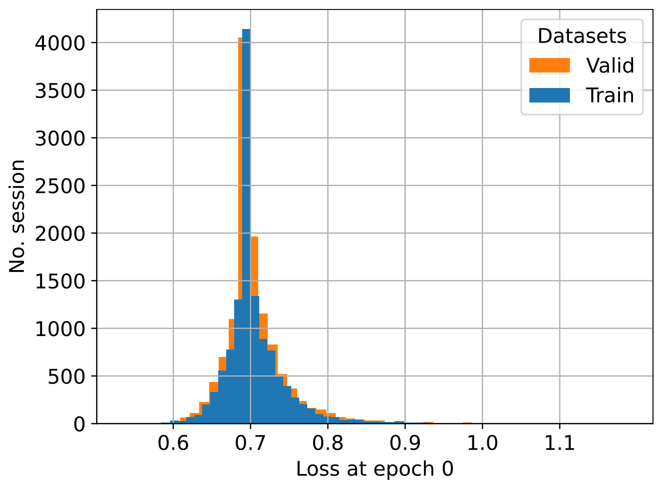


FIGURE 18. Loss distribution without training. Mean: 0.705, standard deviation: 0.042.

B. ACCURACY DISTRIBUTION WITHOUT TRAINING

Figure 19 gives the equivalent of Figure 18 for the accuracy. The result is quite different and highlights properties of the datasets and the metric itself.

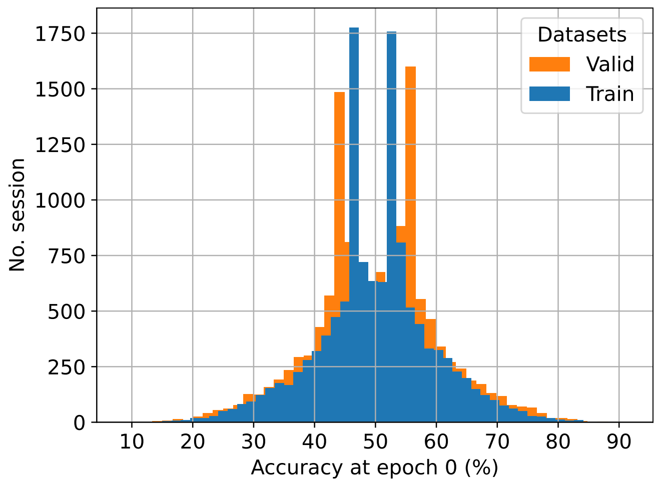


FIGURE 19. Accuracy distribution without training. Mean: 50.0%, standard deviation: 9.95%.

The average obtained is 50%, which is usual for a binary random classification. However, the distribution does not have a Gaussian shape due to the presence of two peaks around the mean value. One possible explanation lies in the correlation between the patches in large images used in the different datasets. If a patch is classified in a certain way, it is likely that the rest of the patches will also be classified in the same way, resulting in an entire part of the dataset. The imbalance arises from the fact that the number of patches extracted is not the same for each WSIs.

Therefore, the binary nature of the accuracy affects the distribution. However, this is not the case for the distribution according to the loss function, as the loss is continuous.

C. LOSS AND ACCURACY ON A 2D DISTRIBUTION

Figure 20 shows a 2D histogram describing the distribution of the pairs (accuracy, loss) for common epochs through all training session performed. A strong match is indicated by lighter colours. It can be seen that the relationship between accuracy and loss is non-linear and quite diffuse (represented by purple areas), although a trend can be discerned in yellow-orange.

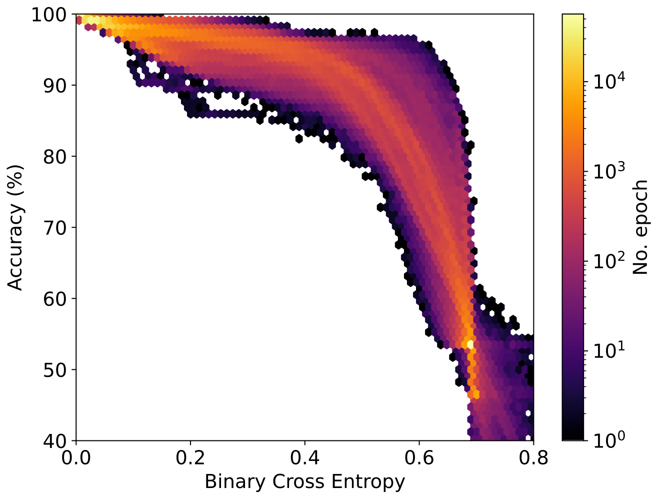


FIGURE 20. Loss and Accuracy distribution on a 2D histogram with a logarithmic scale.

This clearly shows that these two metrics are different and both deserve to be analysed with their various properties in mind.

X. CONCLUSION

The state of the art proposes solutions for classifying neuroblastoma images with scores ranging from 84% to 90%. The present study reaches this level with a score of 88.43% for binary classification.

The approach studied uses CNN, and requires datasets to train, supervise and test models. These datasets are made from a database with certain characteristics that require particular attention: variable image dimensions, and a low number of images for Deep Learning applications. This led to the development of a partitioning algorithm that takes into account target proportion and diversity criteria, before increasing the data. These constraints are in addition to respecting dataset independence. The underlying question behind this independence criterion is as follows: can non-overlapping patches from the same image distort the interpretation of standard metrics such as accuracy or the loss function? A simple experiment was set up, and the conclusion was clear: Yes, because these patches are still correlated. They should therefore be assigned to the same dataset (training, validation, or test) and not mixed.

The selected model is based on the architecture of Inception v.3, but it is worth mentioning that VGG 19 also yields interesting results. Both models have demonstrated good performance, albeit employing different strategies. The metrics on Inception v.3 are refined by adding non-linearities at the top of the network while freezing the entire backbone with the pre-trained weights from ImageNet. On the other hand, the most relevant approach for VGG 19 is to retain the original architecture while fine-tuning not only the last layer but all classifier layers and four layers of the convolutional part.

XI. FUTURE WORK

It should be noted that this study is not exhaustive, as it only explores three major families of CNN, while other families such as EfficientNet or DenseNet exist. Additionally, the classification performed in this study is binary, whereas the Schimada system recommends three classes: Poorly Differentiated, Differentiated, and Undifferentiated for the studied tumour. It would be beneficial to consider adding the missing class to provide a comprehensive model.

XII. LIMITATIONS

It is important to acknowledge that the dataset used in this study consisted of only 61 WSIs, which were later transformed into 1,570 patches and further augmented to 8,608 samples. This database size is relatively small for training or fine-tuning certain CNN architectures that typically possess millions of parameters.

Only SGD was utilized for model optimizations. Other optimizers, such as Adagrad or Adam, exist that could potentially enhance model performance. However, exhaustively testing all available optimizers would require significant time and/or higher computational power. Therefore, we chose to focus on SGD as the baseline optimizer, striking a balance between expected performance and available resources.

REFERENCES

[1] National Cancer Institute. Neuroblastoma treatment (pdq) - health professional version. https://www.cancer.gov/types/neuroblastoma/patient/neuroblastoma-treatment-pdq.

[2] Hiroyuki Shimada, Inge M. Ambros, Louis P. Dehner, Jun-ichi Hata, Vijay V. Joshi, Borghild Roald, Daniel O. Stram, Robert B. Gerbing, John N. Lukens, Katherine K. Matthay, and Robert P. Castleberry. The international neuroblastoma pathology classification (the shimada system). Cancer, 86(2):364–372, 1999.

[3] J. Kong, O. Sertel, H. Shimada, K.L. Boyer, J.H. Saltz, and M.N. Gurcan. Computer-aided evaluation of neuroblastoma on whole-slide histology images: Classifying grade of neuroblastic differentiation. Pattern Recognition, 42(6):1080–1092, 10 2008. Digital Image Processing and Pattern Recognition Techniques for the Detection of Cancer.

[4] Soheila Gheisari, Daniel Catchpoole, Amanda Charlton, and Paul Kennedy. Patched Completed Local Binary Pattern is an Effective Method for Neuroblastoma Histological Image Classification, pages 57–71. Springer, Singapore, 04 2018.

[5] Soheila Gheisari, Daniel R. Catchpoole, Amanda Charlton, and Paul J. Kennedy. Convolutional deep belief network with feature encoding for classification of neuroblastoma histological images. Journal of Pathology Informatics, 9(1):17, 05 2018.

[6] Panta Adhish, Matloob Khushi, Naseem Usman, Kennedy Paul, and Catchpoole Daniel. Classification of neuroblastoma histopathological images using machine learning. In Neural Information Processing, pages 3–14, Cham, 11 2020. Springer International Publishing.

[7] Yanfei Liu, Yuxia Jia, Chongzhi Hou, Nan Li, Na Zhang, Xiaosong Yan, Li Yang, Yong Guo, Huangtao Chen, Jun Li, et al. Pathological prognosis

classification of patients with neuroblastoma using computational pathology analysis. Computers in Biology and Medicine, 149:105980, 2022.

[8] Neha Bhardwaj, Manish Rohilla, Amita Trehan, Deepak Bansal, Nandita Kakkar, and Radhika Srinivasan. Mitosis-karyorrhexis index evaluation by digital image visual analysis for application of international neuroblastoma pathology classification in fna biopsy. Cancer Cytopathology, 130(2):128–135, 2022.

[9] Irene Donato, Kiran Velpula, Andrew Tsung, Jack Tuszynski, and Consolato Sergi. Demystifying neuroblastoma malignancy through fractal dimension, entropy, and lacunarity. Tumori Journal, page 030089162211462, 01 2023.

[10] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[12] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014.

[13] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. CoRR, abs/1905.11946, 09 2020.

[14] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks, 2018.